Cisco APIC Layer 2 Networking Configuration Guide, Release 3.x and Earlier

First Published: 2017-04-11

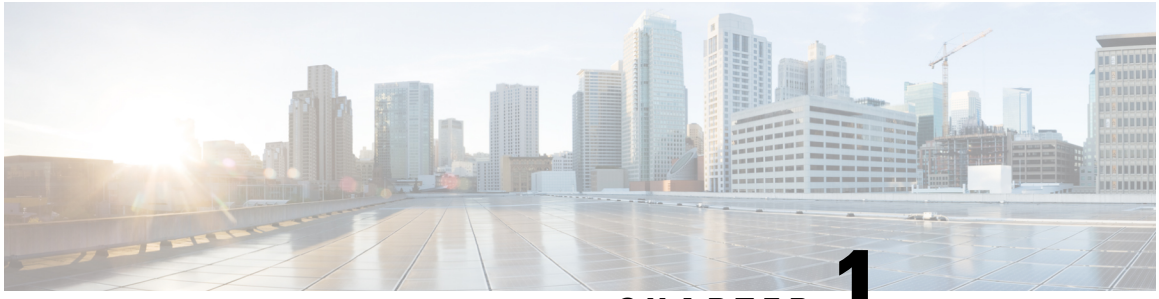
Last Modified: 2024-01-19

Americas Headquarters
Cisco Systems, Inc.
170 West Tasman Drive
San Jose, CA 95134-1706
USA
http://www.cisco.com
Tel: 408 526-4000
800 553-NETS (6387)
Fax: 408 527-0883

Trademarks

Trademarks

CONTENTS



CHAPTER 1

New and Changed

This chapter contains the following section:

New and Changed Information

The following table provides an overview of the significant changes to this guide for this release. The table does not provide an exhaustive list of all changes made to the guide or of the new features up to this release.

Table 1: New Features and Changed Information for Cisco APIC 3.2(5)

Feature	Decription	Where Documented
Support in flood for encapsulation for domains with VXLAN encapsulation	You can configure flood in encapsulation for endpoint groups (EPGs) attached to virtual domains with VXLAN encapsulation. Previously, only VLANs were supported for flood in encapsulation for virtual domains. Flood in encapsulation is used to limit flooding traffic inside a bridge domain to a single encapsulation. You configure flood in encapsulation when you create or modify a bridge domain or an EPG.	*Bridging*

Table 2: New Features and Changed Information for Cisco APIC 3.2(1)

Feature	Description	Where Documented
Enhanced Breakout Support on Profiled QSFP Ports on N9K-C93180YC-FX Switches	Support is added for 100 Gigabit (Gb) (4X25Gb) and 40Gb (4X10Gb) dynamic breakouts on profiled QSFP ports on the N9K-C93180YC-FX switch (in ACI mode).	*Dynamic Breakout Ports*

Feature	Description	Where Documented
Enhanced Port Profile Support on N9K-C93180YC-FX Switches	Support is added on the N9K-C93180YC-FX switch for port profiles to change ports from uplink to downlink or downlink to uplink.	*Configuring Port Profiles to Change Ports from Uplink to Downlink or Downlink to Uplink*
Fibre Channel NPV	Support for FC traffic over the Fabric	*Fibre Channel Connectivity*
Cloning Port Configurations	Support for cloning port configurations is added. After you configure a leaf switch port, you can copy the configuration and apply it to other ports.	*Access Interfaces*
Flood in encapsulation enhancements	Information is added about protocols supporting the **Flood in Encapsulation** option for bridge domains or EPGs.	*Bridging*

Table 3: New Features and Changed Information for Cisco APIC 3.1(2)

Feature	Description	Where Documented
Enhanced Breakout Support on Profiled QSFP Ports on N9K-C9336C-FX2 Switches	Support is added for 100Gb (4X25Gb) and 40Gb(4X10Gb) dynamic breakouts on profiled QSFP ports on the N9K-C9336C-FX2 switch (in ACI mode).	*Dynamic Breakout Ports*

Table 4: New Features and Changed Information for Cisco APIC 3.1(1)

Feature or Change	Description	Where Documented
Disabling IP Learning per Bridge Domain	IP learning per bridge domain is disabled when two hosts are connected as active and standby hosts to the Cisco ACI switches. The MAC learning still occurs in the hardware but the IP learning only occurs from the ARP/GARP/ND processes. This enhancement allows for flexible deployments, for example, firewalls or local gateways.	*Bridge Domain Options*

Feature or Change	Description	Where Documented
MACsec	MACsec provides MAC-layer encryption over wired networks by using out-of-band methods for encryption keying. The MACsec Key Agreement (MKA) Protocol provides the required session keys and manages the required encryption keys.	*MACsec*
Flood in Encapsulation	Beginning with Cisco ACI Release 3.1(1) on the Cisco ACI switches with the next generation ASICs and onwards, all protocols are flooded in encapsulation. Multiple EPGs are now supported under one bridge domain with an external switch. When two EPGs share the same BD and the Flood in Encapsulation option is turned on, the EPG flooding traffic does not reach the other EPG. It overcomes the challenges of using the Cisco ACI switches with the Virtual Connect (VC) tunnel network.	*Flood in Encapsulation*
Basic GUI topics removed	Basic GUI procedures are no longer supported	The following topics have been removed: • *Creating a Tenant, VRF, and Bridge Domain Using the Basic GUI* • *Configuring an Enforced Bridge Domain Using the Basic GUI* • *Deploying FCoE vFC Ports Using the Basic GUI* • *Creating a VSAN Domain Using the APIC Basic GUI* • *Configuring 802.1Q Tunnel Interfaces Using the Basic GUI* • *Configuring Dynamic Breakout Ports with the Basic GUI*

Feature or Change	Description	Where Documented
Breakout Support for 100Gb (4X25Gb) Ports on N9K-C93180LC-EX Switches	Support is added for 100Gb (4X25Gb) dynamic breakouts on the N9K-C93180LC-EX switch (in ACI mode).	*Dynamic Breakout Ports*

Table 5: New Features and Changed Information for Cisco APIC 3.0(1)

Feature or Change	Description	Where Documented
Q-in-Q Encapsulation Mapping for EPGs	**With this release, you can map double-tagged VLAN traffic ingressing on a regular interface, PC, or VPC to an EPG.**	*Q-in-Q Encapsulation Mapping for EPGs*
Enforced Bridge Domain	Enforced bridge domain is supported, in which an endpoint in a subject endpoint group (EPG) can only ping subnet gateways within the associated bridge domain. With this configuration enabled, you can create a global exception list of IP addresses which can ping any subnet gateway.	*Enforced Bridge Domain* in *Bridging*
Configuring Leaf Switch Using Port Association	Using these GUI steps, you can now choose the port and then apply a policy to it.	*Configuring Leaf Switch Physical Ports Using Port Association*

Table 6: New Features and Changed Information for Cisco APIC 2.3(1) Release

Feature or Change	Description	Where Documented
802.1Q Tunnel enhancements	Now you can configure ports on core-switches for use in **Dot1q Tunnels** for multiple customers. You can also define access VLANs to distinguish between customers consuming the corePorts. You can also disable MAC learning on **Dot1q Tunnels**.	*802.1Q Tunnels*
Symmetric hashing	Symmetric hashing is now supported on port channels.	*Port Channels*

Feature or Change	Description	Where Documented
Reflective relay (802.1Qbg)	Reflective relay transfers switching for virtual machines out of the host server to an external network switch. It provides connectivity between VMs on the same physical server and the rest of the network. It allows policies that you configure on the Cisco APIC to apply to traffic between the VMs on the same server.	*Access Interfaces*
Traffic Storm control Unicast/Multicast differentiation	Allows you to configure Storm Control on each traffic type separately.	*Traffic Storm Control*

Table 7: New Features and Changed Information in Cisco APIC Release 2.2(2)

Feature or Change	Description	Where Documented
Document Reorganization	The topics in this guide were collected from *Cisco APIC Basic Configuration Guide* and the following APIC articles: • *Cisco ACI and 802.1Q Tunnels* • *Cisco APIC EPG Deployment to Specific Ports Using Domains, Attach Entity Profiles, and VLANs* • *Cisco APIC and Traffic Storm Control* • *Cisco APIC and Dynamic Breakout Ports* • *Cisco APIC and Proxy ARP* • *Deploying an EPG on a Specific Port Using Cisco APIC*	*Cisco APIC Layer 2 Configuration Guide*

Table 8: New Features and Changed Behavior in Cisco APIC 2.2 (1) Release

Feature	Description	Where Documented
FCoE over FEX	You can now configure FCoE over FEX ports.	*FCoE Connections*

Feature	Description	Where Documented
CDP supported in policies on interfaces to FEX devices	In this release, support is added for CDP on interfaces to FEX devices.	*Access Interfaces*
802.1Q Tunnels	You can now configure 802.1Q tunnels on edge ports to enable point-to-multi-point tunneling of Ethernet frames in the fabric, with Quality of Service (QoS) priority settings.	*802.1Q Tunnels*
Dynamic Breakout Ports	To enable a 40 Gigabit Ethernet (GE) leaf switch port to be connected to 4-10GE capable (downlink) devices (connected with Cisco 40-Gigabit to 4X10-Gigabit breakout cables), you configure the 40GE port to breakout (split) to 4-10GE ports.	*Dynamic Breakout Ports*.

Table 9: New Features and Changed Information for Cisco APIC 2.1(1) Release

Revision to topic *Configuring a Traffic Storm Control Policy Using the CLI*	The topic was replaced with a new topic that uses the NX-OS Style CLI.	See *Configuring a Traffic Storm Control Policy Using the NX-OS Style CLI* in *Traffic Storm Control*.

Table 10: New Features and Changed Information for Cisco APIC 2.0(2) Release

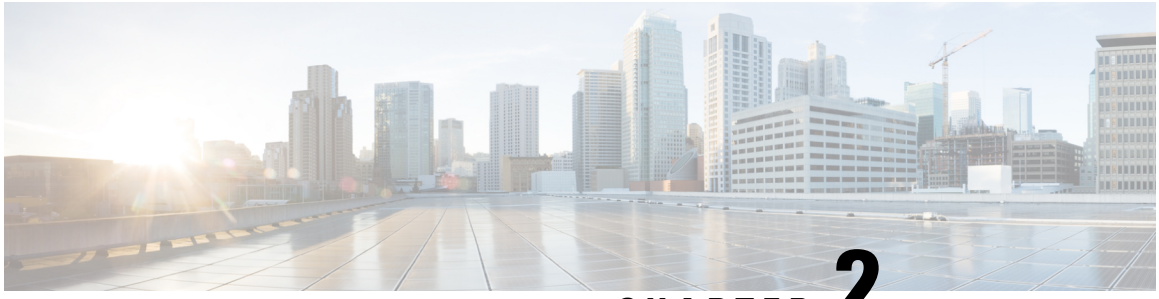
Feature or Change	Description	Where Documented
Revision to topic *Configuring a Traffic Storm Control Policy Using the Advanced GUI*.	Advanced GUI revisions have changed the navigation path to the **Storm Control Interface Policy** work pane.	*Traffic Storm Control*
Proxy ARP	Proxy ARP in Cisco ACI enables endpoints within a network or subnet to communicate with other endpoints without knowing the real MAC address of the endpoints. Proxy ARP is aware of the location of the traffic destination, and offers its own MAC address as the final destination instead.	*Proxy ARP*

Table 11: New Features and Changed Behavior in Cisco APIC for Cisco APIC Release 2.0(1)

Feature or Change	Description	Where Documented
Fibre Channel over Ethernet (FCoE) support	An overview and configuration topics for implementing FCoE connectivity over the ACI fabric.	*FCoE Connections*

Table 12: New Features and Changed Behavior in Cisco APIC for Cisco APIC Release 1.3(1)

Feature or Change	Description	Where Documented
-	Removed object model CLI procedure and replaced with NX-OS-Style CLI procedures.	*Creating AEP, Domains, and VLANs to Deploy an EPG on a Specific Port Using the NX-OS CLI* *Deploying and EPG on a Specific Port with APIC Using the NX-OS Style CLI*



CHAPTER 2

Cisco ACI Forwarding

This chapter contains the following sections:

ACI Fabric Optimizes Modern Data Center Traffic Flows

The Cisco ACI architecture addresses the limitations of traditional data center design, and provides support for the increased east-west traffic demands of modern data centers.

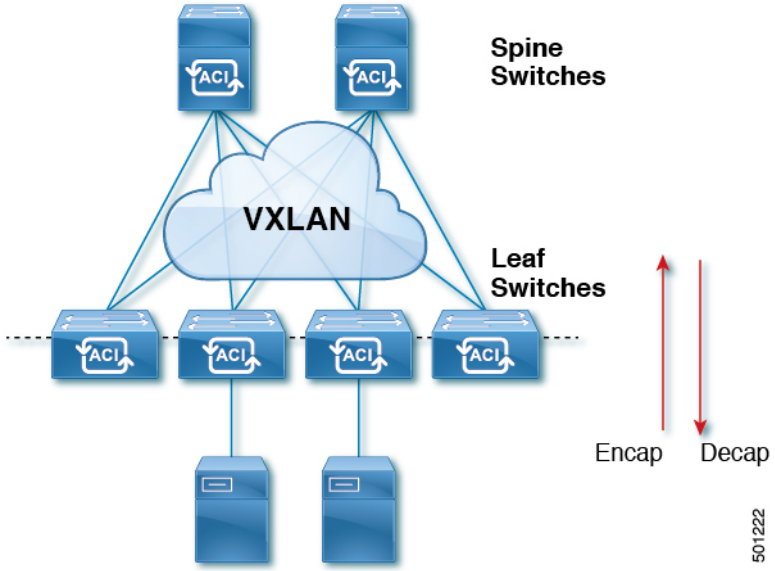
Today, application design drives east-west traffic from server to server through the data center access layer. Applications driving this shift include big data distributed processing designs like Hadoop, live virtual machine or workload migration as with VMware vMotion, server clustering, and multi-tier applications.

North-south traffic drives traditional data center design with core, aggregation, and access layers, or collapsed core and access layers. Client data comes in from the WAN or Internet, a server processes it, and then it exits the data center, which permits data center hardware oversubscription due to WAN or Internet bandwidth constraints. However, Spanning Tree Protocol is required to block loops. This limits available bandwidth due to blocked links, and potentially forces traffic to take a suboptimal path.

In traditional data center designs, IEEE 802.1Q VLANs provide logical segmentation of Layer 2 boundaries or broadcast domains. However, VLAN use of network links is inefficient, requirements for device placements in the data center network can be rigid, and the VLAN maximum of 4094 VLANs can be a limitation. As IT departments and cloud providers build large multi-tenant data centers, VLAN limitations become problematic.

A spine-leaf architecture addresses these limitations. The ACI fabric appears as a single switch to the outside world, capable of bridging and routing. Moving Layer 3 routing to the access layer would limit the Layer 2 reachability that modern applications require. Applications like virtual machine workload mobility and some clustering software require Layer 2 adjacency between source and destination servers. By routing at the access layer, only servers connected to the same access switch with the same VLANs trunked down would be Layer 2-adjacent. In ACI, VXLAN solves this dilemma by decoupling Layer 2 domains from the underlying Layer 3 network infrastructure.

Figure 1: ACI Fabric



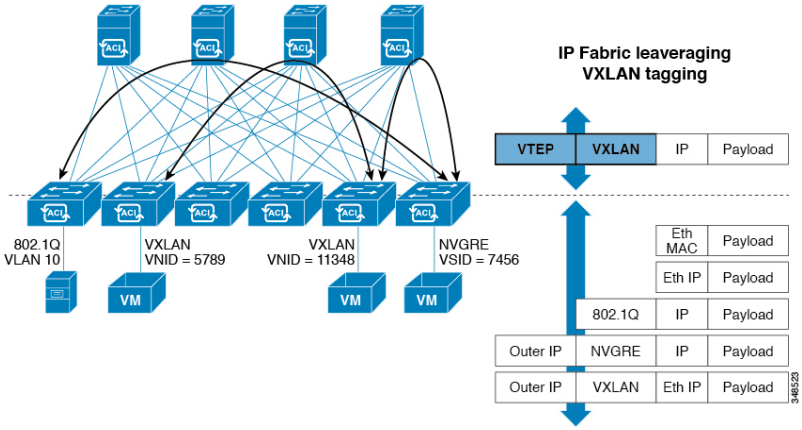
As traffic enters the fabric, ACI encapsulates and applies policy to it, forwards it as needed across the fabric through a spine switch (maximum two-hops), and de-encapsulates it upon exiting the fabric. Within the fabric, ACI uses Intermediate System-to-Intermediate System Protocol (IS-IS) and Council of Oracle Protocol (COOP) for all forwarding of endpoint to endpoint communications. This enables all ACI links to be active, equal cost multipath (ECMP) forwarding in the fabric, and fast-reconverging. For propagating routing information between software defined networks within the fabric and routers external to the fabric, ACI uses the Multiprotocol Border Gateway Protocol (MP-BGP).

VXLAN in ACI

VXLAN is an industry-standard protocol that extends Layer 2 segments over Layer 3 infrastructure to build Layer 2 overlay logical networks. The ACI infrastructure Layer 2 domains reside in the overlay, with isolated broadcast and failure bridge domains. This approach allows the data center network to grow without the risk of creating too large a failure domain.

All traffic in the ACI fabric is normalized as VXLAN packets. At ingress, ACI encapsulates external VLAN, VXLAN, and NVGRE packets in a VXLAN packet. The following figure shows ACI encapsulation normalization.

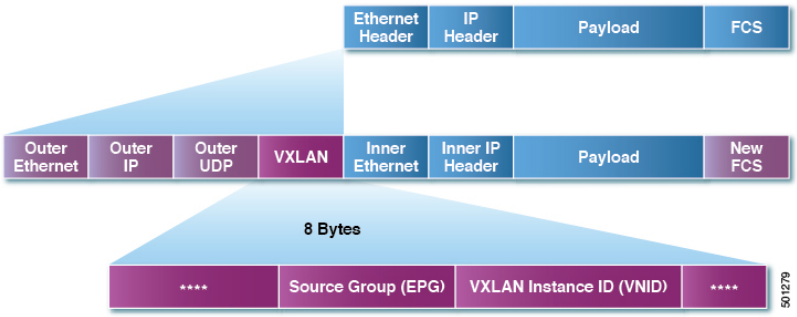
Figure 2: ACI Encapsulation Normalization



Forwarding in the ACI fabric is not limited to or constrained by the encapsulation type or encapsulation overlay network. An ACI bridge domain forwarding policy can be defined to provide standard VLAN behavior where required.

Because every packet in the fabric carries ACI policy attributes, ACI can consistently enforce policy in a fully distributed manner. ACI decouples application policy EPG identity from forwarding. The following illustration shows how the ACI VXLAN header identifies application policy within the fabric.

Figure 3: ACI VXLAN Packet Format



The ACI VXLAN packet contains both Layer 2 MAC address and Layer 3 IP address source and destination fields, which enables efficient and scalable forwarding within the fabric. The ACI VXLAN packet header source group field identifies the application policy endpoint group (EPG) to which the packet belongs. The VXLAN Instance ID (VNID) enables forwarding of the packet through tenant virtual routing and forwarding (VRF) domains within the fabric. The 24-bit VNID field in the VXLAN header provides an expanded address space for up to 16 million unique Layer 2 segments in the same network. This expanded address space gives IT departments and cloud providers greater flexibility as they build large multitenant data centers.

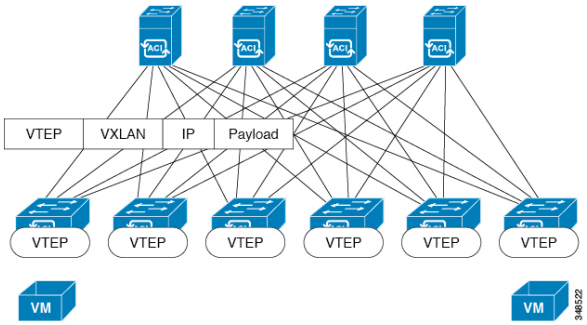
VXLAN enables ACI to deploy Layer 2 virtual networks at scale across the fabric underlay Layer 3 infrastructure. Application endpoint hosts can be flexibly placed in the data center network without concern for the Layer 3 boundary of the underlay infrastructure, while maintaining Layer 2 adjacency in a VXLAN overlay network.

Layer 3 VNIDs Facilitate Transporting Inter-subnet Tenant Traffic

The ACI fabric provides tenant default gateway functionality that routes between the ACI fabric VXLAN networks. For each tenant, the fabric provides a virtual default gateway that spans all of the leaf switches assigned to the tenant. It does this at the ingress interface of the first leaf switch connected to the endpoint. Each ingress interface supports the default gateway interface. All of the ingress interfaces across the fabric share the same router IP address and MAC address for a given tenant subnet.

The ACI fabric decouples the tenant endpoint address, its identifier, from the location of the endpoint that is defined by its locator or VXLAN tunnel endpoint (VTEP) address. Forwarding within the fabric is between VTEPs. The following figure shows decoupled identity and location in ACI.

Figure 4: ACI Decouples Identity and Location



VXLAN uses VTEP devices to map tenant end devices to VXLAN segments and to perform VXLAN encapsulation and de-encapsulation. Each VTEP function has two interfaces:

- A switch interface on the local LAN segment to support local endpoint communication through bridging
- An IP interface to the transport IP network

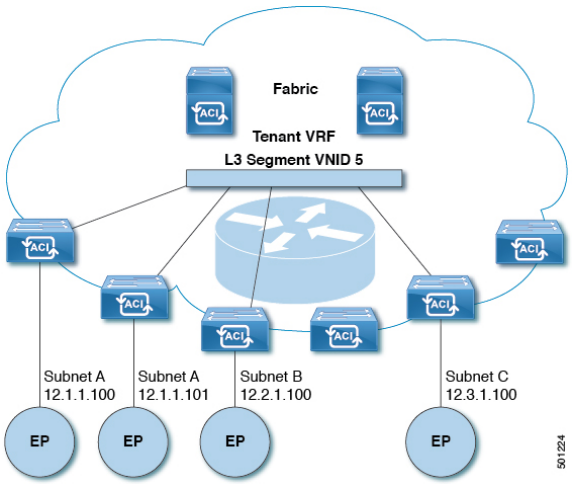
The IP interface has a unique IP address that identifies the VTEP device on the transport IP network known as the infrastructure VLAN. The VTEP device uses this IP address to encapsulate Ethernet frames and transmit the encapsulated packets to the transport network through the IP interface. A VTEP device also discovers the remote VTEPs for its VXLAN segments and learns remote MAC Address-to-VTEP mappings through its IP interface.

The VTEP in ACI maps the internal tenant MAC or IP address to a location using a distributed mapping database. After the VTEP completes a lookup, the VTEP sends the original data packet encapsulated in VXLAN with the destination address of the VTEP on the destination leaf switch. The destination leaf switch de-encapsulates the packet and sends it to the receiving host. With this model, ACI uses a full mesh, single hop, loop-free topology without the need to use the spanning-tree protocol to prevent loops.

The VXLAN segments are independent of the underlying network topology; conversely, the underlying IP network between VTEPs is independent of the VXLAN overlay. It routes the encapsulated packets based on the outer IP address header, which has the initiating VTEP as the source IP address and the terminating VTEP as the destination IP address.

The following figure shows how routing within the tenant is done.

Figure 5: Layer 3 VNIDs Transport ACI Inter-subnet Tenant Traffic



For each tenant VRF in the fabric, ACI assigns a single L3 VNID. ACI transports traffic across the fabric according to the L3 VNID. At the egress leaf switch, ACI routes the packet from the L3 VNID to the VNID of the egress subnet.

Traffic arriving at the fabric ingress that is sent to the ACI fabric default gateway is routed into the Layer 3 VNID. This provides very efficient forwarding in the fabric for traffic routed within the tenant. For example, with this model, traffic between 2 VMs belonging to the same tenant, on the same physical host, but on different subnets, only needs to travel to the ingress switch interface before being routed (using the minimal path cost) to the correct destination.

To distribute external routes within the fabric, ACI route reflectors use multiprotocol BGP (MP-BGP). The fabric administrator provides the autonomous system (AS) number and specifies the spine switches that become route reflectors.

Transmission of STP BPDU

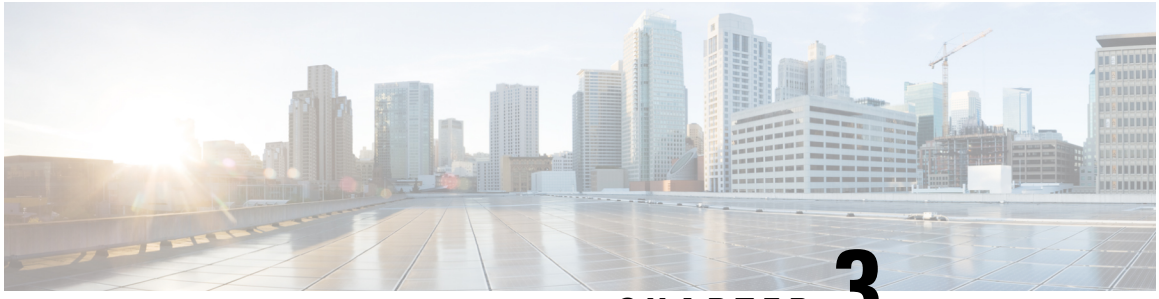
When there are two or more switches running STP connected to Cisco ACI in an EPG and if the static port assignments are as follows:

- All statically assigned ports under EPG are access untagged—STP BPDUs are received and sent without a tag.
- For a mix of statically assigned trunk ports and statically assigned access untagged ports—STP BPDUs received on a trunk port will be sent to the access untagged port with the `dot1q` tag. Hence the access port will go into a inconsistent state.
- For a mix of statically assigned trunk ports and statically assigned access ports under an EPG—Cisco ACI sends STP BPDU with `dot1q` tag and the access ports use `802.1p` access.

  In this case you must use `802.1p` access on the Layer 2 switch to receive and process the tagged STP packets.

  In case `802.1p` is not allowed on the Layer 2 switch, then use the trunk port access.
- Cisco ACI acts as a full-duplex hub and will flood spanning tree BPDUs within the VxLAN VNID associated to the encapsulation VLAN on which the BPDUs were received. Because Cisco ACI is a

full-duplex medium, external switches running versions of rapid spanning tree protocol (RSTP) or rapid per VLAN spanning tree (RPVST) will default to point-to-point link types. As a result, if there are more than 2 external switches that run STP and connect to the same encapsulation VLAN and EPG VNID, you should set the link type to "shared" on the external switch interfaces to avoid problems with convergence and instability. These problems can occur because the switches will receive BPDUs from all bridges (or STP-enabled switches) that are connected to this encapsulation.



CHAPTER 3

Prerequisites for Configuring Layer 2 Networks

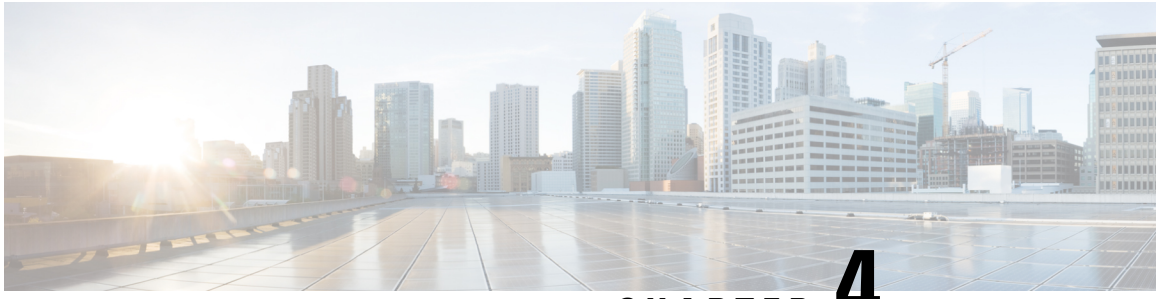
Layer 2 Prerequisites

Before you begin to perform the tasks in this guide, complete the following:

- Install the ACI fabric and ensure that the APIC controllers are online, and the APIC cluster is formed and healthy—For more information, see *Cisco APIC Getting Started Guide, Release 2.x*.
- Create fabric administrator accounts for the administrators that will configure Layer 2 networks—For instructions, see the *User Access, Authentication, and Accounting* and *Management* chapters in *Cisco APIC Basic Configuration Guide*.
- Install and register the target leaf switches in the ACI fabric—For more information, see *Cisco APIC Getting Started Guide, Release 2.x*.

  For information about installing and registering virtual switches, see *Cisco ACI Virtualization Guide*.
- Configure the tenants, VRFs, and EPGs (with application profiles and contracts) that will consume the Layer 2 networks—For instructions, see the *Basic User Tenant Configuration* chapter in *Cisco APIC Basic Configuration Guide*.

Caution If you install 1 Gigabit Ethernet (GE) or 10GE links between the leaf and spine switches in the fabric, there is risk of packets being dropped instead of forwarded, because of inadequate bandwidth. To avoid the risk, use 40GE or 100GE links between the leaf and spine switches.



CHAPTER 4

Networking Domains

This chapter contains the following sections:

Networking Domains

A fabric administrator creates domain policies that configure ports, protocols, VLAN pools, and encapsulation. These policies can be used exclusively by a single tenant, or shared. Once a fabric administrator configures domains in the ACI fabric, tenant administrators can associate tenant endpoint groups (EPGs) to domains.

The following networking domain profiles can be configured:

- VMM domain profiles (`vmmDomP`) are required for virtual machine hypervisor integration.
- Physical domain profiles (`physDomP`) are typically used for bare metal server attachment and management access.
- Bridged outside network domain profiles (`l2extDomP`) are typically used to connect a bridged external network trunk switch to a leaf switch in the ACI fabric.
- Routed outside network domain profiles (`l3extDomP`) are used to connect a router to a leaf switch in the ACI fabric.
- Fibre Channel domain profiles (`fcDomP`) are used to connect Fibre Channel VLANs and VSANs.

A domain is configured to be associated with a VLAN pool. EPGs are then configured to use the VLANs associated with a domain.

Note EPG port and VLAN configurations must match those specified in the domain infrastructure configuration with which the EPG associates. If not, the APIC will raise a fault. When such a fault occurs, verify that the domain infrastructure configuration matches the EPG port and VLAN configurations.

Related Documents

For more information about Layer 3 Networking, see *Cisco APIC Layer 3 Networking Configuration Guide*.

For information about configuring VMM Domains, see *Cisco ACI Virtual Machine Networking* in *Cisco ACI Virtualization Guide.*

Bridge Domains

About Bridge Domains

A bridge domain (BD) represents a Layer 2 forwarding construct within the fabric. One or more endpoint groups (EPGs) can be associated with one bridge domain or subnet. A bridge domain can have one or more subnets that are associated with it. One or more bridge domains together form a tenant network. When you insert a service function between two EPGs, those EPGs must be in separate BDs. To use a service function between two EPGs, those EPGs must be isolated; this follows legacy service insertion based on Layer 2 and Layer 3 lookups.

VMM Domains

Virtual Machine Manager Domain Main Components

ACI fabric virtual machine manager (VMM) domains enable an administrator to configure connectivity policies for virtual machine controllers. The essential components of an ACI VMM domain policy include the following:

- **Virtual Machine Manager Domain Profile**—Groups VM controllers with similar networking policy requirements. For example, VM controllers can share VLAN pools and application endpoint groups (EPGs). The APIC communicates with the controller to publish network configurations such as port groups that are then applied to the virtual workloads. The VMM domain profile includes the following essential components:
  - **Credential**—Associates a valid VM controller user credential with an APIC VMM domain.
  - **Controller**—Specifes how to connect to a VM controller that is part of a policy enforcement domain. For example, the controller specifies the connection to a VMware vCenter that is part a VMM domain.

Note A single VMM domain can contain multiple instances of VM controllers, but they must be from the same vendor (for example, from VMware or from Microsoft.

- **EPG Association**—Endpoint groups regulate connectivity and visibility among the endpoints within the scope of the VMM domain policy. VMM domain EPGs behave as follows:
  - The APIC pushes these EPGs as port groups into the VM controller.

- An EPG can span multiple VMM domains, and a VMM domain can contain multiple EPGs.

- **Attachable Entity Profile Association**—Associates a VMM domain with the physical network infrastructure. An attachable entity profile (AEP) is a network interface template that enables deploying VM controller policies on a large set of leaf switch ports. An AEP specifies which switches and ports are available, and how they are configured.
- **VLAN Pool Association**—A VLAN pool specifies the VLAN IDs or ranges used for VLAN encapsulation that the VMM domain consumes.

Virtual Machine Manager Domains

An APIC VMM domain profile is a policy that defines a VMM domain. The VMM domain policy is created in APIC and pushed into the leaf switches.

VMM domains provide the following:

- A common layer in the ACI fabric that enables scalable fault-tolerant support for multiple VM controller platforms.
- VMM support for multiple tenants within the ACI fabric.

VMM domains contain VM controllers such as VMware vCenter or Microsoft SCVMM Manager and the credential(s) required for the ACI API to interact with the VM controller. A VMM domain enables VM mobility within the domain but not across domains. A single VMM domain can contain multiple instances of VM controllers but they must be the same kind. For example, a VMM domain can contain many VMware vCenters managing multiple controllers each running multiple VMs but it may not also contain SCVMM Managers. A VMM domain inventories controller elements (such as pNICs, vNICs, VM names, and so forth) and pushes policies into the controller(s), creating port groups, and other necessary elements. The ACI VMM domain listens for controller events such as VM mobility and responds accordingly.

Configuring Physical Domains

Configuring a Physical Domain

Physical domains control the scope of where a given VLAN namespace is used. The VLAN namespace that is associated with the physical domain is for non-virtualized servers, although it can also be used for static mapping of port-groups from virtualized servers. You can configure a physical domain for physical device types.

Before you begin

- Configure a tenant.

Procedure

Step 1 On the menu bar, click **Fabric**.

Step 2 On the submenu bar, click **External Access Policies**.

Step 3 In the **Navigation** pane, expand **Physical and External Domains** and click **Physical Domains**.

Step 4 From the **Actions** drop-down list, choose **Create Physical Domain**. The **Create Physical Domain** dialog box appears.

Step 5 Complete the following fields:

Name	Description
Name	The name of the physical domain profile.
Associate Attachable Entity Profiles	Choose the attachable entity profiles to be associated to this domain.
VLAN Pool	The VLAN pool used by the physical domain. The VLAN pool specifies the range or pool for VLANs that is allocated by the APIC for the service graph templates that are using this physical domain. Click **Dynamic** or **Static** allocation.

Step 6 (Optional) Add a AAA security domain and click the **Select** check box.

Step 7 Click **Submit**.

Configuring a Physical Domain Using the REST API

A physical domain acts as the link between the VLAN pool and the Access Entity Profile (AEP). The domain also ties the fabric configuration to the tenant configuration, as the tenant administrator is the one who associates domains to EPGs, while the domains are created under the fabric tab. When configuring in this order, only the profile name and the VLAN pool are configured.

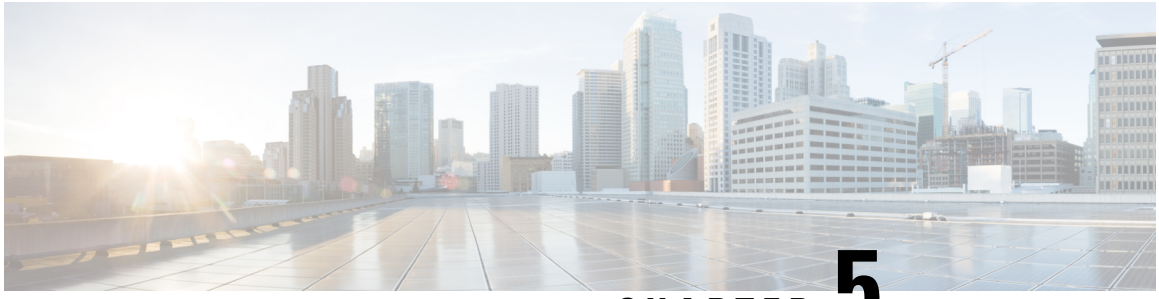
Procedure

Configure a physical domain by sending a post with XML such as the following example:

Example:

```
<physDomP dn="uni/phys-bsprint-PHY" lcOwn="local" modTs="2015-02-23T16:13:21.906-08:00"
   monPolDn="uni/fabric/monfab-default" name="bsprint-PHY" ownerKey="" ownerTag="" status=""
 uid="8131">
      <infraRsVlanNs childAction="" forceResolve="no" lcOwn="local"
modTs="2015-02-23T16:13:22.065-08:00"
         monPolDn="uni/fabric/monfab-default" rType="mo" rn="rsvlanNs" state="formed"
stateQual="none"
         status="" tCl="fvnsVlanInstP" tDn="uni/infra/vlanns-[bsprint-vlan-pool]-static"
tType="mo" uid="8131"/>
     <infraRsVlanNsDef forceResolve="no" lcOwn="local" modTs="2015-02-23T16:13:22.065-08:00"
 rType="mo"
         rn="rsvlanNsDef" state="formed" stateQual="none" status="" tCl="fvnsAInstP"
         tDn="uni/infra/vlanns-[bsprint-vlan-pool]-static" tType="mo"/>
      <infraRtDomP lcOwn="local" modTs="2015-02-23T16:13:52.945-08:00"
rn="rtdomP-[uni/infra/attentp-bsprint-AEP]"
```

```
        status="" tCl="infraAttEntityP" tDn="uni/infra/attentp-bsprint-AEP"/>
</physDomP>
```



CHAPTER 5

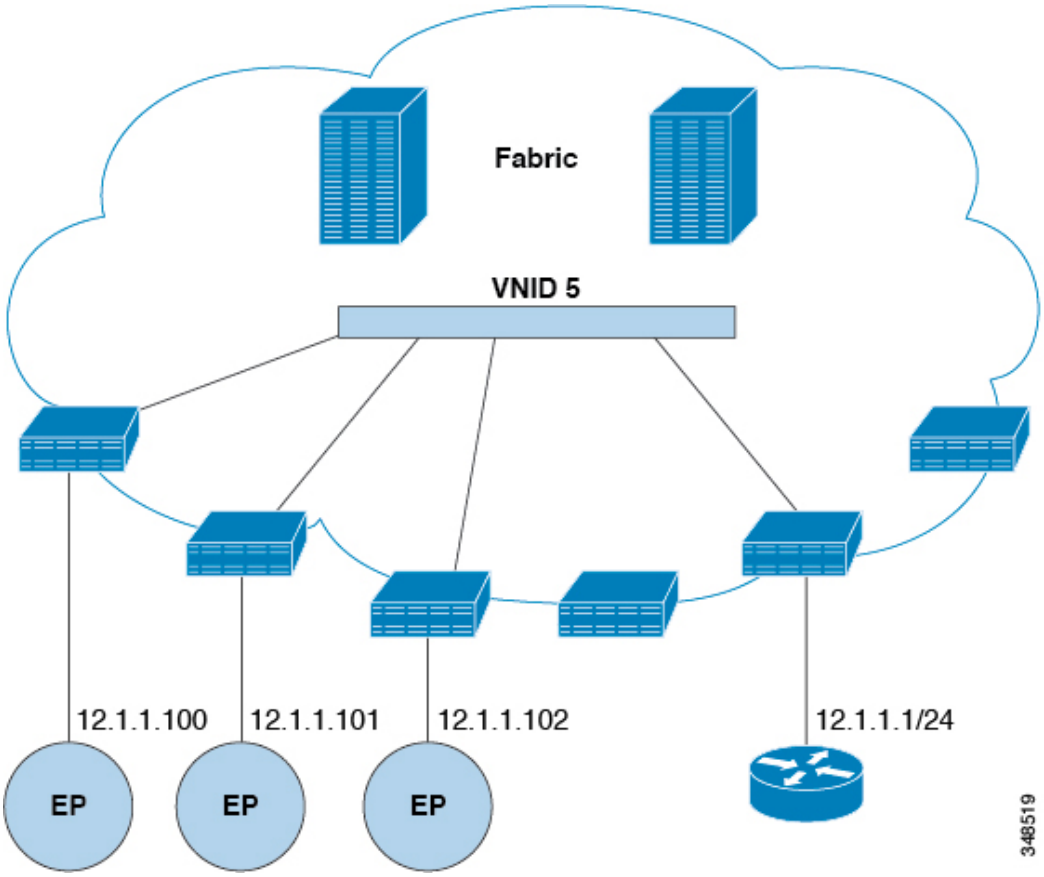
Bridging

This chapter contains the following sections:

Bridged Interface to an External Router

As shown in the figure below, when the leaf switch interface is configured as a bridged interface, the default gateway for the tenant VNID is the external router.

Figure 6: Bridged External Router

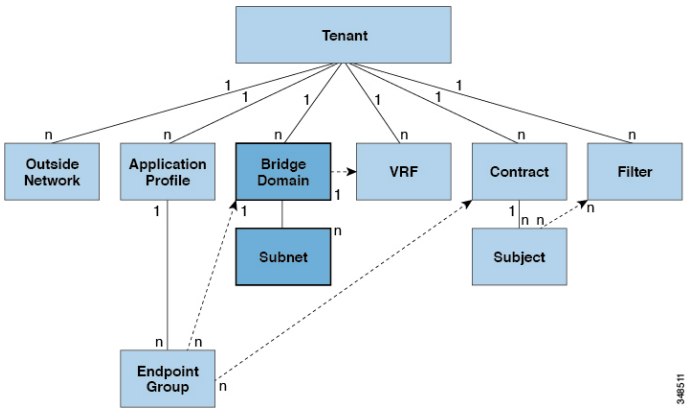


The ACI fabric is unaware of the presence of the external router and the APIC statically assigns the leaf switch interface to its EPG.

Bridge Domains and Subnets

A bridge domain (`fvBD`) represents a Layer 2 forwarding construct within the fabric. The following figure shows the location of bridge domains in the management information tree (MIT) and their relation to other objects in the tenant.

Figure 7: Bridge Domains



A bridge domain must be linked to a VRF instance (also known as a context or private network). With the exception of a Layer 2 VLAN, it must have at least one subnet (`fvSubnet`) associated with it. The bridge domain defines the unique Layer 2 MAC address space and a Layer 2 flood domain if such flooding is enabled. While a VRF instance defines a unique IP address space, that address space can consist of multiple subnets. Those subnets are defined in one or more bridge domains that reference the corresponding VRF instance.

The options for a subnet under a bridge domain or under an EPG are as follows:

- *Public*: The subnet can be exported to a routed connection.
- *Private*: The subnet applies only within its tenant.
- *Shared*: The subnet can be shared with and exported to multiple VRF instances in the same tenant or across tenants as part of a shared service. An example of a shared service is a routed connection to an EPG present in another VRF instance in a different tenant. This enables traffic to pass in both directions across VRF instances. An EPG that provides a shared service must have its subnet configured under that EPG (not under a bridge domain), and its scope must be set to advertised externally, and shared between VRF instances.

Note Shared subnets must be unique across the VRF instance involved in the communication. When a subnet under an EPG provides a Layer 3 external network shared service, such a subnet must be globally unique within the entire Cisco Application Centric Infrastructure (ACI) fabric.

Bridge domain packet behavior can be controlled in the following ways:

Packet Type	Mode
ARP	You can enable or disable **ARP Flooding**; without flooding, ARP packets are sent with unicast. **Note** If the `limitIpLearnToSubnets` in `fvBD` is set, endpoint learning is limited to the bridge domain only if the IP address is in a configured subnet of the bridge domain or an EPG subnet that is a shared service provider.
Unknown Unicast	**L2 Unknown Unicast**, which can be **Flood** or **Hardware Proxy**. **Note** When the bridge domain has **L2 Unknown Unicast** set to **Flood**, if an endpoint is deleted the system deletes it from both the local leaf switches as well as the remote leaf switches where the bridge domain is deployed, by selecting **Clear Remote MAC Entries**. Without this feature, the remote leaf continues to have this endpoint learned until the timer expires. Modifying the **L2 Unknown Unicast** setting causes traffic to bounce (go down and up) on interfaces to devices attached to EPGs associated with this bridge domain.
Unknown IP Multicast	**L3 Unknown Multicast Flooding** **Flood**: Packets are flooded on ingress and border leaf switch nodes only. With N9K-93180YC-EX, packets are flooded on all the nodes where a bridge domain is deployed. **Optimized**: Only 50 bridge domains per leaf are supported. This limitation is not applicable for N9K-93180YC-EX.
L2 Multicast, Broadcast, Unicast	**Multi-Destination Flooding**, which can be one of the following: • **Flood in BD**: Flood in bridge domain • **Flood in Encapsulation**: Flood in encapsulation • **Drop**: Drop the packets

Note Beginning with Cisco APIC release 3.1(1), on the Cisco Nexus 9000 series switches (with names ending with EX and FX and onwards), the following protocols can be flooded in encapsulation or flooded in a bridge domain: OSPF/OSPFv3, BGP, EIGRP, LACP, ISIS, IGMP, PIM, ST-BPDU, ARP/GARP, RARP, and ND.

Bridge domains can span multiple switches. A bridge domain can contain multiple subnets, but a subnet is contained within a single bridge domain. If the bridge domain (fvBD) `limitIPLearnToSubnets` property is set to `yes`, endpoint learning will occur in the bridge domain only if the IP address is within any of the configured subnets for the bridge domain or within an EPG subnet when the EPG is a shared service provider. Subnets can span multiple EPGs; one or more EPGs can be associated with one bridge domain or subnet. In hardware proxy mode, ARP traffic is forwarded to an endpoint in a different bridge domain when that endpoint has been learned as part of the Layer 3 lookup operation.

Bridge Domain Options

A bridge domain can be set to operate in flood mode for unknown unicast frames or in an optimized mode that eliminates flooding for these frames. When operating in flood mode, Layer 2 unknown unicast traffic is flooded over the multicast tree of the bridge domain (GIPo). For the bridge domain to operate in optimized mode you should set it to hardware-proxy. In this case, Layer 2 unknown unicast frames are sent to the spine-proxy anycast VTEP address.

Caution Changing from unknown unicast flooding mode to hw-proxy mode is disruptive to the traffic in the bridge domain.

If IP routing is enabled in the bridge domain, the mapping database learns the IP address of the endpoints in addition to the MAC address.

The **Layer 3 Configurations** tab of the bridge domain panel allows the administrator to configure the following parameters:

- **Unicast Routing**: If this setting is enabled and a subnet address is configured, the fabric provides the default gateway function and routes the traffic. Enabling unicast routing also instructs the mapping database to learn the endpoint IP-to-VTEP mapping for this bridge domain. The IP learning is not dependent upon having a subnet configured under the bridge domain.
- **Subnet Address**: This option configures the SVI IP addresses (default gateway) for the bridge domain.
- **Limit IP Learning to Subnet**: This option is similar to a unicast reverse-forwarding-path check. If this option is selected, the fabric will not learn IP addresses from a subnet other than the one configured on the bridge domain.

Caution Enabling **Limit IP Learning to Subnet** is disruptive to the traffic in the bridge domain.

Disabling IP Learning per Bridge Domain

You can disable IP dataplane learning for a bridge domain. The MAC learning still occurs in the hardware, but the IP learning only occurs from the ARP/GARP/ND processes. This functionality was introduced in the

Cisco APIC 3.1 releases primarily for service graph policy-based redirect (PBR) deployments. We do not recommend using this option and it is not supported except when used with PBR.

See the following guidelines and limitations for disabling IP learning per bridge domain:

- Layer 3 multicast is not supported because the source IP address is not learned to populate the S,G information in the remote top-of-rack (ToR) switches.
- As the DL bit is set in the iVXLAN header, the MAC address is also not learned from the data path in the remote TORs. It results in flooding of the unknown unicast traffic from the remote TOR to all TORs in the fabric where this BD is deployed. It is recommended to configure the BD in proxy mode to overcome this situation if endpoint dataplane learning is disabled.
- ARP should be in flood mode and GARP based detection should be enabled.
- When IP learning is disabled, Layer 3 endpoints are not flushed in the corresponding VRF. It may lead to the endpoints pointing to the same TOR forever. To resolve this issue, flush all the remote IP endpoints in this VRF on all TORs.
- On Cisco ACI switches with Application Leaf Engine (ALE), the inner MAC address is not learned from the VXLAN packets.
- When dataplane learning is disabled on a BD, the existing local endpoints learned via dataplane in that BD are not flushed. If the data traffic is flowing, the existing local endpoints do not age out.

When IP learning is disabled, you have to enable the **Global Subnet Prefix** check option in **System** > **System Settings** > **Fabric Wide Setting** > **Enforce Subnet Check** in the Online Help.

Creating a Tenant, VRF, and Bridge Domain Using the GUI

If you have a public subnet when you configure the routed outside, you must associate the bridge domain with the outside configuration.

Procedure

Step 1 On the menu bar, choose **Tenants** > **Add Tenant**.

Step 2 In the **Create Tenant** dialog box, perform the following tasks:

a) In the **Name** field, enter a name.
b) Click the **Security Domains +** icon to open the **Create Security Domain** dialog box.
c) In the **Name** field, enter a name for the security domain. Click **Submit**.
d) In the **Create Tenant** dialog box, check the check box for the security domain that you created, and click **Submit**.

Step 3 In the **Navigation** pane, expand **Tenant-name** > **Networking**, and in the **Work** pane, drag the **VRF** icon to the canvas to open the **Create VRF** dialog box, and perform the following tasks:

a) In the **Name** field, enter a name.
b) Click **Submit** to complete the VRF configuration.

Step 4 In the **Networking** pane, drag the **BD** icon to the canvas while connecting it to the **VRF** icon. In the **Create Bridge Domain** dialog box that displays, perform the following tasks:

a) In the **Name** field, enter a name.

b) Click the **L3 Configurations** tab.
c) Expand **Subnets** to open the **Create Subnet** dialog box, enter the subnet mask in the **Gateway IP** field and click **OK**.
d) Click **Submit** to complete bridge domain configuration.

Step 5 In the **Networks** pane, drag the **L3** icon down to the canvas while connecting it to the **VRF** icon. In the **Create Routed Outside** dialog box that displays, perform the following tasks:

a) In the **Name** field, enter a name.
b) Expand **Nodes And Interfaces Protocol Profiles** to open the **Create Node Profile** dialog box.
c) In the **Name** field, enter a name.
d) Expand **Nodes** to open the **Select Node** dialog box.
e) In the **Node ID** field, choose a node from the drop-down list.
f) In the **Router ID** field, enter the router ID.
g) Expand **Static Routes** to open the **Create Static Route** dialog box.
h) In the **Prefix** field, enter the IPv4 or IPv6 address.
i) Expand **Next Hop Addresses** and in the **Next Hop IP** field, enter the IPv4 or IPv6 address.
j) In the **Preference** field, enter a number, then click **UPDATE** and then **OK**.
k) In the **Select Node** dialog box, click **OK**.
l) In the **Create Node Profile** dialog box, click **OK**.
m) Check the **BGP**, **OSPF**, or **EIGRP** check boxes if desired, and click **NEXT**. Click **OK** to complete the Layer 3 configuration.

To confirm L3 configuration, in the **Navigation** pane, expand **Networking** > **VRFs**.

Creating a Tenant, VRF, and Bridge Domain Using the NX-OS Style CLI

This section provides information on how to create tenants, VRFs, and bridge domains.

Note Before creating the tenant configuration, you must create a VLAN domain using the **vlan-domain** command and assign the ports to it.

Procedure

Step 1 Create a VLAN domain (which contains a set of VLANs that are allowable in a set of ports) and allocate VLAN inputs, as follows:

Example:

In the following example ("exampleCorp"), note that VLANs 50 - 500 are allocated.

```
apic1# configure
apic1(config)# vlan-domain dom_exampleCorp
apic1(config-vlan)# vlan 50-500
apic1(config-vlan)# exit
```

Step 2 Once the VLANs have been allocated, specify the leaf (switch) and interface for which these VLANs can be used. Then, enter "vlan-domain member" and then the name of the domain you just created.

Example:

In the following example, these VLANs (50 - 500) have been enabled on leaf 101 on interface ethernet 1/2-4 (three ports including 1/2, 1/3, and 1/4). This means that if you are using this interface, you can use VLANS 50-500 on this port for any application that the VLAN can be used for.

```
apic1(config-vlan)# leaf 101
apic1(config-vlan)# interface ethernet 1/2-4
apic1(config-leaf-if)# vlan-domain member dom_exampleCorp
apic1(config-leaf-if)# exit
apic1(config-leaf)# exit
```

Step 3 Create a tenant in global configuration mode, as shown in the following example:

Example:

```
apic1(config)# tenant exampleCorp
```

Step 4 Create a private network (also called VRF) in tenant configuration mode as shown in the following example:

Example:

```
apic1(config)# tenant exampleCorp
apic1(config-tenant)# vrf context exampleCorp_v1
apic1(config-tenant-vrf)# exit
```

Step 5 Create a bridge domain (BD) under the tenant, as shown in the following example:

Example:

```
apic1(config-tenant)# bridge-domain exampleCorp_b1
apic1(config-tenant-bd)# vrf member exampleCorp_v1
apic1(config-tenant-bd)# exit
```

Note In this case, the VRF is "exampleCorp_v1".

Step 6 Allocate IP addresses for the BD (ip and ipv6), as shown in the following example.

Example:

```
apic1(config-tenant)# interface bridge-domain exampleCorp_b1
apic1(config-tenant-interface)# ip address 172.1.1.1/24
apic1(config-tenant-interface)# ipv6 address 2001:1:1::1/64
apic1(config-tenant-interface)# exit
```

What to do next

The next section describes how to add an application profile, create an application endpoint group (EPG), and associate the EPG to the bridge domain.

Related Topics

Configuring a VLAN Domain Using the NX-OS Style CLI

Creating a Tenant, VRF, and Bridge Domain Using the REST API

Procedure

Step 1 Create a tenant.

Example:

```
 POST https://apic-ip-address/api/mo/uni.xml
<fvTenant name="ExampleCorp"/>
```

When the POST succeeds, you see the object that you created in the output.

Step 2 Create a VRF and bridge domain.

Note The Gateway Address can be an IPv4 or an IPv6 address. For more about details IPv6 gateway address, see the related KB article, *KB: Creating a Tenant, VRF, and Bridge Domain with IPv6 Neighbor Discovery* .

Example:

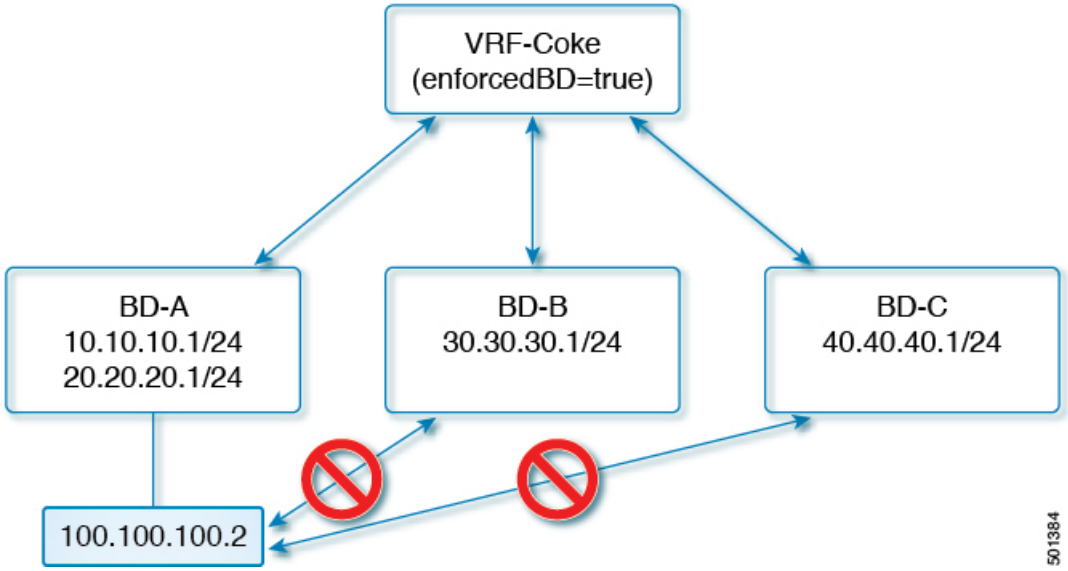
```
 URL for POST: https://apic-ip-address/api/mo/uni/tn-ExampleCorp.xml

<fvTenant name="ExampleCorp">
   <fvCtx name="pvn1"/>
   <fvBD name="bd1">
      <fvRsCtx tnFvCtxName="pvn1"/>
      <fvSubnet ip="10.10.100.1/24"/>
   </fvBD>
</fvTenant>
```

Note If you have a public subnet when you configure the routed outside, you must associate the bridge domain with the outside configuration.

Configuring an Enforced Bridge Domain

An enforced bridge domain configuration entails creating an endpoint in a subject endpoint group (EPG) that can only ping subnet gateways within the associated bridge domain. With this configuration, you can then create a global exception list of IP addresses that can ping any subnet gateway.

Figure 8: Enforced Bridge Domain



Note

- The exception IP addresses can ping all of the bridge domain gateways across all of your VRF instances.
- A loopback interface configured for an L3Out does not enforce reachability to the IP address that is configured for the subject loopback interface.
- When an eBGP peer IP address exists in a different subnet than the subnet of the L3Out interface, you must add the peer subnet to the allowed exception subnets. Otherwise, eBGP traffic is blocked because the source IP address exists in a different subnet than the L3Out interface subnet.
- For a BGP prefixed-based peer, you must add the peer subnet to the list of allowed exception subnets. For example, if 20.1.1.0/24 is configured as BGP prefixed-based peer, you must add 20.1.1.0/24 to the list of allowed exception subnets.
- An enforced bridge domain is not supported with the Management tenant, regardless if the VRF instances are in-band or out-of-band, and any rules to control the traffic to these VRF instances should be configured using regular contracts.

Configuring an Enforced Bridge Domain Using the NX-OS Style CLI

This section provides information on how to configure your enforced bridge domain using the NX-OS style command line interface (CLI).

Procedure

Step 1 Create and enable the tenant:

Example:

In the following example ("cokeVrf") is created and enabled.

```
apic1(config-tenant)# vrf context cokeVrf
apic1(config-tenant-vrf)# bd-enforce enable
apic1(config-tenant-vrf)# exit
apic1(config-tenant)#exit
```

Step 2 Add the subnet to the exception list.

Example:

```
apic1(config)#bd-enf-exp-ip add1.2.3.4/24
apic1(config)#exit
```

You can confirm if the enforced bridge domain is operational using the following type of command:

```
apic1# show running-config all | grep bd-enf
bd-enforce enable
bd-enf-exp-ip add 1.2.3.4/24
```

Example

The following command removes the subnet from the exception list:

```
apic1(config)# no bd-enf-exp-ip 1.2.3.4/24
apic1(config)#tenant coke
apic1(config-tenant)#vrf context cokeVrf
```

What to do next

To disable the enforced bridge domain run the following command:

```
apic1(config-tenant-vrf)# no bd-enforce enable
```

Configuring an Enforced Bridge Domain Using the REST API

Procedure

	Command or Action	Purpose
Step 1	Create a tenant. **Example:** `POST https://apic-ip-address/api/mo/uni.xml <fvTenant name="ExampleCorp"/>`	When the POST succeeds, you see the object that you created in the output.

	Command or Action	Purpose

Step 2	Create a VRF and bridge domain.
	Example:
	URL for POST: https://*apic-ip-address*/api/mo/uni/tn-ExampleCorp.xml
	`<fvTenant name="ExampleCorp">`
	`  <fvCtx name="pvn1"/>`
	`  <fvBD name="bd1">`
	`  <fvRsCtx tnFvCtxName="pvn1" bdEnforceEnable="yes"/>`
	`  <fvSubnet ip="10.10.100.1/24"/>`
	`  </fvBD>`
	`</fvTenant>`
	For adding an exception IP, use the following post:
	https://apic-ip-address/api/node/mo/uni/infra.xml
	`<bdEnforceExceptionCont>`
	`<bdEnforceExceptIp ip="11.0.1.0/24"/>`
	`</bdEnforceExceptionCont>`
	Note If you have a public subnet when you configure the routed outside, you must associate the bridge domain with the outside configuration.
		Note The Gateway Address can be an IPv4 or an IPv6 address. For more about details IPv6 gateway address, see the related KB article, *KB: Creating a Tenant, VRF, and Bridge Domain with IPv6 Neighbor Discovery* .

Configuring Flood in Encapsulation for All Protocols and Proxy ARP Across Encapsulations

Cisco Application Centric Infrastructure (ACI) uses the bridge domain as the Layer 2 broadcast boundary. Each bridge domain can include multiple endpoint groups (EPGs), and each EPG can be mapped to multiple virtual or physical domains. Each EPG can also use different VLAN encapsulation pools in each domain. Each EPG can also use different VLAN or VXLAN encapsulation pools in each domain.

Ordinarily, when you put multiple EPGs within bridge domains, broadcast flooding sends traffic to all the EPGs in the bridge domain. Because EPGs are used to group endpoints and manage traffic to fulfill specific functions, sending the same traffic to all the EPGs in the bridge domain is not always practical.

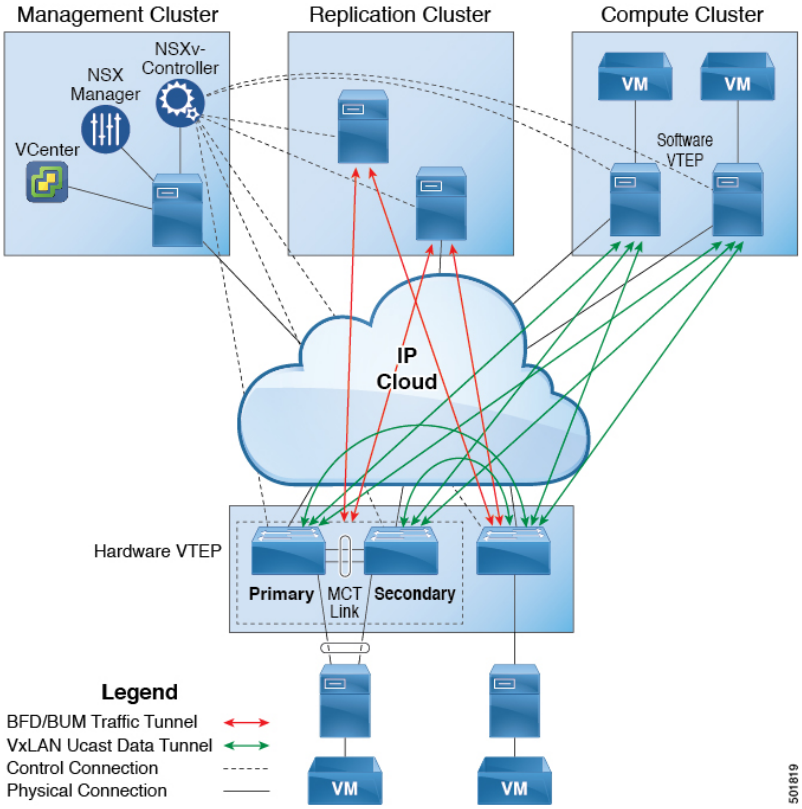
The flood in encapsulation feature helps to consolidate bridge domains in your network. The feature does so by enabling you to control broadcast flooding to endpoints within the bridge domain based on the encapsulation of the virtual or physical domain that the EPGs are associated with.

Flood in encapsulation requires the bridge domain to be configured with a subnet and with IP routing because in order to allow communication between endpoints of different EPGs in the same bridge domain Cisco ACI performs proxy ARP.

Using multiple VLANs in tunnel mode can introduce a few challenges. In a typical deployment using Cisco ACI with a single tunnel, as illustrated in the following figure, there are multiple EPGs under one bridge

domain. In this case, certain traffic is flooded within the bridge domain (and thus in all the EPGs), with the risk of MAC address learning ambiguities that can cause forwarding errors.

Figure 9: Challenges of Cisco ACI with VLAN Tunnel Mode



In this topology, the fabric has a single tunnel network defined that uses one uplink to connect with the Cisco ACI leaf node. Two user VLANs, VLAN 10 and VLAN 11 are carried over this link. The bridge domain is set in flooding mode as the servers' gateways are outside the Cisco ACI cloud. ARP negotiations occur in the following process:

- The server sends one ARP broadcast request over the VLAN 10 network.
- The ARP packet travels through the tunnel network to the external server, which records the source MAC address, learned from its downlink.
- The server then forwards the packet out its uplink to the Cisco ACI leaf switch.
- The Cisco ACI fabric sees the ARP broadcast packet entering on access port VLAN 10 and maps it to EPG1.
- Because the bridge domain is set to flood ARP packets, the packet is flooded within the bridge domain and thus to the ports under both EPGs as they are in the same bridge domain.
- The same ARP broadcast packet comes back over the same uplink.
- The external server sees the original source MAC address from this uplink.

Result: the external device has the same MAC address learned from both the downlink port and uplink port within its single MAC forwarding table, causing traffic disruptions.

Recommended Solution

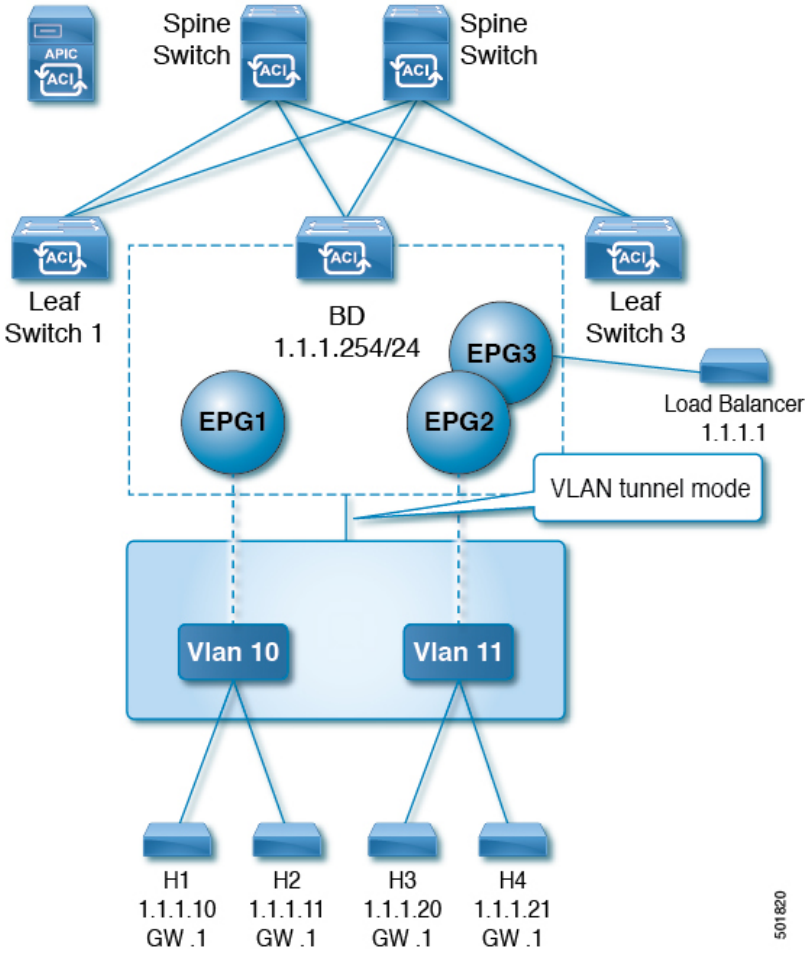
The **Flood in Encapsulation** option is used to limit flooding traffic inside the bridge domain to a single encapsulation. When two EPGs share the same bridge domain and **Flood in Encapsulation** is enabled, the EPG flooding traffic does not reach the other EPG.

Beginning with Cisco Application Policy Infrastructure Controller (APIC) release 3.1(1), on the Cisco Nexus 9000 series switches (with names ending with EX and FX and onwards), all protocols are flooded in encapsulation. Also when enabling **Flood in Encapsulation** for any inter-VLAN traffic, Proxy ARP ensures that the MAC flap issue does not occur, and it limits all flooding (ARP, GARP, and BUM) to the encapsulation. This applies for all EPGs under the bridge domain where it is enabled.

Note Before Cisco APIC release 3.1(1), these features are not supported (Proxy ARP and all protocols being included when flooding within encapsulation). In an earlier Cisco APIC release or earlier generation switches (without EX or FX on their names), if you enable **Flood in Encapsulation** it does not function, no informational fault is generated, but Cisco APIC decreases the health score by 1.

The recommended solution is to support multiple EPGs under one bridge domain by adding an external switch. This design with multiple EPGs under one bridge domain with an external switch is illustrated in the following figure.

Figure 10: Design with Multiple EPGs Under one Bridge Domain with an External Switch



Within the same bridge domain, some EPGs can be service nodes and other EPGs can have flood in encapsulation configured. A Load Balancer resides on a different EPG. The load balancer receives packets from the EPGs and sends them to the other EPGs (there is no proxy ARP and flood within encapsulation does not take place).

If you want to add flood in encapsulation only for selected EPGs, using the NX-OS style CLI, enter the **flood-on-encapsulation enable** command under EPGs.

If you want to add flood in encapsulation for all EPGs, you can use the **multi-destination encap-flood** CLI command under the bridge domain.

Using the CLI, flood in encapsulation configured for an EPG takes precedence over flood in encapsulation that is configured for a bridge domain.

When both bridge domains and EPGs are configured, the behavior is described as follows:

Table 13: Behavior When Both Bridge Domains and EPGs Are Configured

Configuration	Behavior
Flood in encapsulation at the EPG and flood in encapsulation at the bridge domain	Flood in encapsulation takes place for the traffic on all VLANs within the bridge domain.
No flood in encapsulation at the EPG and flood in encapsulation at the bridge domain	Flood in encapsulation takes place for the traffic on all VLANs within the bridge domain.
Flood in encapsulation at the EPG and no flood in encapsulation at the bridge domain	Flood in encapsulation takes place for the traffic on that VLAN within the EPG of the bridge domain.
No flood in encapsulation at the EPG and no flood in encapsulation at the bridge domain	Flooding takes place within the entire bridge domain.

Multi-Destination Protocol Traffic

The EPG/bridge domain level broadcast segmentation is supported for the following network control protocols:

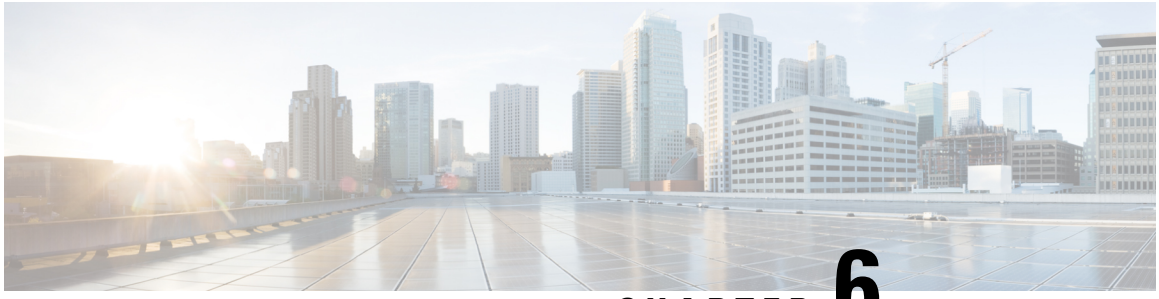
- OSPF
- EIGRP
- LACP
- IS-IS
- BGP
- IGMP
- PIM
- STP-BPDU (flooded within EPG)
- ARP/GARP (controlled by ARP Proxy)
- ND

Flood in Encapsulation Limitations

The following limitations apply when using flood in encapsulation for all protocols:

- Flood in encapsulation does not work in ARP unicast mode.
- Neighbor Solicitation (NS/ND) is not supported for this release.
- You must enable per-port CoPP with flood in encapsulation.
- Flood in encapsulation is supported only in bridge domain in flood mode and ARP in flood mode. Bridge domain spine proxy mode is not supported.
- IPv4 Layer 3 multicast is not supported.
- IPv6 is not supported.
- Virtual machine migration to a different VLAN has momentary issues (60 seconds).

- A load balancer acting as a gateway is supported, for example, in one to one communication between virtual machines and the load balancer in non-proxy mode. No Layer 3 communication is supported. The traffic between virtual machines and the load balancer is on Layer 2. However, if intra-EPG communication passes through the load balancer, then the load balancer changes the SIP and SMAC; otherwise it can lead to a MAC flap. Therefore, Dynamic Source Routing (DSR) mode is not supported in the load balancer.
- Setting up communication between virtual machines through a firwall, as a gateway, is not recommended because if the virtual machine IP address changes to the gateway IP address instead of the firewall IP address, then the firewall can be bypassed.
- Prior releases are not supported (even interoperating between prior and current releases).
- Prior to the 3.2(5) release, the proxy ARP and flood in encapsulation features are not supported for VXLAN encapsulation.
- A mixed-mode topology with Application Leaf Engine (ALE) and Application Spine Engine (ASE) is not recommended and is not supported with flood in encapsulation. Enabling them together can prevent QoS priorities from being enforced.
- Flood in encapsulation is not supported with remote leaf switches and Cisco ACI Multi-Site.
- Flood in encapsulation is not supported for Common Pervasive Gateway (CPGW).
- Flood in encapsulation is not supported on EPGs where microsegmentation is configured.
- If you configure the flood in encapsulation on all EPGs of a bridge domain, ensure that you configure the flood in encapsulation on the bridge domain as well.
- IGMP snooping is not supported with flood in encapsulation.
- There is a condition that causes Cisco ACI to flood in the bridge domain (instead of the encapsulation) packets that are received on an EPG that is configured for flood in encapsulation. This happens regardless of whether the administrator configured flood in encapsulation directly on the EPG or on the bridge domain. The condition for this forwarding behavior is if the ingress leaf node has a remote endpoint for the destination MAC address while the egress leaf node does not have a corresponding local endpoint. This can happen due to reasons such as an interface flapping, an endpoint flush due to STP TCN, learning being disabled on the bridge domain due to an excessive amount of moves, and so on.
- A Layer 3 gateway must be in the Cisco ACI fabric.



CHAPTER 6

EPGs

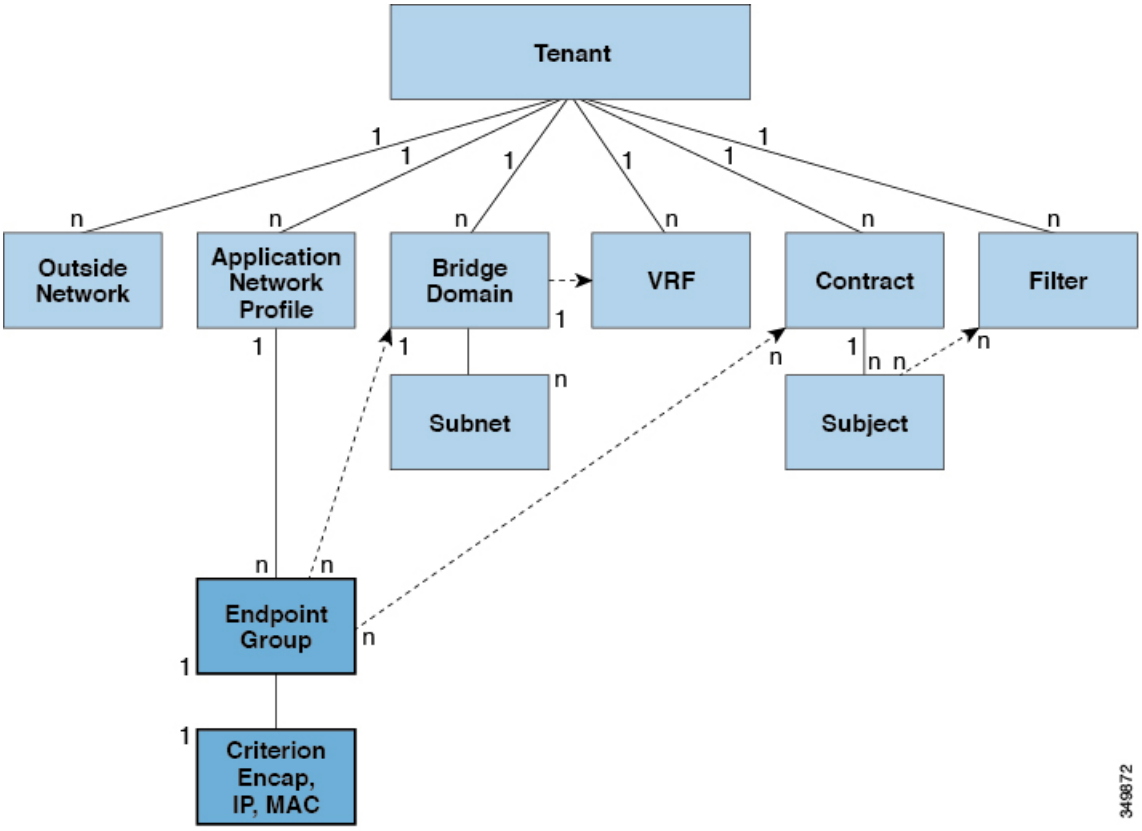
This chapter contains the following sections:

About Endpoint Groups

Endpoint Groups

The endpoint group (EPG) is the most important object in the policy model. The following figure shows where application EPGs are located in the management information tree (MIT) and their relation to other objects in the tenant.

Figure 11: Endpoint Groups



An EPG is a managed object that is a named logical entity that contains a collection of endpoints. Endpoints are devices that are connected to the network directly or indirectly. They have an address (identity), a location, attributes (such as version or patch level), and can be physical or virtual. Knowing the address of an endpoint also enables access to all its other identity details. EPGs are fully decoupled from the physical and logical topology. Endpoint examples include servers, virtual machines, network-attached storage, or clients on the Internet. Endpoint membership in an EPG can be dynamic or static.

The ACI fabric can contain the following types of EPGs:

- Application endpoint group (`fvAEPg`)
- Layer 2 external outside network instance endpoint group (`l2extInstP`)
- Layer 3 external outside network instance endpoint group (`l3extInstP`)
- Management endpoint groups for out-of-band (`mgmtOoB`) or in-band (`mgmtInB`) access.

EPGs contain endpoints that have common policy requirements such as security, virtual machine mobility (VMM), QoS, or Layer 4 to Layer 7 services. Rather than configure and manage endpoints individually, they are placed in an EPG and are managed as a group.

Policies apply to EPGs, never to individual endpoints. An EPG can be statically configured by an administrator in the APIC, or dynamically configured by an automated system such as vCenter or OpenStack.

Note When an EPG uses a static binding path, the encapsulation VLAN associated with this EPG must be part of a static VLAN pool. For IPv4/IPv6 dual-stack configurations, the IP address property is contained in the `fvStIp` child property of the `fvStCEp` MO. Multiple `fvStIp` objects supporting IPv4 and IPv6 addresses can be added under one `fvStCEp` object. When upgrading ACI from IPv4-only firmware to versions of firmware that support IPv6, the existing IP property is copied to an `fvStIp` MO.

Regardless of how an EPG is configured, EPG policies are applied to the endpoints they contain.

WAN router connectivity to the fabric is an example of a configuration that uses a static EPG. To configure WAN router connectivity to the fabric, an administrator configures an `l3extInstP` EPG that includes any endpoints within an associated WAN subnet. The fabric learns of the EPG endpoints through a discovery process as the endpoints progress through their connectivity life cycle. Upon learning of the endpoint, the fabric applies the `l3extInstP` EPG policies accordingly. For example, when a WAN connected client initiates a TCP session with a server within an application (`fvAEPg`) EPG, the `l3extInstP` EPG applies its policies to that client endpoint before the communication with the `fvAEPg` EPG web server begins. When the client server TCP session ends and communication between the client and server terminate, that endpoint no longer exists in the fabric.

Note If a leaf switch is configured for *static binding (leaf switches)* under an EPG, the following restrictions apply:

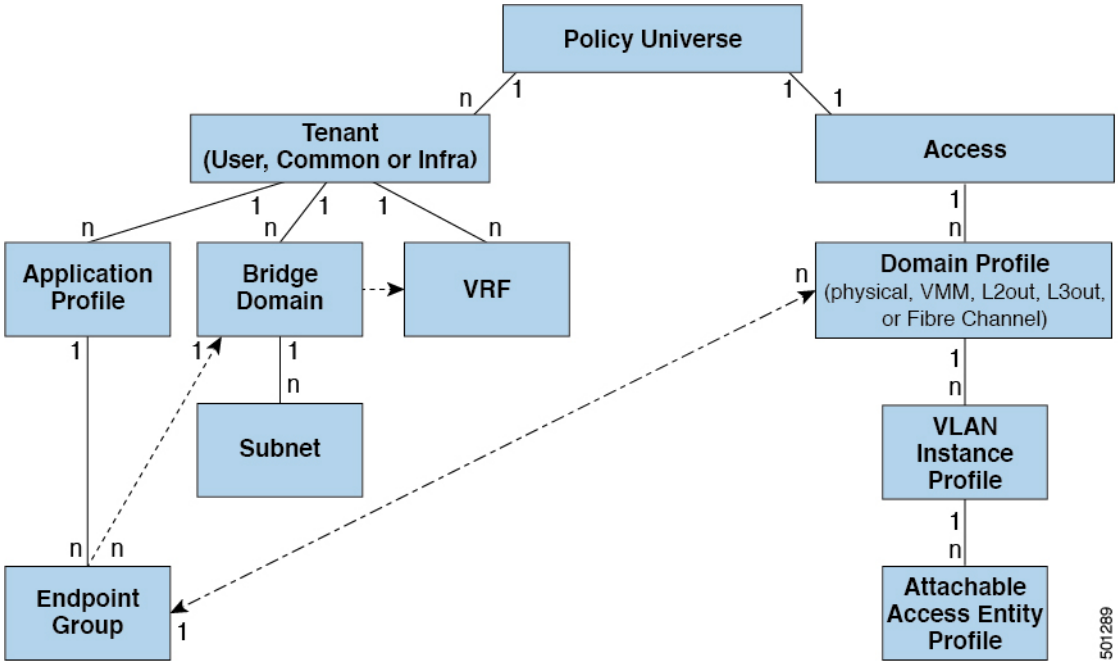
- The static binding cannot be overridden with a static path.
- Interfaces in that switch cannot be used for routed external network (L3out) configurations.
- Interfaces in that switch cannot be assigned IP addresses.

Virtual machine management connectivity to VMware vCenter is an example of a configuration that uses a dynamic EPG. Once the virtual machine management domain is configured in the fabric, vCenter triggers the dynamic configuration of EPGs that enable virtual machine endpoints to start up, move, and shut down as needed.

Access Policies Automate Assigning VLANs to EPGs

While tenant network policies are configured separately from fabric access policies, tenant policies are not activated unless their underlying access policies are in place. Fabric access external-facing interfaces connect to external devices such as virtual machine controllers and hypervisors, hosts, routers, or Fabric Extenders (FEXs). Access policies enable an administrator to configure port channels and virtual port channels, protocols such as LLDP, CDP, or LACP, and features such as monitoring or diagnostics.

Figure 12: Association of Endpoint Groups with Access Policies



In the policy model, EPGs are tightly coupled with VLANs. For traffic to flow, an EPG must be deployed on a leaf port with a VLAN in a physical, VMM, L2out, L3out, or Fibre Channel domain. For more information, see Networking Domains, on page 17.

In the policy model, the domain profile associated to the EPG contains the VLAN instance profile. The domain profile contains both the VLAN instance profile (VLAN pool) and the attacheable Access Entity Profile (AEP), which are associated directly with application EPGs. The AEP deploys the associated application EPGs to all the ports to which it is attached, and automates the task of assigning VLANs. While a large data center could easily have thousands of active virtual machines provisioned on hundreds of VLANs, the ACI fabric can automatically assign VLAN IDs from VLAN pools. This saves a tremendous amount of time, compared with trunking down VLANs in a traditional data center.

VLAN Guidelines

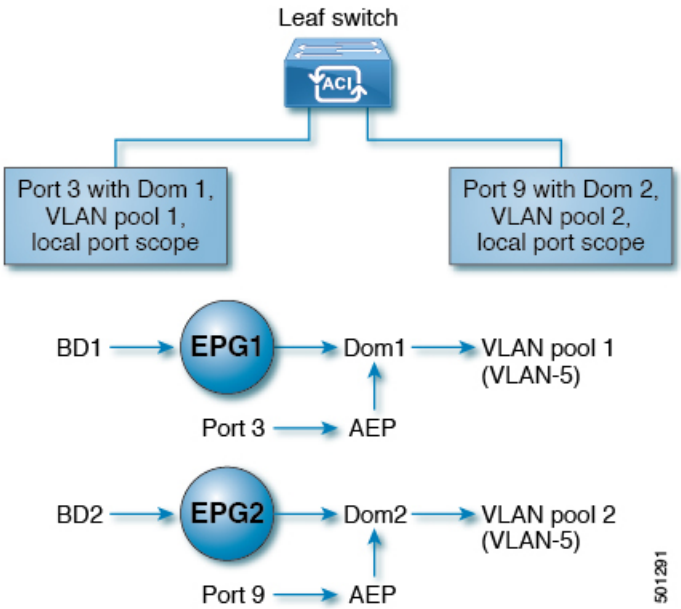
Use the following guidelines to configure the VLANs where EPG traffic will flow.

- Multiple domains can share a VLAN pool, but a single domain can only use one VLAN pool.
- To deploy multiple EPGs with same VLAN encapsulation on a single leaf switch, see Per Port VLAN, on page 44.

Per Port VLAN

In ACI versions prior to the v1.1 release, a given VLAN encapsulation maps to only a single EPG on a leaf switch. If there is a second EPG which has the same VLAN encapsulation on the same leaf switch, the ACI raises a fault.

Starting with the v1.1 release, you can deploy multiple EPGs with the same VLAN encapsulation on a given leaf switch (or FEX), in the Per Port VLAN configuration, similar to the following diagram:



To enable deploying multiple EPGs using the same encapsulation number, on a single leaf switch, use the following guidelines:

- EPGs must be associated with different bridge domains.
- EPGs must be deployed on different ports.
- Both the port and EPG must be associated with the same domain that is associated with a VLAN pool that contains the VLAN number.
- Ports must be configured with `portLocal` VLAN scope.

For example, with Per Port VLAN for the EPGs deployed on ports 3 and 9 in the diagram above, both using VLAN-5, port 3 and EPG1 are associated with Dom1 (pool 1) and port 9 and EPG2 are associated with Dom2 (pool 2).

Traffic coming from port 3 is associated with EPG1, and traffic coming from port 9 is associated with EPG2.

This does not apply to ports configured for Layer 3 external outside connectivity.

When an EPG has more than one physical domain with overlapping VLAN pools, avoid adding more than one domain to the AEP that is used to deploy the EPG on the ports. This avoids the risk of traffic forwarding issues.

When an EPG has only one physical domain with overlapping VLAN pool, you can associate multiple domains with single AEP.

Only ports that have the `vlanScope` set to `portlocal` allow allocation of separate (Port, VLAN) translation entries in both ingress and egress directions. For a given port with the `vlanScope` set to `portGlobal` (the default), each VLAN used by an EPG must be unique on a given leaf switch.

Note Per Port VLAN is not supported on interfaces configured with Multiple Spanning Tree (MST), which requires VLAN IDs to be unique on a single leaf switch, and the VLAN scope to be global.

Reusing VLAN Numbers Previously Used for EPGs on the Same Leaf Switch

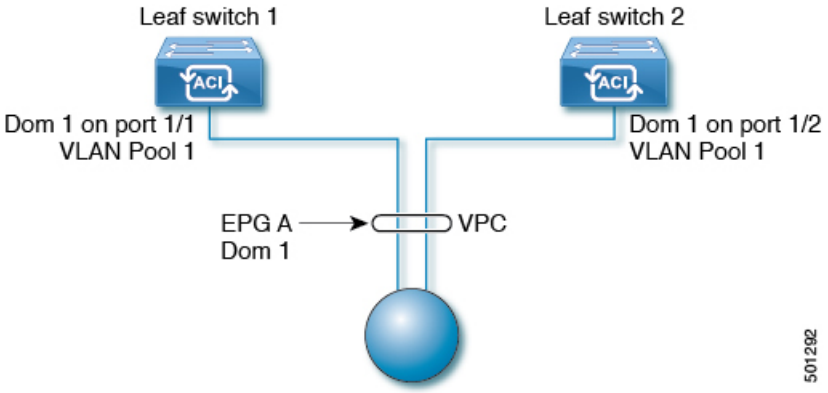
If you have previously configured VLANs for EPGs that are deployed on a leaf switch port, and you want to reuse the same VLAN numbers for different EPGs on different ports on the same leaf switch, use a process, such as the following example, to set them up without disruption:

In this example, EPGs were previously deployed on a port associated with a domain including a VLAN pool with a range of 9-100. You want to configure EPGs using VLAN encapsulations from 9-20.

1. Configure a new VLAN pool on a different port (with a range of, for example, 9-20).
2. Configure a new physical domain that includes leaf ports that are connected to firewalls.
3. Associate the physical domain to the VLAN pool you configured in step 1.
4. Configure the VLAN Scope as `portLocal` for the leaf port.
5. Associate the new EPGs (used by the firewall in this example) to the physical domain you created in step 2.
6. Deploy the EPGs on the leaf ports.

VLAN Guidelines for EPGs Deployed on vPCs

Figure 13: VLANs for Two Legs of a vPC



When an EPG is deployed on a vPC, it must be associated with the same domain (with the same VLAN pool) that is assigned to the leaf switch ports on the two legs of the vPC.

In this diagram, EPG A is deployed on a vPC that is deployed on ports on Leaf switch 1 and Leaf switch 2. The two leaf switch ports and the EPG are all associated with the same domain, containing the same VLAN pool.

Deploying an EPG on a Specific Port

Deploying an EPG on a Specific Node or Port Using the GUI

Before you begin

The tenant where you deploy the EPG is already created.

You can create an EPG on a specific node or a specific port on a node.

Procedure

Step 1 Log in to the Cisco APIC.

Step 2 Choose **Tenants** > ***tenant*** .

Step 3 In the left navigation pane, expand ***tenant*** , **Application Profiles**, and the ***application profile*** .

Step 4 Right-click **Application EPGs** and choose **Create Application EPG**.

Step 5 In the **Create Application EPG STEP 1 > Identity** dialog box, complete the following steps:

a) In the **Name** field, enter a name for the EPG.

b) From the **Bridge Domain** drop-down list, choose a bridge domain.

c) Check the **Statically Link with Leaves/Paths** check box.

This check box allows you to specify on which port you want to deploy the EPG.

d) Click **Next**.

e) From the **Path** drop-down list, choose the static path to the destination EPG.

Step 6 In the **Create Application EPG STEP 2 > Leaves/Paths** dialog box, from the **Physical Domain** drop-down list, choose a physical domain.

Step 7 Complete one of the following sets of steps:

Option	Description
If you want to deploy the EPG on...	Then
A node	**a.** Expand the **Leaves** area. **b.** From the **Node** drop-down list, choose a node. **c.** In the **Encap** field, enter the appropriate VLAN. **d.** (Optional) From the **Deployment Immediacy** drop-down list, accept the default **On Demand** or choose **Immediate**. **e.** (Optional) From the Mode drop-down list, accept the default **Trunk** or choose another mode.
A port on the node	**a.** Expand the **Paths** area. **b.** From the **Path** drop-down list, choose the appropriate node and port.

Option	Description
	c. (Optional) In the **Deployment Immediacy** field drop-down list, accept the default **On Demand** or choose **Immediate**. **d.** (Optional) From the Mode drop-down list, accept the default **Trunk** or choose another mode. **e.** In the **Port Encap** field, enter the secondary VLAN to be deployed. **f.** (Optional) In the **Primary Encap** field, enter the primary VLAN to be deployed.

Step 8 Click **Update** and click **Finish**.

Step 9 In the left navigation pane, expand the EPG that you created.

Step 10 Complete one of the following actions:

- If you created the EPG on a node, click **Static Leafs**, and in the work pane view details of the static binding paths.
- If you created the EPG on a port of the node, click **Static Ports**, and in the work pane view details of the static binding paths.

Deploying an EPG on a Specific Port with APIC Using the NX-OS Style CLI

Procedure

Step 1 Configure a VLAN domain:

Example:

```
apic1(config)# vlan-domain dom1
apic1(config-vlan)# vlan 10-100
```

Step 2 Create a tenant:

Example:

```
apic1# configure
apic1(config)# tenant t1
```

Step 3 Create a private network/VRF:

Example:

```
apic1(config-tenant)# vrf context ctx1
apic1(config-tenant-vrf)# exit
```

Step 4 Create a bridge domain:

Example:

```
apic1(config-tenant)# bridge-domain bd1
```

```
apic1(config-tenant-bd)# vrf member ctx1
apic1(config-tenant-bd)# exit
```

Step 5 Create an application profile and an application EPG:

Example:

```
apic1(config-tenant)# application AP1
apic1(config-tenant-app)# epg EPG1
apic1(config-tenant-app-epg)# bridge-domain member bd1
apic1(config-tenant-app-epg)# exit
apic1(config-tenant-app)# exit
apic1(config-tenant)# exit
```

Step 6 Associate the EPG with a specific port:

Example:

```
apic1(config)# leaf 1017
apic1(config-leaf)# interface ethernet 1/13
apic1(config-leaf-if)# vlan-domain member dom1
apic1(config-leaf-if)# switchport trunk allowed vlan 20 tenant t1 application AP1 epg EPG1
```

Note The vlan-domain and vlan-domain member commands mentioned in the above example are a pre-requisite for deploying an EPG on a port.

Deploying an EPG on a Specific Port with APIC Using the REST API

Before you begin

The tenant where you deploy the EPG is created.

Procedure

Deploy an EPG on a specific port.

Example:

```
<fvTenant name="<tenant_name>" dn="uni/tn-test1" >
    <fvCtx name="<network_name>" pcEnfPref="enforced" knwMcastAct="permit"/>
    <fvBD name="<bridge_domain_name>" unkMcastAct="flood" >
        <fvRsCtx tnFvCtxName="<network_name>"/>
    </fvBD>
    <fvAp name="<application_profile>" >
        <fvAEPg name="<epg_name>" >
            <fvRsPathAtt tDn="topology/pod-1/paths-1017/pathep-[eth1/13]" mode="regular"
instrImedcy="immediate" encap="vlan-20"/>
        </fvAEPg>
    </fvAp>
</fvTenant>
```

Creating Domains, Attach Entity Profiles, and VLANs to Deploy an EPG on a Specific Port

Creating Domains, Attach Entity Profiles, and VLANs to Deploy an EPG on a Specific Port

This topic provides a typical example of how to create physical domains, Attach Entity Profiles (AEP), and VLANs that are mandatory to deploy an EPG on a specific port.

All endpoint groups (EPGs) require a domain. Interface policy groups must also be associated with Attach Entity Profile (AEP), and the AEP must be associated with a domain, if the AEP and EPG have to be in same domain. Based on the association of EPGs to domains and of interface policy groups to domains, the ports and VLANs that the EPG uses are validated. The following domain types associate with EPGs:

- Application EPGs
- Layer 3 external outside network instance EPGs
- Layer 2 external outside network instance EPGs
- Management EPGs for out-of-band and in-band access

The APIC checks if an EPG is associated with one or more of these types of domains. If the EPG is not associated, the system accepts the configuration but raises a fault. The deployed configuration may not function properly if the domain association is not valid. For example, if the VLAN encapsulation is not valid for use with the EPG, the deployed configuration may not function properly.

Note EPG association with the AEP without static binding does not work in a scenario when you configure the EPG as **Trunk** under the AEP with one end point under the same EPG supporting Tagging and the other end point in the same EPG does not support VLAN tagging. While associating AEP under the EPG, you can configure it as Trunk, Access (Tagged) or Access (Untagged).

Creating Domains, and VLANS to Deploy an EPG on a Specific Port Using the GUI

Before you begin

- The tenant where you deploy the EPG is already created.
- An EPG is statically deployed on a specific port.

Procedure

Step 1 On the menu bar, click **Fabric** > **Access Policies**.

Step 2 In the **Navigation** pane, click **Quick Start**.

Step 3 In the **Work** pane, click **Configure an Interface, PC, and vPC**.

Step 4 In the **Configure an Interface, PC, and vPC** dialog box, click the + icon to select switches and perform the following actions:

a) From the **Switches** drop-down list, check the check box for the desired switch.
b) In the **Switch Profile Name** field, a switch name is automatically populated.

Note Optionally, you can enter a modified name.

c) Click the + icon to configure the switch interfaces.
d) In the **Interface Type** field, click the **Individual** radio button.
e) In the **Interfaces** field, enter the range of desired interfaces.
f) In the **Interface Selector Name** field, an interface name is automatically populated.

Note Optionally, you can enter a modified name.

g) In the **Interface Policy Group** field, choose the **Create One** radio button.
h) From the **Link Level Policy** drop-down list, choose the appropriate link level policy.

Note Create additional policies as desired, otherwise the default policy settings are available.

i) From the **Attached Device Type** field, choose the appropriate device type.
j) In the **Domain** field, click the **Create One** radio button.
k) In the **Domain Name** field, enter a domain name.
l) In the **VLAN** field, click the **Create One** radio button.
m) In the **VLAN Range** field, enter the desired VLAN range. Click **Save**, and click **Save** again.
n) Click **Submit**.

Step 5 On the menu bar, click **Tenants**. In the **Navigation** pane, expand the appropriate ***Tenant_name*** > **Application Profiles** > **Application EPGs** > ***EPG_name*** and perform the following actions:

a) Right-click **Domains (VMs and Bare-Metals)**, and click **Add Physical Domain Association**.
b) In the **Add Physical Domain Association** dialog box, from the **Physical Domain Profile** drop-down list, choose the appropriate domain.
c) Click **Submit**.
The AEP is associated with a specific port on a node and with a domain. The physical domain is associated with the VLAN pool and the Tenant is associated with this physical domain.

The switch profile and the interface profile are created. The policy group is created in the port block under the interface profile. The AEP is automatically created, and it is associated with the port block and with the domain. The domain is associated with the VLAN pool and the Tenant is associated with the domain.

Creating AEP, Domains, and VLANs to Deploy an EPG on a Specific Port Using the NX-OS Style CLI

Before you begin

- The tenant where you deploy the EPG is already created.

- An EPG is statically deployed on a specific port.

Procedure

Step 1 Create a VLAN domain and assign VLAN ranges:

Example:

```
apic1(config)# vlan-domain domP
apic1(config-vlan)# vlan 10
apic1(config-vlan)# vlan 25
apic1(config-vlan)# vlan 50-60
apic1(config-vlan)# exit
```

Step 2 Create an interface policy group and assign a VLAN domain to the policy group:

Example:

```
apic1(config)# template policy-group PortGroup
apic1(config-pol-grp-if)# vlan-domain member domP
```

Step 3 Create a leaf interface profile, assign an interface policy group to the profile, and assign the interface IDs on which the profile will be applied:

Example:

```
apic1(config)# leaf-interface-profile InterfaceProfile1
apic1(config-leaf-if-profile)# leaf-interface-group range
apic1(config-leaf-if-group)# policy-group PortGroup
apic1(config-leaf-if-group)# interface ethernet 1/11-13
apic1(config-leaf-if-profile)# exit
```

Step 4 Create a leaf profile, assign the leaf interface profile to the leaf profile, and assign the leaf IDs on which the profile will be applied:

Example:

```
apic1(config)# leaf-profile SwitchProfile-1019
apic1(config-leaf-profile)# leaf-interface-profile InterfaceProfile1
apic1(config-leaf-profile)# leaf-group range
apic1(config-leaf-group)# leaf 1019
apic1(config-leaf-group)#
```

Creating AEP, Domains, and VLANs to Deploy an EPG on a Specific Port Using the REST API

Before you begin

- The tenant where you deploy the EPG is already created.
- An EPG is statically deployed on a specific port.

Procedure

Step 1 Create the interface profile, switch profile and the Attach Entity Profile (AEP).

Example:

```
   <infraInfra>

      <infraNodeP  name="<switch_profile_name>" dn="uni/infra/nprof-<switch_profile_name>"
 >
           <infraLeafS  name="SwitchSeletor" descr="" type="range">
                <infraNodeBlk name="nodeBlk1" descr="" to_="1019" from_="1019"/>
           </infraLeafS>
          <infraRsAccPortP tDn="uni/infra/accportprof-<interface_profile_name>"/>
      </infraNodeP>

      <infraAccPortP name="<interface_profile_name>"
dn="uni/infra/accportprof-<interface_profile_name>" >
           <infraHPortS  name="portSelector"  type="range">
                <infraRsAccBaseGrp tDn="uni/infra/funcprof/accportgrp-<port_group_name>"
fexId="101"/>
               <infraPortBlk name="block2"  toPort="13" toCard="1" fromPort="11"
fromCard="1"/>
          </infraHPortS>
     </infraAccPortP>

    <infraAccPortGrp  name="<port_group_name>"
dn="uni/infra/funcprof/accportgrp-<port_group_name>" >
          <infraRsAttEntP tDn="uni/infra/attentp-<attach_entity_profile_name>"/>
          <infraRsHIfPol tnFabricHIfPolName="1GHifPol"/>
    </infraAccPortGrp>

    <infraAttEntityP  name="<attach_entity_profile_name>"
dn="uni/infra/attentp-<attach_entity_profile_name>" >
         <infraRsDomP tDn="uni/phys-<physical_domain_name>"/>
    </infraAttEntityP>

<infraInfra>
```

Step 2 Create a domain.

Example:

```
<physDomP  name="<physical_domain_name>" dn="uni/phys-<physical_domain_name>">
    <infraRsVlanNs tDn="uni/infra/vlanns-[<vlan_pool_name>]-static"/>
</physDomP>
```

Step 3 Create a VLAN range.

Example:

```
<fvnsVlanInstP  name="<vlan_pool_name>" dn="uni/infra/vlanns-[<vlan_pool_name>]-static"
allocMode="static">
    <fvnsEncapBlk name="" descr="" to="vlan-25" from="vlan-10"/>
</fvnsVlanInstP>
```

Step 4 Associate the EPG with the domain.

Example:

```
<fvTenant  name="<tenant_name>" dn="uni/tn-" >
    <fvAEPg prio="unspecified" name="<epg_name>" matchT="AtleastOne"
dn="uni/tn-test1/ap-AP1/epg-<epg_name>" descr="">
        <fvRsDomAtt tDn="uni/phys-<physical_domain_name>" instrImedcy="immediate"
resImedcy="immediate"/>
```

```
    </fvAEPg>
</fvTenant>
```

Validating Overlapping VLANs

This global feature prevents association of overlapping VLAN pools on a single EPG. If there are any overlapping pools allocated with any EPG in APIC, then this feature cannot be enabled (an error is displayed if there is an attempt to enable it). If no existing overlapping pools are present, then this feature can be enabled. Once enabled, when an attempt to allocate a domain on an EPG is performed, and the domain contains a VLAN pool with a range overlapping with another domain already associate to the EPG, then the configuration is blocked.

When overlapping VLAN pools exist under an EPG, then the FD VNID allocated for the EPG by each switch is non-deterministic and different switches may allocate different VNIDs. This can cause EPM sync failures between leafs within a vPC domain (causing intermittent connectivity for all endpoints within the EPG). It can also cause bridging loops if user is extending STP between the EPG, as the BPDUs will be dropped between switches due to FD VNID mismatch.

Validating Overlapping VLANs Using the GUI

This procedure provides an example of using the APIC GUI to configure overlapping VLAN validation.

Procedure

Step 1 On the menu bar, choose **System** > **System Settings**.

Step 2 In the navigation pane, choose **Fabric Wide Setting**.

Step 3 In the work pane, locate and check **Enforce EPG VLAN Validation**.

Note If overlapping VLAN pools already exist and this parameter is checked, the system returns an error. You must assign VLAN pools that are not overlapping to the EPGs before choosing this feature.

If this parameter is checked and an attempt is made to add an overlapping VLAN pool to an EPG, the system returns an error.

Step 4 Click **Submit**.

Validating Overlapping VLANs Using the REST API

This procedure provides an example of using the REST API to configure overlapping VLAN validation.

Procedure

Step 1 Send this HTTP POST message to enable the validation using the XML API.

Example:

```
POST https://apic-ip-address/api/mo/infra/settings.xml
```

Step 2 Include this XML structure in the body of the POST message.

Example:

```
<?xml version="1.0" encoding="UTF-8"?>
<infraSetPol validateOverlappingVlans=yes />'
```

Note If overlapping VLANs already exist, an error message appears during the POST with the corresponding EPG that has overlapping VLANs:

```
<?xml version="1.0" encoding="UTF-8"?><imdata totalCount="1">
<error code="100" text="Validation failed: Vlan ranges for an EPg cannot overlap
 Dn0=uni/tn-ag/ap-app/epg-e1,
"/></imdata>
```

Deploying EPGs to Multiple Interfaces Through Attached Entity Profiles

Deploying an Application EPG through an AEP or Interface Policy Group to Multiple Ports

Through the APIC Advanced GUI and REST API, you can associate attached entity profiles directly with application EPGs. By doing so, you deploy the associated application EPGs to all those ports associated with the attached entity profile in a single configuration.

Through the APIC REST API or the NX-OS style CLI, you can deploy an application EPG to multiple ports through an Interface Policy Group.

Deploying an EPG through an AEP to Multiple Interfaces Using the APIC GUI

You can quickly associate an application with an attached entity profile to quickly deploy that EPG over all the ports associated with that attached entity profile.

Before you begin

- The target application EPG is created.
- The VLAN pools has been created containing the range of VLANs you wish to use for EPG Deployment on the AEP.
- The physical domain has been created and linked to the VLAN Pool and AEP.
- The target attached entity profile is created and is associated with the ports on which you want to deploy the application EPG.

Procedure

Step 1 Navigate to the target attached entity profile.

a) Open the page for the attached entity profile to use. In the GUI, click **Fabric** > **Access Policies** > **Policies** > **Global** > **Attachable Access Entity Profiles**.

b) Click the target attached entity profile to open its Attachable Access Entity Profile window.

Step 2 Click the **Show Usage** button to view the leaf switches and interfaces associated with this attached entity profile.

the application EPGs associated with this attached entity profile are deployed to all the ports on all the switches associated with this attached entity profile.

Step 3 Use the **Application EPGs** table to associate the target application EPG with this attached entity profile. Click + to add an application EPG entry. Each entry contains the following fields:

Field	Action
Application EPGs	Use the drop down to choose the associated Tenant, Application Profile, and target application EPG.
Encap	Enter the name of the VLAN over which the target application EPG will communicate.
Primary Encap	If the application EPG requires a primary VLAN, enter the name of the primary VLAN.
Mode	Use the drop down to specify the mode in which data is transmitted: • **Trunk** -- Choose if traffic from the host is tagged with a VLAN ID. • **Access** -- Choose if traffic from the host is tagged with an 802.1p tag. • **Access Untagged** -- Choose if the traffic from the host is untagged.

Step 4 Click **Submit**.
the application EPGs associated with this attached entity profile are deployed to all the ports on all the switches associated with this attached entity profile.

Deploying an EPG through an Interface Policy Group to Multiple Interfaces Using the NX-OS Style CLI

In the NX-OS CLI, an attached entity profile is not explicitly defined to associate with an EPG for rapid deployment; instead the interface policy group is defined, assigned a domain, applied to all the ports associated with a VLAN and configured to include the application EPG to be deployed over that VLAN.

Before you begin

- The target application EPG is created.

- The VLAN pools has been created containing the range of VLANs you wish to use for EPG Deployment on the AEP.
- The physical domain has been created and linked to the VLAN Pool and AEP.
- The target attached entity profile is created and is associated with the ports on which you want to deploy the application EPG.

Procedure

Step 1 Associate the target EPG with the interface policy group.

The sample command sequence specifies an interface policy group **pg3** associated with VLAN domain, **domain1**, and with VLAN **1261**. The application EPG, **epg47** is deployed to all interfaces associated with this policy group.

Example:

```
 apic1# configure terminal
apic1(config)# template policy-group pg3
apic1(config-pol-grp-if)# vlan-domain member domain1
apic1(config-pol-grp-if)# switchport trunk allowed vlan 1261 tenant tn10 application pod1-AP

      epg epg47
```

Step 2 Check the target ports to ensure deployment of the policies of the interface policy group associated with application EPG.

The output of the sample **show** command sequence indicates that policy group **pg3** is deployed on Ethernet port **1/20** on leaf switch **1017**.

Example:

```
 apic1# show run leaf 1017 int eth 1/20
# Command: show running-config leaf 1017 int eth 1/20
# Time: Mon Jun 27 22:12:10 2016
  leaf 1017
    interface ethernet 1/20
      policy-group pg3
      exit
    exit
ifav28-ifc1#
```

Deploying an EPG through an AEP to Multiple Interfaces Using the REST API

The interface selectors in the AEP enable you to configure multiple paths for an AEPg. The following can be selected:

1. A node or a group of nodes
2. An interface or a group of interfaces

   The interfaces consume an interface policy group (and so an `infra:AttEntityP`).
3. The infra:AttEntityP is associated to the AEPg, thus specifying the VLANs to use.

An infra:AttEntityP can be associated with multiple AEPgs with different VLANs.

When you associate the infra:AttEntityP with the AEPg, as in 3, this deploys the AEPg on the nodes selected in 1, on the interfaces in 2, with the VLAN provided by 3.

In this example, the AEPg `uni/tn-Coke/ap-AP/epg-EPG1` is deployed on interfaces 1/10, 1/11, and 1/12 of nodes 101 and 102, with vlan-102.

Before you begin

- Create the target application EPG (`AEPg`).
- Create the VLAN pool containing the range of VLANs you wish to use for EPG deployment with the Attached Entity Profile (AEP).
- Create the physical domain and link it to the VLAN pool and AEP.

Procedure

To deploy an AEPg on selected nodes and interfaces, send a post with XML such as the following:

Example:

```
 <infraInfra dn="uni/infra">
   <infraNodeP name="NodeProfile">
       <infraLeafS name="NodeSelector" type="range">
          <infraNodeBlk name="NodeBlok" from_="101" to_="102"/>
          <infraRsAccPortP tDn="uni/infra/accportprof-InterfaceProfile"/>
        </infraLeafS>
   </<infraNodeP>

   <infraAccPortP name="InterfaceProfile">
      <infraHPortS name="InterfaceSelector" type="range">
          <infraPortBlk name=" InterfaceBlock" fromCard="1" toCard="1" fromPort="10"
toPort="12"/>
          <infraRsAccBaseGrp tDn="uni/infra/funcprof/accportgrp-PortGrp" />
      </infraHPortS>
   </infraAccPortP>

   <infraFuncP>
       <infraAccPortGrp name="PortGrp">
           <infraRsAttEntP  tDn="uni/infra/attentp-AttEntityProfile"/>
       </infraAccPortGrp>
   </infraFuncP>

   <infraAttEntityP name="AttEntityProfile" >
       <infraGeneric name="default" >
          <infraRsFuncToEpg tDn="uni/tn-Coke/ap-AP/epg-EPG1" encap="vlan-102"/>
       </infraGeneric>
   </infraAttEntityP>
</infraInfra>
```

Intra-EPG Isolation

Intra-EPG Endpoint Isolation

Intra-EPG endpoint isolation policies provide full isolation for virtual or physical endpoints; no communication is allowed between endpoints in an EPG that is operating with isolation enforced. Isolation enforced EPGs reduce the number of EPG encapsulations required when many clients access a common service but are not allowed to communicate with each other.

An EPG is isolation enforced for all Cisco Application Centric Infrastructure (ACI) network domains or none. While the Cisco ACI fabric implements isolation directly to connected endpoints, switches connected to the fabric are made aware of isolation rules according to a primary VLAN (PVLAN) tag.

Note If an EPG is configured with intra-EPG endpoint isolation enforced, these restrictions apply:

- All Layer 2 endpoint communication across an isolation enforced EPG is dropped within a bridge domain.
- All Layer 3 endpoint communication across an isolation enforced EPG is dropped within the same subnet.
- Preserving QoS CoS priority settings is not supported when traffic is flowing from an EPG with isolation enforced to an EPG without isolation enforced.

BPDUs are not forwarded through EPGs with intra-EPG isolation enabled. Therefore, when you connect an external Layer 2 network that runs spanning tree in a VLAN that maps to an isolated EPG on Cisco ACI, Cisco ACI might prevent spanning tree in the external network from detecting a Layer 2 loop. You can avoid this issue by ensuring that there is only a single logical link between Cisco ACI and the external network in these VLANs.

Intra-EPG Isolation for Bare Metal Servers

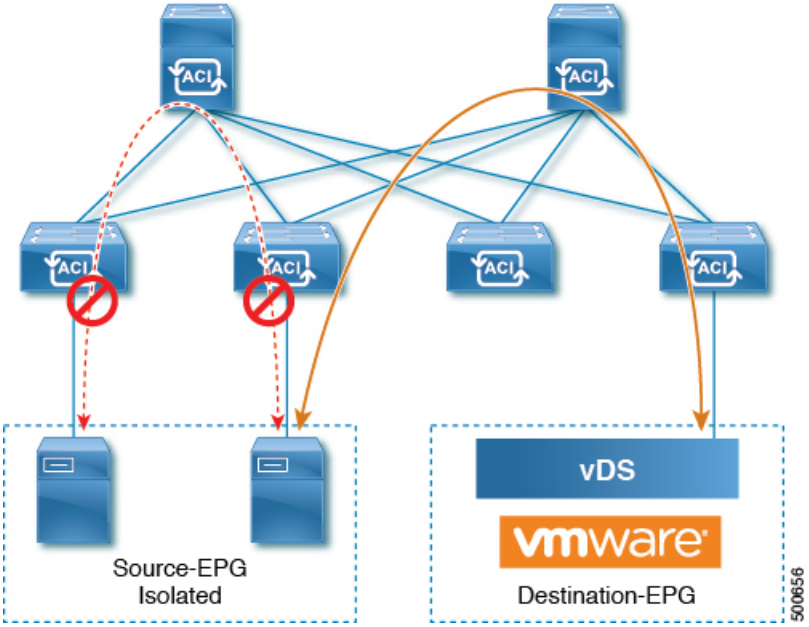
Intra-EPG Isolation for Bare Metal Servers

Intra-EPG endpoint isolation policies can be applied to directly connected endpoints such as bare metal servers.

Examples use cases include the following:

- Backup clients have the same communication requirements for accessing the backup service, buy they don't need to communicate with each other.
- Servers behind a load balancer have the same communication requirements, but isolating them from each other protects against a server that is compromised or infected.

Figure 14: Intra-EPG Isolation for Bare Metal Servers



Bare metal EPG isolation is enforced at the leaf switch. Bare metal servers use VLAN encapsulation. All unicast, multicast and broadcast traffic is dropped (denied) within isolation enforced EPGs. ACI bridge-domains can have a mix of isolated and regular EPGs. Each Isolated EPG can have multiple VLANs where intra-vlan traffic is denied.

Configuring Intra-EPG Isolation for Bare Metal Servers Using the GUI

The port the EPG uses must be associated with a bare metal server interface in the physical domain that is used to connect the bare metal servers directly to leaf switches.

Procedure

Step 1 In a tenant, right click on an **Application Profile**, and open the **Create Application EPG** dialog box to perform the following actions:

a) In the **Name** field, add the EPG name (intra_EPG-deny).
b) For **Intra EPG Isolation**, click **Enforced**.
c) In the **Bridge Domain** field, choose the bridge domain from the drop-down list (bd1).
d) Check the **Statically Link with Leaves/Paths** check box.
e) Click **Next**.

Step 2 In the **Leaves/Paths** dialog box, perform the following actions:

a) In the **Path** section, choose a path from the drop-down list (Node-107/eth1/16) in Trunk Mode.

Specify the **Port Encap** (vlan-102) for the secondary VLAN.

Note If the bare metal server is directly connected to a leaf switch, only the Port Encap secondary VLAN is specified.

Specify the **Primary Encap** (vlan-103) for the primary VLAN.

b) Click **Update**.
c) Click **Finish**.

Configuring Intra-EPG Isolation for Bare Metal Servers Using the NX-OS Style CLI

Procedure

	Command or Action	Purpose

Step 1	In the CLI, create an intra-EPG isolation EPG:
	Example:
	The VMM case is below.
	`ifav19-ifc1(config)# tenant Test_Isolation`
	`ifav19-ifc1(config-tenant)# application PVLAN`
	`ifav19-ifc1(config-tenant-app)# epg EPG1`
	
	`ifav19-ifc1(config-tenant-app-epg)# show running-config`
	`# Command: show running-config`
	`tenant Test_Isolation`
	`application PVLAN epg EPG1`
	`tenant Test_Isolation`
	`application PVLAN`
	`epg EPG1`
	`bridge-domain member BD1`
	`contract consumer bare-metal`
	`contract consumer default`
	`contract provider Isolate_EPG`
	`isolation enforce <---- This enables EPG isolation mode.`
	`exit`
	`exit`
	`ifav19-ifc1(config)# leaf ifav19-leaf3`
	`ifav19-ifc1(config-leaf)# interface ethernet 1/16`
	`ifav19-ifc1(config-leaf-if)# show running-config`
	`ifav19-ifc1(config-leaf-if)# switchport trunk native vlan 101 tenant Test_Isolation application PVLAN epg StaticEPG primary-vlan 100`
	`exit`	

Step 2	Verify the configuration:
	Example:
	`show epg StaticEPG detail`
	`Application EPg Data:`
	`Tenant : Test_Isolation`
	`Application : PVLAN`
	`AEPg : StaticEPG`
	`BD : BD1`
	`uSeg EPG : no`
	`Intra EPG Isolation : enforced`
	`Vlan Domains : phys`	

	Command or Action	Purpose
	Consumed Contracts : bare-metal Provided Contracts : default,Isolate_EPG Denied Contracts : Qos Class : unspecified Tag List : VMM Domains: Domain Type Deployment Immediacy Resolution Immediacy State Encap Primary Encap ---------- --------- ---------- ---------- ---------- ---------- ---------- DVS1 VMware On Demand immediate formed auto auto Static Leaves: Node Encap Deployment Immediacy Mode Modification Time ---------- ---------- ---------- ---------- ---------- Static Paths: Node Interface Encap Modification Time ---------- ---------- ---------- ---------- 1018 eth101/1/1 vlan-100 2016-02-11T18:39:02.337-08:00 1019 eth1/16 vlan-101 2016-02-11T18:39:02.337-08:00 Static Endpoints: Node Interface Encap End Point MAC End Point IP Address Modification Time ---------- ---------- ---------- ---------- ---------- ----------	

Configuring Intra-EPG Isolation for Bare Metal Servers Using the REST API

Before you begin

The port the EPG uses must be associated with a bare metal server interface in the physical domain.

Procedure

Step 1 Send this HTTP POST message to deploy the application using the XML API.

Example:

```
POST https://apic-ip-address/api/mo/uni/tn-ExampleCorp.xml
```

Step 2 Include this XML structure in the body of the POST message.

Example:

```
<fvTenant name="Tenant_BareMetal" >
  <fvAp name="Web">
    <fvAEPg name="IntraEPGDeny" pcEnfPref="enforced">
          <!-- pcEnfPref="enforced" ENABLES ISOLATION-->
          <fvRsBd tnFvBDName="bd" />
          <fvRsDomAtt tDn="uni/phys-Dom1" />
          <!-- PATH ASSOCIATION -->
          <fvRsPathAtt tDn="topology/pod-1/paths-1017/pathep-[eth1/2]" encap="vlan-51"
primaryEncap="vlan-100" instrImedcy='immediate'/>
    </fvAEPg>
 </fvAp>
</fvTenant>
```

Intra-EPG Isolation for VMWare vDS

Intra-EPG Isolation for VMware VDS or Microsoft Hyper-V Virtual Switch

Intra-EPG Isolation is an option to prevent physical or virtual endpoint devices that are in the same base EPG or uSeg EPG from communicating with each other. By default, endpoint devices included in the same EPG are allowed to communicate with one another. However, conditions exist in which total isolation of the endpoint devices from on another within an EPG is desirable. For example, you may want to enforce intra-EPG isolation if the endpoint VMs in the same EPG belong to multiple tenants, or to prevent the possible spread of a virus.

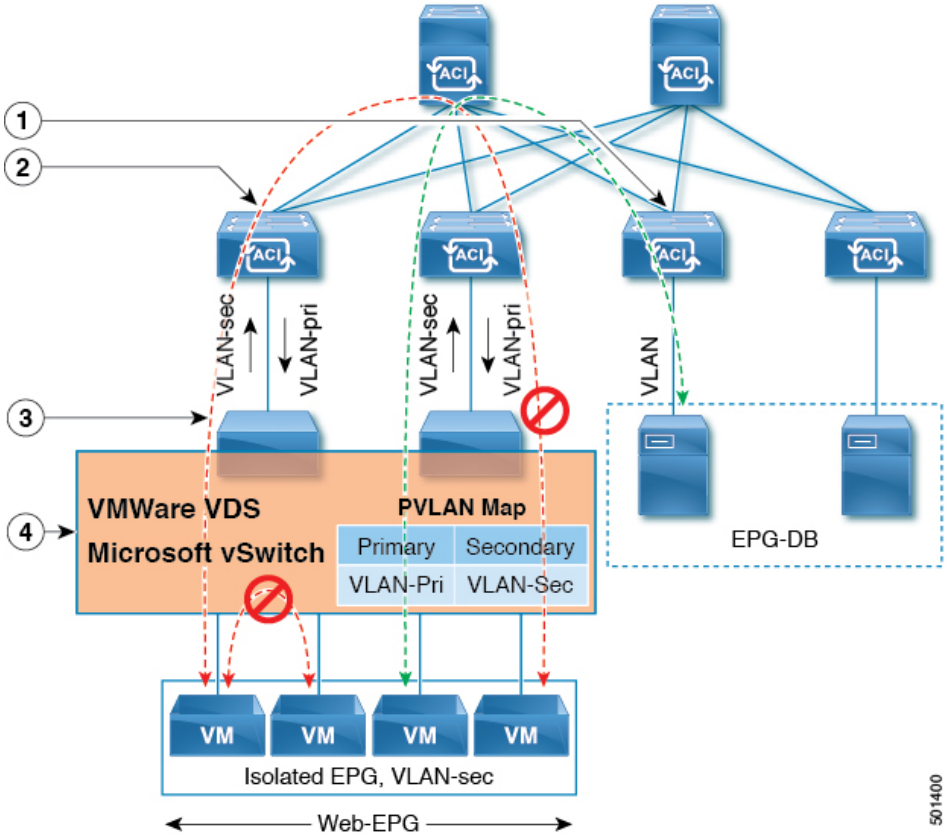
A Cisco ACI virtual machine manager (VMM) domain creates an isolated PVLAN port group at the VMware VDS or Microsoft Hyper-V Virtual Switch for each EPG that has intra-EPG isolation enabled. A fabric administrator specifies primary encapsulation or the fabric dynamically specifies primary encapsulation at the time of EPG-to-VMM domain association. When the fabric administrator selects the VLAN-pri and VLAN-sec values statically, the VMM domain validates that the VLAN-pri and VLAN-sec are part of a static block in the domain pool.

Note When intra-EPG isolation is not enforced, the VLAN-pri value is ignored even if it is specified in the configuration.

VLAN-pri/VLAN-sec pairs for the VMware VDS or Microsoft Hyper-V Virtual Switch are selected per VMM domain during the EPG-to-domain association. The port group created for the intra-EPG isolation EPGs uses the VLAN-sec tagged with type set to `PVLAN`. The VMware VDS or the Microsoft Hyper-V Virtual Switch and fabric swap the VLAN-pri/VLAN-sec encapsulation:

- Communication from the Cisco ACI fabric to the VMware VDS or Microsoft Hyper-V Virtual Switch uses VLAN-pri.
- Communication from the VMware VDS or Microsoft Hyper-V Virtual Switch to the Cisco ACI fabric uses VLAN-sec.

Figure 15: Intra-EPG Isolation for VMware VDS or Microsoft Hyper-V Virtual Switch



Note these details regarding this illustration:

1. EPG-DB sends VLAN traffic to the Cisco ACI leaf switch. The Cisco ACI egress leaf switch encapsulates traffic with a primary VLAN (PVLAN) tag and forwards it to the Web-EPG endpoint.
2. The VMware VDS or Microsoft Hyper-V Virtual Switch sends traffic to the Cisco ACI leaf switch using VLAN-sec. The Cisco ACI leaf switch drops all intra-EPG traffic because isolation is enforced for all intra VLAN-sec traffic within the Web-EPG.
3. The VMware VDS or Microsoft Hyper-V Virtual Switch VLAN-sec uplink to the Cisco ACI Leaf is in isolated trunk mode. The Cisco ACI leaf switch uses VLAN-pri for downlink traffic to the VMware VDS or Microsoft Hyper-V Virtual Switch.

4. The PVLAN map is configured in the VMware VDS or Microsoft Hyper-V Virtual Switch and Cisco ACI leaf switches. VM traffic from WEB-EPG is encapsulated in VLAN-sec. The VMware VDS or Microsoft Hyper-V Virtual Switch denies local intra-WEB EPG VM traffic according to the PVLAN tag. All intra-ESXi host or Microsoft Hyper-V host VM traffic is sent to the Cisco ACI leaf using VLAN-Sec.

Related Topics

For information on configuring intra-EPG isolation in a Cisco AVS environment, see Intra-EPG Isolation Enforcement for Cisco AVS, on page 68.

Configuring Intra-EPG Isolation for VMware VDS or Microsoft Hyper-V Virtual Switch using the GUI

Procedure

Step 1 Log into Cisco APIC.

Step 2 Choose **Tenants** > ***tenant*** .

Step 3 In the left navigation pane expand the **Application Profiles** folder and appropriate application profile.

Step 4 Right-click the **Application EPGs** folder and then choose **Create Application EPG**.

Step 5 In the **Create Application EPG** dialog box, complete the following steps:

a) In the **Name** field, add the EPG name.
b) In the **Intra EPG Isolation** area, click **Enforced**.
c) In the **Bridge Domain** field, choose the bridge domain from the drop-down list.
d) Associate the EPG with a bare metal/physical domain interface or with a VM Domain.
   - For the VM Domain case, check the **Associate to VM Domain Profiles** check box.
   - For the bare metal case, check the **Statically Link with Leaves/Paths** check box.
e) Click **Next**.
f) In the **Associated VM Domain Profiles** area, click the + icon.
g) From the **Domain Profile** drop-down list, choose the desired VMM domain.

   For the static case, in the **Port Encap (or Secondary VLAN for Micro-Seg)** field, specify the secondary VLAN, and in the **Primary VLAN for Micro-Seg** field, specify the primary VLAN. If the Encap fields are left blank, values will be allocated dynamically.

   Note For the static case, a static VLAN must be available in the VLAN pool.

Step 6 Click **Update** and click **Finish**.

Configuring Intra-EPG Isolation for VMware VDS or Microsoft Hyper-V Virtual Switch using the NX-OS Style CLI

Procedure

Step 1 In the CLI, create an intra-EPG isolation EPG:

Example:

The following example is for VMware VDS:

```
apic1(config)# tenant Test_Isolation
apic1(config-tenant)# application PVLAN
apic1(config-tenant-app)# epg EPG1
apic1(config-tenant-app-epg)# show running-config
# Command: show running-config tenant Tenant_VMM application Web epg intraEPGDeny
  tenant Tenant_VMM
    application Web
      epg intraEPGDeny
        bridge-domain member VMM_BD
        vmware-domain member PVLAN encap vlan-2001 primary-encap vlan-2002 push on-demand
        vmware-domain member mininet
          exit
        isolation enforce
        exit
      exit
    exit
apic1(config-tenant-app-epg)#
```

Example:

The following example is for Microsoft Hyper-V Virtual Switch:

```
apic1(config)# tenant Test_Isolation
apic1(config-tenant)# application PVLAN
apic1(config-tenant-app)# epg EPG1
apic1(config-tenant-app-epg)# show running-config
# Command: show running-config tenant Tenant_VMM application Web epg intraEPGDeny
  tenant Tenant_VMM
    application Web
      epg intraEPGDeny
        bridge-domain member VMM_BD
        microsoft-domain member domain1 encap vlan-2003 primary-encap vlan-2004
        microsoft-domain member domain2
          exit
        isolation enforce
        exit
      exit
    exit
apic1(config-tenant-app-epg)#
```

Step 2 Verify the configuration:

Example:

```
 show epg StaticEPG detail
Application EPg Data:
Tenant              : Test_Isolation
Application         : PVLAN
AEPg                : StaticEPG
BD                  : VMM_BD
uSeg EPG            : no
```

```
Intra EPG Isolation : enforced
Vlan Domains        : VMM
Consumed Contracts  : VMware_vDS-Ext
Provided Contracts  : default,Isolate_EPG
Denied Contracts    :
Qos Class           : unspecified
Tag List            :
VMM Domains:
Domain                Type       Deployment Immediacy  Resolution Immediacy  State
   Encap       Primary
Encap
 -------------------- ---------  --------------------  --------------------  --------------
  ----------  ----------
 DVS1                  VMware     On Demand             immediate             formed
    auto        auto

Static Leaves:
 Node        Encap             Deployment Immediacy  Mode                Modification Time

----------  ----------------  --------------------  ------------------
------------------------------

Static Paths:
 Node        Interface                        Encap             Modification Time

 ----------  ------------------------------  ----------------  ------------------------------

 1018        eth101/1/1                      vlan-100          2016-02-11T18:39:02.337-08:00

 1019        eth1/16                         vlan-101          2016-02-11T18:39:02.337-08:00


Static Endpoints:
 Node        Interface          Encap             End Point MAC      End Point IP Address
         Modification Time
 ----------  ------------------------------  ----------------  ------------------
------------------------------  ------------------------------

Dynamic Endpoints:
Encap: (P):Primary VLAN, (S):Secondary VLAN
 Node        Interface            Encap             End Point MAC      End Point IP Address
         Modification Time
 ----------  ------------------------------  ----------------  ------------------
------------------------------  ------------------------------
1017        eth1/3               vlan-943(P)       00:50:56:B3:64:C4  ---
      2016-02-17T18:35:32.224-08:00
                                 vlan-944(S)
```

Configuring Intra-EPG Isolation for VMware VDS or Microsoft Hyper-V Virtual Switch using the REST API

Procedure

Step 1 Send this HTTP POST message to deploy the application using the XML API.

Example:

```
POST https://apic-ip-address/api/mo/uni/tn-ExampleCorp.xml
```

Step 2 For a VMware VDS or Microsoft Hyper-V Virtual Switch deployment, include one of the following XML structures in the body of the POST message.

Example:

The following example is for VMware VDS:

```
<fvTenant name="Tenant_VMM" >
  <fvAp name="Web">
    <fvAEPg name="IntraEPGDeny" pcEnfPref="enforced">
          <!-- pcEnfPref="enforced" ENABLES ISOLATION-->
          <fvRsBd tnFvBDName="bd" />
          <!-- STATIC ENCAP ASSOCIATION TO VMM DOMAIN-->
          <fvRsDomAtt encap="vlan-2001" instrImedcy="lazy" primaryEncap="vlan-2002"
resImedcy="immediate" tDn="uni/vmmp-VMware/dom-DVS1”>
    </fvAEPg>
 </fvAp>
</fvTenant>
```

Example:

The following example is for Microsoft Hyper-V Virtual Switch:

```
<fvTenant name="Tenant_VMM" >
  <fvAp name="Web">
    <fvAEPg name="IntraEPGDeny" pcEnfPref="enforced">
          <!-- pcEnfPref="enforced" ENABLES ISOLATION-->
          <fvRsBd tnFvBDName="bd" />
         <!-- STATIC ENCAP ASSOCIATION TO VMM DOMAIN-->
       <fvRsDomAtt tDn="uni/vmmp-Microsoft/dom-domain1”>
 <fvRsDomAtt encap="vlan-2004" instrImedcy="lazy" primaryEncap="vlan-2003"
 resImedcy="immediate" tDn="uni/vmmp-Microsoft/dom-domain2”>
   </fvAEPg>
 </fvAp>
</fvTenant>
```

Intra-EPG Isolation for AVS

Intra-EPG Isolation Enforcement for Cisco AVS

By default, endpoints with an EPG can communicate with each other without any contracts in place. However, you can isolate endpoints within an EPG from each other. In some instances, you might want to enforce endpoint isolation within an EPG to prevent a VM with a virus or other problem from affecting other VMs in the EPG.

You can configure isolation on all or none of the endpoints within an application EPG; you cannot configure isolation on some endpoints but not on others.

Isolating endpoints within an EPG does not affect any contracts that enable the endpoints to communicate with endpoints in another EPG.

Isolating endpoints within an EPG will trigger a fault when the EPG is associated with Cisco AVS domains in VLAN mode.

Note Using intra-EPG isolation on a Cisco AVS microsegment (uSeg) EPG is not currently supported. Communication is possible between two endpoints that reside in separate uSeg EPGs if either has intra-EPG isolation enforced, regardless of any contract that exists between the two EPGs.

Configuring Intra-EPG Isolation for Cisco AVS Using the GUI

Follow this procedure to create an EPG in which the endpoints of the EPG are isolated from each other.

The port that the EPG uses must belong to one of the VM Managers (VMMs).

Note This procedure assumes that you want to isolate endpoints within an EPG when you create the EPG. If you want to isolate endpoints within an existing EPG, select the EPG in Cisco APIC, and in the **Properties** pane, in the **Intra EPG Isolation** area, choose **Enforced**, and then click **SUBMIT**.

Before you begin

Make sure that Cisco AVS is in VXLAN mode.

Procedure

Step 1 Log in to Cisco APIC.

Step 2 Choose **Tenants**, expand the folder for the tenant, and then expand the **Application Profiles** folder.

Step 3 Right-click an application profile, and choose **Create Application EPG**.

Step 4 In the **Create Application EPG** dialog box, complete the following actions:

a) In the **Name** field, enter the EPG name.
b) In the **Intra EPG Isolation** area, click **Enforced**.
c) From the **Bridge Domain** drop-down list, choose the bridge domain.
d) Check the **Associate to VM Domain Profiles** check box.
e) Click **Next**.
f) In the **Associate VM Domain Profiles** area, click the plus icon, and from the **Domain Profile** drop-down list, choose the desired VMM domain.
g) Click **Update** and click **FINISH**.

What to do next

You can select statistics and view them to help diagnose problems involving the endpoint. See the sections Choosing Statistics to View for Isolated Endpoints on Cisco AVS and Viewing Statistics for Isolated Endpoints on Cisco AVS in this guide.

Configuring Intra-EPG Isolation for Cisco AVS Using the NX-OS Style CLI

Before you begin

Make sure that Cisco AVS is in VXLAN mode.

Procedure

In the CLI, create an intra-EPG isolation EPG:

Example:

```
# Command: show running-config
tenant TENANT1
  application APP1
    epg EPG1
      bridge-domain member VMM_BD
      vmware-domain member VMMDOM1
      isolation enforce <---- This enables EPG into isolation mode.
      exit
   exit
exit
```

What to do next

You can select statistics and view them to help diagnose problems involving the endpoint. See the sections Choosing Statistics to View for Isolated Endpoints on Cisco AVS and Viewing Statistics for Isolated Endpoints on Cisco AVS in this guide.

Configuring Intra-EPG Isolation for Cisco AVS Using the REST API

Before you begin

Make sure that Cisco AVS is in VXLAN mode.

Procedure

Step 1 Send this HTTP POST message to deploy the application using the XML API.

Example:

```
   POST
https://192.0.20.123/api/mo/uni/tn-ExampleCorp.xml
```

Step 2 For a VMM deployment, include the XML structure in the following example in the body of the POST message.

Example:

```
Example:
<fvTenant name="Tenant_VMM" >
  <fvAp name="Web">
    <fvAEPg name="IntraEPGDeny" pcEnfPref="enforced">
      <!-- pcEnfPref="enforced" ENABLES ISOLATION-->
```

```
      <fvRsBd tnFvBDName="bd" />
      <fvRsDomAtt encap="vlan-2001" tDn="uni/vmmp-VMware/dom-DVS1”>
    </fvAEPg>
  </fvAp>
</fvTenant>
```

What to do next

You can select statistics and view them to help diagnose problems involving the endpoint. See the sections Choosing Statistics to View for Isolated Endpoints on Cisco AVS and Viewing Statistics for Isolated Endpoints on Cisco AVS in this guide.

Choosing Statistics to View for Isolated Endpoints on Cisco AVS

If you configured intra-EPG isolation on a Cisco AVS, you need to choose statistics—such as denied connections, received packets, or transmitted multicast packets—for the endpoints before you can view them.

Procedure

Step 1 Log into Cisco APIC.

Step 2 Choose **Tenants** > ***tenant*** .

Step 3 In the tenant navigation pane, choose **Application Profiles** > ***profile*** > **Application EPGs**, and then choose the EPG containing the endpoint the statistics for which you want to view.

Step 4 In the EPG **Properties** work pane, click the **Operational** tab to display the endpoints in the EPG.

Step 5 Double-click the endpoint.

Step 6 In the **Properties** dialog box for the endpoint, click the **Stats** tab and then click the check icon.

Step 7 In the **Select Stats** dialog box, in the **Available** pane, choose the statistics that you want to view for the endpoint and then use the right-pointing arrow to move them into the **Selected** pane.

Step 8 Click **SUBMIT**.

Viewing Statistics for Isolated Endpoints on Cisco AVS

If you configured intra-EPG isolation on a Cisco AVS, once you have chosen statistics for the endpoints, you can view them.

Before you begin

You must have chosen statistics to view for isolated endpoints. See "Choosing Statistics to View for Isolated Endpoints for Cisco AVS" in this guide for instructions.

Procedure

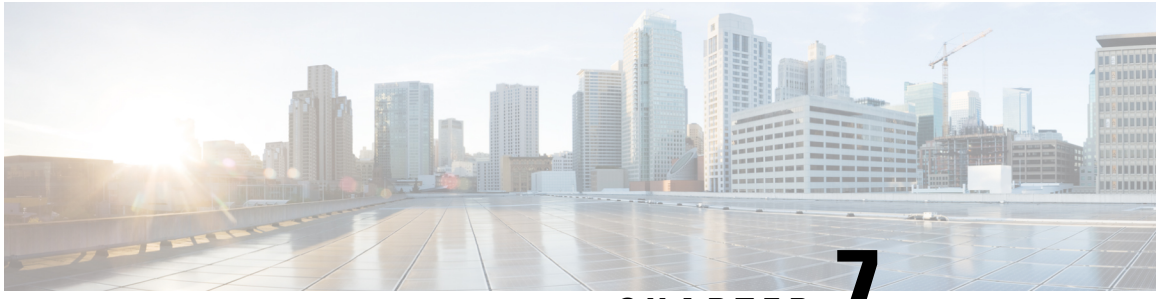
Step 1 Log into Cisco APIC.

Step 2 Choose **Tenants** > ***tenant*** .

Step 3 In the tenant navigation pane, choose **Application Profiles** > ***profile*** > **Application EPGs**, and then choose the EPG containing the endpoint the statistics for which you want to view.

Step 4 In the EPG **Properties** work pane, click the **Stats** tab to display the statistics for the EPG.

The central pane displays the statistics that you chose earlier. You can change the view by clicking the table view or chart view icon on the upper right side of the work pane.



CHAPTER 7

Access Interfaces

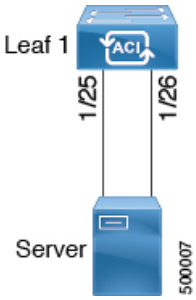
Physical Ports

Configuring Leaf Switch Physical Ports Using Policy Association

The procedure below uses a Quick Start wizard.

Note This procedure provides the steps for attaching a server to an ACI leaf switch interface. The steps would be the same for attaching other kinds of devices to an ACI leaf switch interface.

Figure 16: Switch Interface Configuration for Bare Metal Server



Before you begin

- The ACI fabric is installed, APIC controllers are online, and the APIC cluster is formed and healthy.

- An APIC fabric administrator account is available that will enable creating the necessary fabric infrastructure configurations.
- The target leaf switches are registered in the ACI fabric and available.

Procedure

Step 1 On the APIC menu bar, navigate to **Fabric** > **External Access Policies** > **Quick Start**, and click *Configure an interface, PC, and VPC*.

Step 2 In the **Select Switches To Configure Interfaces** work area, click the large **+** to select switches to configure. In the *Switches* section, click the **+** to add switch ID(s) from the drop-down list of available switch IDs and click **Update**.

Step 3 Click the large **+** to configure switch interfaces.

The interface policy group is a named policy that specifies the group of interface policies you will apply to the selected interfaces of the switch. Examples of interface policies include Link Level Policy (for example, 1gbit port speed), Storm Control Interface Policy, and so forth.

Note The *Attached Device Type* domain is required for enabling an EPG to use the interfaces specified in the switch profile.

a) Specify *individual* as the interface type to use.
b) Specify the interface ID to use.
c) Specify the interface policies to use.
d) Specify the attached device type to use. Choose **`Bare Metal`** for connecting bare metal servers. Bare metal uses the **`phys`** domain type.
e) Click **Save** to update the policy details, then click **Submit** to submit the switch profile to the APIC. The APIC creates the switch profile, along with the interface, selector, and attached device type policies.

Verification: Use the CLI **show int** command on the switch where the server is attached to verify that the switch interface is configured accordingly.

What to do next

This completes the basic leaf interface configuration steps.

Note While this configuration enables hardware connectivity, no data traffic can flow without a valid application profile, EPG, and contract that is associated with this hardware configuration.

Configuring Leaf Switch Physical Ports Using Port Association

This procedure provides the steps for attaching a server to an ACI leaf switch interface. The steps would be the same for attaching other kinds of devices to an ACI leaf switch interface.

Before you begin

- The ACI fabric is installed, APIC controllers are online, and the APIC cluster is formed and healthy.
- An APIC fabric administrator account is available that will enable creating the necessary fabric infrastructure configurations.
- The target leaf switches are registered in the ACI fabric and available.

Procedure

Step 1 On the APIC menu bar, navigate to **Fabric** > **Inventory** > **Inventory**, choose a pod and navigate to the **Configure** tab .

A graphical representation of the switch appears. Choose the port to configure. Once selected, port configure type will appear at the top in the form of a highlighted port configuration type. Choose configuraton type and those configuration parameters will appear.

Step 2 Once you have assigned the appropriate fields to the configuration, click **Submit**.

In this configuration, all changes to the leaf switch are done by selecting the port and applying the policy to it. All leaf switch configuration is done right here on this page.

You have selected the port *then* applied a policy to it.

What to do next

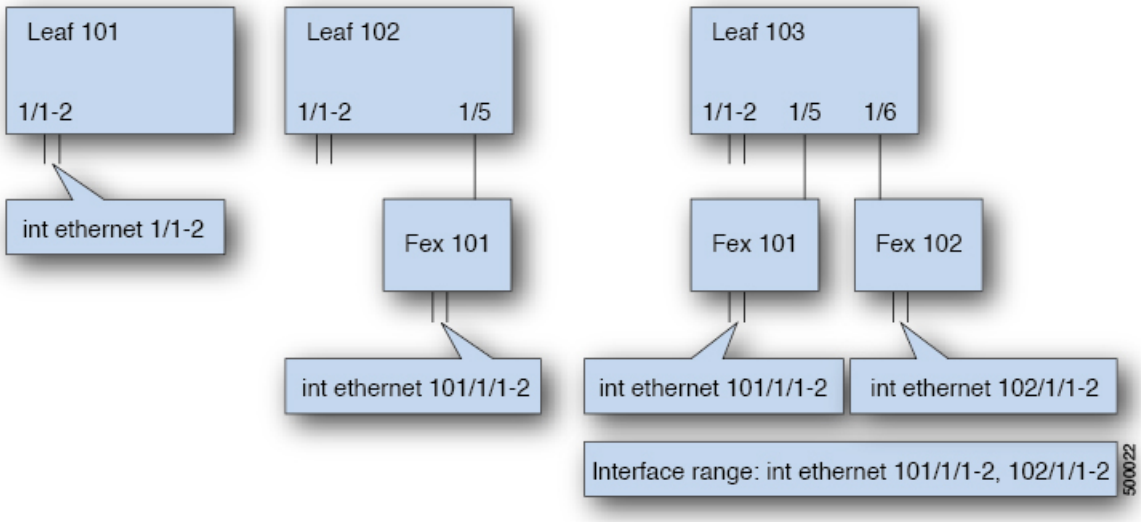
This completes the basic leaf interface configuration steps.

Configuring Physical Ports in Leaf Nodes and FEX Devices Using the NX-OS CLI

The commands in the following examples create many managed objects in the Cisco Application Centric Infrastructure (ACI) policy model that are fully compatible with the REST API/SDK and GUI. However, the CLI user can focus on the intended network configuration instead of Cisco ACI model internals.

Figure 17: Example of leaf node ports and FEX ports in Cisco ACI, on page 76 shows examples of Ethernet ports directly on leaf nodes or FEX modules attached to leaf nodes and how each is represented in the CLI. For FEX ports, the *fex-id* is included in the naming of the port itself as in **ethernet 101/1/1**. While describing an interface range, the **ethernet** keyword need not be repeated as in NX-OS. Example: **interface ethernet 101/1/1-2, 102/1/1-2**.

Figure 17: Example of leaf node ports and FEX ports in Cisco ACI



- Leaf node ID numbers are global.
- The *fex-id* numbers are local to each leaf node.
- Note the space after the keyword **ethernet**.

Procedure

Step 1 **configure**

Enters global configuration mode.

Example:

```
apic1# configure
```

Step 2 **leaf** *node-id*

Specifies the leaf nodes to be configured. The *node-id* can be a single node ID or a range of IDs, in the form *node-id1–node-id2*, to which the configuration will be applied.

Example:

```
apic1(config)# leaf 102
```

Step 3 **interface** *type*

Specifies the interface that you are configuring. You can specify the interface type and identity. For an Ethernet port, use “ethernet slot / port.”

Example:

```
apic1(config-leaf)# interface ethernet 1/2
```

Step 4 (Optional) **fex associate** *node-id*

If the interface or interfaces to be configured are FEX interfaces, you must use this command to attach the FEX module to a leaf node before configuration.

Note This step is required before creating a port channel using FEX ports.

Example:

```
apic1(config-leaf-if)# fex associate 101
```

Step 5 **speed** *speed*

The speed setting is shown as an example. At this point you can configure any of the interface settings shown in the table below.

Example:

```
apic1(config-leaf-if)# speed 10G
```

The following table shows the interface settings that can be configured at this point:

Command	Purpose
[no] shut	Shut down physical interface
[no] speed *speedValue*	Set the speed for physical interface
[no] link debounce time *time*	Set link debounce
[no] negotiate auto	Configure negotiate
[no] cdp enable	Disable/enable Cisco Discovery Protocol (CDP)
[no] mcp enable	Disable/enable Mis-cabling Protocol (MCP)
[no] lldp transmit	Set the transmit for physical interface
[no] lldp receive	Set the LLDP receive for physical interface
spanning-tree {bpduguard | bpdufilter} {enable | disable}	Configure spanning tree BPDU
[no] storm-control level *percentage* [burst-rate *percentage*]	Storm-control configuration (percentage)
[no] storm-control pps *packets-per-second* burst-rate *packets-per-second*	Storm-control configuration (packets-per-second)

Examples

Configure one port in a leaf node. The following example shows how to configure the interface eth1/2 in leaf 101 for the following properties: speed, cdp, and admin state.

```
apic1# configure
apic1(config)# leaf 101
apic1(config-leaf)# interface ethernet 1/2
apic1(config-leaf-if)# speed 10G
```

```
apic1(config-leaf-if)# cdp enable
apic1(config-leaf-if)# no shut
```

Configure multiple ports in multiple leaf nodes. The following example shows the configuration of speed for interfaces eth1/1-10 for each of the leaf nodes 101-103.

```
apic1(config)# leaf 101-103
apic1(config-leaf)# interface eth 1/1-10
apic1(config-leaf-if)# speed 10G
```

Attach a FEX to a leaf node. The following example shows how to attach a FEX module to a leaf node. Unlike in NX-OS, the leaf node port Eth1/5 is implicitly configured as fabric port and a FEX fabric port channel is created internally with the FEX uplink port(s). In Cisco ACI, the FEX fabric port channels use default configuration and no user configuration is allowed.

Note This step is required before creating a port channel using FEX ports, as described in the next example.

```
apic1(config)# leaf 102
apic1(config-leaf)# interface eth 1/5
apic1(config-leaf-if)# fex associate 101
```

Configure FEX ports attached to leaf nodes. This example shows configuration of speed for interfaces eth1/1-10 in FEX module 101 attached to each of the leaf nodes 102-103. The FEX ID 101 is included in the port identifier. FEX IDs start with 101 and are local to a leaf node.

```
apic1(config)# leaf 102-103
apic1(config-leaf)# interface eth 101/1/1-10
apic1(config-leaf-if)# speed 1G
```

Port Cloning

Cloning Port Configurations

In the Cisco APIC Release 3.2 and later, the Port Cloning feature is supported. After you configure a leaf switch port, you can copy the configuration and apply it to other ports. This is only supported in the APIC GUI (not in the NX-OS style CLI).

Port cloning is used for small numbers of leaf switch ports (interfaces) that are individually configured, not for interfaces configured using Fabric Access Policies, that you deploy on multiple nodes in the fabric.

Port cloning is only supported for Layer 2 configurations.

The following policies are not supported on a cloned port:

- Attachable Access Entity
- Storm Control
- DWDM
- MACsec

Cloning a Configured Leaf Switch Port Using the APIC GUI

This task describes how to clone a leaf switch port that you previously configured. For more information about configuring ports, see *Cisco APIC Layer 2 Networking Configuration Guide*.

Before you begin

Configure a leaf switch port (with supported Layer 2 policies) in the GUI under **Fabric** > **Inventory**, and one of the following:

- **Topology** > **Interface** > **Configuration Mode**
- **Pod** > **Interface** > **Configuration Mode**
- **Pod**> **Leaf** > **Interface** > **Configuration Mode**

Procedure

Step 1 On the Menu bar, choose **Fabric** > **Inventory**.

Step 2 Navigate to the location where you configured the first port.

Step 3 For example, expand **Pod** and choose **Leaf**.

Step 4 Click **Interface** and choose **Configuration** from the drop-down list under **Mode**.

Step 5 Click the + icon on the interface menu bar to choose the leaf switch where the port to clone is located.

Step 6 Right-click the port you previously configured and choose **Copy**.

Step 7 Right-click the port on which you want to copy the configuration and choose **Paste**.

Port Channels

PC Host Load Balancing Algorithms

The following table provides the configurable port channel hash algorithms used in PC host load balancing across Cisco Application Centric Infrastructure (ACI) leaf node downlinks. This feature was introduced in Cisco Application Policy Infrastructure Controller (APIC), release 2.3(1e).

Table 14: PC Host Load Balancing Algorithms

Traffic Type	Hashing Data Points
End Host PC traffic (configurable)	**For layer 2 frames**: Source MAC address and Destination MAC address **For IP Traffic**: • Source IP address • Destination IP address • Layer 4 Source Port • Layer 4 Destination Port

When there is more than one port channel on a leaf switch, such as Po1 and Po2, then the following scenario is supported:

- Po1: Enable symmetric hash with SIP only.
- Po2: Do not enable symmetric hash. Use default hashing.

However, the following scenario is not supported because the second port channel Po2 has a different hash parameter:

- Po1: Enable symmetric hash with SIP only.
- Po2: Enable symmetric hash with DIP only.

That is, on a single leaf switch, all port channels that require symmetric hashing should use the same hash policy/parameter or use the default hashing.

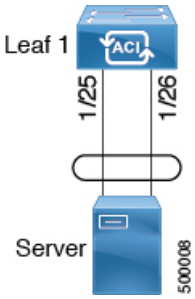
Note Port channel hash algorithms are applied at each individual leaf node independently. The algorithms do not have influence on load balancing within the fabric, such as load balancing to leaf nodes in a vPC pair. Thus, symmetrical hashing is not supported on a vPC.

ACI Leaf Switch Port Channel Configuration Using the GUI

The procedure below uses a Quick Start wizard.

Note This procedure provides the steps for attaching a server to an ACI leaf switch interface. The steps would be the same for attaching other kinds of devices to an ACI leaf switch interface.

Figure 18: Switch Port Channel Configuration



Before you begin

- The ACI fabric is installed, APIC controllers are online, and the APIC cluster is formed and healthy.
- An APIC fabric administrator account is available that will enable creating the necessary fabric infrastructure configurations.
- The target leaf switches are registered in the ACI fabric and available.

Procedure

Step 1 On the APIC menu bar, navigate to **Fabric** > **External Access Policies** > **Quick Start**, and click *Configure an interface, PC, and vPC*.

Step 2 In the **Select Switches To Configure Interfaces** work area, click the large + to select switches to configure. In the *Switches* section, click the + to add switch ID(s) from the drop-down list of available switch IDs and click **Update**.

Step 3 Click the large + to configure switch interfaces.

The interface policy group is a named policy that specifies the group of interface policies you will apply to the selected interfaces of the switch. Examples of interface policies include Link Level Policy (for example, 1gbit port speed), Storm Control Interface Policy, and so forth.

Note The *Attached Device Type* is required for enabling an EPG to use the interfaces specified in the switch profile.

a) Specify *pc* as the interface type to use.
b) Specify the interface IDs to use.
c) Specify the interface policies to use. For example, click the **Port Channel Policy** drop-down arrow to choose an existing port channel policy or to create a new port channel policy.

Note

- Choosing to create a port channel policy displays the **Create Port Channel Policy** dialog box where you can specify the policy details and enable features such as symmetric hashing. Also note that choosing the **Symmetric hashing** option displays the **Load Balance Hashing** field, which enables you to configure hash tuple. However, only one customized hashing option can be applied on the same leaf switch.
- Symmetric hashing is not supported on the following switches:
  - Cisco Nexus 93128TX
  - Cisco Nexus 9372PX
  - Cisco Nexus 9372PX-E
  - Cisco Nexus 9372TX
  - Cisco Nexus 9372TX-E
  - Cisco Nexus 9396PX
  - Cisco Nexus 9396TX

d) Specify the attached device type to use. Choose **`Bare Metal`** for connecting bare metal servers. Bare metal uses the **`phys`** domain type.

e) Click **Save** to update the policy details, then click **Submit** to submit the switch profile to the APIC. The APIC creates the switch profile, along with the interface, selector, and attached device type policies.

Verification: Use the CLI **show int** command on the switch where the server is attached to verify that the switch interface is configured accordingly.

What to do next

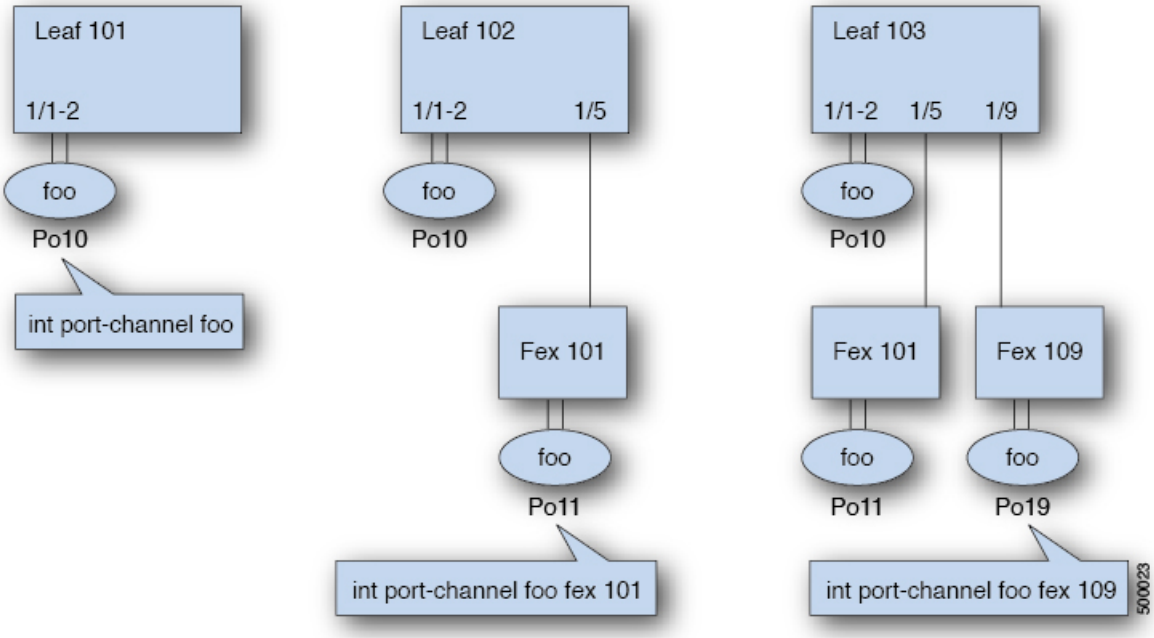
This completes the port channel configuration steps.

Note While this configuration enables hardware connectivity, no data traffic can flow without a valid application profile, EPG, and contract that is associated with this hardware configuration.

Configuring Port Channels in Leaf Nodes and FEX Devices Using the NX-OS CLI

Port-channels are logical interfaces in NX-OS used to aggregate bandwidth for multiple physical ports and also for providing redundancy in case of link failures. In NX-OS, port-channel interfaces are identified by user-specified numbers in the range 1 to 4096 unique within a node. Port-channel interfaces are either configured explicitly (using the **interface port-channel** command) or created implicitly (using the **channel-group** command). The configuration of the port-channel interface is applied to all the member ports of the port-channel. There are certain compatibility parameters (speed, for example) that cannot be configured on the member ports.

In the ACI model, port-channels are configured as logical entities identified by a name to represent a collection of policies that can be assigned to set of ports in one or more leaf nodes. Such assignment creates one port-channel interface in each of the leaf nodes identified by an auto-generated number in the range 1 to 4096 within the leaf node, which may be same or different among the nodes for the same port-channel name. The membership of these port-channels may be same or different as well. When a port-channel is created on the FEX ports, the same port-channel name can be used to create one port-channel interface in each of the FEX devices attached to the leaf node. Thus, it is possible to create up to N+1 unique port-channel interfaces (identified by the auto-generated port-channel numbers) for each leaf node attached to N FEX modules. This is illustrated with the examples below. Port-channels on the FEX ports are identified by specifying the *fex-id* along with the port-channel name (**interface port-channel foo fex 101** , for example).



- N+1 instances per leaf of port-channel foo are possible when each leaf is connected to N FEX nodes.
- Leaf ports and FEX ports cannot be part of the same port-channel instance.
- Each FEX node can have only one instance of port-channel foo.

Procedure

	Command or Action	Purpose
Step 1	**configure** **Example:** `apic1# configure`	Enters global configuration mode.
Step 2	**template port-channel** *channel-name* **Example:** `apic1(config)# template port-channel foo`	Creates a new port-channel or configures an existing port-channel (global configuration).

	Command or Action	Purpose
Step 3	[**no**] **switchport access vlan** *vlan-id* **tenant** *tenant-name* **application** *application-name* **epg** *epg-name* **Example:** `apic1(config-po-ch-if)# switchport access vlan 4 tenant ExampleCorp application Web epg webEpg`	Deploys the EPG with the VLAN on all ports with which the port-channel is associated.

Step 4	**channel-mode active**
	Example:
	`apic1(config-po-ch-if)# channel-mode active`
	Note To enable symmetric hashing, enter the **lacp symmetric-hash** command:
	`apic1(config-po-ch-if)# lacp symmetric-hash`
		Note The **channel-mode** command is equivalent to the mode option in the channel-group command in NX-OS. In ACI, however, this is supported for the port-channel (not on a member port).
		Symmetric hashing is not supported on the following switches:
		• Cisco Nexus 93128TX
		• Cisco Nexus 9372PX
		• Cisco Nexus 9372PX-E
		• Cisco Nexus 9372TX
		• Cisco Nexus 9372TX-E
		• Cisco Nexus 9396PX
		• Cisco Nexus 9396TX

Step 5	**exit** **Example:** `apic1(config-po-ch-if)# exit`	Returns to configure mode.
Step 6	**leaf** *node-id* **Example:** `apic1(config)# leaf 101`	Specifies the leaf switches to be configured. The *node-id* can be a single node ID or a range of IDs, in the form *node-id1* – *node-id2* , to which the configuration will be applied.
Step 7	**interface** *type* **Example:** `apic1(config-leaf)# interface ethernet 1/1-2`	Specifies the interface or range of interfaces that you are configuring to the port-channel.
Step 8	[**no**] **channel-group** *channel-name* **Example:** `apic1(config-leaf-if)# channel-group foo`	Assigns the interface or range of interfaces to the port-channel. Use the keyword **no** to remove the interface from the port-channel. To change the port-channel assignment on an

	Command or Action	Purpose
		interface, you can enter the **channel-group** command without first removing the interface from the previous port-channel.

Step 9	(Optional) **lacp port-priority** *priority*
	Example:
	`apic1(config-leaf-if)# lacp port-priority 1000`
	`apic1(config-leaf-if)# lacp rate fast`
		This setting and other per-port LACP properties can be applied to member ports of a port-channel at this point.
		Note In the ACI model, these commands are allowed only after the ports are member of a port channel. If a port is removed from a port channel, configuration of these per-port properties are removed as well.

The following table shows various commands for global configurations of port channel properties in the ACI model. These commands can also be used for configuring overrides for port channels in a specific leaf in the (config-leaf-if) CLI mode. The configuration made on the port-channel is applied to all member ports.

CLI Syntax	Feature
[no] speed <speedValue>	Set the speed for port-channel
[no] link debounce time <time>	Set Link Debounce for port-channel
[no] negotiate auto	Configure Negotiate for port-channel
[no] cdp enable	Disable/Enable CDP for port-channel
[no] mcp enable	Disable/Enable MCP for port-channel
[no] lldp transmit	Set the transmit for port-channel
[no] lldp receive	Set the lldp receive for port-channel
spanning-tree <bpduguard | bpdufilter> <enable | disable>	Configure spanning tree BPDU
[no] storm-control level <percentage> [burst-rate <percentage>]	Storm-control configuration (percentage)
[no] storm-control pps <packet-per-second> burst-rate <packets-per-second>	Storm-control configuration (packets-per-second)
[no] channel-mode { active | passive | on| mac-pinning }	LACP mode for the link in port-channel l
[no] lacp min-links <value>	Set minimum number of links
[no] lacp max-links <value>	Set maximum number of links
[no] lacp fast-select-hot-standby	LACP fast select for hot standby ports

CLI Syntax	Feature
[no] lacp graceful-convergence	LACP graceful convergence
[no] lacp load-defer	LACP load defer member ports
[no] lacp suspend-individual	LACP individual Port suspension
[no] lacp port-priority	LACP port priority
[no] lacp rate	LACP rate

Examples

Configure a port channel (global configuration). A logical entity foo is created that represents a collection of policies with two configurations: speed and channel mode. More properties can be configured as required.

Note The channel mode command is equivalent to the mode option in the channel group command in NX-OS. In ACI, however, this supported for the port-channel (not on member port).

```
apic1(config)# template port-channel foo
apic1(config-po-ch-if)# switchport access vlan 4 tenant ExampleCorp application Web epg
webEpg
apic1(config-po-ch-if)# speed 10G
apic1(config-po-ch-if)# channel-mode active
```

Configure ports to a port-channel in a FEX. In this example, port channel foo is assigned to ports Ethernet 1/1-2 in FEX 101 attached to leaf node 102 to create an instance of port channel foo. The leaf node will auto-generate a number, say 1002 to identify the port channel in the switch. This port channel number would be unique to the leaf node 102 regardless of how many instance of port channel foo are created.

Note The configuration to attach the FEX module to the leaf node must be done before creating port channels using FEX ports.

```
apic1(config)# leaf 102
apic1(config-leaf)# interface ethernet 101/1/1-2
apic1(config-leaf-if)# channel-group foo
```

In Leaf 102, this port channel interface can be referred to as interface port-channel foo FEX 101.

```
apic1(config)# leaf 102
apic1(config-leaf)# interface port-channel foo fex 101
apic1(config-leaf)# shut
```

Configure ports to a port channel in multiple leaf nodes. In this example, port channel foo is assigned to ports Ethernet 1/1-2 in each of the leaf nodes 101-103. The leaf nodes will auto generate a number

unique in each node (which may be same or different among nodes) to represent the port-channel interfaces.

```
apic1(config)# leaf 101-103
apic1(config-leaf)# interface ethernet 1/1-2
apic1(config-leaf-if)# channel-group foo
```

Add members to port channels. This example would add two members eth1/3-4 to the port-channel in each leaf node, so that port-channel foo in each node would have members eth 1/1-4.

```
apic1(config)# leaf 101-103
apic1(config-leaf)# interface ethernet 1/3-4
apic1(config-leaf-if)# channel-group foo
```

Remove members from port channels. This example would remove two members eth1/2, eth1/4 from the port channel foo in each leaf node, so that port channel foo in each node would have members eth 1/1, eth1/3.

```
 apic1(config)# leaf 101-103
apic1(config-leaf)# interface eth 1/2,1/4
apic1(config-leaf-if)# no channel-group foo
```

Configure port-channel with different members in multiple leaf nodes. This example shows how to use the same port-channel foo policies to create a port-channel interface in multiple leaf nodes with different member ports in each leaf. The port-channel numbers in the leaf nodes may be same or different for the same port-channel foo. In the CLI, however, the configuration will be referred as interface port-channel foo. If the port-channel is configured for the FEX ports, it would be referred to as interface port-channel foo fex <fex-id>.

```
apic1(config)# leaf 101
apic1(config-leaf)# interface ethernet 1/1-2
apic1(config-leaf-if)# channel-group foo
apic1(config-leaf-if)# exit
apic1(config-leaf)# exit
apic1(config)# leaf 102
apic1(config-leaf)# interface ethernet 1/3-4
apic1(config-leaf-if)# channel-group foo
apic1(config-leaf-if)# exit
apic1(config-leaf)# exit
apic1(config)# leaf 103
apic1(config-leaf)# interface ethernet 1/5-8
apic1(config-leaf-if)# channel-group foo
apic1(config-leaf-if)# exit
apic1(config-leaf)# interface ethernet 101/1/1-2
apic1(config-leaf-if)# channel-group foo
```

Configure per port properties for LACP. This example shows how to configure member ports of a port-channel for per-port properties for LACP.

Note In ACI model, these commands are allowed only after the ports are member of a port channel. If a port is removed from a port channel, configuration of these per-port properties would be removed as well.

```
apic1(config)# leaf 101
apic1(config-leaf)# interface ethernet 1/1-2
```

```
apic1(config-leaf-if)# channel-group foo
apic1(config-leaf-if)# lacp port-priority 1000
apic1(config-leaf-if)# lacp rate fast
```

Configure admin state for port channels. In this example, a port-channel foo is configured in each of the leaf nodes 101-103 using the channel-group command. The admin state of port-channel(s) can be configured in each leaf using the port-channel interface. In ACI model, the admin state of the port-channel cannot be configured in the global scope.

```
// create port-channel foo in each leaf
apic1(config)# leaf 101-103
apic1(config-leaf)# interface ethernet 1/3-4
apic1(config-leaf-if)# channel-group foo

// configure admin state in specific leaf
apic1(config)# leaf 101
apic1(config-leaf)# interface port-channel foo
apic1(config-leaf-if)# shut
```

Override config is very helpful to assign specific vlan-domain, for example, to the port-channel interfaces in each leaf while sharing other properties.

```
// configure a port channel global config
apic1(config)# interface port-channel foo
apic1(config-if)# speed 1G
apic1(config-if)# channel-mode active

// create port-channel foo in each leaf
apic1(config)# leaf 101-103
apic1(config-leaf)# interface ethernet 1/1-2
apic1(config-leaf-if)# channel-group foo

// override port-channel foo in leaf 102
apic1(config)# leaf 102
apic1(config-leaf)# interface port-channel foo
apic1(config-leaf-if)#  speed 10G
apic1(config-leaf-if)#  channel-mode on
apic1(config-leaf-if)#  vlan-domain dom-foo
```

This example shows how to change port channel assignment for ports using the channel-group command. There is no need to remove port channel membership before assigning to other port channel.

```
apic1(config)# leaf 101-103
apic1(config-leaf)# interface ethernet 1/3-4
apic1(config-leaf-if)# channel-group foo
apic1(config-leaf-if)# channel-group bar
```

Configuring Two Port Channels Applied to Multiple Switches Using the REST API

This example creates two port channels (PCs) on leaf switch 17, another port channel on leaf switch 18, and a third one on leaf switch 20. On each leaf switch, the same interfaces will be part of the PC (interface 1/10 to 1/15 for port channel 1 and 1/20 to 1/25 for port channel 2). The policy uses two switch blocks because

each a switch block can contain only one group of consecutive switch IDs. All these PCs will have the same configuration.

Note Even though the PC configurations are the same, this example uses two different interface policy groups. Each Interface Policy Group represents a PC on a switch. All interfaces associated with a given interface policy group are part of the same PCs.

Before you begin

- The ACI fabric is installed, APIC controllers are online, and the APIC cluster is formed and healthy.
- An APIC fabric administrator account is available that will enable creating the necessary fabric infrastructure configurations.
- The target leaf switch and protocol(s) are configured and available.

Procedure

To create the two PCs, send a post with XML such as the following:

Example:

```
<infraInfra dn="uni/infra">

       <infraNodeP name=”test">
             <infraLeafS name="leafs" type="range">
                  <infraNodeBlk name="nblk”
                       from_=”17" to_=”18”/>
                  <infraNodeBlk name="nblk”
                       from_=”20" to_=”20”/>
             </infraLeafS>
             <infraRsAccPortP tDn="uni/infra/accportprof-test1"/>
             <infraRsAccPortP tDn="uni/infra/accportprof-test2"/>
       </infraNodeP>

       <infraAccPortP name="test1">
             <infraHPortS name="pselc" type="range">
                  <infraPortBlk name="blk1”
                       fromCard="1" toCard="1"
                       fromPort="10" toPort=”15”/>
             <infraRsAccBaseGrp
                       tDn="uni/infra/funcprof/accbundle-bndlgrp1"/>
              </infraHPortS>
       </infraAccPortP>

       <infraAccPortP name="test2">
             <infraHPortS name="pselc" type="range">
                  <infraPortBlk name="blk1”
                       fromCard="1" toCard="1"
                       fromPort=“20" toPort=”25”/>
                  <infraRsAccBaseGrp
                       tDn="uni/infra/funcprof/accbundle-bndlgrp2" />
             </infraHPortS>
       </infraAccPortP>

       <infraFuncP>
```

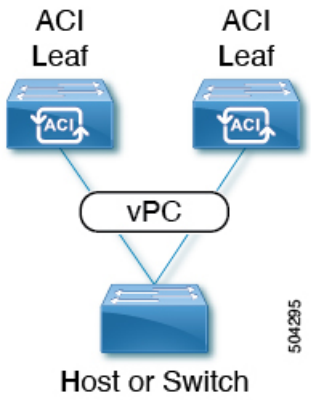
```
            <infraAccBndlGrp name="bndlgrp1" lagT="link">
                  <infraRsHIfPol  tnFabricHIfPolName="default"/>
                  <infraRsCdpIfPol tnCdpIfPolName="default"/>
                  <infraRsLacpPol tnLacpLagPolName="default"/>
            </infraAccBndlGrp>

            <infraAccBndlGrp name="bndlgrp2" lagT="link">
                  <infraRsHIfPol  tnFabricHIfPolName="default"/>
                  <infraRsCdpIfPol tnCdpIfPolName="default"/>
                  <infraRsLacpPol tnLacpLagPolName="default"/>
            </infraAccBndlGrp>
      </infraFuncP>

</infraInfra>
```

Virtual Port Channels

About Virtual Port Channels in Cisco ACI

A virtual port channel (vPC) allows links that are physically connected to two different Cisco Application Centric Infrastructure (ACI) leaf nodes to appear as a single port channel (PC) to a third device, such as a network switch, server, any other networking device that supports link aggregation technology. vPCs consist of two Cisco ACI leaf switches designated as vPC peer switches. Of the vPC peers, one is primary and one is secondary. The system formed by the switches is referred to as a vPC domain.

Figure 19: vPC Domain



The following behavior is specific to the Cisco ACI vPC implementation:

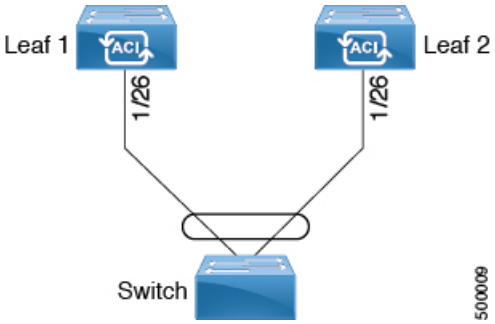
- No dedicated peer-link between the vPC peers. Instead, the fabric itself serves as the Multi-Chassis Trunking (MCT).
- Peer reachability protocol: Cisco ACI uses the Zero Message Queue (ZMQ) instead of Cisco Fabric Services (CFS).
    - ZMQ is an open-source, high-performance messaging library that uses TCP as the transport.
    - This library is packaged as libzmq on the switch and linked into each application that needs to communicate with a vPC peer.

- Peer reachability is not handled using a physical peer link. Instead, routing triggers are used to detect peer reachability.
  - The vPC manager registers with Unicast Routing Information Base (URIB) for peer route notifications.
  - When IS-IS discovers a route to the peer, URIB notifies the vPC manager, which in turn attempts to open a ZMQ socket with the peer.
  - When the peer route is withdrawn by IS-IS, the vPC manager is again notified by URIB, and the vPC manager brings down the MCT link.

ACI Virtual Port Channel Workflow

This workflow provides an overview of the steps required to configure a virtual port channel (vPC).

Figure 20: Virtual port channel configuration



1. Prerequisites

- Ensure that you have read/write access privileges to the infra security domain.
- Ensure that the target leaf switches with the necessary interfaces are available.

Note When creating a vPC domain between two Leaf nodes, please considering the hardware model limitations:

- Generation 1 switches are compatible only with other Generation 1 Switches. These switch models can be identified by the lack of “EX”, or “FX” at the end of the switch name. For example N9K-9312TX.

  Generation 2 and later switches can be mixed together in a vPC domain. These switch models can be identified with “EX”, “FX” or “FX2” at the end of the switch name. For example N9K-93108TC-EX, or N9K-9348GC-FXP.

Example:

Compatible vPC Switch Pairs:

- N9K-9312TX & N9K-9312TX
- N9K-93108TC-EX & N9K-9348GC-FXP
- Nexus 93180TC-FX & Nexus 93180YC-FX
- Nexus 93180YC-FX & Nexus 93180YC-FX

Incompatible vPC Switch Pairs:

- N9K-9312TX & N9K-93108TC-EX
- N9K-9312TX & Nexus 93180YC-FX

2. Configure the Virtual Port Channel

1. On the APIC menu bar, navigate to **Fabric** > **External Access Policies** > **Quick Start**, and click **Configure an interface, PC, and vPC** to open the quick start wizard.
2. Provide the specifications for the policy name, switch IDs and the interfaces the virtual port channel will use. Add the Interface Policy parameters, such as group port speed, storm control, CDP, LLDP. Add the Attached Device Type as an **External Bridged Device** and specify the VLAN and domain that will be used.
3. Use the CLI **show int** command on the ACI leaf switches where the external switch is attached to verify that the switches and virtual port channel are configured accordingly.

Note While this configuration enables hardware connectivity, no data traffic can flow without a valid application profile, EPG, and contract that is associated with this hardware configuration.

Configure the Application Profile

1. On the APIC menu bar, navigate to **Tenant** > ***tenant-name*** > **Quick Start**, and click **Create an application profile** under the tenant quick start wizard.
2. Configure the endpoint groups (EPGs), contracts, bridge domain, subnet, and context.
3. Associate the application profile EPGs with the virtual port channel switch profile created above.

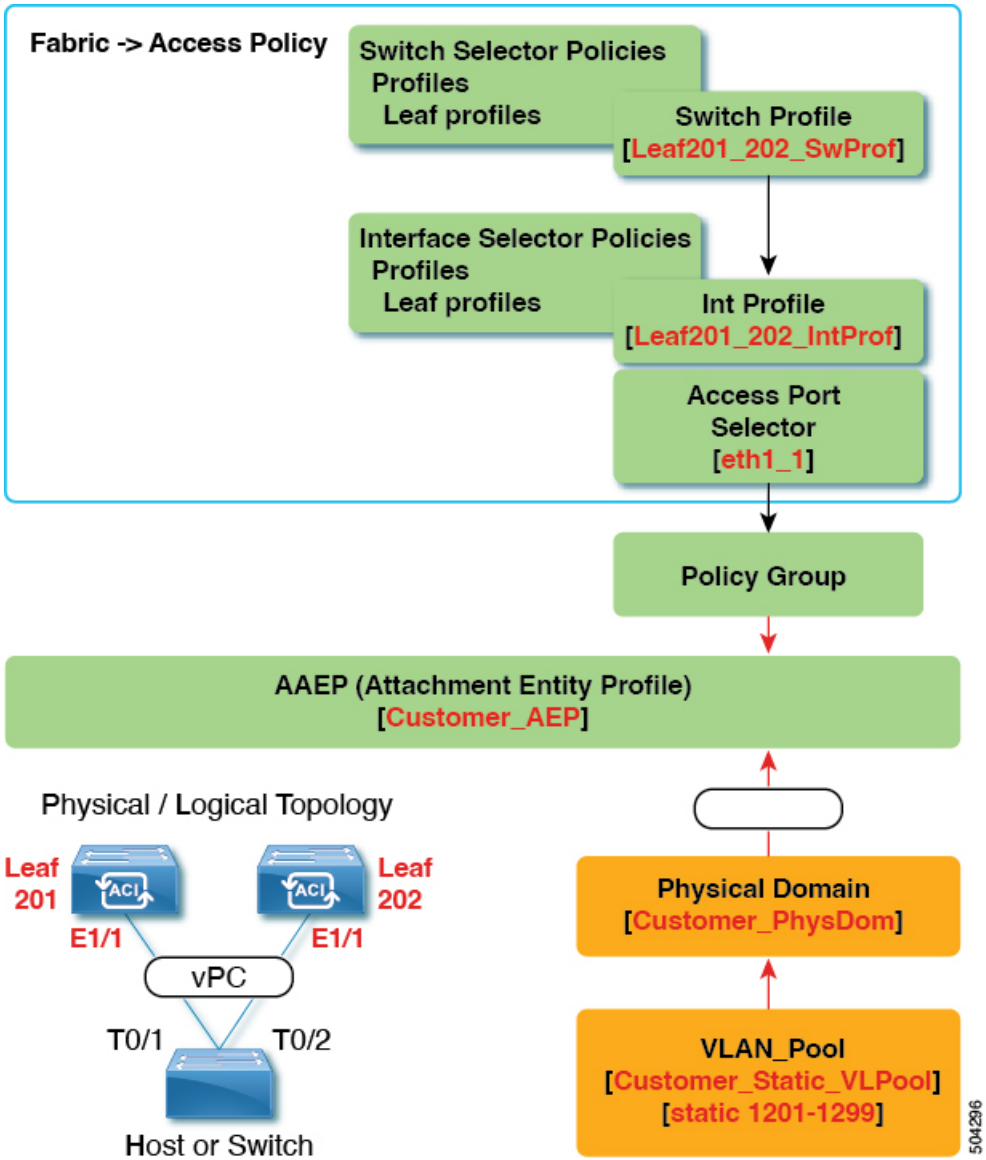
Virtual Port Channel Use Cases

vPC With the Same Leaf Switch Interfaces Across Two Leaf Switches With Combined Profiles

For the example of this use case, you define the following things:

- A combined switch profile called `Leaf201_202_SwProf` (node 201 and node 202).
- A combined interface profile called `Leaf201_202_IntProf` (node 201 and node 202).
- An access port selector called `Eth1_1` (under the `Leaf201_202` interface profile) is pointing toward a vPC interface policy group.
- The vPC interface policy group is pointing toward an AAEP called `Customer_AEP`.
- The AEP (`Customer_AEP`) has an association with the `Customer_PhysDom`.
- The `Customer_PhysDom` has an association with a VLAN pool called `Customer_Static_VLPool`.

Figure 21: vPC With the Same Leaf Switch Interfaces Across Two Leaf Switches With Combined Profiles



What This Configuration Does

On switches `Leaf201` and `Leaf202`, configure port Eth1/1 to be part of a vPC. This vPC interface will have access to VLANs 1201 through 1299. Depending on the interface policy group, you can enable LACP Active and other interface specific policy configurations.

When to Use This Configuration

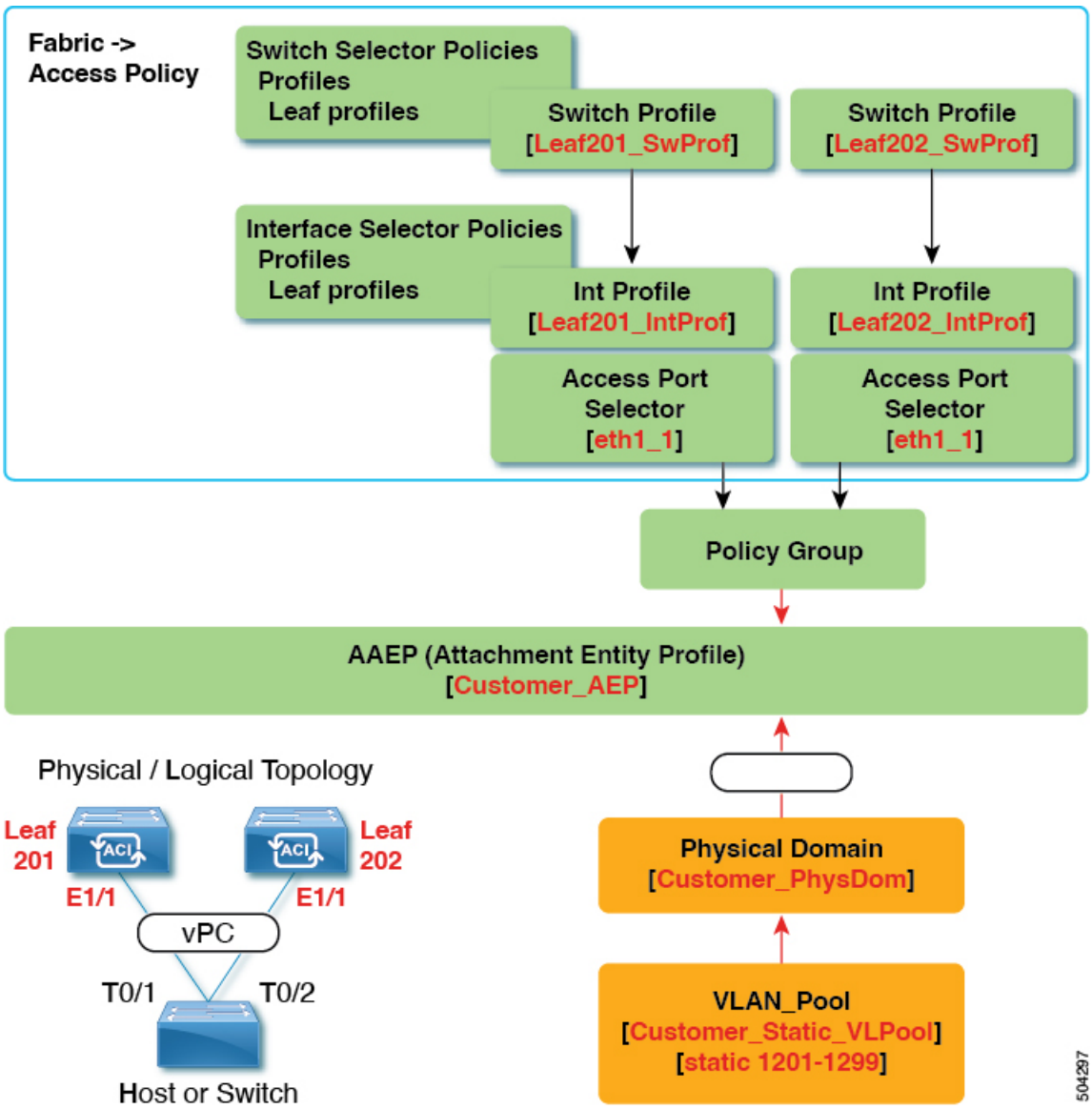
If you have dedicated pairs of compute leaf switches with nothing but vPC-connected servers, for example, this would be a solid use case for using combined-switch/interface profiles under your fabric access policies for those switches. You could preconfigure your switch, interface, access port selector, and vPC interface policy groups in such a way that allowed you to plug in 48 chassis-type servers with minimal effort.

vPC With the Same Leaf Switch Interfaces Across Two Leaf Switches with Individual Profiles

For the example of this use case, you define the following things:

- Individual switch profiles called `Leaf201_SwProf` and `Leaf202_SwProf` (node 201 and node 202).
- Individual interface profiles called `Leaf201_IntProf` and `Leaf202_IntProf` (node 201 and node 202)
- Access port selectors called `Eth1_1` (under the `Leaf201` and `Leaf202` interface profiles) is pointing toward the same vPC interface policy group.
- The vPC interface policy group is pointing toward an AAEP called `Customer_AEP`.
- The AEP (`Customer_AEP`) has an association with the `Customer_PhysDom`.
- The `Customer_PhysDom` has an association with a VLAN pool called `Customer_Static_VLPool`.

Figure 22: vPC With the Same Leaf Switch Interfaces Across Two Leaf Switches with Individual Profiles



What This Configuration Does

On switches `Leaf201` and `Leaf202`, configure port Eth1/1 to be a part of a vPC. This vPC interface will have access to VLANs 1201 through 1299. Depending on the interface policy group, you can enable LACP active and other interface specific policy configurations.

When to Use This Configuration

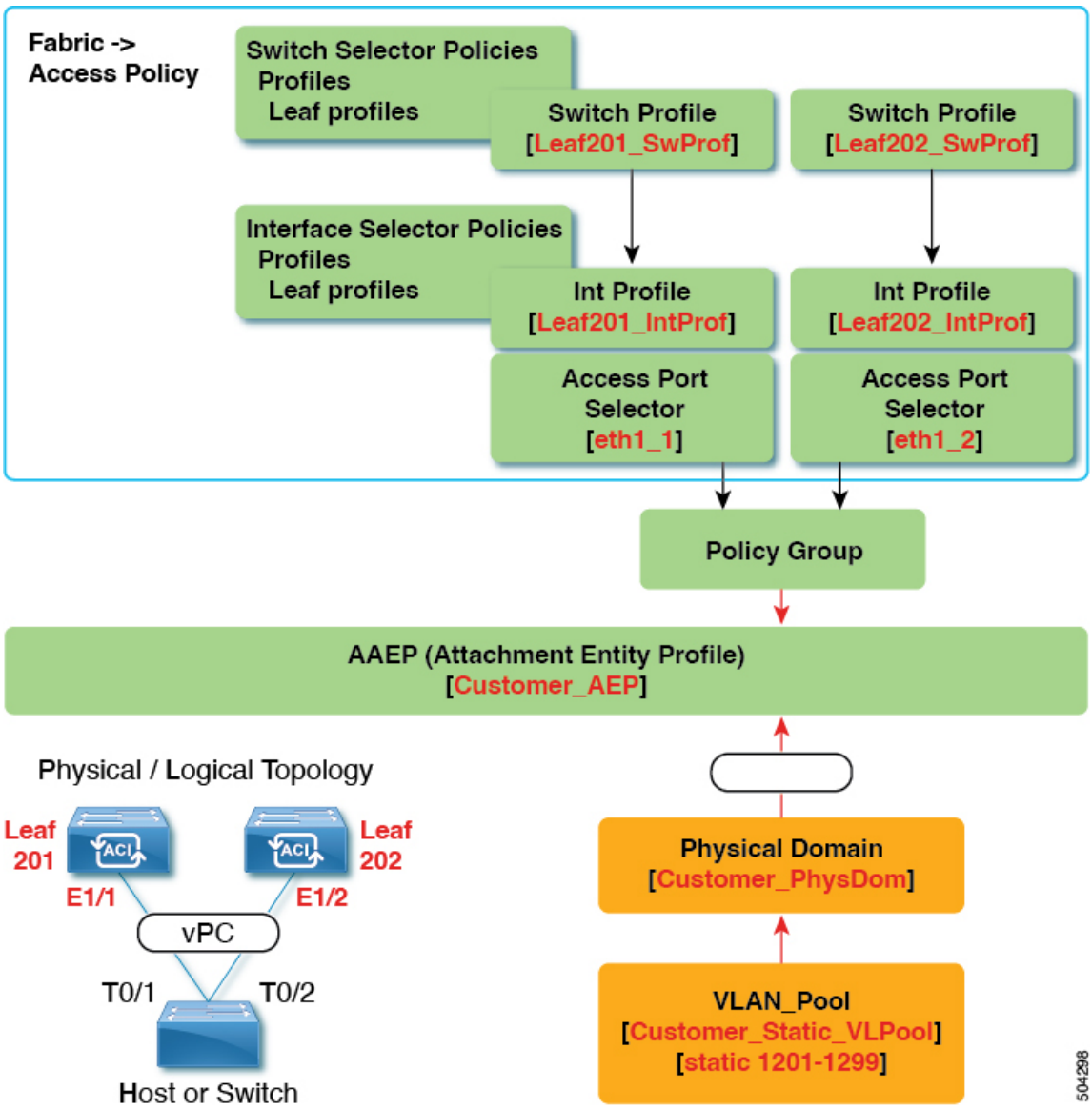
Use this configuration when you have leaf switches that support mixed workloads, such as compute, services, or Cisco Application Policy Infrastructure Controllers (APICs). In this case, having individual interface profiles allows for the most amount of flexibility, while allowing you to keep your **Fabric** > **Access Policies** configuration as clean and manageable as possible.

vPC With Different Leaf Switch Interfaces Across Two Leaf Switches With Individual Profiles

For the example of this use case, you define the following things:

- Individual switch profiles called `Leaf201_SwProf` and `Leaf202_SwProf` (node 201 and node 202).
- Individual interface profiles called `Leaf201_IntProf` and `Leaf202_IntProf` (node 201 and node 202)
- An access port selector called `Eth1_1` (under the `Leaf201` interface profile) is pointing toward the same vPC interface policy group.
- An access port selector called `Eth1_2` (under the `Leaf202` interface profile) is pointing toward the same vPC interface policy group.
- The vPC interface policy group is pointing toward an AAEP called `Customer_AEP`.
- The AEP (`Customer_AEP`) has an association with the `Customer_PhysDom`.
- The `Customer_PhysDom` has an association with a VLAN pool called `Customer_Static_VLPool`.

Figure 23: vPC With Different Leaf Switch Interfaces Across Two Leaf Switchs With Individual Profiles



What This Configuration Does

On ports Eth1/1 on Leaf201 and Eth 1/2 on Leaf202, configure those ports to be a part of a vPC. This vPC interface will have access to VLANs 1201 through 1299. Depending on the interface policy group, you can enable LACP active and other interface specific policy configurations.

When to Use This Configuration

Use this configuration in a lab environment where you cannot use interfaces that match up. However, you must constantly refer to the GUI to determine where you plugged in your server. As a result, this configuration is cumbersome and is not ideal.

Note Do not use this configuration in production.

Defining vPC Switch Pairs Using the GUI

This procedure defines vPC switch pairs using the GUI. We recommend that you keep the leaf switch peer group names simple as shown in the following example:

- Leaf201_202
- Leaf203_204
- Leaf205_206

For naming and numbering best practices, see the *Cisco ACI Object Naming and Numbering: Best Practices* document:

https://www.cisco.com/c/en/us/td/docs/switches/datacenter/aci/apic/sw/kb/b-Cisco-ACI-Naming-and-Numbering.html

Procedure

Step 1 On the menu bar, choose **Fabric** > **Access Policies**.

Step 2 In the Navigation pane, choose **Policies** > **Switch** > **Virtual Port Channel default**.

Step 3 In the **Explicit vPC Protection Groups** table, click + and fill out the fields as follows:

a) In the **Name** field, enter the vPC pair name.

Example name: `Leaf201_202`. A name similar to the example easily identifies which two fabric nodes are vPC peers.

b) In the **ID** field, enter the vPC pair ID (logical peer ID).

Example ID: `201`. The example uses the first node ID number of the pair to make it easier to correlate the ID with the vPC pair.

c) In the **Switch 1** and **Switch 2** fields, choose the leaf switches for the vPC switch pair.

d) Click **Submit**.

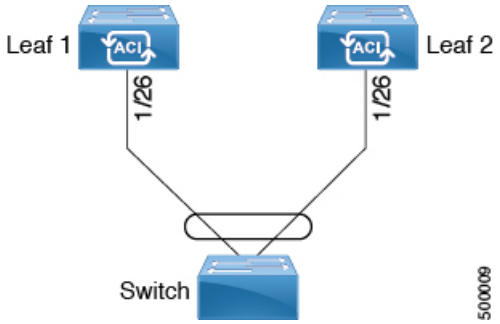
The vPC pair gets added to the **Explicit vPC Protection Groups** table. The **Virtual IP** value is an auto-generated IP address from the system tunnel endpoint (TEP) pool, and represents the virtual shared (Anycast) TEP of the vPC switch pair. That is, packets destined to vPC-connected endpoints of the vPC pair will use this Anycast VTEP to send the packets.

ACI Leaf Switch Virtual Port Channel Configuration Using the GUI

The procedure below uses a Quick Start wizard.

Note This procedure provides the steps for attaching a trunked switch to a ACI leaf switch virtual port channel. The steps would be the same for attaching other kinds of devices to an ACI leaf switch interface.

Figure 24: Switch Virtual Port Channel Configuration



Note LACP sets a port to the suspended state if it does not receive an LACP PDU from the peer. This can cause some servers to fail to boot up as they require LACP to logically bring-up the port. You can tune behavior to individual use by disabling **LACP suspend individual**. To do so, create a port channel policy in your vPC policy group, and after setting the mode to LACP active, remove **Suspend Individual Port**. Now the ports in the vPC will stay active and continue to send LACP packets.

Note Adaptive Load Balancing (ALB) (based on ARP Negotiation) across virtual port channels is not supported in the ACI.

Before you begin

- The ACI fabric is installed, APIC controllers are online, and the APIC cluster is formed and healthy.
- An APIC fabric administrator account is available that will enable creating the necessary fabric infrastructure configurations.
- The target leaf switches are registered in the ACI fabric and available.

Note When creating a vPC domain between two leaf switches, both switches must be in the same switch generation, one of the following:

- Generation 1 - Cisco Nexus N9K switches without “EX” on the end of the switch name; for example, N9K-9312TX
- Generation 2 – Cisco Nexus N9K switches with “EX” on the end of the switch model name; for example, N9K-93108TC-EX

Switches such as these two are not compatible vPC peers. Instead, use switches of the same generation.

Procedure

Step 1 On the APIC menu bar, navigate to **Fabric** > **Access Policies** > **Quick Start**, and click *Configure an interface, PC, and vPC*.

Step 2 In the *Configure an interface, PC, and vPC* work area, click the large green **+** to select switches.
The **Select Switches To Configure Interfaces** work area opens with the **Quick** option selected by default.

Step 3 Select switch IDs from the **Switches** drop-down list, name the profile, then click **Save**.
The saved policy displays in the *Configured Switch Interfaces* list.

Step 4 Configure the *Interface Policy Group* and *Attached Device Type* that the virtual port channel will use for the selected switches.

The interface policy group is a named policy that specifies the group of interface policies you will apply to the selected interfaces of the switch. Examples of interface policies include Link Level Policy (for example, 1gbit port speed), Storm Control Interface Policy, and so forth.

Note The *Attached Device Type* domain is required for enabling an EPG to use the interfaces specified in the switch profile.

a) Specify *vpc* the interface type (individual, PC, or vPC) to use.
b) Specify the interface IDs to use.
c) Specify the interface policies to use.
d) Specify the attached device type to use. Choose **`External Bridged Devices`** for connecting a switch.
e) Specify the *Domain*, and *VLAN Range*.
f) Click **Save** to update the policy details, then click **Submit** to submit the switch profile to the APIC.
The APIC creates the switch profile, along with the interface, selector, and attached device type policies.

Verification: Use the CLI **show int** command on the leaf switches where the external switch is attached to verify that the vPC is configured accordingly.

What to do next

This completes the switch virtual port channel configuration steps.

Note While this configuration enables hardware connectivity, no data traffic can flow without a valid application profile, EPG, and contract that is associated with this hardware configuration.

Configuring Virtual Port Channels in Leaf Nodes and FEX Devices Using the NX-OS CLI

A virtual port channel (vPC) is an enhancement to port-channels that allows connection of a host or switch to two upstream leaf nodes to improve bandwidth utilization and availability. In NX-OS, vPC configuration is done in each of the two upstream switches and configuration is synchronized using peer link between the switches.

Note When creating a vPC domain between two leaf switches, both switches must be in the same switch generation, one of the following:

- Generation 1 - Cisco Nexus N9K switches without “EX” on the end of the switch name; for example, N9K-9312TX
- Generation 2 – Cisco Nexus N9K switches with “EX” on the end of the switch model name; for example, N9K-93108TC-EX

Switches such as these two are not compatible vPC peers. Instead, use switches of the same generation.

The Cisco Application Centric Infrastructure (ACI) model does not require a peer link and vPC configuration can be done globally for both the upstream leaf nodes. A global configuration mode called **vpc context** is introduced in Cisco ACI and vPC interfaces are represented using a type **interface vpc** that allows global configuration applicable to both leaf nodes.

Two different topologies are supported for vPC in the Cisco ACI model: vPC using leaf ports and vPC over FEX ports. It is possible to create many vPC interfaces between a pair of leaf nodes and similarly, many vPC interfaces can be created between a pair of FEX modules attached to the leaf node pairs in a straight-through topology.

vPC considerations include:

- The vPC name used is unique between leaf node pairs. For example, only one vPC 'corp' can be created per leaf pair (with or without FEX).
- Leaf ports and FEX ports cannot be part of the same vPC.
- Each FEX module can be part of only one instance of vPC corp.
- vPC context allows configuration
- The vPC context mode allows configuration of all vPCs for a given leaf pair. For vPC over FEX, the *fex-id* pairs must be specified either for the vPC context or along with the vPC interface, as shown in the following two alternative examples.

```
(config)# vpc context leaf 101 102
(config-vpc)# interface vpc Reg fex 101 101
```

or

```
(config)# vpc context leaf 101 102 fex 101 101
(config-vpc)# interface vpc Reg
```

In the Cisco ACI model, vPC configuration is done in the following steps (as shown in the examples below).

Note A VLAN domain is required with a VLAN range. It must be associated with the port-channel template.

1. VLAN domain configuration (global config) with VLAN range
2. vPC domain configuration (global config)
3. Port-channel template configuration (global config)

4. Associate the port channel template with the VLAN domain
5. Port-channel configuration for vPC (global config)
6. Configure ports to vPC in leaf nodes
7. Configure Layer 2, Layer 3 for vPC in the vPC context

Procedure

Step 1 **configure**

Enters global configuration mode.

Example:

```
apic1# configure
```

Step 2 **vlan-domain***name*[**dynamic**] [**type** *domain-type*]

Configures a VLAN domain for the virtual port-channel (here with a port-channel template).

Example:

```
apic1(config)# vlan-domain dom1 dynamic
```

Step 3 **vlan***range*

Configures a VLAN range for the VLAN domain and exits the configuration mode. The range can be a single VLAN or a range of VLANs.

Example:

```
apic1(config-vlan)# vlan 1000-1999
apic1(config-vlan)# exit
```

Step 4 **vpc domain explicit** *domain-id* **leaf** *node-id1 node-id2*

Configures a vPC domain between a pair of leaf nodes. You can specify the vPC domain ID in the explicit mode along with the leaf node pairs.

Alternative commands to configure a vPC domain are as follows:

- **vpc domain** [**consecutive** | **reciprocal**]

  The consecutive and reciprocal options allow auto configuration of a vPC domain across all leaf nodes in the Cisco ACI fabric.

- **vpc domain consecutive** *domain-start* **leaf** *start-node end-node*

  This command configures a vPC domain consecutively for a selected set of leaf node pairs.

Example:

```
apic1(config)# vpc domain explicit 1 leaf 101 102
```

Step 5 **peer-dead-interval** *interval*

Configures the time delay the Leaf switch waits to restore the vPC before receiving a response from the peer. If it does not receive a response from the peer within this time, the Leaf switch considers the peer dead and brings up the vPC with the role as a master. If it does receive a response from the peer it restores the vPC at that point. The range is from 5 seconds to 600 seconds. The default is 200 seconds.

Example:

```
apic1(config-vpc)# peer-dead-interval 10
```

Step 6 **exit**

Returns to global configuration mode.

Example:

```
apic1(config-vpc)# exit
```

Step 7 **template port-channel** *channel-name*

Creates a new port-channel or configures an existing port-channel (global configuration).

All vPCs are configured as port-channels in each leaf pair. The same port-channel name must be used in a leaf pair for the same vPC. This port-channel can be used to create a vPC among one or more pairs of leaf nodes. Each leaf node will have only one instance of this vPC.

Example:

```
apic1(config)# template port-channel corp
```

Step 8 **vlan-domain member***vlan-domain-name*

Associates the port channel template with the previously configured VLAN domain.

Example:

```
vlan-domain member dom1
```

Step 9 **switchport access vlan** *vlan-id* **tenant** *tenant-name* **application** *application-name* **epg** *epg-name*

Deploys the EPG with the VLAN on all ports with which the port-channel is associated.

Example:

```
apic1(config-po-ch-if)# switchport access vlan 4 tenant ExampleCorp application Web epg
webEpg
```

Step 10 **channel-mode active**

Note A port-channel must be in active channel-mode for a vPC.

Example:

```
apic1(config-po-ch-if)# channel-mode active
```

Step 11 **exit**

Returns to configure mode.

Example:

```
apic1(config-po-ch-if)# exit
```

Step 12 **leaf** *node-id1 node-id2*

Specifies the pair of leaf switches to be configured.

Example:

```
apic1(config)# leaf 101-102
```

Step 13 **interface** *typeleaf/interface-range*

Specifies the interface or range of interfaces that you are configuring to the port-channel.

Example:

```
apic1(config-leaf)# interface ethernet 1/3-4
```

Step 14 [**no**] **channel-group** *channel-name* **vpc**

Assigns the interface or range of interfaces to the port-channel. Use the keyword **no** to remove the interface from the port-channel. To change the port-channel assignment on an interface, you can enter the **channel-group** command without first removing the interface from the previous port-channel.

Note The **vpc** keyword in this command makes the port-channel a vPC. If the vPC does not already exist, a vPC ID is automatically generated and is applied to all member leaf nodes.

Example:

```
apic1(config-leaf-if)# channel-group corp vpc
```

Step 15 **exit**

Example:

```
apic1(config-leaf-if)# exit
```

Step 16 **exit**

Example:

```
apic1(config-leaf)# exit
```

Step 17 **vpc context leaf** *node-id1 node-id2*

The vPC context mode allows configuration of vPC to be applied to both leaf node pairs.

Example:

```
apic1(config)# vpc context leaf 101 102
```

Step 18 **interface vpc** *channel-name*

Example:

```
apic1(config-vpc)# interface vpc blue fex 102 102
```

Step 19 (Optional) [**no**] **shutdown**

Administrative state configuration in the vPC context allows changing the admin state of a vPC with one command for both leaf nodes.

Example:

```
apic1(config-vpc-if)# no shut
```

Example

This example shows how to configure a basic vPC.

```
apic1# configure
apic1(config)# vlan-domain dom1 dynamic
apic1(config-vlan)# vlan 1000-1999
apic1(config-vlan)# exit
apic1(config)# vpc domain explicit 1 leaf 101 102
apic1(config-vpc)# peer-dead-interval 10
```

```
apic1(config-vpc)# exit
apic1(config)# template port-channel corp
apic1(config-po-ch-if)# vlan-domain member dom1
apic1(config-po-ch-if)# channel-mode active
apic1(config-po-ch-if)# exit
apic1(config)# leaf 101-102
apic1(config-leaf)# interface ethernet 1/3-4
apic1(config-leaf-if)# channel-group corp vpc
apic1(config-leaf-if)# exit
apic1(config)# vpc context leaf 101 102
```

This example shows how to configure vPCs with FEX ports.

```
apic1(config-leaf)# interface ethernet 101/1/1-2
apic1(config-leaf-if)# channel-group Reg vpc
apic1(config)# vpc context leaf 101 102
apic1(config-vpc)# interface vpc corp
apic1(config-vpc-if)# exit
apic1(config-vpc)# interface vpc red fex 101 101
apic1(config-vpc-if)# switchport
apic1(config-vpc-if)# exit
apic1(config-vpc)# interface vpc blue fex 102 102
apic1(config-vpc-if)# shut
```

Configuring a Single Virtual Port Channel Across Two Switches Using the REST API

The two steps for creating a virtual port channel across two switches are as follows:

- Create a `fabricExplicitGEp`: this policy specifies the leaf switch that pairs to form the virtual port channel.
- Use the infra selector to specify the interface configuration.

The APIC performs several validations of the `fabricExplicitGEp` and faults are raised when any of these validations fail. A leaf can be paired with only one other leaf. The APIC rejects any configuration that breaks this rule. When creating a `fabricExplicitGEp`, an administrator must provide the IDs of both of the leaf switches to be paired. The APIC rejects any configuration which breaks this rule. Both switches must be up when `fabricExplicitGEp` is created. If one switch is not up, the APIC accepts the configuration but raises a fault. Both switches must be leaf switches. If one or both switch IDs corresponds to a spine, the APIC accepts the configuration but raises a fault.

Before you begin

- The ACI fabric is installed, APIC controllers are online, and the APIC cluster is formed and healthy.
- An APIC fabric administrator account is available that will enable creating the necessary fabric infrastructure configurations.
- The target leaf switch and protocol(s) are configured and available.

Procedure

To create the `fabricExplicitGEp` policy and use the intra selector to specify the interface, send a post with XML such as the following example:

Example:

```
<fabricProtPol pairT="explicit">
<fabricExplicitGEp name="tG" id="2">
 <fabricNodePEp id=”18”/>
 <fabricNodePEp id=”25"/>
        </fabricExplicitGEp>
   </fabricProtPol>
```

Configuring a Virtual Port Channel on Selected Port Blocks of Two Switches Using the REST API

This policy creates a single virtual port channel (vPC) on leaf switches 18 and 25, using interfaces 1/10 to 1/15 on leaf 18, and interfaces 1/20 to 1/25 on leaf 25.

Before you begin

- The ACI fabric is installed, APIC controllers are online, and the APIC cluster is formed and healthy.
- An APIC fabric administrator account is available that will enable creating the necessary fabric infrastructure configurations.
- The target leaf switch and protocol(s) are configured and available.

Note When creating a vPC domain between two leaf switches, both switches must be in the same switch generation, one of the following:

- Generation 1 - Cisco Nexus N9K switches without “EX” on the end of the switch name; for example, N9K-9312TX
- Generation 2 – Cisco Nexus N9K switches with “EX” on the end of the switch model name; for example, N9K-93108TC-EX

Switches such as these two are not compatible vPC peers. Instead, use switches of the same generation.

Procedure

To create the vPC send a post with XML such as the following example:

Example:

```
<infraInfra dn="uni/infra">

       <infraNodeP name=”test1">
```

```
            <infraLeafS name="leafs" type="range">
                 <infraNodeBlk name="nblk”
                      from_=”18" to_=”18”/>
            </infraLeafS>
            <infraRsAccPortP tDn="uni/infra/accportprof-test1"/>
        </infraNodeP>

        <infraNodeP name=”test2">
            <infraLeafS name="leafs" type="range">
                 <infraNodeBlk name="nblk”
                      from_=”25" to_=”25”/>
            </infraLeafS>
            <infraRsAccPortP tDn="uni/infra/accportprof-test2"/>
        </infraNodeP>

        <infraAccPortP name="test1">
            <infraHPortS name="pselc" type="range">
                 <infraPortBlk name="blk1”
                      fromCard="1" toCard="1"
                      fromPort="10" toPort=”15”/>
                 <infraRsAccBaseGrp
                      tDn="uni/infra/funcprof/accbundle-bndlgrp" />
             </infraHPortS>
        </infraAccPortP>

        <infraAccPortP name="test2">
             <infraHPortS name="pselc" type="range">
                  <infraPortBlk name="blk1”
                       fromCard="1" toCard="1"
                       fromPort=“20" toPort=”25”/>
             <infraRsAccBaseGrp
                  tDn="uni/infra/funcprof/accbundle-bndlgrp" />
             </infraHPortS>
         </infraAccPortP>

         <infraFuncP>
               <infraAccBndlGrp name="bndlgrp" lagT=”node">
                    <infraRsHIfPol  tnFabricHIfPolName=“default"/>
                    <infraRsCdpIfPol tnCdpIfPolName=”default”/>
                    <infraRsLacpPol tnLacpLagPolName=”default"/>
               </infraAccBndlGrp>
         </infraFuncP>

</infraInfra>
```

Reflective Relay

Reflective Relay (802.1Qbg)

Reflective relay is a switching option beginning with Cisco APIC Release 2.3(1). Reflective relay—the tagless approach of IEEE standard 802.1Qbg—forwards all traffic to an external switch, which then applies policy and sends the traffic back to the destination or target VM on the server as needed. There is no local switching. For broadcast or multicast traffic, reflective relay provides packet replication to each VM locally on the server.

One benefit of reflective relay is that it leverages the external switch for switching features and management capabilities, freeing server resources to support the VMs. Reflective relay also allows policies that you configure on the Cisco APIC to apply to traffic between the VMs on the same server.

In the Cisco ACI, you can enable reflective relay, which allows traffic to turn back out of the same port it came in on. You can enable reflective relay on individual ports, port channels, or virtual port channels as a Layer 2 interface policy using the APIC GUI, NX-OS CLI, or REST API. It is disabled by default.

The term *Virtual Ethernet Port Aggregator* (VEPA) is also used to describe 802.1Qbg functionality.

Reflective Relay Support

Reflective relay supports the following:

- IEEE standard 802.1Qbg tagless approach, known as reflective relay.

  Cisco APIC Release 2.3(1) release does not support the IEE standard 802.1Qbg S-tagged approach with multichannel technology.

- Physical domains.

  Virtual domains are not supported.

- Physical ports, port channels (PCs), and virtual port channels (vPCs).

  Cisco Fabric Extender (FEX) and blade servers are not supported. If reflective relay is enabled on an unsupported interface, a fault is raised, and the last valid configuration is retained. Disabling reflective relay on the port clears the fault.

- Cisco Nexus 9000 series switches with *EX* or *FX* at the end of their model name.

Enabling Reflective Relay Using the GUI

Reflective relay is disabled by default; however, you can enable it on a port, port channel, or virtual port channel as a Layer 2 interface policy on the switch. You first configure a policy and then associate the policy with a policy group.

Before you begin

This procedure assumes that you have set up the Cisco Application Centric Infrastructure (ACI) fabric and installed the physical switches.

Procedure

Step 1 Choose **Fabric** > **External Access Policies** > > **Interface Policies** and then open the **Policies** folder.

Step 2 Right-click the **L2 Interface** folder and choose **Create L2 Interface Policy**.

Step 3 In the **Create L2 Interface Policy** dialog box, enter a name in the **Name** field.

Step 4 In the **Reflective Relay (802.1Qbg)** area, click **enabled**.

Step 5 Choose other options in the dialog box as needed.

Step 6 Click **SUBMIT**.

Step 7 In the **Policies** navigation pane, open the **Policy Groups** folder and click the **Leaf Policy Groups** folder.

Step 8 In the **Leaf Policy Groups** central pane, expand the **ACTIONS** drop-down list, and choose **Create Leaf Access Port Policy Group**, **Create PC Interface Policy Group**, **Create vPC Interface Policy Group**, or **Create PC/vPC Override Policy Group**.

Step 9 In the policy group dialog box, enter a name in the **Name field**.

Step 10 From the **L2 Interface Policy** drop-down list, choose the policy that you just created to enable Reflective Relay.

Step 11 Click **Submit**.

Enabling Reflective Relay Using the NX-OS CLI

Reflective relay is disabled by default; however, you can enable it on a port, port channel, or virtual port channel as a Layer 2 interface policy on the switch. In the NX-OS CLI, you can use a template to enable reflective relay on multiple ports or you can enable it on individual ports.

Before you begin

This procedure assumes that you have set up the Cisco Application Centric Infrastructure (ACI) fabric and installed the physical switches.

Procedure

Enable reflective relay on one or multiple ports:

Example:

This example enables reflective relay on a single port:

```
apic1(config)# leaf 101
apic1(config-leaf)# interface ethernet 1/2
apic1(config-leaf-if)# switchport vepa enabled
apic1(config-leaf-if)# exit
apic1(config-leaf)# exit
```

Example:

This example enables reflective relay on multiple ports using a template:

```
apic1(config)# template policy-group grp1
apic1(config-pol-grp-if)# switchport vepa enabled
apic1(config-pol-grp-if)# exit
apic1(config)# leaf 101
apic1(config-leaf)# interface ethernet 1/2-4
apic1(config-leaf-if)# policy-group grp1
```

Example:

This example enables reflective relay on a port channel:

```
apic1(config)# leaf 101
apic1(config-leaf)# interface port-channel po2
apic1(config-leaf-if)# switchport vepa enabled
apic1(config-leaf-if)# exit
apic1(config-leaf)# exit
apic1(config)#
```

Example:

This example enables reflective relay on multiple port channels:

```
apic1(config)# template port-channel po1
apic1(config-if)# switchport vepa enabled
apic1(config-if)# exit
apic1(config)# leaf 101
apic1(config-leaf)# interface ethernet 1/3-4
apic1(config-leaf-if)# channel-group po1
apic1(config-leaf-if)# exit
apic1(config-leaf)# exit
```

Example:

This example enables reflective relay on a virtual port channel:

```
apic1(config)# vpc domain explicit 1 leaf 101 102
apic1(config-vpc)# exit
apic1(config)# template port-channel po4
apic1(config-if)# exit
apic1(config)# leaf 101-102
apic1(config-leaf)# interface eth 1/11-12
apic1(config-leaf-if)# channel-group po4 vpc
apic1(config-leaf-if)# exit
apic1(config-leaf)# exit
apic1(config)# vpc context leaf 101 102
apic1(config-vpc)# interface vpc po4
apic1(config-vpc-if)# switchport vepa enabled
```

Enabling Reflective Relay Using the REST API

Reflective relay is disabled by default; however, you can enable it on a port, port channel, or virtual port channel as a Layer 2 interface policy on the switch.

Before you begin

This procedure assumes that you have set up the Cisco Application Centric Infrastructure (ACI) fabric and installed the physical switches.

Procedure

Step 1 Configure a Layer 2 Interface policy with reflective relay enabled.

Example:

```
<l2IfPol name="VepaL2IfPol" vepa="enabled"  />
```

Step 2 Apply the Layer 2 interface policy to a leaf access port policy group.

Example:

```
<infraAccPortGrp name="VepaPortG">
    <infraRsL2IfPol tnL2IfPolName="VepaL2IfPol"/>
</infraAccPortGrp>
```

Step 3 Configure an interface profile with an interface selector.

Example:

```
<infraAccPortP name="vepa">
        <infraHPortS name="pselc" type="range">
             <infraPortBlk name="blk"
                 fromCard="1" toCard="1" fromPort="20" toPort="22">
             </infraPortBlk>
             <infraRsAccBaseGrp tDn="uni/infra/funcprof/accportgrp-VepaPortG" />
         </infraHPortS>
 </infraAccPortP>
```

Step 4 Configure a node profile with node selector.

Example:

```
<infraNodeP name="VepaNodeProfile">
        <infraLeafS name="VepaLeafSelector" type="range">
                 <infraNodeBlk name="VepaNodeBlk" from_="101" to_="102"/>
        </infraLeafS>
               <infraRsAccPortP tDn="uni/infra/accportprof-vepa"/>
    </infraNodeP>
```

FEX Interfaces

Configuring Port, PC, and vPC Connections to FEX Devices

FEX connections and the profiles used to configure them can be created using the GUI, NX-OS Style CLI, or the REST API.

Interface profiles for configuring FEX connections are supported since Cisco APIC, Release 3.0(1k).

For information on how to configure them using the NX-OS style CLI, see the topics about configuring ports, PCs and vPCs using the NX-OS style CLI.

ACI FEX Guidelines

Observe the following guidelines when deploying a FEX:

- Assuming that no leaf switch front panel ports are configured to deploy and EPG and VLANs, a maximum of 10,000 port EPGs are supported for being deployed using a FEX.
- For each FEX port or vPC that includes FEX ports as members, a maximum of 20 EPGs per VLAN are supported.
- A vPC with FEX interfaces ignores the minimum and maximum number of links configured in its port-channel policy. The vPC remains up even if the number of links is less than the minimum or greater than the maximum.

FEX Virtual Port Channels

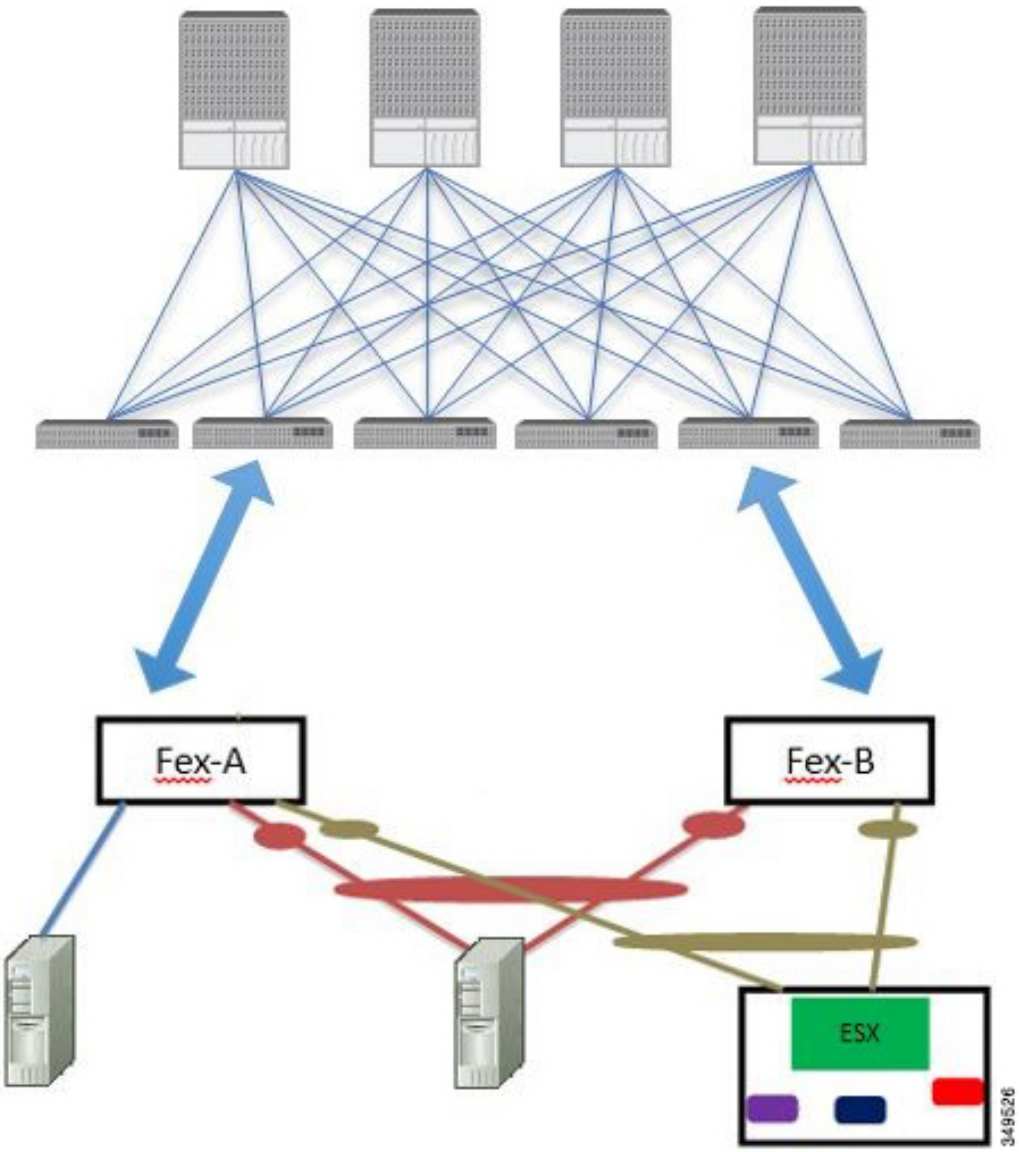
The ACI fabric supports Cisco Fabric Extender (FEX) server-side virtual port channels (vPC), also known as an FEX straight-through vPC.

Note When creating a vPC domain between two leaf switches, both switches must be in the same switch generation, one of the following:

- Generation 1 - Cisco Nexus N9K switches without “EX” or "FX" on the end of the switch name; for example, N9K-9312TX
- Generation 2 – Cisco Nexus N9K switches with “EX” or "FX" on the end of the switch model name; for example, N9K-93108TC-EX

Switches such as these two are not compatible vPC peers. Instead, use switches of the same generation.

Figure 25: Supported FEX vPC Topologies



Supported FEX vPC port channel topologies include the following:

- Both VTEP and non-VTEP hypervisors behind a FEX.
- Virtual switches (such as AVS or VDS) connected to two FEXs that are connected to the ACI fabric (vPCs directly connected on physical FEX ports is not supported - a vPC is supported only on port channels).

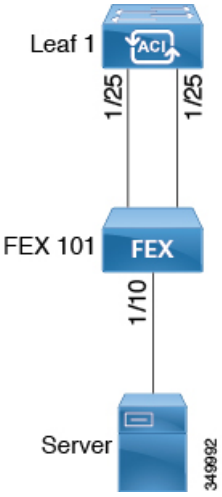
Note When using GARP as the protocol to n.jpgy of IP to MAC binding changes to different interfaces on the same FEX you must set the bridge domain mode to **ARP Flooding** and enable **EP Move Detection Mode**: **GARP-based Detection**, on the **L3 Configuration** page of the bridge domain wizard. This workaround is only required with Generation 1 switches. With Generation 2 switches or later, this is not an issue.

Configuring a Basic FEX Connection Using the GUI

The procedure below uses a Quick Start wizard that automatically creates some necessary policies for FEX deployment. The main steps are as follows:

1. Configure a switch profile that includes an auto-generated FEX profile.
2. Customize the auto-generated **FEX Profile** to enable attaching a server to a single FEX port.

Figure 26: Basic FEX Configuration



Note This procedure provides the steps for attaching a server to the FEX. The steps would be the same for attaching any device to an ACI attached FEX.

Note Configuring FEX connections with FEX IDs `165` to `199` is not supported in the APIC GUI. To use one of these FEX IDs, configure the profile using the NX-OS style CLI. For more information, see *Configuring FEX Connections Using Interface Profiles with the NX-OS Style CLI*.

Before you begin

- The ACI fabric is installed, APIC controllers are online, and the APIC cluster is formed and healthy.
- An APIC fabric administrator account is available that will enable creating the necessary fabric infrastructure configurations.
- The target leaf switches, interfaces, and protocol(s) are configured and available.
- The FEX is powered on and connected to the target leaf interfaces

Note A maximum of eight members is supported in fabric port-channels connected to FEXs.

Procedure

Step 1 On the APIC, create a switch profile using the **Fabric** > **Access Policies** > **Quick Start Configure Interface, PC, And vPC** wizard.

a) On the APIC menu bar, navigate to **Fabric** > **Access Policies** > **Quick Start**.
b) In the **Quick Start** page, click the **Configure an interface, PC, and vPC** option to open the **Configure Interface, PC And vPC** wizard.
c) In the **Configure an interface, PC, and vPC** work area, click the **+** to add a new switch profile.
d) In the **Select Switches To Configure Interfaces** work area, click the **Advanced** radio button.
e) Select the switch from the drop-down list of available switch IDs.

Troubleshooting Tips

In this procedure, one switch is included in the profile. Selecting multiple switches allows the same profile to be used on multiple switches.

f) Provide a name in the *Switch Profile Name* field.
g) Click the **+** above the **Fexes** list to add a FEX ID and the switch ports to which it will connect to the switch profile.

You must configure FEX IDs 165 - 199, using the NX-OS style CLI. See *Configuring FEX Connections Using Interface Profiles with the NX-OS Style CLI*.

h) Click **Save** to save the changes. Click **Submit** to submit the switch profile to the APIC.
The APIC auto-generates the necessary FEX profile (*<switch policy name>_FexP<FEX ID>*) and selector (*<switch policy name>_ifselctor*).

Verification: Use the CLI **show fex** command on the switch where the FEX is attached to verify that the FEX is online.

Step 2 Customize the auto-generated **FEX Profile** to enable attaching a server to a single FEX port.

a) In the **Navigation** pane, locate the switch policy you just created in the policies list. You will also find the auto-generated FEX the *<switch policy name>_FexP<FEX ID>* profile.
b) In the work pane of the *<switch policy name>_FexP<FEX ID>* profile, click the **+** to add a new entry to the *Interface Selectors For FEX* list.
The **Create Access Port Selector** dialog opens.
c) Provide a name for the selector.
d) Specify the FEX interface IDs to use.
e) Select an existing *Interface Policy Group* from the list or *Create Access Port Policy Group*.

The access port policy group is a named policy that specifies the group of interface policies you will apply to the selected interfaces of the FEX. Examples of interface policies include Link Level Policy (for example, 1gbit port speed), Attach Entity Profile, Storm Control Interface Policy, and so forth.

Note Within the interface policy group, the *Attached Entity Profile* is required for enabling an EPG to use the interfaces specified in the FEX port selector.

f) Click **Submit** to submit the FEX profile to the APIC.
The APIC updates the FEX profile.

Verification: Use the CLI **show int** command on the switch where the FEX is attached to verify that the FEX interface is configured accordingly.

This completes the basic FEX configuration steps.

What to do next

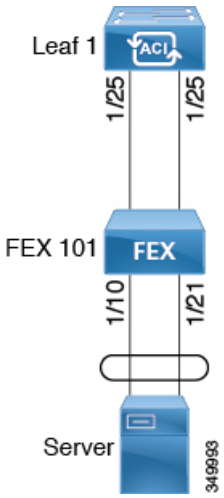
Note While this configuration enables hardware connectivity, no data traffic can flow without a valid application profile, EPG, and contract that is associated with this hardware configuration.

Configuring FEX Port Channel Connections Using the GUI

The main steps are as follows:

1. Configure an FEX profile to use FEX ports to form a port channel.
2. Configure the port channel to enable attaching a server.

Figure 27: FEX port channel



Note This procedure provides the steps for attaching a server to the FEX port channel. The steps would be the same for attaching any device to an ACI attached FEX.

Before you begin

- The ACI fabric is installed, APIC controllers are online, and the APIC cluster is formed and healthy.
- An APIC fabric administrator account is available that will enable creating the necessary fabric infrastructure configurations.
- The target leaf switch, interfaces, and protocol(s) are configured and available.
- The FEX is configured, powered on, and connected to the target leaf interfaces

Procedure

Step 1 On the APIC, add a port channel to a FEX profile.

a) On the APIC menu bar, navigate to **Fabric** > **Access Policies** > **Interfaces** > **Leaf Interfaces** > **Profiles**.

b) In the **Navigation Pane**, select the FEX profile.

APIC auto-generated FEX profile names are formed as follows: *<switch policy name>_FexP<FEX ID>*.

c) In the **FEX Profile** work area, click the +to add a new entry to the *Interface Selectors For FEX* list. The **Create Access Port Selector** dialog opens.

Step 2 Customize the **Create Access Port Selector** to enable attaching a server to the FEX port channel.

a) Provide a name for the selector.

b) Specify the FEX interface IDs to use.

c) Select an existing *Interface Policy Group* from the list or *Create PC Interface Policy Group*.

The port channel interface policy group specifies the group of policies you will apply to the selected interfaces of the FEX. Examples of interface policies include Link Level Policy (for example, 1gbit port speed), Attach Entity Profile, Storm Control Interface Policy, and so forth.

Note Within the interface policy group, the *Attached Entity Profile* is required for enabling an EPG to use the interfaces specified in the FEX port selector.

d) In the *Port Channel Policy* option, select static or dynamic LACP according to the requirements of your configuration.
e) Click **Submit** to submit the updated FEX proflle to the APIC.
The APIC updates the FEX profile.

Verification: Use the CLI **show port-channel summary** command on the switch where the FEX is attached to verify that the port channel is configured accordingly.

What to do next

This completes the FEX port channel configuration steps.

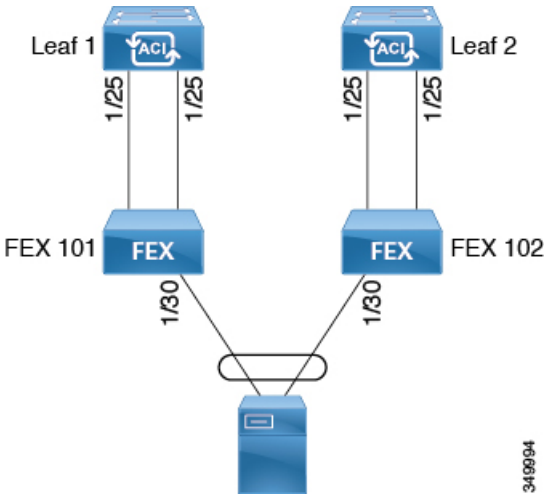
Note While this configuration enables hardware connectivity, no data traffic can flow without a valid application profile, EPG, and contract that is associated with this hardware configuration.

Configuring FEX vPC Connections Using the GUI

The main steps are as follows:

1. Configure two existing FEX profiles to form a virtual port channel.
2. Configure the virtual port channel to enable attaching a server to the FEX port channel.

Figure 28: FEX virtual port channel



Note This procedure provides the steps for attaching a server to the FEX virtual port channel. The steps would be the same for attaching any device to an ACI attached FEX.

Before you begin

- The ACI fabric is installed, APIC controllers are online, and the APIC cluster is formed and healthy.
- An APIC fabric administrator account is available that will enable creating the necessary fabric infrastructure configurations.
- The target leaf switch, interfaces, and protocol(s) are configured and available.
- The FEXes are configured, powered on, and connected to the target leaf interfaces

Note When creating a vPC domain between two leaf switches, both switches must be in the same switch generation, one of the following:

- Generation 1 - Cisco Nexus N9K switches without “EX” on the end of the switch name; for example, N9K-9312TX
- Generation 2 – Cisco Nexus N9K switches with “EX” on the end of the switch model name; for example, N9K-93108TC-EX

Switches such as these two are not compatible vPC peers. Instead, use switches of the same generation.

Procedure

Step 1 On the APIC, add a virtual port channel to two FEX profiles.

a) On the APIC menu bar, navigate to **Fabric** > **External Access Policies** > **Interfaces** > **Leaf Interfaces** > **Profiles**.

b) In the **Navigation Pane**, select the first FEX profile.

APIC auto-generated FEX profile names are formed as follows: *<switch policy name>_FexP<FEX ID>*.

c) In the **FEX Profile** work area, click the **+**to add a new entry to the *Interface Selectors For FEX* list. The **Create Access Port Selector** dialog opens.

Step 2 Customize the **Create Access Port Selector** to enable attaching a server to the FEX virtual port channel.

a) Provide a name for the selector.

b) Specify the FEX interface ID to use.

Typically, you will use the same interface ID on each FEX to form the virtual port channel.

c) Select an existing *Interface Policy Group* from the list or *Create VPC Interface Policy Group*.

The virtual port channel interface policy group specifies the group of policies you will apply to the selected interfaces of the FEX. Examples of interface policies include Link Level Policy (for example, 1gbit port speed), Attach Entity Profile, Storm Control Interface Policy, and so forth.

Note Within the interface policy group, the *Attached Entity Profile* is required for enabling an EPG to use the interfaces specified in the FEX port selector.

d) In the *Port Channel Policy* option, select static or dynamic LACP according to the requirements of your configuration.
e) Click **Submit** to submit the updated FEX proflle to the APIC.
The APIC updates the FEX profile.

Verification: Use the CLI **show port-channel summary** command on the switch where the FEX is attached to verify that the port channel is configured accordingly.

Step 3 Configure the second FEX to use the same *Interface Policy Group* just specified for the first FEX.

a) In the **FEX Profile** work area of the second FEX profile, click the +to add a new entry to the *Interface Selectors For FEX* list.
The **Create Access Port Selector** dialog opens.
b) Provide a name for the selector.
c) Specify the FEX interface ID to use.

Typically, you will use the same interface ID on each FEX to form the virtual port channel.

d) From the drop-down list, select the same virtual port channel *Interface Policy Group* just used in the first FEX profile.

The virtual port channel interface policy group specifies the group of policies you will apply to the selected interfaces of the FEX. Examples of interface policies include Link Level Policy (for example, 1gbit port speed), Attach Entity Profile, Storm Control Interface Policy, and so forth.

Note Within the interface policy group, the *Attached Entity Profile* is required for enabling an EPG to use the interfaces specified in the FEX port selector.

e) Click **Submit** to submit the updated FEX proflle to the APIC.
The APIC updates the FEX profile.

Verification: Use the CLI **show vpc extended** command on the switch where one of the FEXes is attached to verify that the virtual port channel is configured accordingly.

What to do next

This completes the FEX virtual port channel configuration steps.

Note While this configuration enables hardware connectivity, no data traffic can flow without a valid application profile, EPG, and contract that is associated with this hardware configuration.

Configuring an FEX VPC Policy Using the REST API

This task creates a FEX virtual port channel (VPC) policy.

Before you begin

- The ACI fabric is installed, APIC controllers are online, and the APIC cluster is formed and healthy.
- An APIC fabric administrator account is available that will enable creating the necessary fabric infrastructure configurations.
- The target leaf switch, interfaces, and protocol(s) are configured and available.
- The FEXes are configured, powered on, and connected to the target leaf interfaces

Note When creating a VPC domain between two leaf switches, both switches must be in the same switch generation, one of the following:

- Generation 1 - Cisco Nexus N9K switches without “EX” on the end of the switch name; for example, N9K-9312TX
- Generation 2 – Cisco Nexus N9K switches with “EX” on the end of the switch model name; for example, N9K-93108TC-EX

Switches such as these two are not compatible VPC peers. Instead, use switches of the same generation.

Procedure

To create the policy linking the FEX through a VPC to two switches, send a post with XML such as the following example:

Example:

```
<polUni>
<infraInfra dn="uni/infra">

<infraNodeP name="fexNodeP105">
    <infraLeafS name="leafs" type="range">
        <infraNodeBlk name="test" from_="105" to_="105"/>
    </infraLeafS>
    <infraRsAccPortP tDn="uni/infra/accportprof-fex116nif105" />
</infraNodeP>

<infraNodeP name="fexNodeP101">
    <infraLeafS name="leafs" type="range">
        <infraNodeBlk name="test" from_="101" to_="101"/>
    </infraLeafS>
    <infraRsAccPortP tDn="uni/infra/accportprof-fex113nif101" />
</infraNodeP>

<infraAccPortP name="fex116nif105">
   <infraHPortS name="pselc" type="range">
   <infraPortBlk name="blk1"
        fromCard="1" toCard="1" fromPort="45" toPort="48" >
   </infraPortBlk>
   <infraRsAccBaseGrp tDn="uni/infra/fexprof-fexHIF116/fexbundle-fex116" fexId="116" />
</infraHPortS>
</infraAccPortP>

<infraAccPortP name="fex113nif101">
```

```
   <infraHPortS name="pselc" type="range">
   <infraPortBlk name="blk1"
        fromCard="1" toCard="1" fromPort="45" toPort="48" >
   </infraPortBlk>
   <infraRsAccBaseGrp tDn="uni/infra/fexprof-fexHIF113/fexbundle-fex113" fexId="113" />
</infraHPortS>
</infraAccPortP>

<infraFexP name="fexHIF113">
    <infraFexBndlGrp name="fex113"/>
    <infraHPortS name="pselc-fexPC" type="range">
        <infraPortBlk name="blk"
            fromCard="1" toCard="1" fromPort="15" toPort="16" >
        </infraPortBlk>
        <infraRsAccBaseGrp tDn="uni/infra/funcprof/accbundle-fexPCbundle" />
    </infraHPortS>
        <infraHPortS name="pselc-fexVPC" type="range">
        <infraPortBlk name="blk"
            fromCard="1" toCard="1" fromPort="1" toPort="8" >
        </infraPortBlk>
        <infraRsAccBaseGrp tDn="uni/infra/funcprof/accbundle-fexvpcbundle" />
    </infraHPortS>
    <infraHPortS name="pselc-fexaccess" type="range">
        <infraPortBlk name="blk"
            fromCard="1" toCard="1" fromPort="47" toPort="47">
        </infraPortBlk>
        <infraRsAccBaseGrp tDn="uni/infra/funcprof/accportgrp-fexaccport" />
     </infraHPortS>

</infraFexP>

<infraFexP name="fexHIF116">
    <infraFexBndlGrp name="fex116"/>
    <infraHPortS name="pselc-fexPC" type="range">
        <infraPortBlk name="blk"
            fromCard="1" toCard="1" fromPort="17" toPort="18" >
        </infraPortBlk>
        <infraRsAccBaseGrp tDn="uni/infra/funcprof/accbundle-fexPCbundle" />
    </infraHPortS>
        <infraHPortS name="pselc-fexVPC" type="range">
        <infraPortBlk name="blk"
            fromCard="1" toCard="1" fromPort="1" toPort="8" >
        </infraPortBlk>
        <infraRsAccBaseGrp tDn="uni/infra/funcprof/accbundle-fexvpcbundle" />
      </infraHPortS>
      <infraHPortS name="pselc-fexaccess" type="range">
        <infraPortBlk name="blk"
            fromCard="1" toCard="1" fromPort="47" toPort="47">
        </infraPortBlk>
        <infraRsAccBaseGrp tDn="uni/infra/funcprof/accportgrp-fexaccport" />
     </infraHPortS>


</infraFexP>

<infraFuncP>
<infraAccBndlGrp name="fexPCbundle" lagT="link">
    <infraRsLacpPol tnLacpLagPolName='staticLag'/>
    <infraRsHIfPol tnFabricHIfPolName="1GHIfPol" />
    <infraRsAttEntP tDn="uni/infra/attentp-fexvpcAttEP"/>
</infraAccBndlGrp>

<infraAccBndlGrp name="fexvpcbundle" lagT="node">
    <infraRsLacpPol tnLacpLagPolName='staticLag'/>
```

```
    <infraRsHIfPol tnFabricHIfPolName="1GHIfPol" />
    <infraRsAttEntP tDn="uni/infra/attentp-fexvpcAttEP"/>
</infraAccBndlGrp>
</infraFuncP>


<fabricHIfPol name="1GHIfPol" speed="1G" />
<infraAttEntityP name="fexvpcAttEP">
    <infraProvAcc name="provfunc"/>
    <infraRsDomP tDn="uni/phys-fexvpcDOM"/>
</infraAttEntityP>

<lacpLagPol dn="uni/infra/lacplagp-staticLag"
    ctrl="susp-individual,graceful-conv"
    minLinks="2"
    maxLinks="16">
</lacpLagPol>
```

Configuring FEX Connectivity to an ACI leaf switch Using Profiles with the NX-OS-Style CLI

Use this procedure to configure FEX connections to leaf nodes using the NX-OS style CLI.

Note Configuring FEX connections with FEX IDs `165` to `199` is not supported in the Cisco Application Policy Infrastructure Controller (APIC) GUI. To use one of these FEX IDs, configure the profile using the following commands.

Procedure

Step 1 **configure**

Enters global configuration mode.

Example:

```
apic1# configure
```

Step 2 **leaf-interface-profile** *name*

Specifies the leaf interface profile to be configured.

Example:

```
apic1(config)# leaf-interface-profile fexIntProf1
```

Step 3 **leaf-interface-group** *name*

Specifies the interface group to be configured.

Example:

```
apic1(config-leaf-if-profile)# leaf-interface-group leafIntGrp1
```

Step 4 **fex associate** *fex-id* [**template** *template-typefex-template-name*]

Attaches a FEX module to a leaf node. Use the optional template keyword to specify a template to be used. If it does not exist, the system creates a template with the name and type you specified.

Example:

```
apic1(config-leaf-if-group)# fex associate 101
```

Example

This merged example configures a leaf interface profile for FEX connections with ID 101.

```
apic1# configure
apic1(config)# leaf-interface-profile fexIntProf1
apic1(config-leaf-if-profile)# leaf-interface-group leafIntGrp1
apic1(config-leaf-if-group)# fex associate 101
```

Configuring Port Profiles to Change Ports from Uplink to Downlink or Downlink to Uplink

Configuring Port Profiles

Prior to Cisco Application Policy Infrastructure Controller (APIC) release 3.1(1), conversion from uplink port to downlink port or downlink port to uplink port (in a port profile) was not supported on Cisco Application Centric Infrastructure (ACI) leaf switches. Starting with Cisco APIC release 3.1(1), uplink and downlink conversion is supported on Cisco Nexus 9000 series switches with names that end in EX or FX, and later (for example, N9K-C9348GC-FXP). A FEX connected to converted downlinks is also supported.

This functionality is supported on the following Cisco switches:

- N9K-C9348GC-FXP
- N9K-C93180LC-EX and N9K-C93180YC-FX
- N9K-C93108TC-EX and N9K-C93108TC-FX (only uplink to downlink conversion is supported)
- N9K-C9336C-FX2 (only downlink to uplink conversion is supported)

Restrictions

- Fast Link Failover policies and port profiles are not supported on the same port. If port profile is enabled, Fast Link Failover cannot be enabled or vice versa.
- The last 2 uplink ports of supported TOR switches cannot be converted to downlink ports (they are reserved for uplink connections.)
- Up to Cisco APIC release 3.2, port profiles and breakout ports are not supported on the same ports.
- With Cisco APIC release 3.2 and later, dynamic breakouts (both 100Gb and 40Gb) are supported on profiled QSFP ports on the N9K-C93180YC-FX switch. Breakout and port profile are supported together

for conversion of uplink to downlink on ports 49-52. Breakout (both **10g-4x** or **25g-4x** options) is supported on downlink profiled ports.

- Reloading a switch after changing a switch's port profile configuration interrupts traffic through the data plane.

Guidelines

In converting uplinks to downlinks and downlinks to uplinks, consider the following guidelines.

Subject	Guideline
Decommissioning nodes with port profiles	If a decommissioned node has the Port Profile feature deployed on it, the port conversions are not removed even after decommissioning the node. It is necessary to manually delete the configurations after decommission, for the ports to return to the default state. To do this, log onto the switch, run the **setup-clean-config.sh** script, and wait for it to run. Then, enter the **reload** command.
FIPS	When you enable or disable Federal Information Processing Standards (FIPS) on a Cisco ACI fabric, you must reload each of the switches in the fabric for the change to take effect. The configured scale profile setting is lost when you issue the first reload after changing the FIPS configuration. The switch remains operational, but it uses the default scale profile. This issue does not happen on subsequent reloads if the FIPS configuration has not changed. FIPS is supported on Cisco NX-OS release 13.1(1) or later. If you must downgrade the firmware from a release that supports FIPS to a release that does not support FIPS, you must first disable FIPS on the Cisco ACI fabric and reload all the switches in the fabric for the FIPS configuration change.

Subject	Guideline
Maximum uplink port limit	When the maximum uplink port limit is reached and ports 25 and 27 are converted from uplink to downlink and back to uplink on Cisco 93180LC-EX switches: On Cisco 93180LC-EX Switches, ports 25 and 27 are the native uplink ports. Using the port profile, if you convert port 25 and 27 to downlink ports, ports 29, 30, 31, and 32 are still available as four native uplink ports. Because of the threshold on the number of ports (which is maximum of 12 ports) that can be converted, you can convert 8 more downlink ports to uplink ports. For example, ports 1, 3, 5, 7, 9, 13, 15, 17 are converted to uplink ports and ports 29, 30, 31 and 32 are the 4 native uplink ports (the maximum uplink port limit on Cisco 93180LC-EX switches). When the switch is in this state and if the port profile configuration is deleted on ports 25 and 27, ports 25 and 27 are converted back to uplink ports, but there are already 12 uplink ports on the switch (as mentioned earlier). To accommodate ports 25 and 27 as uplink ports, 2 random ports from the port range 1, 3, 5, 7, 9, 13, 15, 17 are denied the uplink conversion and this situation cannot be controlled by the user. Therefore, it is mandatory to clear all the faults before reloading the leaf node to avoid any unexpected behavior regarding the port type. It should be noted that if a node is reloaded without clearing the port profile faults, especially when there is a fault related to limit-exceed, the port might not be in an expected operational state.

Breakout Limitations

Switch	Releases	Limitations
N9K-C9332PQ	Cisco APIC 2.2 (1) and later	• 40Gb dynamic breakouts into 4X10Gb ports are supported. • Ports 13 and 14 do not support breakouts. • Port profiles and breakouts are not supported on the same port.
N9K-C93180LC-EX	Cisco APIC 3.1(1) and later	• 40Gb and 100Gb dynamic breakouts are supported on ports 1 through 24 on odd numbered ports. • When the top ports (odd ports) are broken out, then the bottom ports (even ports) are error disabled. • Port profiles and breakouts are not supported on the same port.

Switch	Releases	Limitations
N9K-C9336C-FX2	Cisco APIC 3.1(2) and later	• 40Gb and 100Gb dynamic breakouts are supported on ports 1 through 30. • Port profiles and breakouts are not supported on the same port.

N9K-C93180YC-FX	Cisco APIC 3.2(1) and later	• 40Gb and 100Gb dynamic breakouts are supported on ports 49 though 52, when they are on profiled QSFP ports. To use them for dynamic breakout, perform the following steps:
		  • Convert ports 49-52 to front panel ports (downlinks).
		  • Perform a port-profile reload, using one of the following methods:
		  • In the Cisco APIC GUI, navigate to **Fabric** > **Inventory** > **Pod** > **Leaf**, right-click **Chassis** and choose **Reload**.
		  • In the iBash CLI, enter the **reload** command.
		  • Apply breakouts on the profiled ports 49-52.
		• Ports 53 and 54 do not support either port profiles or breakouts.

Port Profile Configuration Summary

The following table summarizes supported uplinks and downlinks for the switches that support port profile conversions from Uplink to Downlink and Downlink to Uplink.

Switch Model	Default Links	Max Uplinks (Fabric Ports)	Max Downlinks (Server Ports)	Release Supported
N9K-C9348GC-FXP	48 x 100M/1G BASE-T downlinks 4 x 10/25-Gbps SFP28 downlinks 2 x 40/100-Gbps QSFP28 uplinks	48 x 100M/1G BASE-T downlinks 4 x 10/25-Gbps SFP28 uplinks 2 x 40/100-Gbps QSFP28 uplinks	Same as default port configuration	3.1(1i)

Switch Model	Default Links	Max Uplinks (Fabric Ports)	Max Downlinks (Server Ports)	Release Supported
N9K-C93180LC-EX	24 x 40-Gbps QSFP28 downlinks(1-24) 2 x 40/100-Gbps QSFP28 uplinks(25, 27) 4 x 40/100-Gbps QSFP28 uplinks(29-32) Or 12 x 100-Gbps QSFP28 downlinks(odd number from 1-24) 2 x 40/100-Gbps QSFP28 uplinks(25, 27) 4 x 40/100-Gbps QSFP28 uplinks(29-32)	18 x 40-Gbps QSFP28 downlinks (from 1-24) 6 x 40-Gbps QSFP28 uplinks(from 1-24) 2 x 40/100-Gbps QSFP28 uplinks(25, 27) 4 x 40/100-Gbps QSFP28 uplinks(29-32) Or 6 x 100-Gbps QSFP28 downlinks(odd number from 1-24) 6 x 100-Gbps QSFP28 uplinks(odd number from 1-24) 2 x 40/100-Gbps QSFP28 uplinks(25, 27) 4 x 40/100-Gbps QSFP28 uplinks(29-32)	24 x 40-Gbps QSFP28 downlinks(1-24) 2 x 40/100-Gbps QSFP28 downlinks(25, 27) 4 x 40/100-Gbps QSFP28 uplinks(29-32) Or 12 x 100-Gbps QSFP28 downlinks(odd number from 1-24) 2 x 40/100-Gbps QSFP28 downlinks (25, 27) 4 x 40/100-Gbps QSFP28 uplinks(29-32)	3.1(1i)
N9K-C93180YC-EX N9K-C93180YC-FX	48 x 10/25-Gbps fiber downlinks 6 x 40/100-Gbps QSFP28 uplinks	Same as default port configuration	48 x 10/25-Gbps fiber downlinks 4 x 40/100-Gbps QSFP28 downlinks 2 x 40/100-Gbps QSFP28 uplinks	3.1(1i)
		48 x 10/25-Gbps fiber uplinks 6 x 40/100-Gbps QSFP28 uplinks		4.0(1)
N9K-C93108TC-EX N9K-C93108TC-FX	48 x 10GBASE-T downlinks 6 x 40/100-Gbps QSFP28 uplinks	Same as default port configuration	48 x 10/25-Gbps fiber downlinks 4 x 40/100-Gbps QSFP28 downlinks 2 x 40/100-Gbps QSFP28 uplinks	3.1

Switch Model	Default Links	Max Uplinks (Fabric Ports)	Max Downlinks (Server Ports)	Release Supported
N9K-C9336C-FX2	30 x 40/100-Gbps QSFP28 downlinks 6 x 40/100-Gbps QSFP28 uplinks	18 x 40/100-Gbps QSFP28 downlinks 18 x 40/100-Gbps QSFP28 uplinks	Same as default port configuration	3.2(1i)
		18 x 40/100-Gbps QSFP28 downlinks 18 x 40/100-Gbps QSFP28 uplinks	34 x 40/100-Gbps QSFP28 downlinks 2 x 40/100-Gbps QSFP28 uplinks	3.2(3i)
		36 x 40/100-Gbps QSFP28 uplinks	34 x 40/100-Gbps QSFP28 downlinks 2 x 40/100-Gbps QSFP28 uplinks	4.1
N9K-93240YC-FX2	48 x 10/25-Gbps fiber downlinks 12 x 40/100-Gbps QSFP28 uplinks	Same as default port configuration	48 x 10/25-Gbps fiber downlinks 10 x 40/100-Gbps QSFP28 downlinks 2 x 40/100-Gbps QSFP28 uplinks	4.0(1)
		48 x 10/25-Gbps fiber uplinks 12 x 40/100-Gbps QSFP28 uplinks		4.1
N9K-C93216TC-FX2	96 x 10G BASE-T downlinks 12 x 40/100-Gbps QSFP28 uplinks	Same as default port configuration	96 x 10G BASE-T downlinks 10 x 40/100-Gbps QSFP28 downlinks 2 x 40/100-Gbps QSFP28 uplinks	4.1.2
N9K-C93360YC-FX2	96 x 10/25-Gbps SFP28 downlinks 12 x 40/100-Gbps QSFP28 uplinks	44 x 10/25Gbps SFP28 downlinks 52 x 10/25Gbps SFP28 uplinks 12 x 40/100Gbps QSFP28 uplinks	96 x 10/25-Gbps SFP28 downlinks 10 x 40/100-Gbps QSFP28 downlinks 2 x 40/100-Gbps QSFP28 uplinks	4.1.2

Configuring a Port Profile Using the GUI

This procedure explains how to configure a port profile, which determines the port type: uplink or downlink.

Before you begin

- The ACI fabric is installed, APIC controllers are online, and the APIC cluster is formed and healthy.

- An APIC fabric administrator account is available that will enable creating or modifying the necessary fabric infrastructure configurations.
- The target leaf switches are registered in the ACI fabric and available.

Procedure

Step 1 From the **Fabric** menu, select **Inventory**.

Step 2 In the left navigation pane of the **Inventory** screen, select **Topology**.

Step 3 Under **Topology** tab, select the **Interface** tab in the right navigation pane.

Step 4 Select the mode as **Configuration**.

Step 5 Add a leaf switch by clicking on the + icon (**Add Switches**) in the table menu.

Step 6 In the **Add Switches** table, select the **Switch ID** and click **Add Selected**.

When you select the port, the available option is highlighted.

Step 7 Select the ports and choose the new port type as **Uplink** or **Downlink**.

The last two ports are reserved for uplink. These cannot be converted to downlink ports.

Step 8 After clicking uplink or downlink, click **Submit** (reload the switch on your own later) or **Submit and Reload Switch**.

Note Reload the switch for the change in the uplink or downlink configuration to take effect.

Configuring a Port Profile Using the NX-OS Style CLI

To configure a port profile using the NX-OS style CLI, perform the following steps:

Before you begin

- The ACI fabric is installed, APIC controllers are online, and the APIC cluster is formed and healthy.
- An APIC fabric administrator account is available that will enable creating or modifying the necessary fabric infrastructure configurations.
- The target leaf switches are registered in the ACI fabric and available.

Procedure

Step 1 **configure**

Enters global configuration mode.

Example:

```
apic1# configure
```

Step 2 **leaf** *node-id*

Specifies the leaf or leaf switches to be configured.

Example:

```
apic1(config)# leaf 102
```

Step 3 **interface** *type*

Specifies the interface that you are configuring. You can specify the interface type and identity. For an Ethernet port, use ethernet *slot* / *port*.

Example:

```
apic1(config-leaf)# interface ethernet 1/2
```

Step 4 **port-direction** {**uplink** | **downlink**}

Determines the port direction or changes it. This example configures the port to be a downlink.

Note On the N9K-C9336C-FX switch, changing a port from uplink to downlink is not supported.

Example:

```
apic1(config-leaf-if)# port-direction downlink
```

Step 5 Log on to the leaf switch where the port is located and enter the **reload** command.

Configuring a Port Profile Using the REST API

Before you begin

- The ACI fabric is installed, APIC controllers are online, and the APIC cluster is formed and healthy.
- An APIC fabric administrator account is available that will enable creating or modifying the necessary fabric infrastructure configurations.
- The target leaf switches are registered in the ACI fabric and available.

Procedure

Step 1 To create a port profile that converts a downlink to an uplink, send a post with XML such as the following:

```
<!-- /api/node/mo/uni/infra/prtdirec.xml -->
<infraRsPortDirection tDn="topology/pod-1/paths-106/pathep-[eth1/7]" direc="UpLink" />
```

Step 2 To create a port profile that converts an uplink to a downlink, send a post with XML such as the following:

Example:

```
<!-- /api/node/mo/uni/infra/prtdirec.xml -->
<infraRsPortDirection tDn="topology/pod-1/paths-106/pathep-[eth1/52]" direc="DownLink" />
```

Verifying Port Profile Configuration and Conversion Using the NX-OS Style CLI

You can verify the configuration and the conversion of the ports using the **show interface brief** CLI command.

Note Port profile can be deployed only on the top ports of a Cisco N9K-C93180LC-EX switch, for example, 1, 3, 5, 7, 9, 11, 13, 15, 17, 19, 21, and 23. When the top port is converted using the port profile, the bottom ports are hardware disabled. For example, if Eth 1/1 is converted using the port profile, Eth 1/2 is hardware disabled.

Procedure

Step 1 This example displays the output for converting an uplink port to downlink port. Before converting an uplink port to downlink port, the output is displayed in the example. The keyword **routed** denotes the port as uplink port.

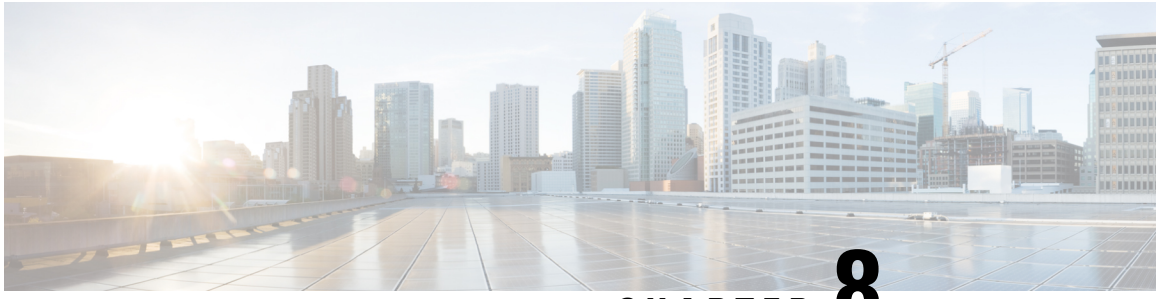
Example:

```
switch# show interface brief
<snip>
Eth1/49          --      eth  routed  down   sfp-missing              100G(D)   --
Eth1/50          --      eth  routed  down   sfp-missing              100G(D)   --
<snip>
```

Step 2 After configuring the port profile and reloading the switch, the output is displayed in the example. The keyword **trunk** denotes the port as downlink port.

Example:

```
switch# show interface brief
<snip>
Eth1/49          0       eth  trunk   down   sfp-missing              100G(D)   --
Eth1/50          0       eth  trunk   down   sfp-missing              100G(D)   --
<snip>
```



CHAPTER 8

FCoE Connections

This chapter contains the following sections:

Supporting Fibre Channel over Ethernet Traffic on the ACI Fabric

Cisco ACI enables you to configure and manage support for Fibre Channel over Ethernet (FCoE) traffic on the ACI fabric.

FCoE is a protocol that encapsulates Fibre Channel (FC) packets within Ethernet packets, thus enabling storage traffic to move seamlessly between a Fibre Channel SAN and an Ethernet network.

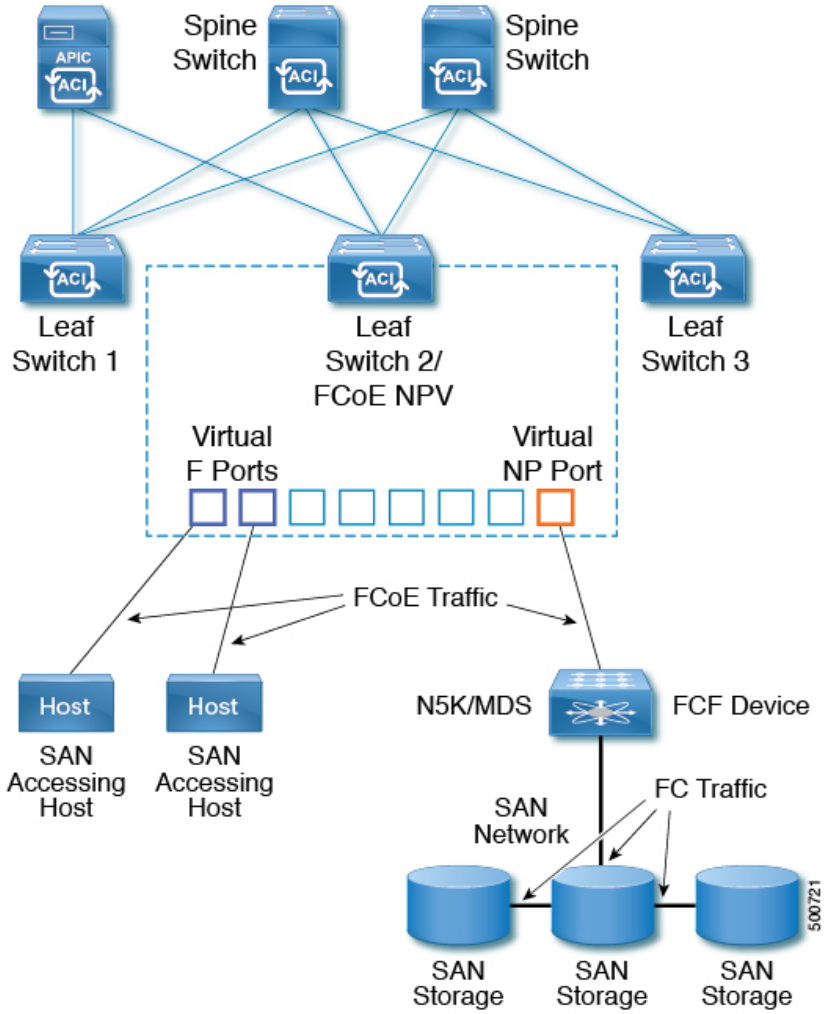
A typical implementation of FCoE protocol support on the ACI fabric enables hosts located on the Ethernet-based ACI fabric to communicate with SAN storage devices located on an FC network. The hosts are connecting through virtual F ports deployed on an ACI leaf switch. The SAN storage devices and FC network are connected through a Fibre Channel Forwarding (FCF) bridge to the ACI fabric through a virtual NP port, deployed on the same ACI leaf switch as is the virtual F port. Virtual NP ports and virtual F ports are also referred to generically as virtual Fibre Channel (vFC) ports.

Note In the FCoE topology, the role of the ACI leaf switch is to provide a path for FCoE traffic between the locally connected SAN hosts and a locally connected FCF device. The leaf switch does not perform local switching between SAN hosts, and the FCoE traffic is not forwarded to a spine switch.

Topology Supporting FCoE Traffic Through ACI

The topology of a typical configuration supporting FCoE traffic over the ACI fabric consists of the following components:

Figure 29: ACI Topology Supporting FCoE Traffic



- One or more ACI leaf switches configured through FC SAN policies to function as an NPV backbone.
- Selected interfaces on the NPV-configured leaf switches configured to function as virtual F ports, which accommodate FCoE traffic to and from hosts running SAN management or SAN-consuming applications.
- Selected interfaces on the NPV-configured leaf switches configured to function as virtual NP ports, which accommodate FCoE traffic to and from a Fibre Channel Forwarding (FCF) bridge.

The FCF bridge receives FC traffic from fibre channel links typically connecting SAN storage devices and encapsulates the FC packets into FCoE frames for transmission over the ACI fabric to the SAN management or SAN Data-consuming hosts. It receives FCoE traffic and repackages it back to FC for transmission over the fibre channel network.

Note In the above ACI topology, FCoE traffic support requires direct connections between the hosts and virtual F ports and direct connections between the FCF device and the virtual NP port.

APIC servers enable an operator to configure and monitor the FCoE traffic through the APIC GUI, the APIC NX-OS style CLI, or through application calls to the APIC REST API.

Topology Supporting FCoE Initialization

In order for FCoE traffic flow to take place as described, you must also set up separate VLAN connectivity over which SAN Hosts broadcast FCoE Initialization protocol (FIP) packets to discover the interfaces enabled as F ports.

vFC Interface Configuration Rules

Whether you set up the vFC network and EPG deployment through the APIC GUI, NX-OS style CLI, or the REST API, the following general rules apply across platforms:

- F port mode is the default mode for vFC ports. NP port mode must be specifically configured in the Interface policies.
- The load balancing default mode is for leaf-switch or interface level vFC configuration is src-dst-ox-id.
- One VSAN assignment per bridge domain is supported.
- The allocation mode for VSAN pools and VLAN pools must always be static.
- vFC ports require association with a VSAN domain (also called Fibre Channel domain) that contains VSANs mapped to VLANs.

Fibre Channel over Ethernet Guidelines and Limitations

The VLAN used for FCoE should have vlanScope set to `Global`. Setting vlanScope to `portLocal` is not supported for FCoE. The value is set using the Layer 2 interface policy (l2IfPol).

Fibre Channel over Ethernet Supported Hardware

FCoE is supported on the following switches:

- N9K-C93180LC-EX (When 40 Gigabit Ethernet (GE) ports are enabled as FCoE F or NP ports, they cannot be enabled for 40GE port breakout. FCoE is not supported on breakout ports.)
- N9K-C93180YC-EX
- N9K-C93180LC-EX (FCoE support on FEX ports)
- N9K-C93180YC-FX (FCoE support on FEX ports, 40G ports (1/49-54), and 4x10G breakout ports)

FCoE is supported on the following Nexus FEX devices:

- N2K-C2348UPQ-10GE
- N2K-C2348TQ-10GE
- N2K-C2232PP-10GE
- N2K-B22DELL-P

- N2K-B22HP-P
- N2K-B22IBM-P
- N2K-B22DELL-P-FI

Configuring FCoE Using the APIC GUI

FCoE GUI Configuration

FCoE Policy, Profile, and Domain Configurations

You can use the APIC GUI under the Fabric Access Policies tab to configure policies, policy groups, and profiles to enable customized and scaled-out deployment and assignment of FCoE supporting F and NP ports on your ACI leaf switches. Then, under the APIC the Tenant tab, you can configure EPG access to those ports.

Policies and Policy Groups

APIC policies and policy groups you create or configure for FCoE support include the following:

Access Switch Policy Group

The combination of switch-level policies that support FCoE traffic through ACI leaf switches.

You can associate this policy group with a leaf profile to enable FCoE support on designated ACI leaf switches.

This policy group consists of the following policies:

- **Fibre Channel SAN Policy**

  Specifies the EDTOV, RATOV, and MAC Address prefix (also called the FC map) values used by the NPV leaf.

- **Fibre Channel Node Policy**

  Specifies the load balance options and FIP keep alive intervals that apply to FCoE traffic associated with this switch policy group.

Interface Policy Groups

The combination of interface-level policies that support FCoE traffic through interfaces on ACI leaf switches.

You can associate this policy group with an FCoE supportive interface profile to enable FCoE support on designated interfaces.

You configure two interface policy groups: One policy group for F ports, and one policy group for NP ports.

The following policies in the interface policy group apply to FCoE enablement and traffic:

- **Priority Flow Control Policy**

Specifies the state of priority flow control (PFC) on the interfaces to which this policy group is applied.

This policy specifies under what circumstances QoS-level priority flow control will be applied to FCoE traffic.

- **Fibre Channel Interface Policy**

  Specifies whether the interfaces to which this policy group is applied are to be configured as F ports or NP ports.

- **Slow Drain Policy**

  Specifies the policy for handling FCoE packets that are causing traffic congestion on the ACI Fabric.

Global Policies

The APIC global policies whose settings can affect the performance characteristics of FCoE traffic on the ACI fabric.

The Global **QOS Class Policies** for **Level1**, **Level2**, **Level4**, **Level5**, or **Level6** connections, contain the following settings that affect FCoE traffic on the ACI fabric:

- **PFC Admin State must be set to Auto**

  Specifies whether to enable priority flow control to this level of FCoE traffic (default value is false).

- **No Drop COS**

  Specifies whether to enable a no-drop policy for this level of FCoE traffic designated with a certain Class of Service (CoS) level.

  Note: QoS level enabled for PFC and FCoE no-drop must match with the Priority Group ID enabled for PFC on CNA.

  Note: Only one QoS level can be enabled for no-drop and PFC, and the same QoS level must be associated with FCoE EPGs.

- QoS Class—Priority flow control requires that CoS levels be globally enabled for the fabric and assigned to the profiles of applications that generate FCoE traffic.

  CoS Preservation must also be enabled—Navigate to **Fabric** > **Access Policies** > **Policies** > **Global** > **QoS Class** and enable **Preserve COS Dot1p Preserve**.

Note Some legacy CNAs may require the **Level2** Global QoS Policy to be used as the **No Drop** PFC, FCoE (Fibre Channel over Ethernet) QoS Policy. If your Converged Network Adapters (CNAs) are not logging into the fabric, and you have noticed that no FCoE Initiation Protocol (FIP) frames are being sent by the CNAs, try enabling **Level2** as the FCoE QoS policy. The **Level2** policy must be attached to the FCoE EPGs in use and only one QoS level can be enabled for PFC no-drop.

Profiles

APIC profiles that you can create or configure for FCoE support include the following:

Leaf Profile

Specifies the ACI Fabric leaf switches on which to configure support of FCoE traffic.

The combination of policies contained in the access switch policy group can be applied to the leaf switches included in this profile.

Interface Profiles

Specifies a set of interfaces on which to deploy F Ports or NP Ports.

You configure at least two leaf interface profiles: One interface profile for F ports, and one interface profile for NP ports.

The combination of policies contained in the interface policy group for F ports can be applied to the set of interfaces included in the interface profile for F ports.

The combination of policies contained in the interface policy group for NP ports can be applied to the set of interfaces included in the interface profile for NP ports.

Attached Entity Profile

Binds the interface policy group settings with the Fibre Channel domain mapping.

Domains

Domains that you create of configure for FCoE support include the following:

Physical Domain

A virtual domain created to support LANs for FCoE VLAN Discovery. The Physical domain will specify the VLAN pool to support FCoE VLAN discovery.

Fibre Channel Domain

A virtual domain created to support virtual SANs for FCoE connections.

A Fibre Channel domain specifies a VSAN pool, VLAN pool and the VSAN Attribute over which the FCoE traffic is carried.

- **VSAN pool** - a set of virtual SANs which you associate with existing VLANs. Individual VSANs can be assigned to associated FCoE-enabled interfaces in the same way that VLANs can be assigned to those interfaces for Ethernet connectivity.
- **VLAN pool** - the set of VLANs available to be associated with individual VSANs.
- **VSAN Attribute** - The mapping of a VSAN to a VLAN.

Tenant Entities

Under the Tenant tab, you configure bridge domain and EPG entities to access the FCoE ports and exchange the FCoE traffic.

The entities include the following:

Bridge Domain (configured for FCoE support)

A bridge domain created and configured under a tenant to carry FCoE traffic for applications that use FCoE connections.

Application EPG

The EPG under the same tenant to be associated with the FCoE bridge domain.

Fibre Channel Path

Specifies the interfaces enabled as FCoE F ports or NP ports to be associated with the selected EPG. After you associate the Fibre Channel path with an EPG the FCoE interface is deployed in the specified VSAN.

Deploying FCoE vFC Ports Using the APIC GUI

The APIC GUI enables you to create customized node policy groups, leaf profiles, interface policy groups, interface profiles, and virtual SAN domains that system administrators can re-use to ensure that all interfaces they designate as F ports or NP ports to handle FCoE traffic have consistent FCoE-related policies applied.

Before you begin

- The ACI fabric is installed.
- If you deploy over a port channel (PC) topology, the port channel is set up as described in ACI Leaf Switch Port Channel Configuration Using the GUI, on page 80.
- If you deploy over a virtual port channel (vPC) topology, the vPC is set up as described in ACI Leaf Switch Virtual Port Channel Configuration Using the GUI, on page 99.

Procedure

Step 1 Create an FCoE supportive switch policy group to specify and combine all the leaf switch policies that support FCoE configuration.

This policy group will be applied to the leaf switches that you want to serve as NPV hosts.

a) In the APIC GUI, starting on the APIC menu bar, click **Fabric** > **Access Policies** > **Switches** > **Leaf Switches** > **Policy Groups**.
b) Right-click **Policy Groups** and click **Create Access Switch Policy Group**.
c) In the **Create Access Switch Policy Group** dialog, specify the settings described below and then click **Submit**.

Policy	Description
Name	Identifies the switch policy group. Enter a name that indicates the FCoE supportive function of this switch policy group. For example, **`fcoe_switch_policy_grp`**.
Fibre Channel SAN Policy	Specifies the following SAN Policy values: • FC Protocol EDTOV (default: 2000) • FC Protocol RATOV (default : 10000) • MAC address prefix (also called FC map) used by the leaf switch. This value should match the value of the peer device connected on the same port. Typically the default value OE:FC:00 is used. Click the drop-down option box. • To use the default EDTOV, RATOV, and MAC address prefix values, click **default**.

Policy	Description
	• To use the value specified in an existing policy, click that policy. • To create a new policy to specify a new customized MAC address prefix, click **Create Fibre Channel SAN Policy** and follow the prompts.

Step 2 Create a leaf profile for leaf switches to support FCoE traffic.

This profile specifies a switch or set of leaf switches to assign the switch policy group that was configured in the previous step. This association enables that set of switches to support FCoE traffic with pre-defined policy settings.

a) Starting at the APIC menu bar, click **Fabric** > **Access Policies** > **Switches** > **Leaf Switches** > **Profiles**
b) Right-click **Leaf Profiles**, then click **Create Leaf Profile**.
c) In the **Create Leaf Profile** dialog create and name the leaf profile (for example: NPV-1)
d) Also in the **Create Leaf Profile** dialog, locate the **Leaf Selectors** table, click +to create a new table row and specify the leaf switches to serve as NPV devices.
e) In the new table row choose a leaf name, blocks, and assign the switch policy group that you created in the previous step.
f) Click **Next** and then click **Finish**.

Step 3 Create at least two FCoE-supportive interface policy groups: one to combine all policies that support FCoE F port interfaces, and one to combine all policies that support FCoE NP port interfaces.

These interface policy groups are to be applied to the interface profiles that are applied to interfaces that are to serve as F ports and NP ports.

a) On the APIC menu bar, click **Fabric** > **Access Policies** > **Interfaces** > **Leaf Interfaces** > **Policy Groups**.
b) Right-click **Policy Groups**, then, depending on how port access is configured, click one of the following options: **Create Leaf Access Port Policy Group**, **Create PC Interface Port Policy**, or **Create vPC Interface Port Policy Group**.

Note
- If you deploy over a PC interface, view ACI Leaf Switch Port Channel Configuration Using the GUI, on page 80 for additional information.
- If you deploy over a vPC interface, view ACI Leaf Switch Virtual Port Channel Configuration Using the GUI, on page 99 for additional information.

c) In the policy group dialog, specify for inclusion the Fibre Channel Interface policy, the slow drain policy, and the priority flow control policy you configure.

Policy	Description
Name	Name of this policy group. Enter a name that indicates the FCoE supportive function of this Leaf Access Port Policy Group and the port type, (F or NP) that it is intended to support, for example: **`fcoe_f_port_policy`** or **`fcoe_np_port_policy`**.
Priority Flow Control Policy	Specifies the state of the Priority Flow Control (PFC) on the interfaces to which this policy group is applied. Options include the following:

Policy	Description
	• **Auto** (the default value) Enables priority flow control (PFC) on local port on the no-drop CoS as configured, on the condition that values advertised by the DCBX and negotiated with the peer succeed. Failure causes priority flow control to be disabled on the no-drop CoS. • **Off** disables FCoE priority flow control on the local port under all circumstances. • **On** enables FCoE PFC on the local port under all circumstances. Click the drop-down option box: • To use the default values, click **default**. • To use the value specified in an existing policy, click that policy. • To create a new policy specifying different values, click **Create Priority Flow Control Policy** and follow the prompts. **Note** PFC requires that Class of Service (CoS) levels be globally enabled for the fabric and assigned to the profiles of applications that generate FCoE traffic. Also CoS Preservation must be enabled. To enable it, navigate to **Fabric** > **Access Policies** > **Policies** > **Global** > **QoS Class** and enable **Preserve COS Dot1p Preserve**.
Slow Drain Policy	Specifies how to handle FCoE packets that are causing traffic congestion on the ACI fabric. Options include the following: • Congestion Clear Action (default: disabled) Action to be taken during FCoE traffic congestion. Options include: • Err - disable - Disable the port. • Log - Record congestion in the Event Log. • Disabled- Take no action. • Congestion Detect Multiplier (default: 10) The number of pause frames received on a port that triggers a congestion clear action to address FCoE traffic congestion. • Flush Admin State • Enabled - Flush the buffer. • Disabled - Don't flush the buffer. • Flush Timeout (default: 500 milliseconds) Threshold in milliseconds to trigger buffer flush drop during congestion. • To use the default values, click **default**. • To use the value specified in an existing policy, click that policy.

Policy	Description
	• To create a new policy specifying different values, click **Create Slow Drain Policy** and follow the prompts.

Step 4 Create at least two interface profiles: one profile to support F port connections, one profile to support NP port connections, and optional additional profiles to be associated with additional port policy variations.

a) Starting at the APIC bar menu click **Fabric** > **Access Policies** > **Interfaces** > **Leaf Interfaces** > **Profiles**.

b) Right-click **Profiles** and choose **Create Leaf Interface Profile**.

c) In the **Create Leaf Interface Profile** dialog, enter a descriptive name for the profile, for example,**`FCoE_F_port_Interface_profile-1`**.

d) Locate the **Interface Selectors** table and click **+** to display the **Create Access Port Selector** dialog. This dialog enables you to display a range of interfaces and apply settings to the fields described in the following table.

Option	Description
Name	A descriptive name for this port selector.
Interface IDs	Specifies the set of interfaces to which this range applies. • To include all interfaces in the switch, choose **All**. • To include an individual interface in this range, specify single Interface ID, for example: **`1/20`**. • To include a range of interfaces in this range, enter the lower and upper values separated by a dash, for example: **`1/10 - 1/15`**. **Note** Specify separate, non-overlapping ranges of interfaces when configuring interface profiles for F ports and an NP port.
Interface Policy Group	The name of either the F port interface policy group or the NP port policy group that you configured in the previous step. • To designate the interfaces included in this profile as F ports, choose the interface policy group that you configured for F ports. • To designate the interfaces included in the profile as NP ports, choose the interface policy group that you configured for NP ports.

Step 5 Click **Submit**. Repeat the previous step so that you at least have interface profiles for both F ports and an NP port.

Step 6 Configure whether to apply global QoS policies to FCoE traffic.

You can specify different QoS policies to different levels (1, 2, 4, 5, or 6) of FCoE traffic.

a) Starting at the APIC bar menu click **Fabric** > **Access Policies** > **Policies** > **Global** > **QoS Class** and enable the **Preserve CoS** flag in the **QoS Class** pane.

b) In the **QoS Class - Level 1**, **QoS Class - Level 2**, **QoS Class - Level 4**, **QoS Class - Level 5**, or **QoS Class - Level 6** dialog, edit the following fields to specify the PFC and no-drop CoS. Then click **Submit.**

Note Only 1 Level can be configured for PFC and no-drop CoS.

Policy	Description
PFC Admin State	Whether to enable priority flow control to this level of FCoE traffic (default value is false). Enabling priority flow control sets the **Congestion Algorithm** for this level of FCoE traffic to **no-drop**.
No-Drop-CoS	The CoS level to impose no drop FCoE packet handling even in the case of FCoE traffic congestion.

Step 7 Define a Fibre Channel domain. Create a set of virtual SANs (VSANs) and map them to set of existing VLANs.

a) Starting at the APIC bar menu click **Fabric** > **Access Policies** > **Physical and External Domains** > **Fibre Channel Domains**.

b) Right-click **Fibre Channel Domains** and click **Create Fibre Channel Domain**.

c) In the **Fibre Channel Domain** dialog, specify the following settings:

Option	Description/Action
Name	Specifies the name or label you want to assign the VSAN domain you are creating. (For example: vsan-dom2)
VSAN Pool	The pool of VSANs assigned to this domain. • To select an existing VSAN pool, click the drop-down and choose a listed pool. If you want to revise it, click the Edit icon. • To create a VSAN pool, click **Create a VSAN Pool**. If you open the dialog to create a VSAN pool, follow the prompts configure the following: • A **Static** resource allocation method to support FCoE. • a range of VSANs that will be available to assign to FCoE F port interfaces and NP port interfaces. **Note** Minimum range value is 1. Maximum range value is 4078. Configure multiple ranges of VSANs if necessary.
VLAN Pool	The pool of VLANS available to be mapped to by the members of the VSAN pool. A VLAN pool specifies numerical ranges of VLANs you want available to support FCoE connections for this domain. The VLANs in the ranges you specify are available for VSANs to map to them. • To select an existing VLAN pool, click the drop-down and choose a listed pool. If you want to revise it, click the Edit icon. • To create a VLAN pool, click **Create a VLAN Pool**. If you open the dialog to create a VLAN pool, follow the prompts configure the following: • A **Static** resource allocation method to support FCoE. • a range of VLANs that will be available for VSANs to map to.

Option	Description/Action
	Note Minimum range value is 1. Maximum range value is 4094. Configure multiple ranges of VLANs if necessary.
VSAN Attr	The VSAN Attributes map for this domain. The VSAN Attributes map VSANs in the VSAN pool to VLANs in the VLAN pool. • To select an existing VSAN Attributes map, click the drop-down and choose a listed map. If you want to revise it, click the Edit icon. • To create a VSAN Attributes map, click **Create VSAN Attributes**. If you open the dialog to configure the VSAN attributes, follow the prompts configure the following: • The appropriate load balancing option (src-dst-ox-id or src-dst-id). • Mapping of individual VSANs to individual VLANs, for example: vsan-8 to vlan-10 **Note** Only VSANs and VLANs in the ranges you specified for this domain can be mapped to each other.

Step 8 Create an attached entity profile to bind the Fibre Channel domain with the interface policy group.

a) On the APIC menu bar, click **Fabric** > **Access Policies** > **Interfaces** > **Leaf Interfaces** > **Policy Groups** > ***interface_policy_group_name*** .

In this step *interface_policy_group_name* is the interface policy group that you defined in Step 3.

b) In the interface policy group dialog, Click the Attached Entity Profile drop-down and choose an existing Attached Entity Profile or click **Create Attached Entity Profile** to create a new one.

c) In the Attached Entity Profile dialog specify the following settings:

Field	Description
Name	A name for this Attached Entity Profile.
Domains To Be Associated To Interfaces	Lists the domain to be associated with the interface policy group. In this case, choose the Fibre Channel domain you configured in Step 7. Click **Submit**.

Step 9 Associate the leaf profile and the F port and NP port interface profiles.

a) Starting at the APIC menu bar, click **Fabric** > **Access Policies** > **Switches** > **Leaf Switches** > **Profiles** then click the name of the leaf profile you configured in Step 2.

b) In the **Create Leaf Profile** dialog, locate the **Associated Interface Selector Profiles** table, click +to create a new table row and choose the F port interface profile you created in Step 4.

c) Again on the **Associated Interface Selector Profiles** table, click +to create a new table row and choose the NP port interface profile you created in Step 4.

d) Click **Submit**.

What to do next

After successful deployment of virtual F ports and NP ports to interfaces on the ACI fabric, the next step is for system administrators to enable EPG access and connection over those interfaces.

For more information, see Deploying EPG Access to vFC Ports Using the APIC GUI, on page 145.

Deploying EPG Access to vFC Ports Using the APIC GUI

After you have configured ACI fabric entities to support FCoE traffic and F port and NP port functioning of designated interfaces, your next step is to configure EPG access to those ports.

Before you begin

- The ACI fabric is installed.
- A Fibre Channel Forwarding (FCF) switch, connected to a FC network (for example, SAN storage), is physically attached by Ethernet to an ACI leaf switch port.
- A host application that needs to access the FC network is physically attached by Ethernet to a port on the same ACI leaf switch.
- Leaf policy groups, leaf profiles, interface policy groups, interface profiles, and Fibre Channel domains have all been configured to support FCoE traffic.

Procedure

Step 1 Under an appropriate tenant configure an existing bridge domain to support FCoE or create a bridge domain to support FCoE.

Option:	Actions
To configure an existing bridge domain for FCoE	**a.** Click **Tenant** > ***tenant_name*** > **Networking** > **Bridge Domains** > ***bridge_domain_name*** . **b.** In the **Type** field of the bridge domain's **Properties** panel, click **fc**. **c.** Click **Submit**.
To create a new bridge domain for FCoE	**a.** Click **Tenant** > ***tenant_name*** > **Networking** > **Bridge Domains** > **Actions** > **Create a Bridge Domain**. **b.** In the **Name** field of the **Specify Bridge Domain for the VRF** dialog, enter a bridge domain name. **c.** In the **Type** field of **Specify Bridge Domain for the VRF dialog**, click **fc**. **d.** In VRF field select a VRF from the drop-down or click **Create VRF** to create and configure a new VRF.

Option:	Actions
	e. Finish the bridge domain configuration. **f.** Click **Submit**.

Step 2 Under the same tenant, configure an existing EPG or create a new EPG to associate with the FCoE-configured bridge domain.

Option:	Actions
To associate an existing EPG	**a.** Click **Tenant** > ***<tenant_name>*** > **Application Profiles** > ***<application_profile_name>*** > **Application EPGs** > ***<epg_name>*** . **b.** In the **QoS class** field choose the quality of service (**Level1**, **Level2**, **Level4**, **Level5**, or **Level6**) to assign to traffic generated by this EPG. If you configured one of the QoS levels for priority-flow control no-drop congestion handling and you want FCoE traffic handled with no-dropped packet priority, assign that QoS level to this EPG. **c.** In the **Bridge Domain** field of the EPG's **Properties** panel, click the drop-down list and choose the name of a bridge domain configured for Type: fcoe. **d.** Click **Submit**. **Note** If you change the **Bridge Domain** field, you must wait 30-35 seconds between changes. Changing the Bridge Domain field too rapidly causes vFC interfaces on the NPV Switch to fail and a switch reload must be executed.
To create and associate a new EPG	**a.** Click **Tenant** > ***<tenant_name>*** > **Application Profiles** > ***<application_profile_name>*** > **Application EPGs**. **b.** Right-click **Application EPGs** and click **Create Application EPG**. **c.** In the **QoS class** field choose the quality of service (**Level1**, **Level2**, **Level4**, **Level5**, or **Level6**) to assign to traffic generated by this EPG. If you configured one of the QoS levels for priority-flow control no-drop congestion handling and you want FCoE traffic handled with no-dropped packet priority, assign that QoS level to this EPG. **d.** In the **Bridge Domain** field of the **Specify the EPG Identity** dialog, click the drop-down list and choose the name of a bridge domain configured for Type: fcoe. **Note** If you change the **Bridge Domain** field, you must wait 30-35 seconds between changes. Changing the Bridge Domain field too rapidly causes vFC interfaces on the NPV Switch to fail and a switch reload must be executed. **e.** Finish the bridge domain configuration. **f.** Click **Finish**.

Step 3 Add a Fibre Channel Domain association with the EPG.

a) Click **Tenant** > ***<tenant_name>*** > **Application Profiles** > ***<application_profile_name>*** > **Application EPGs** > ***<epg_name>*** > **Domains (VMs and Bare Metal)**.
b) Right-click **Domains (VMs and Bare Metal)** and click **Add Fibre Channel Domain Association**.
c) In the **Add Fibre Channel Domain Association** dialog, locate the **Fibre Channel Domain Profile Field.**
d) Click the drop-down list and choose the name of the Fibre Channel domain that you previously configured.
e) Click **Submit**.

Step 4 Under the associated EPG define a Fibre Channel path.

The Fibre Channel path specifies the interfaces enabled as FCoE F ports or NP ports to be associated with the selected EPG.

a) Click **Tenant** > ***<tenant_name>*** > **Application Profiles** > ***<application_profile_name>*** > **Application EPGs** > ***<epg_name>*** > **Fibre Channel (Paths)**.
b) Right-click **Fibre Channel (Paths)** and click **Deploy Fibre Channel**.
c) In the **Deploy Fibre Channel** dialog configure the following settings:

Option:	Actions
Path Type	The type of interface (Port, Direct Port Channel, or Virtual Port Channel) being accessed for sending and receiving FCoE traffic.
Path	The Node-interface path through which FCoE traffic associated with the selected EPG will flow. Click the drop-down list and choose from the listed interfaces. . **Note** Choose only the interfaces previously configured as F ports or NP ports. Choosing interfaces that you did not configure causes only default values to apply to those interfaces. **Note** To deploy FCoE over FEX, select the FEX ports previously configured.
VSAN	The VSAN which will use the interface selected in the **Path** field. **Note** The specified VSAN must be in the range of VSANs that was designated for the VSAN pool. In most cases, all interfaces that this EPG is configured to access must be assigned the same VSAN, unless you specify a Fibre Channel path over a Virtual Port Channel (VPC) connection. In that case, you can specify two VSANs, one for each leg of the connection.
VSAN Mode	The mode (**Native** or **Regular**) in which the selected VSAN accesses the selected interface. Every interface configured for FCoE support, requires one VSAN and only one VSAN configured for Native mode. Any additional VSANs assigned to the same interface must access it in Regular mode.
Pinning label	(Optional) This option applies only if you are mapping access to an F port and it is necessary to bind this F port with a specific uplink NP port. It associates a pinning label (pinning label 1 or pinning label 2) with a specific NP port. You can then assign that pinning label to the target F port. This association causes the associated NP port to serve in all cases as the uplink port to the target F Port.

Option:	Actions
	Choose a pinning label and associate it with an interface configured as an NP port. This option implements what is also referred to as "traffic-mapping." **Note** The F port and the associated Pinning Label NP port must be on the same Leaf switch.

Step 5 Click **Submit**.

Step 6 Repeat Steps 4 and 5 for every FCoE enabled interface to which you are mapping EPG access.

Step 7 Verify successful deployment, as follows:

a) Click **Fabric** > **Inventory** > ***Pod_name*** > ***leaf_name*** > **Interfaces** > **VFC interfaces**.

The interfaces on which you deployed ports are listed under VFC Interfaces.

What to do next

After you have set up EPG access to the vFC interfaces, the final step is to set up the network supporting the FCoE initialization protocol (FIP), which enables discovery of those interfaces.

For more information, see Deploying the EPG to Support the FCoE Initiation Protocol, on page 148.

Deploying the EPG to Support the FCoE Initiation Protocol

After you have configured FCoE EPG access to your server ports, you must also configure EPG access to support the FCoE Initiation Protocol (FIP).

Before you begin

- The ACI fabric is installed.
- A host application that needs to access the FC network is physically attached by Ethernet to a port on the same ACI leaf switch.
- Leaf policy groups, leaf profiles, interface policy groups, interface profiles, and Fibre Channel domains have all been configured to support FCoE traffic as described in the topic Deploying EPG Access to vFC Ports Using the APIC GUI, on page 145.
- EPG access to the vFC ports is enabled as described in the topic Deploying EPG Access to vFC Ports Using the APIC GUI, on page 145.

Procedure

Step 1 Under the same tenant configure an existing bridge domain to support FIP or create a regular bridge domain to support FIP.

Option:	Actions
To configure an existing bridge domain for FCoE	**a.** Click **Tenant** > ***tenant_name*** > **Networking** > **Bridge Domains** > ***bridge_domain_name*** .

Option:	Actions
	b. In the **Type** field of the bridge domain's **Properties** panel, click **Regular**. **c.** Click **Submit**.
To create a new bridge domain for FCoE	**a.** Click **Tenant** > ***tenant_name*** > **Networking** > **Bridge Domains** > **Actions** > **Create a Bridge Domain**. **b.** In the **Name** field of the **Specify Bridge Domain for the VRF** dialog, enter a bridge domain name. **c.** In the **Type** field of **Specify Bridge Domain for the VRF dialog**, click **Regular**. **d.** In VRF field select a VRF from the drop-down or click **Create VRF** to create and configure a new VRF. **e.** Finish the bridge domain configuration. **f.** Click **Submit**.

Step 2 Under the same tenant, configure an existing EPG or create a new EPG to associate with the regular-type bridge domain.

Option:	Actions
To associate an existing EPG	**a.** Click **Tenant** > ***tenant_name*** > **Application Profiles** > **ap1** > **Application EPGs** > ***epg_name*** . **b.** In the **Bridge Domain** field of the EPG's **Properties** panel, click the drop-down list and choose the name of the regular bridge domain that you just configured to support FIP. **c.** Click **Submit**.
To create and associate a new EPG	**a.** Click **Tenant** > ***tenant_name*** > **Application Profiles** > **ap1** > **Application EPGs**. **b.** Right-click **Application EPGs** and click **Create Application EPG**. **c.** In the **Bridge Domain** field of the **Specify the EPG Identity** dialog, click the drop-down list and choose the name of the regular bridge domain that you just configured to support FIP. **d.** Finish the bridge domain configuration. **e.** Click **Finish**.

Step 3 Add a Physical Domain association with the EPG.

a) Click **Tenant** > ***tenant_name*** > **Application Profiles** > **ap1** > **Application EPGs** > ***epg_name*** > **Domains & Bare Metal**.
b) Right-click **Domains & Bare Metal** and click **Add Physical Domain Association**.
c) In the **Add Physical Domain Association** dialog, locate Physical Domain Profile Field.

d) Click the drop-down list and choose the name of the physical domain that contains the LAN that intended for use in FIP support.
e) Click **Submit**.

Step 4 Under the associated EPG define a path.

The path specifies the interfaces enabled as FCoE F ports or NP ports to be associated with the selected EPG.

a) Click **Tenant** > ***tenant_name*** > **Application Profiles** > **ap1** > **Application EPGs** > ***epg_name*** > **Static Ports**.
b) Right-click **Static Ports** and click **Deploy Static EPG on PC, VPC, or Interface**.
c) In the **Path Type** field, specify the port type (Port, Direct Port Channel, or Virtual Port Channel) on which you want to deploy an F mode vFC.
d) In the **Path** field, specify all the paths on which are deployed the F ports.
e) Choose the VLAN Encap that you want to use as your FCoE VLAN discovery and 802.1p(access) as port mode.
f) Click **Submit**.

The FCoE components will begin the discovery process to initiate the operation of the FCoE network.

Undeploying FCoE Connectivity Using the APIC GUI

To undo FCoE enablement of leaf switch interfaces on the ACI fabric, delete the Fibre Channel path and Fibre Channel domain and its elements that you defined in Deploying FCoE vFC Ports Using the APIC GUI, on page 139.

Note If during clean up you delete the Ethernet configuration object (infraHPortS) for a vFC port (for example, in the **Interface Selector** table on the **Leaf Interface Profiles** page of the GUI), the default vFC properties remain associated with that interface. For example it the interface configuration for vFC NP port 1/20 is deleted, that port remains a vFC port but with default F port setting rather than non-default NP port setting applied.

Before you begin

You must know the name of the Fibre Channel path and Fibre Channel domain including its associated VSAN pool, VLAN pool, and VSAN Attributes map that you specified during FCoE deployment.

Procedure

Step 1 Delete the associated Fibre Channel path to undeploy vFC from the port/vsan whose path was specified on this deployment.

This action removes vFC deployment from the port/vsan whose path was specified on this deployment.

a) Click **Tenants** > ***tenant_name*** > **Application Profiles** > ***app_profile_name*** > **Application EPGs** > ***app_epg_name*** > **Fibre Channel (Paths)**. Then right-click the name of the target Fibre Channel path and choose **Delete**.
b) Click **Yes** to confirm the deletion.

Step 2 Delete the VLAN to VSAN map that you configured when you defined the Fibre Channel domain.

This action removes vFC deployment from all the elements defined in the map.

a) Click **Fabric** > **External Access Policies** > **Pools** > **VSAN Attributes**. Then right-click the name of the target map and choose **Delete**.
b) Click **Yes** to confirm the deletion.

Step 3 Delete the VLAN and VSAN pools that you defined when you defined the Fibre Channel domain.

This action eliminates all vFC deployment from the ACI fabric.

a) Click **Fabric** > **External Access Policies** > **Pools** > **VSAN** and then, right-click the name of the target VSAN pool name and choose **Delete**.
b) Click **Yes** to confirm the deletion.
c) Click **Fabric** > **External Access Policies** > **Pools** > **VLAN** then, right-click the target VLAN pool name and choose **Delete**.
d) Click **Yes** to confirm the deletion.

Step 4 Delete the Fibre Channel Domain that contained the VSAN pool, VLAN pool, and Map elements you just deleted.

a) Click **Tenants** > **tenant_name** > **Application Profiles** > **Fibre Channel Domains**. Then right-click the name of the target Fibre Channel Domain and choose **Delete**.
b) Click **Yes** to confirm the deletion.

Step 5 You can delete the tenant/EPG/App and the selectors if you don't need them.

Option	Action
If you want to delete the associated application EPG but save the associated tenant and application profile:	Click **Tenants** > ***tenant_name*** > **Application Profiles** > ***app_profile_name*** > **Application EPGs**, right-click the name of the target application EPG, choose **Delete**, then click **Yes** to confirm deletion.
If you want to delete the associated application profile but save the associated tenant:	Click **Tenants** > ***tenant_name*** > **Application Profiles**, right-click the name of the target application profile, choose **Delete**, then click **Yes** to confirm deletion.
If you want to delete the associated tenant:	Click **Tenants** > , right-click the name of the target tenant, choose **Delete**, then click **Yes** to confirm deletion.

Configuring FCoE Using the NX_OS Style CLI

FCoE NX-OS Style CLI Configuration

Configuring FCoE Connectivity Without Policies or Profiles Using the NX-OS Style CLI

The following sample NX-OS style CLI sequences configure FCoE connectivity for EPG **e1** under tenant **t1** without configuring or applying switch-level and interface-level policies and profiles.

Procedure

	Command or Action	Purpose

Step 1	Under the target tenant configure a bridge domain to support FCoE traffic.
	Example:
	`apic1(config)# tenant t1`
	`apic1(config-tenant)# vrf context v1`
	`apic1(config-tenant-vrf)# exit`
	`apic1(config-tenant)# bridge-domain b1`
	`apic1(config-tenant-bd)# fc`
	`apic1(config-tenant-bd)# vrf member v1`
	`apic1(config-tenant-bd)# exit`
	`apic1(config-tenant)# exit`
		The sample command sequence creates bridge domain **b1** under tenant **t1** configured to support FCoE connectivity.

Step 2	Under the same tenant, associate the target EPG with the FCoE-configured bridge domain.
	Example:
	`apic1(config)# tenant t1`
	`apic1(config-tenant)# application a1`
	`apic1(config-tenant-app)# epg e1`
	`apic1(config-tenant-app-epg)# bridge-domain member b1`
	`apic1(config-tenant-app-epg)# exit`
	`apic1(config-tenant-app)# exit`
	`apic1(config-tenant)# exit`
		The sample command sequence creates EPG **e1** and associates that EPG with the FCoE-configured bridge domain **b1**.

Step 3	Create a VSAN domain, VSAN pools, VLAN pools and VSAN to VLAN mapping.
	Example:
	A
	`apic1(config)# vsan-domain dom1`
	`apic1(config-vsan)# vsan 1-10`
	`apic1(config-vsan)# vlan 1-10`
	`apic1(config-vsan)# fcoe vsan 1 vlan 1 loadbalancing src-dst-ox-id`
	`apic1(config-vsan)# fcoe vsan 2 vlan 2`
	Example:
	B
	`apic1(config)# template vsan-attribute pol1`
	`apic1(config-vsan-attr)#  fcoe vsan 2 vlan 12 loadbalancing src-dst-ox-id`
	`apic1(config-vsan-attr)#  fcoe vsan 3 vlan 13 loadbalancing src-dst-ox-id`
	`apic1(config-vsan-attr)#  exit`
	`apic1(config)#  vsan-domain dom1`
	`apic1(config-vsan)# vsan 1-10`
	`apic1(config-vsan)# vlan 1-10`
	`apic1(config-vsan)# inherit vsan-attribute pol1`
	`apic1(config-vsan)# exit`
		In **Example A**, the sample command sequence creates VSAN domain, **dom1** with VSAN pools and VLAN pools, maps VSAN 1 to VLAN 1 and maps VSAN 2 to VLAN 2
		In **Example B**, an alternate sample command sequence creates a reusable VSAN attribute template **pol1** and then creates VSAN domain **dom1**, which inherits the attributes and mappings from that template.

	Command or Action	Purpose

Step 4	Create the physical domain to support the FCoE Initialization (FIP) process.
	Example:
	`apic1(config)# vlan-domain fipVlanDom`
	`apic1(config-vlan)# vlan 120`
	`apic1(config-vlan)# exit`
		In the example, the command sequence creates a regular VLAN domain, **fipVlanDom**, which includes VLAN **120** to support the FIP process.

Step 5	Under the target tenant configure a regular bridge domain.
	Example:
	`apic1(config)# tenant t1`
	`apic1(config-tenant)# vrf context v2`
	`apic1(config-tenant-vrf)# exit`
	`apic1(config-tenant)# bridge-domain fip-bd`
	`apic1(config-tenant-bd)# vrf member v2`
	`apic1(config-tenant-bd)# exit`
	`apic1(config-tenant)# exit`
		In the example, the command sequence creates bridge domain **fip-bd**.

Step 6	Under the same tenant, associate this EPG with the configured regular bridge domain.
	Example:
	`apic1(config)# tenant t1`
	`apic1(config-tenant)# application a1`
	`apic1(config-tenant-app)# epg epg-fip`
	`apic1(config-tenant-app-epg)# bridge-domain member fip-bd`
	`apic1(config-tenant-app-epg)# exit`
	`apic1(config-tenant-app)# exit`
	`apic1(config-tenant)# exit`
		In the example, the command sequence associates EPG **epg-fip** with bridge domain **fip-bd**.

Step 7	Configure a VFC interface with F mode.
	Example:
	A
	`apic1(config)# leaf 101`
	`apic1(config-leaf)# interface ethernet 1/2`
	`apic1(config-leaf-if)# vlan-domain member fipVlanDom`
	`apic1(config-leaf-if)# switchport trunk native vlan 120 tenant t1 application a1 epg epg-fip`
	`apic1(config-leaf-if)# exit`
	
	`apic1(config-leaf)# exit`
	`apic1(config-leaf)# interface vfc 1/2`
	`apic1(config-leaf-if)# switchport mode f`
	`apic1(config-leaf-if)# vsan-domain member dom1`
	`apic1(config-leaf-if)# switchport vsan 2 tenant t1 application a1 epg e1`
	`apic1(config-leaf-if)# switchport trunk`
		In example **A** the command sequence enables interface **1/2** on leaf switch **101** to function as an **F** port and associates that interface with VSAN domain **dom1**.
		Each of the targeted interfaces must be assigned one (and only one) VSAN in native mode. Each interface may be assigned one or more additional VSANs in regular mode.
		The sample command sequence associates the target interface **1/2** with:
		• VLAN **120** for FIP discovery and associates it with EPG **epg-fip** and application **a1** under tenant **t1**.
		• VSAN **2** as a native VSAN and associates it with EPG **e1** and application **a1** under tenant **t1**.
		• VSAN **3** as a regular VSAN.

	Command or Action	Purpose
	allowed vsan 3 tenant t1 application a1 epg e2 apic1(config-leaf-if)# exit **Example:** **B** apic1(config)# vpc context leaf 101 102 apic1(config-vpc)# interface vpc vpc1 apic1(config-vpc-if)# vlan-domain member vfdom100 apic1(config-vpc-if)# vsan-domain member dom1 apic1(config-vpc-if)# #For FIP discovery apic1(config-vpc-if)# switchport trunk native vlan 120 tenant t1 application a1 epg epg-fip apic1(config-vpc-if)# switchport vsan 2 tenant t1 application a1 epg e1 apic1(config-vpc-if)# exit apic1(config-vpc)# exit apic1(config)# leaf 101-102 apic1(config-leaf)# interface ethernet 1/3 apic1(config-leaf-if)# channel-group vpc1 vpc apic1(config-leaf-if)# exit apic1(config-leaf)# exit **Example:** **C** apic1(config)# leaf 101 apic1(config-leaf)# interface vfc-po pc1 apic1(config-leaf-if)# vsan-domain member dom1 apic1(config-leaf-if)# switchport vsan 2 tenant t1 application a1 epg e1 apic1(config-leaf-if)# exit apic1(config-leaf)# interface ethernet 1/2 apic1(config-leaf-if)# channel-group pc1 apic1(config-leaf-if)# exit apic1(config-leaf)# exit	In example **B**, the command sequence configures a vFC over a vPC with the same VSAN on both the legs. From the CLI you cannot specify different VSANs on each log. The alternate configuration can be carried out in the APIC advanced GUI.
Step 8	Configure a VFC interface with NP mode. **Example:** apic1(config)# leaf 101 apic1(config-leaf)# interface vfc 1/4 apic1(config-leaf-if)# switchport mode np apic1(config-leaf-if)# vsan-domain member dom1	The sample command sequence enables interface **1/4** on leaf switch **101** to function as an **NP** port and associates that interface with VSAN domain **dom1**.
Step 9	Assign the targeted FCoE-enabled interfaces a VSAN. **Example:** apic1(config-leaf-if)# switchport trunk allowed vsan 1 tenant t1 application a1	Each of the targeted interfaces must be assigned one (and only one) VSAN in native mode. Each interface may be assigned one or more additional VSANs in regular mode.

	Command or Action	Purpose
	`epg e1` `apic1(config-leaf-if)# switchport vsan 2 tenant t4 application a4 epg e4`	The sample command sequence assigns the target interface to VSAN **1** and associates it with EPG **e1** and application **a1** under tenant **t1**. "trunk allowed" assigns vsan 1 regular mode status. The command sequence also assigns the interface a required **native mode** VSAN **2**. As this example shows, it is permissible for different VSANs to provide different EPGs running under different tenants access to the same interfaces.

Configuring FCoE Connectivity With Policies and Profiles Using the NX-OS Style CLI

The following sample NX-OS style CLI sequences create and use policies to configure FCoE connectivity for EPG **e1** under tenant **t1**.

Procedure

	Command or Action	Purpose

Step 1	Under the target tenant configure a bridge domain to support FCoE traffic.
	Example:
	`apic1# configure`
	`apic1(config)# tenant t1`
	`apic1(config-tenant)# vrf context v1`
	`apic1(config-tenant-vrf)# exit`
	`apic1(config-tenant)# bridge-domain b1`
	
	`apic1(config-tenant-bd)# fc`
	`apic1(config-tenant-bd)# vrf member v1`
	`apic1(config-tenant-bd)# exit`
	`apic1(config-tenant)# exit`
	`apic1(config)#`
		The sample command sequence creates bridge domain **b1** under tenant **t1** configured to support FCoE connectivity.

Step 2	Under the same tenant, associate your target EPG with the FCoE configured bridge domain.
	Example:
	`apic1(config)# tenant t1`
	`apic1(config-tenant)# application a1`
	`apic1(config-tenant-app)# epg e1`
	`apic1(config-tenant-app-epg)# bridge-domain member b1`
	`apic1(config-tenant-app-epg)# exit`
	`apic1(config-tenant-app)# exit`
	`apic1(config-tenant)# exit`
	`apic1(config)#`
		The sample command sequence creates EPG **e1** associates that EPG with FCoE-configured bridge domain **b1**.

	Command or Action	Purpose

Step 3	Create a VSAN domain, VSAN pools, VLAN pools and VSAN to VLAN mapping.
	Example:
	A
	`apic1(config)# vsan-domain dom1`
	`apic1(config-vsan)# vsan 1-10`
	`apic1(config-vsan)# vlan 1-10`
	`apic1(config-vsan)# fcoe vsan 1 vlan 1 loadbalancing src-dst-ox-id`
	`apic1(config-vsan)# fcoe vsan 2 vlan 2`
	Example:
	B
	`apic1(config)# template vsan-attribute pol1`
	`apic1(config-vsan-attr)# fcoe vsan 2 vlan 12 loadbalancing src-dst-ox-id`
	`apic1(config-vsan-attr)# fcoe vsan 3 vlan 13 loadbalancing src-dst-ox-id`
	`apic1(config-vsan-attr)# exit`
	`apic1(config)# vsan-domain dom1`
	`apic1(config-vsan)# inherit vsan-attribute pol1`
	`apic1(config-vsan)# exit`
		In **Example A**, the sample command sequence creates VSAN domain, **dom1** with VSAN pools and VLAN pools, maps VSAN 1 VLAN 1 and maps VSAN 2 to VLAN 2
		In **Example B**, an alternate sample command sequence creates a reusable vsan attribute template **pol1** and then creates VSAN domain **dom1**, which inherits the attributes and mappings from that template.

Step 4	Create the physical domain to support the FCoE Initialization (FIP) process.
	Example:
	`apic1(config)# vlan-domain fipVlanDom`
	`apic1(config)# vlan-pool fipVlanPool`	

Step 5	Configure a Fibre Channel SAN policy.
	Example:
	`apic1#`
	`apic1# configure`
	`apic1(config)# template fc-fabric-policy ffp1`
	`apic1(config-fc-fabric-policy)# fctimer e-d-tov 1111`
	`apic1(config-fc-fabric-policy)# fctimer r-a-tov 2222`
	`apic1(config-fc-fabric-policy)# fcoe fcmap 0E:FC:01`
	`apic1(config-fc-fabric-policy)# exit`
		The sample command sequence creates Fibre Channel SAN policy **ffp1** to specify a combination of error-detect timeout values (EDTOV), resource allocation timeout values (RATOV), and the default FC map values for FCoE-enabled interfaces on a target leaf switch.

Step 6	Create a Fibre Channel node policy. **Example:**	The sample command sequence creates Fibre Channel node policy **flp1** to specify a

	Command or Action	Purpose

	`apic1(config)# template fc-leaf-policy flp1`
	`apic1(config-fc-leaf-policy)# fcoe fka-adv-period 44`
	`apic1(config-fc-leaf-policy)# exit`
		combination of disruptive load-balancing enablement and FIP keep-alive values. These values also apply to all the FCoE-enabled interfaces on a target leaf switch.

Step 7	Create Node Policy Group.
	Example:
	`apic1(config)# template leaf-policy-group lpg1`
	`apic1(config-leaf-policy-group)# inherit fc-fabric-policy ffp1`
	`apic1(config-leaf-policy-group)# inherit fc-leaf-policy flp1`
	`apic1(config-leaf-policy-group)# exit`
	`apic1(config)# exit`
	`apic1#`
		The sample command sequence creates a Node Policy group, **lpg1**, which combines the values of the Fibre Channel SAN policy **ffp1** and Fibre Channel node policy, **flp1**. The combined values of this node policy group can be applied to Node profiles configured later.

Step 8	Create a Node Profile.
	Example:
	`apic1(config)# leaf-profile lp1`
	`apic1(config-leaf-profile)# leaf-group lg1`
	`apic1(config-leaf-group)# leaf 101`
	`apic1(config-leaf-group)# leaf-policy-group lpg1`
		The sample command sequence creates node profile **lp1** associates it with node policy group **lpg1**, node group **lg1**, and leaf switch **101**.

Step 9	Create an interface policy group for F port interfaces.
	Example:
	`apic1(config)# template policy-group ipg1`
	`apic1(config-pol-grp-if)# priority-flow-control mode auto`
	`apic1(config-pol-grp-if)# switchport mode f`
	`apic1(config-pol-grp-if)# slow-drain pause timeout 111`
	`apic1(config-pol-grp-if)# slow-drain congestion-timeout count 55`
	`apic1(config-pol-grp-if)# slow-drain congestion-timeout action log`
		The sample command sequence creates interface policy group **ipg1** and assigns a combination of values that determine priority flow control enablement, F port enablement, and slow-drain policy values for any interface that this policy group is applied to.

Step 10	Create an interface policy group for NP port interfaces.
	Example:
	`apic1(config)# template policy-group ipg2`
	`apic1(config-pol-grp-if)# priority-flow-control mode auto`
	`apic1(config-pol-grp-if)# switchport mode np`
	`apic1(config-pol-grp-if)# slow-drain pause timeout 111`
	`apic1(config-pol-grp-if)# slow-drain congestion-timeout count 55`
		The sample command sequence creates interface policy group **ipg2** and assigns a combination of values that determine priority flow control enablement, NP port enablement, and slow-drain policy values for any interface that this policy group is applied to.

	Command or Action	Purpose
	`apic1(config-pol-grp-if)# slow-drain congestion-timeout action log`	

Step 11	Create an interface profile for F port interfaces.
	Example:
	`apic1# configure`
	`apic1(config)# leaf-interface-profile lip1`
	`apic1(config-leaf-if-profile)# description 'test description lip1'`
	`apic1(config-leaf-if-profile)# leaf-interface-group lig1`
	`apic1(config-leaf-if-group)# description 'test description lig1'`
	`apic1(config-leaf-if-group)# policy-group ipg1`
	`apic1(config-leaf-if-group)# interface ethernet 1/2-6, 1/9-13`
		The sample command sequence creates an interface profile **lip1** for F port interfaces, associates the profile with F port specific interface policy group **ipg1**, and specifies the interfaces to which this profile and its associated policies applies.

Step 12	Create an interface profile for NP port interfaces.
	Example:
	`apic1# configure`
	`apic1(config)# leaf-interface-profile lip2`
	`apic1(config-leaf-if-profile)# description 'test description lip2'`
	`apic1(config-leaf-if-profile)# leaf-interface-group lig2`
	`apic1(config-leaf-if-group)# description 'test description lig2'`
	`apic1(config-leaf-if-group)# policy-group ipg2`
	`apic1(config-leaf-if-group)# interface ethernet 1/14`
		The sample command sequence creates an interface profile **lip2** for NP port interfaces, associates the profile with NP port specific interface policy group **ipg2**, and specifies the interface to which this profile and its associated policies applies.

Step 13	Configure QoS Class Policy for Level 1.
	Example:
	`apic1(config)# qos parameters level1`
	`apic1(config-qos)# pause no-drop cos 3`
		The sample command sequence specifies the QoS level of FCoE traffic to which priority flow control policy might be applied and pauses no-drop packet handling for Class of Service level 3.

Configuring FCoE Over FEX Using NX-OS Style CLI

FEX ports are configured as port VSANs.

Procedure

Step 1 Configure Tenant and VSAN domain:

Example:

```
apic1# configure
apic1(config)#  tenant t1
apic1(config-tenant)#   vrf context v1
apic1(config-tenant-vrf)#     exit
apic1(config-tenant)#   bridge-domain b1
apic1(config-tenant-bd)#       fc
apic1(config-tenant-bd)#       vrf member v1
apic1(config-tenant-bd)#       exit
apic1(config-tenant)#   application a1
apic1(config-tenant-app)#      epg e1
apic1(config-tenant-app-epg)# bridge-domain member b1
apic1(config-tenant-app-epg)# exit
apic1(config-tenant-app)#      exit
apic1(config-tenant)# exit

apic1(config)#  vsan-domain dom1
apic1(config-vsan)# vlan 1-100
apic1(config-vsan)# vsan 1-100
apic1(config-vsan)# fcoe vsan 2 vlan 2 loadbalancing src-dst-ox-id
apic1(config-vsan)# fcoe vsan 3 vlan 3 loadbalancing src-dst-ox-id
apic1(config-vsan)# fcoe vsan 5 vlan 5
apic1(config-vsan)# exit
```

Step 2 Associate FEX to an interface:

Example:

```
apic1(config)#  leaf 101
apic1(config-leaf)#  interface ethernet 1/12
apic1(config-leaf-if)#  fex associate 111
apic1(config-leaf-if)#  exit
```

Step 3 Configure FCoE over FEX per port, port-channel, and VPC:

Example:

```
apic1(config-leaf)#  interface vfc 111/1/2
apic1(config-leaf-if)#  vsan-domain member dom1
apic1(config-leaf-if)#  switchport vsan 2 tenant t1 application a1 epg e1

apic1(config-leaf-if)# exit

apic1(config-leaf)#  interface vfc-po pc1 fex 111
apic1(config-leaf-if)# vsan-domain member dom1
apic1(config-leaf-if)# switchport vsan 2 tenant t1 application a1 epg e1
apic1(config-leaf-if)# exit
apic1(config-leaf)# interface ethernet 111/1/3
apic1(config-leaf-if)# channel-group pc1
apic1(config-leaf-if# exit
apic1(config-leaf)# exit

apic1(config)# vpc domain explicit 12 leaf 101 102
apic1(config-vpc)# exit
apic1(config)# vpc context leaf 101 102
apic1(config-vpc)#     interface vpc vpc1 fex 111 111
apic1(config-vpc-if)#       vsan-domain member dom1
apic1(config-vpc-if)#       switchport vsan 2 tenant t1 application a1 epg e1
apic1(config-vpc-if)#       exit
apic1(config-vpc)#     exit
apic1(config)# leaf 101-102
apic1(config-leaf)# interface ethernet 1/2
apic1(config-leaf-if)# fex associate 111
apic1(config-leaf-if)# exit
apic1(config-leaf)# interface ethernet 111/1/2
```

```
apic1(config-leaf-if)# channel-group vpc1 vpc
apic1(config-leaf-if)# exit
```

Step 4 Verify the configuration with the following command:

Example:

```
apic1(config-vpc)# show vsan-domain detail
vsan-domain : dom1

vsan : 1-100

vlan : 1-100

Leaf          Interface         Vsan  Vlan  Vsan-Mode    Port-Mode  Usage
Operational State
------------  ----------------  ----  ----  -----------  ---------
---------------------------------------
101           vfc111/1/2        2       2    Native                  Tenant: t1
Deployed
                                                                      App: a1
                                                                      Epg: e1

101           PC:pc1            5       5    Native                Tenant: t1
Deployed
                                                                      App: a1
                                                                      Epg: e1

101           vfc111/1/3        3       3    Native       F          Tenant: t1
Deployed
                                                                      App: a1
                                                                      Epg: e1
```

Verifying FCoE Configuration Using the NX-OS Style CLI

The following **show** command verifies the FCoE configuration on your leaf switch ports.

Procedure

Use the **show vsan-domain** command to verify FCoE is enabled on the target switch.

The command example confirms FCoE enabled on the listed leaf switches and its FCF connection details.

Example:

```
ifav-isim8-ifc1# show vsan-domain detail
vsan-domain : iPostfcoeDomP1

vsan : 1-20  51-52  100-102  104-110  200  1999  3100-3101  3133
      2000

vlan : 1-20  51-52  100-102  104-110  200  1999  3100-3101  3133
      2000

                                  Vsan    Port                    Operational
 Leaf  Interface       Vsan  Vlan  Mode    Mode  Usage             State
 ----  ---------       ----  ----  ------- ----  ----------------  ------------
 101   vfc1/11         1     1     Regular  F    Tenant: iPost101  Deployed
```

```
                                                App: iPost1

                                                Epg: iPost1


 101   vfc1/12         1     1     Regular  NP   Tenant: iPost101  Deployed

                                                App: iPost1

                                                Epg: iPost1


 101   PC:infraAccBndl 4     4     Regular  NP   Tenant: iPost101  Deployed

          Grp_pc01                              App: iPost4

                                                Epg: iPost4


 101   vfc1/30         2000        Native        Tenant: t1        Not deployed
                                                 App: a1           (invalid-path)

                                                Epg: e1
```

Undeploying FCoE Elements Using the NX-OS Style CLI

Any move to undeploy FCoE connectivity from the ACI fabric requires that you remove the FCoE components on several levels.

Procedure

Step 1 List the attributes of the leaf port interface, set its mode setting to default, and then remove its EPG deployment and domain association.

The example sets the port mode setting of interface vfc **1/2** to default and then removes the deployment of EPG **e1** and the association with VSAN Domain **dom1** from that interface.

Example:

```
apic1(config)# leaf 101
apic1(config-leaf)# interface vfc 1/2
apic1(config-leaf-if)# show run
# Command: show running-config leaf 101 interface vfc 1 / 2
# Time: Tue Jul 26 09:41:11 2016
  leaf 101
    interface vfc 1/2
      vsan-domain member dom1
      switchport vsan 2 tenant t1 application a1 epg e1
      exit
    exit
apic1(config-leaf-if)# no switchport mode
apic1(config-leaf-if)# no switchport vsan 2 tenant t1 application a1 epg e1
```

```
apic1(config-leaf-if)# no vsan-domain member dom1
apic1(config-leaf-if)# exit
apic1(config-leaf)# exit
```

Step 2 List and remove the VSAN/VLAN mapping and the VLAN and VSAN pools.

The example removes the VSAN/VLAN mapping for **vsan 2**, VLAN pool **1-10**, and VSAN pool **1-10** from VSAN domain **dom1**.

Example:

```
apic1(config)# vsan-domain dom1
apic1(config-vsan)# show run
# Command: show running-config vsan-domain dom1
# Time: Tue Jul 26 09:43:47 2016
  vsan-domain dom1
    vsan 1-10
    vlan 1-10
    fcoe vsan 2 vlan 2
    exit
apic1(config-vsan)# no fcoe vsan 2
apic1(config-vsan)# no vlan 1-10
apic1(config-vsan)# no vsan 1-10
apic1(config-vsan)# exit

##############################################################################
NOTE: To remove a template-based VSAN to VLAN mapping use an alternate sequence:
##############################################################################

apic1(config)# template vsan-attribute <template_name>
apic1(config-vsan-attr)# no fcoe vsan 2
```

Step 3 Delete the VSAN Domain.

The example deletes VSAN domain **dom1**.

Example:

```
apic1(config)# no vsan-domain dom1
```

Step 4 You can delete the associated tenant, EPG, and selectors if you do not need them.

Configuring FCoE Using the REST API

Configuring FCoE Connectivity Using the REST API

You can configure FCoE-enabled interfaces and EPGs accessing those interfaces using the FCoE protocol with the REST API.

Procedure

Step 1 To create a VSAN pool, send a post with XML such as the following example.

The example creates VSAN pool **vsanPool1** and specifies the range of VSANs to be included.

Example:

```
https://apic-ip-address/api/mo/uni/infra/vsanns-[vsanPool1]-static.xml

<!-- Vsan-pool -->
<fvnsVsanInstP name="vsanPool1" allocMode="static">
     <fvnsVsanEncapBlk name="encap" from="vsan-5" to="vsan-100"/>
</fvnsVsanInstP>
```

Step 2 To create a VLAN pool, send a post with XML such as the following example.

The example creates VLAN pool **vlanPool1** and specifies the range of VLANs to be included.

Example:

```
https://apic-ip-address/api/mo/uni/infra/vlanns-[vlanPool1]-static.xml

<!-- Vlan-pool -->
<fvnsVlanInstP name="vlanPool1" allocMode="static">
 <fvnsEncapBlk name="encap" from="vlan-5" to="vlan-100"/>
</fvnsVlanInstP>
```

Step 3 To create a VSAN-Attribute policy, send a post with XML such as the following example.

The example creates VSAN attribute policy **vsanattri1**, maps **vsan-10** to **vlan-43**, and maps **vsan-11** to **vlan-44**.

Example:

```
https://apic-ip-address/api/mo/uni/infra/vsanattrp-[vsanattr1].xml

<fcVsanAttrP name="vsanattr1">

    <fcVsanAttrPEntry vlanEncap="vlan-43" vsanEncap="vsan-10"/>
    <fcVsanAttrPEntry vlanEncap="vlan-44" vsanEncap="vsan-11"
     lbType="src-dst-ox-id"/>
</fcVsanAttrP>
```

Step 4 To create a Fibre Channel domain, send a post with XML such as the following example.

The example creates VSAN domain **vsanDom1**.

Example:

```
https://apic-ip-address/api/mo/uni/fc-vsanDom1.xml
<!-- Vsan-domain -->
<fcDomP name="vsanDom1">
     <fcRsVsanAttr tDn="uni/infra/vsanattrp-[vsanattr1]"/>
     <infraRsVlanNs tDn="uni/infra/vlanns-[vlanPool1]-static"/>
     <fcRsVsanNs tDn="uni/infra/vsanns-[vsanPool1]-static"/>
</fcDomP>
```

Step 5 To create the tenant, application profile, EPG and associate the FCoE bridge domain with the EPG, send a post with XML such as the following example.

The example creates a bridge domain **bd1** under a target tenant configured to support FCoE and an application EPG **epg1**. It associates the EPG with VSAN domain vsanDom1 and a Fibre Channel path (to interface **1/39** on leaf switch **101**. It deletes a Fibre channel path to interface **1/40** by assigning the <fvRsFcPathAtt> object with "deleted" status. Each interface is associated with a VSAN.

Note Two other possible alternative vFC deployments are also displayed. One sample deploys vFC on a port channel. The other sample deploys vFC on a virtual port channel.

Example:

```
https://apic-ip-address/api/mo/uni/tn-tenant1.xml

<fvTenant
name="tenant1">
  <fvCtx name="vrf1"/>

  <!-- bridge domain -->
  <fvBD name="bd1" type="fc" >
    <fvRsCtx tnFvCtxName="vrf1" />
  </fvBD>

  <fvAp name="app1">
    <fvAEPg name="epg1">
      <fvRsBd tnFvBDName="bd1" />
      <fvRsDomAtt tDn="uni/fc-vsanDom1" />
      <fvRsFcPathAtt tDn="topology/pod-1/paths-101/pathep-[eth1/39]"
       vsan="vsan-11" vsanMode="native"/>
      <fvRsFcPathAtt tDn="topology/pod-1/paths-101/pathep-[eth1/40]"
       vsan="vsan-10" vsanMode="regular" status="deleted"/>
    </fvAEPg>

<!-- Sample deployment of vFC on a port channel -->

      <fvRsFcPathAtt vsanMode="native" vsan="vsan-10"
       tDN="topology/pod-1/paths 101/pathep-pc01"/>

<!-- Sample deployment of vFC on a virtual port channel -->

      <fvRsFcPathAtt vsanMode="native" vsan="vsan-10"
       tDn="topology/pod-1/paths-101/pathep-vpc01"/>
      <fvRsFcPathAtt vsanMode="native" vsan="vsan-10"
       tDn="topology/pod-1/paths-102/pathep-vpc01"/>

  </fvAp>
</fvTenant>
```

Step 6 To create a port policy group and an AEP, send a post with XML such as the following example.

The example executes the following requests:

- Creates a policy group **portgrp1** that includes an FC interface policy **fcIfPol1**, a priority flow control policy **pfcIfPol1** and a slow-drain policy **sdIfPol1**.
- Creates an attached entity profile (AEP) **AttEntP1** that associates the ports in VSAN domain **vsanDom1** with the settings to be specified for **fcIfPol1**, **pfcIfPol1**, and **sdIfPol1**.

Example:

```
https://apic-ip-address/api/mo/uni.xml

<polUni>
  <infraInfra>
    <infraFuncP>
      <infraAccPortGrp name="portgrp1">
        <infraRsFcIfPol tnFcIfPolName="fcIfPol1"/>
        <infraRsAttEntP tDn="uni/infra/attentp-AttEntP1" />
        <infraRsQosPfcIfPol tnQosPfcIfPolName="pfcIfPol1"/>
        <infraRsQosSdIfPol tnQosSdIfPolName="sdIfPol1"/>
      </infraAccPortGrp>
    </infraFuncP>

    <infraAttEntityP name="AttEntP1">
      <infraRsDomP tDn="uni/fc-vsanDom1"/>
    </infraAttEntityP>
```

```
    <qosPfcIfPol dn="uni/infra/pfc-pfcIfPol1" adminSt="on">
    </qosPfcIfPol>
    <qosSdIfPol dn="uni/infra/qossdpol-sdIfPol1" congClearAction="log"
     congDetectMult="5" flushIntvl="100" flushAdminSt="enabled">
    </qosSdIfPol>
    <fcIfPol dn="uni/infra/fcIfPol-fcIfPol1" portMode="np">
    </fcIfPol>

 </infraInfra>
</polUni>
```

Step 7 To create a node selector and a port selector, send a post with XML such as the following example.

The example executes the following requests:

- Creates node selector **leafsel1** that specifies leaf node **101**.
- Creates port selector **portsel1** that specifies port **1/39**.

Example:

```
https://apic-ip-address/api/mo/uni.xml

<polUni>
  <infraInfra>
    <infraNodeP name="nprof1">
      <infraLeafS name="leafsel1" type="range">
        <infraNodeBlk name="nblk1" from_="101" to_="101"/>
      </infraLeafS>
      <infraRsAccPortP tDn="uni/infra/accportprof-pprof1"/>
    </infraNodeP>

    <infraAccPortP name="pprof1">
      <infraHPortS name="portsel1" type="range">
        <infraPortBlk name="blk"
        fromCard="1" toCard="1" fromPort="39" toPort="39">
        </infraPortBlk>

        <infraRsAccBaseGrp tDn="uni/infra/funcprof/accportgrp-portgrp1" />
      </infraHPortS>

    </infraAccPortP>
 </infraInfra>
</polUni>
```

Step 8 To create a vPC, send a post with XML such as the following example.

Example:

```
https://apic-ip-address/api/mo/uni.xml
<polUni>
  <fabricInst>

     <vpcInstPol name="vpc01" />

     <fabricProtPol pairT="explicit" >
        <fabricExplicitGEp name="vpc01" id="100" >
           <fabricNodePEp id="101"/>
           <fabricNodePEp id="102"/>
           <fabricRsVpcInstPol tnVpcInstPolName="vpc01" />
           <!-- <fabricLagId accBndlGrp="infraAccBndlGrp_{{pcname}}" /> -->
        </fabricExplicitGEp>
```

```
    </fabricProtPol>

  </fabricInst>
</polUni>
```

Configuring FCoE Over FEX Using REST API

Before you begin

- Follow the steps 1 through 4 as described in Configuring FCoE Connectivity Using the REST API, on page 162

Procedure

Step 1 Configure FCoE over FEX (Selectors): Port:

Example:

```
<infraInfra dn="uni/infra">
  <infraNodeP name="nprof1">
    <infraLeafS name="leafsel1" type="range">
      <infraNodeBlk name="nblk1" from_="101" to_="101"/>
    </infraLeafS>
    <infraRsAccPortP tDn="uni/infra/accportprof-pprof1" />
  </infraNodeP>

    <infraAccPortP name="pprof1">
      <infraHPortS name="portsel1" type="range">
        <infraPortBlk name="blk"
        fromCard="1" toCard="1" fromPort="17" toPort="17"></infraPortBlk>
       <infraRsAccBaseGrp tDn="uni/infra/fexprof-fexprof1/fexbundle-fexbundle1" fexId="110"
 />
      </infraHPortS>
    </infraAccPortP>

    <infraFuncP>
    <infraAccPortGrp name="portgrp1">
      <infraRsAttEntP tDn="uni/infra/attentp-attentp1" />
    </infraAccPortGrp>
    </infraFuncP>

  <infraFexP name="fexprof1">
    <infraFexBndlGrp name="fexbundle1"/>
    <infraHPortS name="portsel2" type="range">
      <infraPortBlk name="blk2"
        fromCard="1" toCard="1" fromPort="20" toPort="20"></infraPortBlk>
      <infraRsAccBaseGrp tDn="uni/infra/funcprof/accportgrp-portgrp1"/>
    </infraHPortS>
  </infraFexP>

 <infraAttEntityP name="attentp1">
<infraRsDomP tDn="uni/fc-vsanDom1"/>
 </infraAttEntityP>
</infraInfra>
```

Step 2 Tenant configuration:

Example:

```
fvTenant name="tenant1">
<fvCtx name="vrf1"/>

<!-- bridge domain -->
    <fvBD name="bd1" type="fc" >
    <fvRsCtx tnFvCtxName="vrf1" />
    </fvBD>

<fvAp name="app1">
    <fvAEPg name="epg1">
        <fvRsBd tnFvBDName="bd1" />
       <fvRsDomAtt tDn="uni/fc-vsanDom1" />
<fvRsFcPathAtt tDn="topology/pod-1/paths-101/extpaths-110/pathep-[eth1/17]" vsan="vsan-11"
 vsanMode="native"/>
        </fvAEPg>
    </fvAp>
</fvTenant>
```

Step 3 Configure FCoE over FEX (Selectors): Port-Channel:

Example:

```
<infraInfra dn="uni/infra">
  <infraNodeP name="nprof1">
    <infraLeafS name="leafsel1" type="range">
      <infraNodeBlk name="nblk1" from_="101" to_="101"/>
    </infraLeafS>
    <infraRsAccPortP tDn="uni/infra/accportprof-pprof1" />
  </infraNodeP>

    <infraAccPortP name="pprof1">
      <infraHPortS name="portsel1" type="range">
        <infraPortBlk name="blk1"
        fromCard="1" toCard="1" fromPort="18" toPort="18"></infraPortBlk>
       <infraRsAccBaseGrp tDn="uni/infra/fexprof-fexprof1/fexbundle-fexbundle1" fexId="111"
 />
      </infraHPortS>
    </infraAccPortP>

  <infraFexP name="fexprof1">
    <infraFexBndlGrp name="fexbundle1"/>
    <infraHPortS name="portsel1" type="range">
      <infraPortBlk name="blk1"
        fromCard="1" toCard="1" fromPort="20" toPort="20"></infraPortBlk>
      <infraRsAccBaseGrp tDn="uni/infra/funcprof/accbundle-pc1"/>
    </infraHPortS>
  </infraFexP>

  <infraFuncP>
    <infraAccBndlGrp name="pc1">
      <infraRsAttEntP tDn="uni/infra/attentp-attentp1" />
    </infraAccBndlGrp>
  </infraFuncP>

 <infraAttEntityP name="attentp1">
<infraRsDomP tDn="uni/fc-vsanDom1"/>
 </infraAttEntityP>
</infraInfra>
```

Step 4 Tenant configuration:

Example:

```
<fvTenant name="tenant1">
<fvCtx name="vrf1"/>

<!-- bridge domain -->
    <fvBD name="bd1" type="fc" >
    <fvRsCtx tnFvCtxName="vrf1" />
    </fvBD>

<fvAp name="app1">
    <fvAEPg name="epg1">
        <fvRsBd tnFvBDName="bd1" />
       <fvRsDomAtt tDn="uni/fc-vsanDom1" />
<fvRsFcPathAtt tDn="topology/pod-1/paths-101/extpaths-111/pathep-[pc1]" vsan="vsan-11"
vsanMode="native" />
        </fvAEPg>
    </fvAp>
</fvTenant>
```

Step 5 Configure FCoE over FEX (Selectors): vPC:

Example:

```
<polUni>
<fabricInst>
<vpcInstPol name="vpc1" />
<fabricProtPol pairT="explicit" >
<fabricExplicitGEp name="vpc1" id="100" >
<fabricNodePEp id="101"/>
<fabricNodePEp id="102"/>
<fabricRsVpcInstPol tnVpcInstPolName="vpc1" />
</fabricExplicitGEp>
</fabricProtPol>
</fabricInst>
</polUni>
```

Step 6 Tenant configuration:

Example:

```
<fvTenant name="tenant1">
<fvCtx name="vrf1"/>

<!-- bridge domain -->
    <fvBD name="bd1" type="fc" >
    <fvRsCtx tnFvCtxName="vrf1" />
    </fvBD>

<fvAp name="app1">
    <fvAEPg name="epg1">
        <fvRsBd tnFvBDName="bd1" />
       <fvRsDomAtt tDn="uni/fc-vsanDom1" />
<fvRsFcPathAtt vsanMode="native" vsan="vsan-11"
tDn="topology/pod-1/protpaths-101-102/extprotpaths-111-111/pathep-[vpc1]" />
</fvAEPg>
    </fvAp>
</fvTenant>
```

Step 7 Selector configuration:

Example:

```
<polUni>
<infraInfra>
<infraNodeP name="nprof1">
<infraLeafS name="leafsel1" type="range">
<infraNodeBlk name="nblk1" from_="101" to_="101"/>
```

```
</infraLeafS>
<infraRsAccPortP tDn="uni/infra/accportprof-pprof1" />
</infraNodeP>

<infraNodeP name="nprof2">
<infraLeafS name="leafsel2" type="range">
<infraNodeBlk name="nblk2" from_="102" to_="102"/>
</infraLeafS>
<infraRsAccPortP tDn="uni/infra/accportprof-pprof2" />
</infraNodeP>

<infraAccPortP name="pprof1">
<infraHPortS name="portsel1" type="range">
<infraPortBlk name="blk1"
fromCard="1" toCard="1" fromPort="18" toPort="18">
</infraPortBlk>
<infraRsAccBaseGrp tDn="uni/infra/fexprof-fexprof1/fexbundle-fexbundle1" fexId="111" />
</infraHPortS>
</infraAccPortP>
<infraAccPortP name="pprof2">
<infraHPortS name="portsel2" type="range">
<infraPortBlk name="blk2"
fromCard="1" toCard="1" fromPort="18" toPort="18">
</infraPortBlk>
<infraRsAccBaseGrp tDn="uni/infra/fexprof-fexprof2/fexbundle-fexbundle2" fexId="111" />
</infraHPortS>
</infraAccPortP>

<infraFexP name="fexprof1">
<infraFexBndlGrp name="fexbundle1"/>
<infraHPortS name="portsel1" type="range">
<infraPortBlk name="blk1"
fromCard="1" toCard="1" fromPort="20" toPort="20">
</infraPortBlk>
    <infraRsAccBaseGrp tDn="uni/infra/funcprof/accbundle-vpc1"/>
</infraHPortS>
</infraFexP>

<infraFexP name="fexprof2">
<infraFexBndlGrp name="fexbundle2"/>
<infraHPortS name="portsel2" type="range">
<infraPortBlk name="blk2"
fromCard="1" toCard="1" fromPort="20" toPort="20">

</infraPortBlk>
<infraRsAccBaseGrp tDn="uni/infra/funcprof/accbundle-vpc1"/>
</infraHPortS>
</infraFexP>

<infraFuncP>
<infraAccBndlGrp name="vpc1" lagT="node">
    <infraRsAttEntP tDn="uni/infra/attentp-attentp1" />
</infraAccBndlGrp>
</infraFuncP>

<infraAttEntityP name="attentp1">
<infraRsDomP tDn="uni/fc-vsanDom1"/>
</infraAttEntityP>
</infraInfra>
</polUni>
```

Configuring an FCoE vPC Using the REST API

This procedure creates a virtual port channel (vPC).

Procedure

Step 1 Create a vPC domain.

This step creates a virtual port channel security policy (fabric:ProtPol) containing a group policy (fabric:ExplicitGEp) that contains two node policy endpoints (fabric:NodePEp) named "101" and "102."

Example:

```
POST https://apic-ip-address/api/node/mo/uni/fabric/protpol.xml

<fabricProtPol>
    <fabricExplicitGEp name="vpc-explicitGrp1101102" id="100" >
        <fabricNodePEp id="101" />
        <fabricNodePEp id="102" />
    </fabricExplicitGEp>
</fabricProtPol>
```

Step 2 Create a Fibre Channel interface policy.

This step creates a Fibre Channel interface policy (fc:IfPol) named "vpc1" that has trunk mode enabled.

Example:

```
POST https://apic-ip-address/api/node/mo/uni/infra/fcIfPol-vpc1.xml

<fcIfPol name="vpc1" trunkMode="trunk-on" />
```

Step 3 Create an LACP port channel policy.

This step creates an LACP port channel policy (lacp:LagPol) named "vpc1" that has LACP-active mode enabled. The suspend-individual-port control is disabled from the port channel; otherwise the physical interface will be suspended when LACP BPDU is not received from the host.

Example:

```
POST https://apic-ip-address/api/node/mo/uni/infra/lacplagp-vpc1.xml

<lacpLagPol name="vpc1" mode="active" ctrl="graceful-conv,fast-sel-hot-stdby" />
```

Step 4 Create the vPC.

Example:

```
POST https://apic-ip-address/api/node/mo/uni/infra.xml

<infraInfra>
  <infraAccPortP
      name="Switch101-102_Profile_ifselector"
      descr="GUI Interface Selector Generated PortP Profile: Switch101-102_Profile">
    <infraHPortS name="Switch101-102_1-ports-49" type="range">
      <infraPortBlk name="block1" fromPort="49" toPort="49" />
```

```
      <infraRsAccBaseGrp
          tDn="uni/infra/funcprof/accbundle-Switch101-102_1-ports-49_PolGrp" />
    </infraHPortS>
  </infraAccPortP>
  <infraFuncP>
    <infraAccBndlGrp name="Switch101-102_1-ports-49_PolGrp" lagT="node">
      <infraRsAttEntP tDn="uni/infra/attentp-fcDom_AttEntityP" />
      <infraRsFcIfPol tnFcIfPolName="vpc1" />
      <infraRsLacpPol tnLacpLagPolName="vpc1" />
    </infraAccBndlGrp>
  </infraFuncP>
  <infraNodeP
      name="Switch101-102_Profile"
      descr="GUI Interface Selector Generated Profile: Switch101-102_Profile">
    <infraLeafS name="Switch101-102_Profile_selector_101102" type="range">
      <infraNodeBlk name="single0" from_="101" to_="101" />
      <infraNodeBlk name="single1" from_="102" to_="102" />
    </infraLeafS>
    <infraRsAccPortP
        tDn="uni/infra/accportprof-Switch101-102_Profile_ifselector" />>
  </infraNodeP>
</infraInfra>
```

Step 5 Create a native VLAN.

a) Create a bridge domain and associate it with a VRF.

Example:

```
POST https://apic-ip-address/api/node/mo/uni/tn-newtenant/BD-BDnew1.xml

<fvBD name="BDnew1" mac="00:22:BD:F8:19:FF" >
  <fvRsCtx tnFvCtxName="vrf" />
</fvBD>
```

b) Create an application EPG and associate it with the bridge domain.

Example:

```
POST https://apic-ip-address/api/node/mo/uni/tn-newtenant/ap-AP1/epg-epgNew.xml

<fvAEPg name="epgNew" >
  <fvRsBd tnFvBDName="BDnew1" />
</fvAEPg>
```

c) Create a static path and associate it with a VLAN.

Example:

```
POST https://apic-ip-address/api/node/mo/uni/tn-newtenant/ap-AP1/epg-epgNew.xml

<fvRsPathAtt
  encap="vlan-1"
  instrImedcy="immediate"
  mode="native"
  tDn="topology/pod-1/protpaths-101-102/pathep-[Switch101-102_1-ports-49_PolGrp]" />
```

Step 6 Create a vFC.

a) Create a bridge domain and associate it with a VRF.

Example:

```
POST https://apic-ip-address/api/node/mo/uni/tn-newtenant/BD-BD3.xml

<fvBD
    name="BD3"
    mac="00:22:BD:F8:19:FF"
    type="fc"
    unicastRoute="false" >
  <fvRsCtx tnFvCtxName="vrf" />
</fvBD>
```

b) Create an application EPG and associate it with the bridge domain.

Example:

```
POST https://apic-ip-address/api/node/mo/uni/tn-newtenant/ap-AP1/epg-epg3.xml

<fvAEPg name="epg3" >
  <fvRsBd tnFvBDName="BD3" />
</fvAEPg>
```

c) Create a static path and associate it with a VSAN.

Example:

```
POST https://apic-ip-address/api/node/mo/uni/tn-newtenant/ap-AP1/epg-epg3.xml

<fvRsFcPathAtt
    vsan="vsan-3"
    vsanMode="native"
    tDn="topology/pod-1/paths-101/pathep-[eth1/49]" />
```

Undeploying FCoE Connectivity through the REST API or SDK

To undeploy FCoE connectivity through the APIC REST API or SDK , delete the following objects associated with the deployment:

Object	Description
<fvRsFcPathAtt> (Fibre Channel Path)	The Fibre Channel path specifies the vFC path to the actual interface. Deleting each object of this type removes the deployment from that object's associated interfaces.
<fcVsanAttrpP> (VSAN/VLAN map)	The VSAN/VLAN map maps the VSANs to their associated VLANs deleting this object removes the association between the VSANs that support FCoE connectivity and their underlying VSANs.
<fvnsVsanInstP> (VSAN pool)	The VSAN pool specifies the set of VSANs available to support FCoE connectivity. Deleting this pool removes those VSANs.

Object	Description
<fvnsVlanIsntP> ((VLAN pool)	The VLAN pool specifies the set of VLANs available for VSAN mapping. Deleting the associated VLAN pool cleans up after an FCoE undeployment, removing the underlying VLAN entities over which the VSAN entities ran.
<fcDomP> (VSAN or Fibre Channel domain)	The Fibre Channel domain includes all the VSANs and their mappings. Deleting this object undeploys vFC from all interfaces associated with this domain.
<fvAEPg> (application EPG)	The application EPG associated with the FCoE connectivity. If the purpose of the application EPGs was only to support FCoE-related activity, you might consider deleting this object.
<fvAp> (application profile)	The application profile associated with the FCoE connectivity. If the purpose of the application profile was only to support FCoE-related activity, you might consider deleting this object.
<fvTenant> (tenant)	The tenant associated with the FCoE connectivity. If the purpose of the tenant was only to support FCoE-related activity, you might consider deleting this object.

Note If during clean up you delete the Ethernet configuration object (infraHPortS) for a vFC port, the default vFC properties remain associated with that interface. For example it the interface configuration for vFC NP port 1/20 is deleted, that port remains a vFC port but with default F port setting rather than non-default NP port setting applied.

The following steps undeploy FCoE-enabled interfaces and EPGs accessing those interfaces using the FCoE protocol.

Procedure

Step 1 To delete the associated Fibre Channel path objects, send a post with XML such as the following example.

The example deletes all instances of the Fibre Channel path object <fvRsFcPathAtt>.

Note Deleting the Fibre Channel paths undeploys the vFC from the ports/VSANs that used them.

Example:

```
https://apic-ip-address/api/mo/uni/tn-tenant1.xml

<fvTenant
name="tenant1">
  <fvCtx name="vrf1"/>

  <!-- bridge domain -->
  <fvBD name="bd1" type="fc" >
    <fvRsCtx tnFvCtxName="vrf1" />
  </fvBD>

  <fvAp name="app1">
```

```
    <fvAEPg name="epg1">
      <fvRsBd tnFvBDName="bd1" />
      <fvRsDomAtt tDn="uni/fc-vsanDom1" />
      <fvRsFcPathAtt tDn="topology/pod-1/paths-101/pathep-[eth1/39]"
       vsan="vsan-11" vsanMode="native" status="deleted"/>
      <fvRsFcPathAtt tDn="topology/pod-1/paths-101/pathep-[eth1/40]"
       vsan="vsan-10" vsanMode="regular" status="deleted"/>
    </fvAEPg>

<!-- Sample undeployment of vFC on a port channel -->

      <fvRsFcPathAtt vsanMode="native" vsan="vsan-10"
       tDN="topology/pod-1/paths 101/pathep-pc01" status="deleted"/>

<!-- Sample undeployment of vFC on a virtual port channel -->

      <fvRsFcPathAtt vsanMode="native" vsan="vsan-10"
       tDn="topology/pod-1/paths-101/pathep-vpc01" status="deleted"/>
      <fvRsFcPathAtt vsanMode="native" vsan="vsan-10"
       tDn="topology/pod-1/paths-102/pathep-vpc01" status="deleted"/>

  </fvAp>
</fvTenant>
```

Step 2 To delete the associated VSAN/VLAN map, send a post such as the following example.

The example deletes the VSAN/VLAN map **vsanattri1** and its associated <fcVsanAttrpP> object.

Example:

```
https://apic-ip-address/api/mo/uni/infra/vsanattrp-[vsanattr1].xml

<fcVsanAttrP name="vsanattr1" status="deleted">

    <fcVsanAttrPEntry vlanEncap="vlan-43" vsanEncap="vsan-10" status="deleted"/>
    <fcVsanAttrPEntry vlanEncap="vlan-44" vsanEncap="vsan-11"
     lbType="src-dst-ox-id" status="deleted" />
</fcVsanAttrP>
```

Step 3 To delete the associated VSAN pool, send a post such as the following example.

The example deletes the VSAN pool **vsanPool1** and its associated <fvnsVsanInstP> object.

Example:

```
https://apic-ip-address/api/mo/uni/infra/vsanns-[vsanPool1]-static.xml

<!-- Vsan-pool -->
<fvnsVsanInstP name="vsanPool1" allocMode="static" status="deleted">
 <fvnsVsanEncapBlk name="encap" from="vsan-5" to="vsan-100" />
</fvnsVsanInstP>
```

Step 4 To delete the associated VLAN pool, send a post with XML such as the following example.

The example deletes the VLAN pool **vlanPool1** and its associated <fvnsVlanIsntP> object.

Example:

```
https://apic-ip-address/api/mo/uni/infra/vlanns-[vlanPool1]-static.xml

<!-- Vlan-pool -->
<fvnsVlanInstP name="vlanPool1" allocMode="static" status="deleted">
 <fvnsEncapBlk name="encap" from="vlan-5" to="vlan-100" />
</fvnsVlanInstP>
```

Step 5 To delete the associated Fibre Channel domain, send a post with XML such as the following example.

The example deletes the VSAN domain **vsanDom1** and its associated <fcDomP> object.

Example:

```
https://apic-ip-address/api/mo/uni/fc-vsanDom1.xml
<!-- Vsan-domain -->
<fcDomP name="vsanDom1" status="deleted">
     <fcRsVsanAttr tDn="uni/infra/vsanattrp-[vsanattr1]"/>
     <infraRsVlanNs tDn="uni/infra/vlanns-[vlanPool1]-static"/>
     <fcRsVsanNs tDn="uni/infra/vsanns-[vsanPool1]-static"/>
</fcDomP>
```

Step 6 **Optional:** If appropriate, you can delete the associated application EPG, the associated application profile, or the associated tenant.

Example:

In the following sample, the associated application EPG **epg1** and its associated <fvAEPg> object is deleted.

```
https://apic-ip-address/api/mo/uni/tn-tenant1.xml

<fvTenant
name="tenant1"/>
  <fvCtx name="vrf1"/>

  <!-- bridge domain -->
  <fvBD name="bd1" type="fc" >
    <fvRsCtx tnFvCtxName="vrf1" />
  </fvBD>

  <fvAp name="app1">
    <fvAEPg name="epg1" status= "deleted">
      <fvRsBd tnFvBDName="bd1" />
      <fvRsDomAtt tDn="uni/fc-vsanDom1" />
      <fvRsFcPathAtt tDn="topology/pod-1/paths-101/pathep-[eth1/39]"
       vsan="vsan-11" vsanMode="native" status="deleted"/>
      <fvRsFcPathAtt tDn="topology/pod-1/paths-101/pathep-[eth1/40]"
       vsan="vsan-10" vsanMode="regular" status="deleted"/>
    </fvAEPg>

<!-- Sample undeployment of vFC on a port channel -->

      <fvRsFcPathAtt vsanMode="native" vsan="vsan-10"
       tDN="topology/pod-1/paths 101/pathep-pc01" status="deleted"/>

<!-- Sample undeployment of vFC on a virtual port channel -->

      <fvRsFcPathAtt vsanMode="native" vsan="vsan-10"
       tDn="topology/pod-1/paths-101/pathep-vpc01" status="deleted"/>
      <fvRsFcPathAtt vsanMode="native" vsan="vsan-10"
       tDn="topology/pod-1/paths-102/pathep-vpc01" status="deleted"/>

  </fvAp>
</fvTenant>
```

Example:

In the following example, the associated application profile **app1** and its associated <fvAp> object is deleted.

```
https://apic-ip-address/api/mo/uni/tn-tenant1.xml

<fvTenant
name="tenant1">
  <fvCtx name="vrf1"/>

  <!-- bridge domain -->
```

```
  <fvBD name="bd1" type="fc">
    <fvRsCtx tnFvCtxName="vrf1" />
  </fvBD>

  <fvAp name="app1" status="deleted">
    <fvAEPg name="epg1" status= "deleted">
      <fvRsBd tnFvBDName="bd1" />
      <fvRsDomAtt tDn="uni/fc-vsanDom1" />
      <fvRsFcPathAtt tDn="topology/pod-1/paths-101/pathep-[eth1/39]"
       vsan="vsan-11" vsanMode="native" status="deleted"/>
      <fvRsFcPathAtt tDn="topology/pod-1/paths-101/pathep-[eth1/40]"
       vsan="vsan-10" vsanMode="regular" status="deleted"/>
    </fvAEPg>

<!-- Sample undeployment of vFC on a port channel -->

      <fvRsFcPathAtt vsanMode="native" vsan="vsan-10"
       tDN="topology/pod-1/paths 101/pathep-pc01" status="deleted"/>

<!-- Sample undeployment of vFC on a virtual port channel -->

      <fvRsFcPathAtt vsanMode="native" vsan="vsan-10"
       tDn="topology/pod-1/paths-101/pathep-vpc01" status="deleted"/>
      <fvRsFcPathAtt vsanMode="native" vsan="vsan-10"
       tDn="topology/pod-1/paths-102/pathep-vpc01" status="deleted"/>

  </fvAp>
</fvTenant>
```

Example:

In the following example, the entire tenant **tenant1** and its associated <fvTenant> object is deleted.

```
https://apic-ip-address/api/mo/uni/tn-tenant1.xml

<fvTenant
name="tenant1" status="deleted">
  <fvCtx name="vrf1"/>

  <!-- bridge domain -->
  <fvBD name="bd1" type="fc" status="deleted">
    <fvRsCtx tnFvCtxName="vrf1" />
  </fvBD>

  <fvAp name="app1">
    <fvAEPg name="epg1" status= "deleted">
      <fvRsBd tnFvBDName="bd1" />
      <fvRsDomAtt tDn="uni/fc-vsanDom1" />
      <fvRsFcPathAtt tDn="topology/pod-1/paths-101/pathep-[eth1/39]"
       vsan="vsan-11" vsanMode="native" status="deleted"/>
      <fvRsFcPathAtt tDn="topology/pod-1/paths-101/pathep-[eth1/40]"
       vsan="vsan-10" vsanMode="regular" status="deleted"/>
    </fvAEPg>

<!-- Sample undeployment of vFC on a port channel -->

      <fvRsFcPathAtt vsanMode="native" vsan="vsan-10"
       tDN="topology/pod-1/paths 101/pathep-pc01" status="deleted"/>

<!-- Sample undeployment of vFC on a virtual port channel -->

      <fvRsFcPathAtt vsanMode="native" vsan="vsan-10"
       tDn="topology/pod-1/paths-101/pathep-vpc01" status="deleted"/>
      <fvRsFcPathAtt vsanMode="native" vsan="vsan-10"
       tDn="topology/pod-1/paths-102/pathep-vpc01" status="deleted"/>
```

```
  </fvAp>
</fvTenant>
```

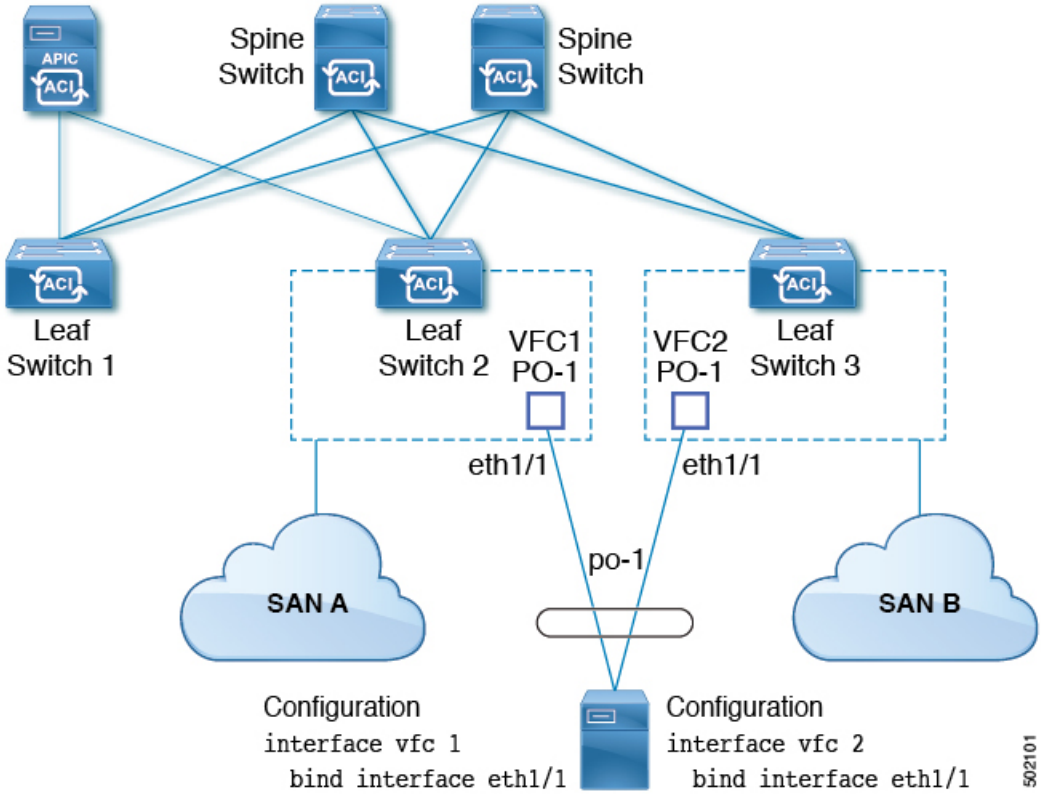
SAN Boot with vPC

Cisco ACI supports the SAN boot of initiators on Link Aggregation Control Protocol (LACP) based vPC. This limitation is specific to LACP-based port channels.

In the normal host-to-vPC topology, the host-facing vFC interface is bound to the vPC, and the vPC must be logically up before the vFC interface can come up. In this topology, a host will not be able to boot from SAN when LACP is configured on the vPC, because LACP on the host is typically implemented in the host driver and not in the adapter firmware.

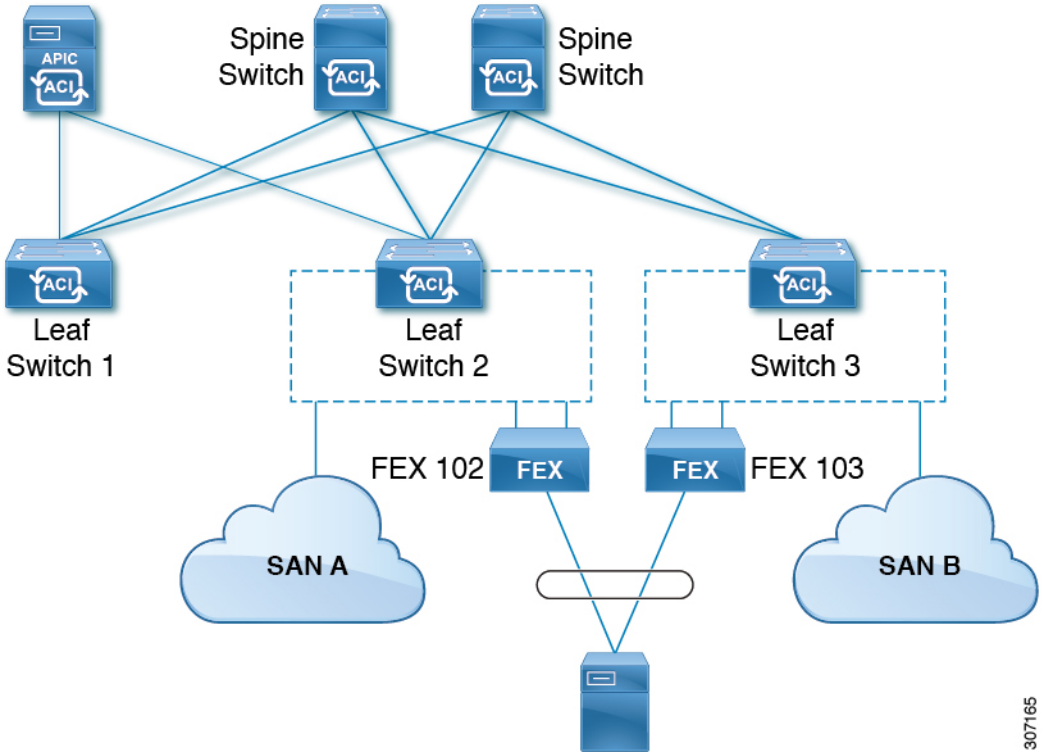
For SAN boot, the host-facing vFC interfaces are bound to port channel members instead of the port channel itself. This binding ensures that the host-side vFC comes up during a SAN boot as soon as the link on the CNA/Host Bus Adapter (HBA) comes up, without relying on the LACP-based port channel to form first.

Figure 30: SAN Boot Topology with vPC



Beginning with Cisco APIC Release 4.0(2), SAN boot is supported through a FEX host interface (HIF) port vPC, as shown in the following figure.

Figure 31: SAN Boot Topology with a FEX host interface (HIF) port vPC



Guidelines and Restrictions for SAN Boot with vPC

- Multi-member port channels are not supported.
- If a vFC is bound to a member port, the port channel cannot have more than 1 member.
- If a vFC is bound to a port channel, the port channel can have only one member port.

Configuring SAN Boot with vPC Using the GUI

To simplify the configuration, this procedure uses the **Configure Interface, PC, and vPC** wizard in **Fabric > Access Policies > Quickstart**.

Before you begin

This procedure assumes that the following items are already configured:

- VSAN Pool
- VLAN Pool
- VSAN Attributes, mapping VSANs in the VSAN pool to VLANs
- Fibre Channel domain (VSAN domain)
- Tenant, Application Profile

- Attached Entity Profile

Procedure

Step 1 On the APIC menu bar, navigate to **Fabric** > **Access Policies** > **Quickstart** and click *Configure an interface, PC, and VPC*.

Step 2 In the *Configure an interface, PC, and VPC* work area, in the **vPC Switch Pairs** toolbar, click **+** to create a switch pair. Perform the following actions:

a) From the **vPC Domain ID** text box, enter a number to designate the switch pair.
b) From the **Switch 1** drop-down list, select a leaf switch.

Only switches with interfaces in the same vPC policy group can be paired together.

c) From the **Switch 2** drop-down list, select a leaf switch.
d) click **Save** to save this switch pair.

Step 3 In the *Configure an interface, PC, and vPC* work area, click the large green **+** to select switches.
The **Select Switches To Configure Interfaces** work area opens with the **Quick** option selected by default.

Step 4 Select two switch IDs from the **Switches** drop-down list, and name the switch profile.

Step 5 Click the large green **+** again to configure the switch interfaces.

Step 6 In the **Interface Type** control, select **vPC**.

Step 7 For **Interfaces**, enter a single port number, such as `1/49`, that will be used on both switches as vPC members.

This action creates an interface selector policy. You can accept or change the name of the policy in the **Interface Selector Name** text box.

Step 8 In the **Interface Policy Group** control, select **Create One**.

Step 9 From the **Fibre Channel Interface Policy** text box, select **Create Fibre Channel Interface Policy** and perform the following actions.

a) In the **Name** field, type a name for the Fibre Channel interface policy.
b) From the **Port Mode** selector, select **F**.
c) From the **Trunk Mode** selector, select **trunk-on**.
d) Click **Submit**.

Step 10 From the **Port Channel Policy** text box, select **Create Port Channel Policy** and perform the following actions.

a) In the **Name** field, type a name for the port channel policy.
b) From the **Mode** drop-down list, select **LACP Active**.
c) From the **Control** selector, delete **Suspend Individual Port**.

Suspend Individual Port must be removed from the port channel; otherwise the physical interface will be suspended when LACP BPDU is not received from the host.

d) Click **Submit**.

Step 11 From the **Attached Device Type** drop-down list, select **Fibre Channel**.

Step 12 From the **Fibre Channel Domain** drop-down list, select your Fibre Channel domain (VSAN domain).

Step 13 Click **Save** to save this vPC configuration.

Step 14 Click **Save** to save this interface configuration.

Step 15 Click **Submit**.

Step 16 Expand **Tenants** > **Tenant** ***name*** > **Application Profiles** > ***name*** > **Application EPGs**.

Step 17 Right-click **Application EPGs**, select **Create Application EPG** and perform the following actions.

This EPG will be the Native EPG, in which the Native VLAN will be configured.

a) In the **Name** field, type a name for the EPG.
b) From the **Bridge Domain** drop-down list, select **Create Bridge Domain**.
c) In the **Name** field, type a name for the bridge domain.
d) From the **Type** control, select **regular**.
e) From the **VRF** drop-down list, choose the tenant VRF. If no VRF exists yet, select **Create VRF**, name the VRF and click **Submit**.
f) Click **Next**, **Next**, and **Finish** to return to **Create Application EPG**.
g) Click **Finish**.

Step 18 Expand the Native EPG created in the previous step.

Step 19 Right-click **Static Ports,** select **Deploy Static EPG On PC, VPC, or Interface** and perform the following actions.

a) From the **Path Type** control, select **Virtual Port Channel**.
b) From the **Path** drop-down list, select the port channel policy created for vPC.
c) From the **Port Encap** drop-down list, select **VLAN** and enter the number of an Ethernet VLAN.
d) From the **Deployment Immediacy** control, select **Immediate**.
e) From the **Mode** control, select **Access (802.1P)**.
f) Click **Submit**.

Step 20 Right-click **Application EPGs**, select **Create Application EPG** and perform the following actions.

This EPG will be the first of two EPGs, one for each SAN.

a) In the **Name** field, type a name for the EPG.
b) From the **Bridge Domain** drop-down list, select **Create Bridge Domain**.
c) In the **Name** field, type a name for the bridge domain.
d) From the **Type** control, select **fc**.
e) From the **VRF** drop-down list, choose the tenant VRF. If no VRF exists yet, select **Create VRF**, name the VRF and click **Submit**.
f) Click **Next**, **Next**, and **Finish** to return to **Create Application EPG**.
g) Click **Finish**.

Step 21 Repeat the previous step to create a second application EPG.

This second EPG will be used for the second SAN.

Step 22 Expand one of the two SAN EPGs, right-click **Fibre Channel (Paths)**, select **Deploy Fibre Channel** and perform the following actions.

a) From the **Path Type** control, select **Port**.
b) From the **Node** drop-down list, select one leaf of your switch pair.
c) From the **Path** drop-down list, select the Ethernet port number of your VPC.
d) In the **VSAN** text box, type the VSAN number prefixed by "vsan-".

   For example, type "vsan-300" for VSAN number 300.

e) In the **VSAN Mode** control, select **Native**.

f) Click **Submit**.

Step 23 Expand the other of the two SAN EPGs and repeat the previous step, selecting the other leaf of your switch pair.

SAN Boot with vPC Configuration Using the CLI

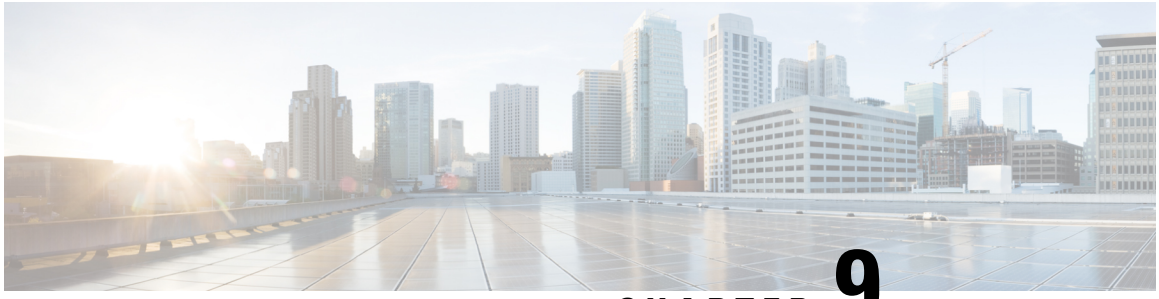
This example assumes that the following items have been configured:

- A VLAN domain
- A tenant, application profile, and an application EPG
- A port channel template "Switch101-102_1-ports-49_PolGrp"

In this example, VSAN 200 is bound to physical Ethernet interface 1/49 on leaf 101 and VSAN 300 is bound to physical Ethernet interface 1/49 on leaf 102. The two interfaces are members of virtual port channel Switch101-102_1-ports-49_PolGrp.

```
apic1(config-leaf)# show running-config
# Command: show running-config leaf 101
# Time: Sat Sep  1 12:51:23 2018
  leaf 101

    interface ethernet 1/49
      # channel-group Switch101-102_1-ports-49_PolGrp vpc
      switchport trunk native vlan 5 tenant newtenant application AP1 epg epgNative
      port-direction downlink
      exit

    # Port-Channel inherits configuration from "template port-channel
Switch101-102_1-ports-49_PolGrp"
    interface port-channel Switch101-102_1-ports-49_PolGrp
      exit

    interface vfc 1/49
     # Interface inherits configuration from "channel-group Switch101-102_1-ports-49_PolGrp"
 applied to interface ethernet 1/49
      switchport vsan 200 tenant newtenant application AP1 epg epg200
      exit

apic1(config-leaf)# show running-config
# Command: show running-config leaf 102
# Time: Sat Sep  1 13:28:02 2018
  leaf 102
    interface ethernet 1/49
      # channel-group Switch101-102_1-ports-49_PolGrp vpc
      switchport trunk native vlan 1 tenant newtenant application AP1 epg epgNative
      port-direction downlink
      exit

    # Port-Channel inherits configuration from "template port-channel
Switch101-102_1-ports-49_PolGrp"
    interface port-channel Switch101-102_1-ports-49_PolGrp
      exit

    interface vfc 1/49
     # Interface inherits configuration from "channel-group Switch101-102_1-ports-49_PolGrp"
 applied to interface ethernet 1/49
```

```
switchport vsan 300 tenant newtenant application AP1 epg epg300
```



CHAPTER 9

Fibre Channel NPV

This chapter contains the following sections:

Fibre Channel Connectivity Overview

A switch is in NPV mode after enabling NPV. NPV mode applies to an entire switch. All end devices connected to a switch that are in NPV mode must log in as an N port to use this feature (loop-attached devices are not supported). All links from the edge switches (in NPV mode) to the NPV core switches are established as NP ports (not E ports), which are used for typical inter-switch links.

FC NPV Benefits

FC NPV provides the following:

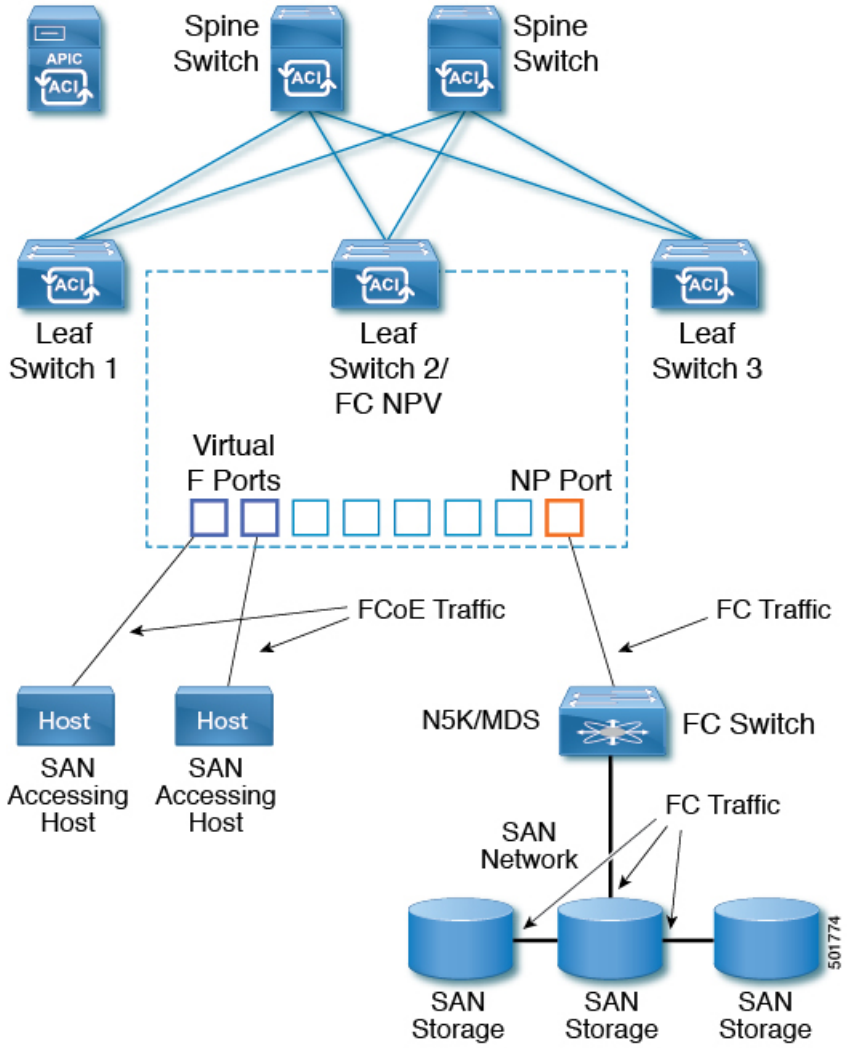
- Increased number of hosts that connect to the fabric without adding domain IDs in the fabric
- Connection of FC and FCoE hosts and targets to SAN fabrics using FC interfaces
- Automatic traffic mapping
- Static traffic mapping
- Disruptive automatic load balancing

FC NPV Mode

Feature-set fcoe-npv in ACI will be enabled automatically by default when first FCoE/FC configuration is pushed.

FC Topology

The topology of a typical configuration supporting FC traffic over the ACI fabric consists of the following components:



- A Leaf can be connected to a FC switch by using FCoE NP port or native FC NP port.
- An ACI Leaf can be directly connected with a server/Storage using FCoE links.
- FC/FCoE traffic is not sent to fabric/spine. A Leaf switch does not do local switching for FCoE traffic. The switching is done by a core switch which is connected with a leaf switch via FC/FCoE NPV link.
- Multiple FDISC followed by Flogi is supported with FCoE host and FC/FCoE NP links.

NPV Traffic Management

In most cases, Cisco recommends allowing all traffic to use all available uplinks. Use FC NPV traffic management only when automatic traffic engineering does not meet your network requirements.

Automatic Uplink Selection

NPV supports automatic selection of external (NP uplink) interfaces. When a server (host) interface is brought up, the external interface with the minimum load is selected from the available external interfaces in the same VSAN as the server interface.

When a new external interface becomes operational, the existing load is not redistributed automatically to include the newly available uplink. Server interfaces that become operational after the external interface can select the new uplink.

Traffic Maps

FC NPV supports traffic maps. A traffic map allows you to specify the external (NP uplink) interfaces that a server (host) interface can use to connect to the core switches.

Note When an FC NPV traffic map is configured for a server interface, the server interface must select only from the external interfaces in its traffic map. If none of the specified external interfaces are operational, the server remains in a non-operational state.

The FC NPV traffic map feature provides the following benefits:

- Facilitates traffic engineering by allowing configuration of a fixed set of external interfaces for a specific server interface (or range of server interfaces).
- Ensures correct operation of the persistent FC ID feature; this is because a server interface will always connect to the same external interface (or one of a specified set of external interfaces) by providing the same traffic path after an interface reinitialization or switch reboot.

Disruptive Auto Load Balancing of Server Logins across NP Links

FC NPV supports disruptive load balancing of server logins. When disruptive load balancing is enabled, FC NPV redistributes the server interfaces across all available NP uplinks when a new NP uplink becomes operational. To move a server interface from one NP uplink to another NP uplink, FC NPV forces reinitialization of the server interface so that the server performs a new login to the core switch.

Only server interfaces that are moved to a different uplink are reinitialized. A system message is generated for each server interface that is moved.

Note Redistributing a server interface causes traffic disruption to the attached end devices. Adding a member to the existing port-channel does not trigger disruptive auto load-balance.

To avoid disruption of server traffic, you should enable this feature only after adding a new NP uplink, and then disable it again after the server interfaces have been redistributed.

If disruptive load balancing is not enabled, you can manually reinitialize some or all of the server interfaces to distribute server traffic to new NP uplink interfaces.

FC NPV Traffic Management Guidelines

When deploying FC NPV traffic management, follow these guidelines:

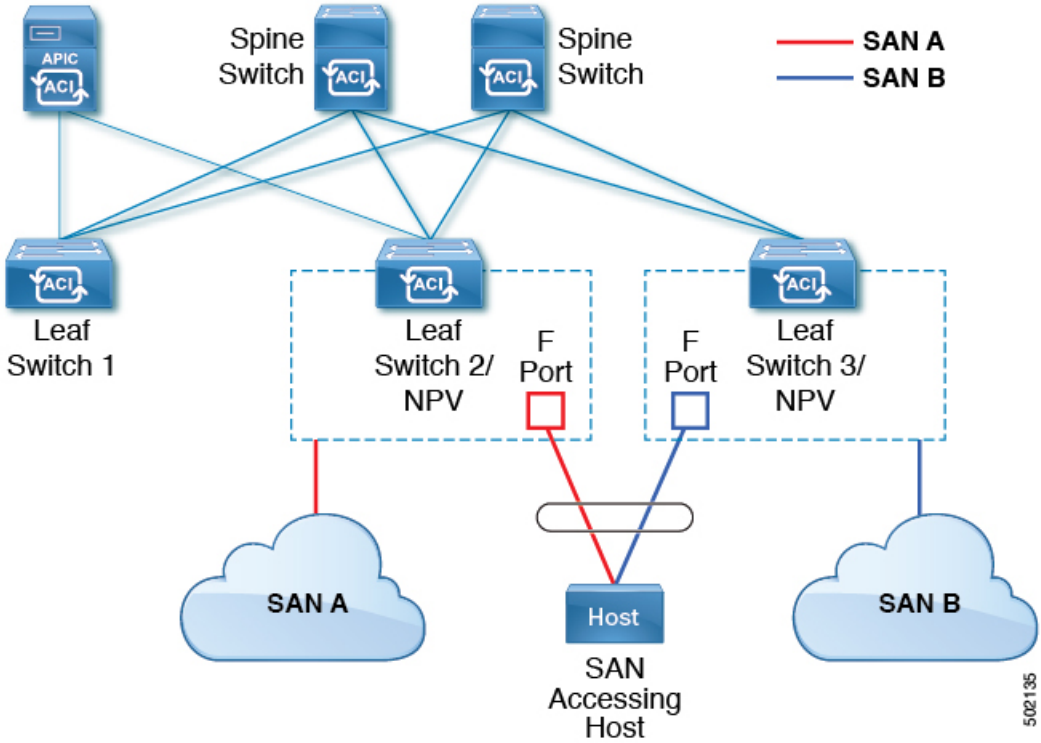
- Use FC NPV traffic management only when automatic traffic engineering does not meet your network requirements.
- You do not need to configure traffic maps for all server interfaces. By default, FC NPV will use automatic traffic management.
- Server interfaces configured to use a set of NP uplink interfaces cannot use any other available NP uplink interfaces, even if none of the configured interfaces are available.
- When disruptive load balancing is enabled, a server interface may be moved from one NP uplink to another NP uplink. Moving between NP uplink interfaces requires FC NPV to relogin to the core switch, causing traffic disruption.
- To link a set of servers to a specific core switch, associate the server interfaces with a set of NP uplink interfaces that all connect to that core switch.
- Configure Persistent FC IDs on the core switch and use the traffic map feature to direct server interface traffic onto NP uplinks that all connect to the associated core switch.
- When initially configuring traffic map pinning, you must shut the server host port before configuring the first traffic map.
- If traffic mapping is configured for more than one uplink, when removing the traffic map through which a host has logged in, you must first shut the host before removing the traffic map.
- While configuring a traffic map for an FCoE host behind a FEX, you can map one host to either multiple FCoE NP/uplinks (VFC or VFC-PO) or to a single Fibre Channel/SAN port channel NP/uplink.

Note When a server is statically mapped to an external interface, the server traffic is not redistributed in the event that the external interface becomes down for any reason.

SAN A/B Separation

SAN A and SAN B separation ensures that SAN connectivity is available even if one of the fabric components fails. SAN A and SAN B separation can be achieved physically or logically by separating the VSANs that are carried across the fabric.

Figure 32: SAN A/B Separation



SAN Port Channels

About SAN Port Channels

- A SAN port channel is a logical interface that combines a set of FC interfaces connected to the same Fibre Channel node and operates as one link.
- SAN port channels support bandwidth utilization and availability.
- SAN port channels on Cisco ACI switches are used to connect to FC core switches and to provide optimal bandwidth utilization and transparent failover between the uplinks of a VSAN.

SAN Port Channel Guidelines and Limitations

- The maximum number of active port channels (SAN port channels plus VFC uplink/NP port channels) on the Cisco ACI switch is seven. Any additional configured port channels remain in the **errdisabled** state until you shut or delete one of the existing active port channels. After you shut/delete an existing active port channel, shut/no shut the **errdisabled** port channel to bring it up.
- The maximum number of FC interfaces that can be combined into a SAN port channel is limited to 16.
- The default channel mode on Cisco ACI switches for SAN port channels is **active**; this cannot be changed.
- When a SAN port channel is connected to a Cisco FC core switch, only channel mode active is supported. Channel mode active must be configured on the Cisco FC core switch.

About SAN Port Channel Modes

A SAN port channel is configured with channel mode active by default. When active, the member ports initiate port channel protocol negotiation with the peer port regardless of the channel-group mode of the peer port. If the peer port, while configured in a channel group, does not support the port-channel protocol, or responds with a nonnegotiable status, the port channel is disabled. The active port channel mode allows automatic recovery without explicitly enabling and disabling the port-channel-member ports at either end.

Fibre Channel N-Port Virtualization Guidelines and Limitations

When configuring Fibre Channel N-Port Virtualization (NPV), note the following guidelines and limitations:

- Fibre Channel NP ports support trunk mode, but Fibre Channel F ports do not.
- On a trunk Fibre Channel port, internal login happens on the highest VSAN.
- On the core switch, the following features must be enabled:

```
feature npiv
feature fport-channel-trunk
```

- To use an 8G uplink speed, you must configure the IDLE fill pattern on the core switch.

Note Following is an example of configuring IDLE fill pattern on a Cisco MDS switch:

```
Switch(config)# int fc2/3
Switch(config)# switchport fill-pattern IDLE speed 8000
Switch(config)# show run int fc2/3

interface fc2/3
switchport speed 8000
switchport mode NP
switchport fill-pattern IDLE speed 8000
no shutdown
```

- You can use ports 1 through 48 for Fibre Channel configuration. Ports 49 through 54 cannot be Fibre Channel ports.
- If you convert a port from Ethernet to Fibre Channel or the other way around, you must reload the switch. Currently, you can convert only one contiguous range of ports to Fibre Channel ports, and this range must be a multiple of 4, ending with a port number that is a multiple of 4. For example, 1-4, 1-8, or 21-24.
- Fibre Channel Uplink (NP) connectivity to Brocade Port Blade Fibre Channel 16-32 is not supported when a Cisco N9K-93180YC-FX leaf switch port is configured in 8G speed.
- The selected port speed must be supported by the SFP. For example, because a 32G SFP supports 8/16/32G, a 4G port speed requires an 8G or 16G SFP. Because a 16G SFP supports 4/8/16G, a 32G port speed requires a 32G SFP.
- Speed autonegotiation is supported. The default speed is 'auto'.
- You cannot use Fibre Channel on 40G and breakout ports.
- FEX cannot be directly connected to FC ports.

- FEX HIF ports cannot be converted to FC.
- Reloading a switch after changing a switch's port profile configuration interrupts traffic through the data plane.

Fibre Channel N-Port Virtualization Supported Hardware

Fibre Channel N-Port Virtualization (FC NPV) is supported on the N9K-C93180YC-FX switch and only the following FC SFPs are supported:

- DS-SFP-FC8G-SW — 2/4/8G (2G is not a supported FC NPV port speed)
- DS-SFP-FC16G-SW — 4/8/16G (not compatible when FC NPV port speed is 32G)
- DS-SFP-FC32G-SW — 8/16/32G (not compatible when FC NPV port speed is 4G)

Supported NPIV core switches are Cisco Nexus 5000 Series, Nexus 6000 Series, Nexus 7000 Series (FCoE), and Cisco MDS 9000 Series Multilayer Switches.

Fibre Channel N-Port Virtualization Interoperability

The following table lists third party products with which the Fibre Channel N-port virtualization (FC NPV) feature of Cisco Application Policy Infrastructure Controller (APIC) was tested for interoperability.

Table 15: Third Party Products That Are Supported With FC NPV

Third Party Switch Vendor	Brocade
Third Party Hardware Model	DS-6620B
Third Party Software Release	8.2.1a
Cisco NX-OS Release	14.1(1) and later
Cisco Nexus 9000 Model	N9K-C93180YC-FX
Interoperability Mode	NA (NPV)
Cisco SFP Module	DS-SFP-FC32G-SW
Third Party SFP Module	Brocade-32G

Fibre Channel NPV GUI Configuration

Configuring a Native Fibre Channel Port Profile Using the GUI

This procedure configures a set of native Fibre Channel (FC) F ports for connecting to Fibre Channel hosts, such as servers.

To simplify the configuration, this procedure uses the **Configure an Interface, PC, and vPC** wizard.

Procedure

Step 1 On the APIC menu bar, navigate to **Fabric** > **Access Policies** > **Quickstart** and click *Configure an interface, PC, and vPC*.

Step 2 In the **Configured Switch Interfaces** toolbar, click **+** to create a switch profile. Perform the following actions:

This switch profile configures your server host ports. Another switch profile configures your uplink ports.

a) From the **Switches** drop-down list, choose your NPV leaf switch.

This action automatically creates a leaf switch profile. You can accept or change the name of the leaf switch profile in the **Switch Profile Name** text box.

b) Click the large green **+** on the ports drawing to open more interface settings.

c) For **Interface Type**, select **FC** to specify Fibre Channel host interface ports (F ports).

d) For **Interfaces**, enter a port range for the FC ports.

Only one contiguous range of ports can be converted to FC ports. This range must be a multiple of 4 ending with a port number that is a multiple of 4 (for example, 1-4, 1-8, and 21-32 are valid ranges).

This action creates an interface selector policy. You can accept or change the name of the policy in the **Interface Selector Name** text box.

Note Port conversion from Ethernet to FC requires a reload of the switch. After the interface policy is applied, a notification alarm appears in the GUI, prompting you to reload the switch. During a switch reload, communication to the switch is interrupted, resulting in timeouts when trying to access the switch.

e) From the **Policy Group Name** drop-down list, select **Create FC Interface Policy Group**.

f) In the **Create FC Interface Policy Group** dialog box, type a name in the **Name** field.

g) In the **Fibre Channel Interface Policy** drop-down list, select **Create Fibre Channel Interface Policy**.

h) In the **Create Fibre Channel Interface Policy** dialog box, type a name in the **Name** field and configure the following settings:

Field	Setting
Port Mode	For host interfaces, select **F**.
Trunk Mode	For host interfaces, select **trunk-off**.
Speed	Select **auto** (default).
Receive Buffer Credit	Select **64**.

i) Click **Submit** to save the Fibre Channel interface policy and return to the **Create FC Interface Policy Group** dialog box.

j) From the **Attached Entity Profile** drop-down list, choose **Create Attachable Access Entity Profile**.

The attachable entity profile option specifies the interfaces where the leaf access port policy is deployed.

k) In the **Name** field, enter a name for the attachable entity policy.

l) In the **Domains (VMM, Physical, or External) To Be Associated To Interfaces** toolbar, click **+** to add a domain profile.
m) From the **Domain Profile** drop-down list, choose **Create Fibre Channel Domain**.
n) In the **Name** field, enter a name for the Fibre Channel domain.
o) From the **VSAN Pool** drop-down list, choose **Create VSAN Pool**.
p) In the **Name** field, enter a name for the VSAN pool.
q) In the **Encap Blocks** toolbar, click **+** to add a VSAN range.
r) In the **Create VSAN Ranges** dialog box, enter **From** and **To** VSAN numbers.
s) For **Allocation Mode**, select **Static Allocation** and click **OK**.
t) In the **Create VSAN Ranges** dialog box, click **Submit**.
u) In the **Create Fibre Channel Domain** dialog box, click **Submit**.

Note In the Fibre Channel Domain, when using native FC ports instead of FCoE, it is not necessary to configure a VLAN pool or VSAN attributes.

v) In the **Create Attachable Access Entity Profile** dialog box, click **Update** to select the Fibre Channel domain profile and click **Submit**.
w) In the **Create FC Policy Group** dialog box, click **Submit**.
x) In the **Configure Interface, PC, and vPC** dialog box, click **Save** to save this switch profile for your server host ports.

Note Port conversion from Ethernet to FC requires a reload of the switch. After the interface policy is applied, a notification alarm appears in the GUI, prompting you to reload the switch. During a switch reload, communication to the switch is interrupted, resulting in timeouts when trying to access the switch.

In **Fabric** > **Access Policies** > **Switches** > **Leaf Switches** > **Profiles** > ***name*** , the Fibre Channel port profile appears in the **Associated Interface Selector Profiles** list in the **Leaf Profiles** work pane.

What to do next

- Configure a Fibre Channel uplink connection profile.
- Deploy the server ports and uplink ports in a tenant to connect to a Fibre Channel core switch.

Configuring a Native FC Port Channel Profile Using the GUI

This procedure configures a native Fibre Channel port channel (FC PC) profile for an uplink connection to a Fibre Channel core switch.

Note This procedure can also be performed using the **Configure Interface, PC, and vPC** wizard.

Before you begin

Configure your uplink connections, including an attachable entity profile.

Procedure

Step 1 Expand **Fabric** > **Access Policies** > **Interfaces** > **Leaf Interfaces** > **Profiles**.

Step 2 Right click **Profiles** and click **Create Leaf Interface Profile**.

Step 3 In the **Create Leaf Interface Profile** dialog box, perform the following steps:

a) In the **Name** field, enter a name for the leaf interface profile.
b) In the **Interface Selectors** toolbar, click **+** to open the **Create Access Port Selector** dialog box.
c) In the **Name** field, enter a name for the port selector.
d) In the **Interface IDs** field, enter a port range for the FC PC ports.

The port channel can have a maximum of 16 ports.

Only one contiguous range of ports can be converted to FC ports. This range must be a multiple of 4 ending with a port number that is a multiple of 4 (for example, 1-4, 1-8, and 21-32 are valid ranges).

Note Port conversion from Ethernet to FC requires a reload of the switch. After the interface policy is applied, a notification alarm appears in the GUI, prompting you to reload the switch manually. During a switch reload, communication to the switch is interrupted, resulting in timeouts when trying to access the switch.

e) From the **Interface Policy Group** drop-down list, choose **Create FC PC Interface Policy Group**.
f) In the **Name** field, enter a name for the FC PC interface policy group.
g) From the **Fibre Channel Interface Policy** drop-down list, choose **Create Fibre Channel Interface Policy**.
h) In the **Name** field, enter a name for the FC PC interface policy.
i) In the **Create Interface FC Policy** dialog box, type a name in the **Name** field and configure the following settings:

Field	Setting
Port Mode	For uplink interfaces, select **NP**.
Trunk Mode	For uplink interfaces, select **trunk-on**.

j) Click **Submit** to save the FC PC interface policy and return to the **Create FC PC Interface Policy Group** dialog box.
k) From the **Port Channel Policy** drop-down list, choose **Create Port Channel Policy**.
l) In the **Name** field, enter a name for the port channel policy.

The other settings in this menu can be ignored.

m) Click **Submit** to save the port channel policy and return to the **Create FC PC Interface Policy Group** dialog box.
n) From the **Attached Entity Profile** drop-down list, choose the existing attachable entity profile.
o) Click **Submit** to return to the **Create Access Port Selector** dialog box.
p) Click **OK** to return to the **Create Leaf Interface Profile** dialog box.
q) Click **OK** to return to the **Leaf Interfaces - Profiles** work pane.

Step 4 Expand **Fabric** > **Access Policies** > **Switches** > **Leaf Switches** > **Profiles**.

Step 5 Right click the leaf switch profile that you created and click **Create Interface Profile**.

Step 6 In the **Create Interface Profile** dialog box, perform the following steps:

a) From the **Interface Select Profile** drop-down list, choose the leaf interface profile that you created for the port channel.
b) Click **Submit** to return to the **Leaf Interfaces - Profiles** work pane.

Note Port conversion from Ethernet to FC requires a reload of the switch. After the interface policy is applied, a notification alarm appears in the GUI, prompting you to reload the switch. During a switch reload, communication to the switch is interrupted, resulting in timeouts when trying to access the switch.

In **Fabric** > **Access Policies** > **Switches** > **Leaf Switches** > **Profiles** > ***name*** , the FC port channel profile appears in the **Associated Interface Selector Profiles** list in the work pane.

What to do next

Deploy the server ports and uplink ports in a tenant to connect to a Fibre Channel core switch.

Deploying Fibre Channel Ports

This procedure activates the Fibre Channel server host ports and uplink ports.

Before you begin

- Configure Fibre Channel (FC) server host port profiles (F ports).
- Configure FC uplink port profiles (NP or TNP ports).
- Configure a leaf switch profile that includes two associated interface selector profiles — one for host ports and one for uplink ports.

Procedure

Step 1 Expand **Tenants** > **Tenant** ***name*** > **Application Profiles**

If the tenant does not exist, you must create a tenant.

Step 2 Right click **Application Profiles**, click **Create Application Profile**, and perform the following actions:
a) In the **Name** field, enter a name for the application profile.
b) Click **Submit**.

Step 3 Expand **Tenants** > **Tenant** ***name*** > **Application Profiles** > ***name*** > **Application EPGs**

Step 4 Right click **Application EPGs** and click **Create Application EPG**, and perform the following actions:

Step 5 In the **Create Application EPG** dialog box, perform the following actions:
a) In the **Name** field, enter a name for the application EPG.
b) Configure the following settings:

Field	Setting
Intra EPG Isolation	Select **Unenforced**.
Preferred Group Member	Select **Exclude**.

Field	Setting
Flood on Encapsulation	Select **Disabled**.

c) From the **Bridge Domain** drop-down list, select **Create Bridge Domain**.
d) In the **Name** field, enter a name for the bridge domain.
e) For **Type**, select **fc** to specify a Fibre Channel bridge domain.
f) From the **VRF** drop-down list, select **Create VRF**.
g) In the **Name** field, enter a name for the VRF.
h) Click **Submit** to return to the **Create Bridge Domain** dialog box.
i) Click **Next**, then **Next**, then **Finish** to return to the **Create Application EPG** dialog box.
j) Click **Finish**.

Step 6 Expand **Tenants** > **Tenant *name*** > **Application Profiles** > ***name*** > **Application EPGs** > ***name*** > **Domains (VMs and Bare-Metals)**.

Step 7 Right click **Domains (VMs and Bare-Metals)** and click **Add Fibre Channel Domain Association**, and perform the following actions:

a) From the **Fibre Channel Domain Profile** drop-down list, select the Fibre Channel domain that you created when you configured your host ports.
b) Click **Submit**.

Step 8 Expand **Tenants** > **Tenant *name*** > **Application Profiles** > ***name*** > **Application EPGs** > ***name*** > **Fibre Channel (Paths)** and perform the following actions:

This step deploys the server host ports.

a) Right click **Fibre Channel (Paths)** and click **Deploy Fibre Channel**.
b) In the **Path Type** control, click **Port**.
c) From the **Node** drop-down list, choose the leaf switch.
d) From the **Path** drop-down list, choose the leaf switch port that is configured as a server host port.
e) In the **VSAN** field, enter the port VSAN.
f) In the **VSAN Mode** control, click **Native**.
g) Verify that the **Type** is fcoe.
h) (Optional) If you require a traffic map, use the **Pinning Label** drop-down list.

Note If multiple uplink ports are available and you want this host port to always direct its FLOGI to a specific uplink, you can create a pinning profile (traffic map) to associate the host port to the uplink port. Otherwise, hosts are load-balanced among the available uplink ports.

i) Click **Submit**.
j) Repeat from **Step a** for each Fibre Channel host port.

Step 9 Expand **Tenants** > **Tenant *name*** > **Application Profiles** > ***name*** > **Application EPGs** > ***name*** > **Fibre Channel (Paths)** and perform the following actions:

This step deploys the uplink port channel.

a) Right click **Fibre Channel (Paths)** and click **Deploy Fibre Channel**.
b) In the **Path Type** control, click **Direct Port Channel**.
c) From the **Path** drop-down list, choose the uplink port channel.
d) In the **VSAN** field, enter the port default VSAN.
e) In the **VSAN Mode** control, click **Native** for a port VSAN or **Regular** for a trunk VSAN.

f) Verify that the **Type** is fcoe.
g) Click **Submit**.
h) Repeat from **Step a** for each Fibre Channel uplink port or port channel.

Configuring a Traffic Map for a Fibre Channel Port

In an application in which multiple uplink ports are available, server traffic by default is load-balanced among the available uplink ports. In some cases, it might be necessary to have a server send its login request (FLOGI) to one or more specific uplink ports or port channels. In such cases, you can create a pinning profile (traffic map) to associate the server port to those uplink ports or port channels.

This procedure assumes that you have already configured one or more server ports and one or more uplink ports or port channels. Because the server ports have already been configured, you must first shut (disable) any server port that is to be mapped to an uplink. After configuring the traffic map, re-enable the port.

Before you begin

This procedure assumes that the following items are already configured:

- Server ports (F ports) and uplink ports or port channels (NP ports)
- A tenant, including an application profile and application EPG

Note Before creating a pinning profile (traffic map), you must shut the server port that is to be mapped to an uplink.

Procedure

Step 1 In the **Fabric** > **Inventory** > **Pod *n*** > **Leaf *n*** > **Interfaces** > **FC Interfaces** work pane, select and disable the server interface port that is to be mapped to an uplink.

Step 2 Expand **Tenants** > **Tenant *name*** > **Application Profiles** > ***application profile name*** > **Application EPGs** > ***EPG name*** > **Fibre Channel (Paths)** and perform the following actions:

a) Right click **Fibre Channel (Paths)** and click **Deploy Fibre Channel**.
b) In the **Path Type** control, click **Port**.
c) From the **Node** drop-down list, choose the leaf switch.
d) From the **Path** drop-down list, choose the server port that is to be mapped to a specific uplink port.
e) In the **VSAN** field, enter the port default VSAN.
f) In the **VSAN Mode** control, click **Native**.
g) Verify that the **Type** is fcoe.
h) From the **Pinning Label** drop-down list, choose **Create Pinning Profile**.
i) In the **Name** field, enter a name for the traffic map.
j) In the **Path Type** control, click **Port** to connect to a single NP uplink port or **Direct Port Channel** to connect to an FC port channel.

   If you choose `Port` for the path type, you must also choose the leaf switch from the **Node** drop-down list that appears.

If you choose **`Direct Port Channel`** for the path type, you must also choose the FC PC you have defined in Interface Policy Group.

k) From the **Path** drop-down list, choose the uplink port or port channel to which the server port will be mapped.
l) Click **Submit** to return to the **Deploy Fibre Channel** dialog box.
m) Click **Submit**.

Step 3 In the **Fabric** > **Inventory** > **Pod *n*** > **Leaf *n*** > **Interfaces** > **FC Interfaces** work pane, select and re-enable the server interface port that is mapped to an uplink.

Fibre Channel NPV NX-OS-Style CLI Configuration

Configuring Fibre Channel Interfaces Using the CLI

On an NPV-enabled leaf switch, you can convert universal ports to Fibre Channel (FC) ports. The FC ports can be either F ports or NP ports, and NP ports can form a port channel.

Procedure

Step 1 Convert a range of ports from Ethernet to Fibre Channel.

Example:

```
apic1(config)# leaf 101
apic1(config-leaf)# slot 1
apic1(config-leaf-slot)# port 1 12 type fc
```

This example converts ports 1/1-12 on leaf 101 to Fibre Channel ports. The **[no]** form of the **port type fc** command converts the ports from Fibre Channel back to Ethernet.

Note The conversion of ports takes place only after a reboot of the leaf switch.

Currently only one contiguous range of ports can be converted to FC ports, and this range must be a multiple of 4 ending with a port number that is a multiple of 4 (for example, 1-4, 1-8, or 21-24).

Step 2 Configure all Fibre channel interfaces.

Example:

```
apic1(config)# leaf 101
apic1(config-leaf)# interface fc 1/1
apic1(config-leaf-fc-if)# switchport mode [f | np]
apic1(config-leaf-fc-if)# switchport rxbbcredit <16-64>
apic1(config-leaf-fc-if)# switchport speed [16G | 32G | 4G | 8G | auto | unknown]
apic1(config-leaf-fc-if)# switchport trunk-mode [ auto | trunk-off | trunk-on | un-init]
apic1(config-leaf-fc-if)# switchport [trunk allowed] vsan <1-4093> tenant <name>  \
                              application <name> epg <name>
```

Note FC host interfaces (F ports) do not support a speed configuration of 8Gbps.

A FC interface can be configured in access mode or trunk mode. To configure the FC port in access mode, use the following command format:

Example:

```
apic1(config-leaf-fc-if)# switchport vsan 2 tenant t1 application a1 epg e1
```

To configure a FC port in trunk mode, use the following command format:

Example:

```
apic1(config-leaf-fc-if)# switchport trunk allowed vsan 4 tenant t1 application a1 epg e1
```

To configure a FC port channel, configure a FC port interface template and apply it to FC interfaces that will be members of the FC port-channel.

The port channel can have a maximum of 16 members.

Example:

```
apic1(config)# template fc-port-channel my-fc-pc
apic1(config-fc-po-ch-if)# lacp max-links 4
apic1(config-fc-po-ch-if)# lacp min-links 1
apic1(config-fc-po-ch-if)# vsan-domain member dom1
apic1(config-fc-po-ch-if)# exit
apic1(config)# leaf 101
apic1(config-leaf)# interface fc 1/1-2
apic1(config-leaf-fc-if)# fc-channel-group my-fc-pc
apic1(config-leaf-fc-if)# exit
apic1(config-leaf)# interface fc-port-channel my-fc-pc
apic1(config-leaf-fc-pc)# switchport mode [f | np]
apic1(config-leaf-fc-pc)# switchport rxbbcredit <16-64>
apic1(config-leaf-fc-pc)# switchport speed [16G | 32G | 4G | 8G | auto | unknown]
apic1(config-leaf-fc-pc)# switchport trunkmode [ auto | trunk-off | trunk-on | un-init]
```

Configuring Fibre Channel NPV Policies Using the CLI

Before you begin

Leaf switch ports to be used in an NPV application have been converted to Fibre Channel (FC) ports.

Procedure

Step 1 Create a template of a Fibre Channel F port policy group.

Example:

```
apic1(config)# template fc-policy-group my-fc-policy-group-f-ports
apic1(config-fc-pol-grp-if)# vsan-domain member dom1
apic1(config-fc-pol-grp-if)# switchport mode f
apic1(config-fc-pol-grp-if)# switchport trunk-mode trunk-off
```

You can configure other switchport settings, such as speed.

Step 2 Create a template of a Fibre Channel NP port policy group.

Example:

```
apic1(config)# template fc-policy-group my-fc-policy-group-np-ports
apic1(config-fc-pol-grp-if)# vsan-domain member dom1
apic1(config-fc-pol-grp-if)# switchport mode np
apic1(config-fc-pol-grp-if)# switchport trunk-mode trunk-on
```

You can configure other switchport settings, such as speed.

Step 3 Create a fabric-wide Fibre Channel policy.

Example:

```
apic1(config)# template fc-fabric-policy my-fabric-fc-policy
apic1(config-fc-fabric-policy)# fctimer e-d-tov 1000
apic1(config-fc-fabric-policy)# fctimer r-a-tov 5000
apic1(config-fc-fabric-policy)# fcoe fcmap 0E:FC:01
```

Step 4 Create a Fibre Channel port channel policy.

Example:

```
apic1(config)# template fc-port-channel my-fc-pc
apic1(config-fc-po-ch-if)# lacp max-links 4
apic1(config-fc-po-ch-if)# lacp min-links 1
apic1(config-fc-po-ch-if)# vsan-domain member dom1
```

Step 5 Create a leaf-wide Fibre Channel policy group.

Example:

```
apic1(config)# template fc-leaf-policy my-fc-leaf-policy
apic1(config-fc-leaf-policy)# npv auto-load-balance disruptive
apic1(config-fc-leaf-policy)# fcoe fka-adv-period 10
```

Note The policy commands that are shown here are only examples, and are not mandatory settings.

Step 6 Create a leaf policy group.

```
apic1(config)# template leaf-policy-group lpg1
apic1(config-leaf-policy-group)# inherit fc-fabric-policy my-fabric-fc-policy
apic1(config-leaf-policy-group)# inherit fc-leaf-policy my-fc-leaf-policy
```

The leaf policy group is created by inheriting FC-related policies.

Step 7 Create a leaf profile to apply a leaf-policy-group to a leaf-group.

Example:

```
apic1(config)# leaf-profile my-leaf-profile
apic1(config-leaf-profile)# leaf-group my-leaf-group
```

```
apic1(config-leaf-group)# leaf 101
apic1(config-leaf-group)# leaf-policy-group lpg1
```

This example applies fabric-wide FC policies and leaf-wide FC policies that are grouped into a leaf policy group lpg1 to leaf 101.

Step 8 Create a leaf interface profile and apply a fc-policy-group to a set of FC interfaces.

Example:

```
apic1(config)# leaf-interface-profile my-leaf-interface-profile
apic1(config-leaf-if-profile)# leaf-interface-group my-leaf-interface-group
apic1(config-leaf-if-group)# fc-policy-group my-fc-policy-group-f-ports
apic1(config-leaf-if-group)# interface fc 1/1-10
```

Configuring an NPV Traffic Map Using the CLI

This procedure maps traffic coming from a FC/FCoE server (host) interface to a FC/FCoE external (uplink) interface configured in NP mode.

Before you begin

All server interfaces must be F ports and all uplink interfaces must be NP ports.

Procedure

Example:

```
apic1(config)# leaf 101
apic1(config-leaf)# npv traffic-map server-interface  \
    { vfc <slot/port> | vfc-po <po-name> |fc <slot/port> } \
    label <name> tenant <tn> app <ap> epg <ep>
apic1(config-leaf)# npv traffic-map external-interface \
    { vfc <slot/port> | vfc-po <po-name> |fc <slot/port> } \
    tenant <tn> label <name>
```

Example:

```
apic1(config)# leaf 101
apic1(config-leaf)# npv traffic-map server-interface vfc 1/1 label serv1 tenant t1 app ap1
 epg epg1
apic1(config-leaf)# npv traffic-map external-interface vfc-po my-fc-pc tenant t1 label ext1
```

Fibre Channel NPV REST API Configuration

Configuring FC Connectivity Using the REST API

You can configure FC-enabled interfaces and EPGs accessing those interfaces using the FC protocol with the REST API.

Procedure

Step 1 To create a VSAN pool, send a post with XML such as the following example. The example creates VSAN pool myVsanPool1 and specifies the range of VSANs to be included as vsan-50 to vsan-60:

Example:

```
https://apic-ip-address/api/mo/uni/infra/vsanns-[myVsanPool1]-static.xml

<fvnsVsanInstP allocMode="static" name="myVsanPool1">
  <fvnsVsanEncapBlk from="vsan-50" name="encap" to="vsan-60"/>
</fvnsVsanInstP>
```

Step 2 To create a Fibre Channel domain, send a post with XML such as the following example. The example creates Fibre Channel domain (VSAN domain) myFcDomain1 and associates it with the VSAN pool myVsanPool1:

Example:

```
https://apic-ip-address/api/mo/uni/fc-myFcDomain1.xml

<fcDomP name="myFcDomain1">
  <fcRsVsanNs tDn="uni/infra/vsanns-[myVsanPool1]-static"/>
</fcDomP>
```

Step 3 To create an Attached Entity Policy (AEP) for the FC ports, send a post with XML such as the following example. The example creates the AEP myFcAEP1 and associates it with the Fibre Channel domain myFcDomain1:

Example:

```
https://apic-ip-address/api/mo/uni.xml

<polUni>
<infraInfra>
    <infraAttEntityP name="myFcAEP1">
        <infraRsDomP tDn="uni/fc-myFcDomain1"/>
    </infraAttEntityP>
</infraInfra>
</polUni>
```

Step 4 To create a FC interface policy and a policy group for server host ports, send a post with XML. This example executes the following requests:

- Creates a FC interface policy myFcHostIfPolicy1 for server host ports. These are F ports with no trunking.
- Creates a FC interface policy group myFcHostPortGroup1 that includes the FC host interface policy myFcHostIfPolicy1.

- Associates the policy group to the FC interface policy to convert these ports to FC ports.
- Creates a host port profile myFcHostPortProfile.
- Creates a port selector myFcHostSelector that specifies ports in range 1/1-8.
- Creates a node selector myFcNode1 that specifies leaf node 104.
- Creates a node selector myLeafSelector that specifies leaf node 104.
- Associates the host ports to the leaf node.

Example:

```
https://apic-ip-address/api/mo/uni.xml

<polUni>
    <infraInfra>
       <fcIfPol name="myFcHostIfPolicy1" portMode="f" trunkMode="trunk-off" speed="auto"/>

        <infraFuncP>
            <infraFcAccPortGrp name="myFcHostPortGroup1">
                <infraRsFcL2IfPol tnFcIfPolName="myFcHostIfPolicy1" />
            </infraFcAccPortGrp>
        </infraFuncP>
        <infraAccPortP name="myFcHostPortProfile">
            <infraHPortS name="myFcHostSelector" type="range">
                <infraPortBlk name="myHostPorts" fromCard="1" toCard="1" fromPort="1"
toPort="8" />
               <infraRsAccBaseGrp tDn="uni/infra/funcprof/fcaccportgrp-myFcHostPortGroup1"
 />
            </infraHPortS>
        </infraAccPortP>
        <infraNodeP name="myFcNode1">
            <infraLeafS name="myLeafSelector" type="range">
                <infraNodeBlk name="myLeaf104" from_="104" to_="104" />
            </infraLeafS>
            <infraRsAccPortP tDn="uni/infra/accportprof-myHostPorts" />
        </infraNodeP>
    </infraInfra>
</polUni>
```

Note When this configuration is applied, a switch reload is required to bring up the ports as FC ports.

Currently only one contiguous range of ports can be converted to FC ports, and this range must be multiple of 4 ending with a port number that is multiple of 4. Examples are 1-4, 1-8, or 21-24.

Step 5 To create a FC uplink port interface policy and a policy group for uplink port channels, send a post with XML. This example executes the following requests:

- Creates a FC interface policy myFcUplinkIfPolicy2 for uplink ports. These are NP ports with trunking enabled.
- Creates a FC interface bundle policy group myFcUplinkBundleGroup2 that includes the FC uplink interface policy myFcUplinkIfPolicy2.
- Associates the policy group to the FC interface policy to convert these ports to FC ports.
- Creates an uplink port profile myFcUplinkPortProfile.
- Creates a port selector myFcUplinkSelector that specifies ports in range 1/9-12.

- Associates the host ports to the leaf node 104.

Example:

```
https://apic-ip-address/api/mo/uni.xml

<polUni>
    <infraInfra>
       <fcIfPol name="myFcUplinkIfPolicy2" portMode="np" trunkMode="trunk-on" speed="auto"/>

        <infraFuncP>
            <infraFcAccBndlGrp name="myFcUplinkBundleGroup2">
                <infraRsFcL2IfPol tnFcIfPolName="myFcUplinkIfPolicy2" />
            </infraFcAccBndlGrp>
        </infraFuncP>
        <infraAccPortP name="myFcUplinkPortProfile">
            <infraHPortS name="myFcUplinkSelector" type="range">
                <infraPortBlk name="myUplinkPorts" fromCard="1" toCard="1" fromPort="9"
toPort="12" />
                <infraRsAccBaseGrp
tDn="uni/infra/funcprof/fcaccportgrp-myFcUplinkBundleGroup2" />
            </infraHPortS>
        </infraAccPortP>
        <infraNodeP name="myFcNode1">
            <infraLeafS name="myLeafSelector" type="range">
                <infraNodeBlk name="myLeaf104" from_="104" to_="104" />
            </infraLeafS>
            <infraRsAccPortP tDn="uni/infra/accportprof-myUplinkPorts" />
        </infraNodeP>
    </infraInfra>
</polUni>
```

Note When this configuration is applied, a switch reload is required to bring up the ports as FC ports.

Currently only one contiguous range of ports can be converted to FC ports, and this range must be multiple of 4 ending with a port number that is multiple of 4. Examples are 1-4, 1-8, or 21-24.

Step 6 To create the tenant, application profile, EPG and associate the FC bridge domain with the EPG, send a post with XML such as the following example. The example creates a bridge domain myFcBD1 under a target tenant configured to support FC and an application EPG epg1. It associates the EPG with Fibre Channel domain myFcDomain1 and a Fibre Channel path to interface 1/7 on leaf switch 104. Each interface is associated with a VSAN.

Example:

```
https://apic-ip-address/api/mo/uni/tn-tenant1.xml

<fvTenant name="tenant1">
  <fvCtx name="myFcVRF"/>
  <fvBD name="myFcBD1" type="fc">
    <fvRsCtx tnFvCtxName="myFcVRF"/>
  </fvBD>
  <fvAp name="app1">
    <fvAEPg name="epg1">
      <fvRsBd tnFvBDName="myFcBD1"/>
      <fvRsDomAtt tDn="uni/fc-myFcDomain1"/>
      <fvRsFcPathAtt tDn="topology/pod-1/paths-104/pathep-[fc1/1]" vsan="vsan-50"
vsanMode="native"/>
      <fvRsFcPathAtt tDn="topology/pod-1/paths-104/pathep-[fc1/2]" vsan="vsan-50"
vsanMode="native"/>
    </fvAEPg>
  </fvAp>
```

```
</fvTenant>
```

Step 7 To create a traffic map to pin server ports to uplink ports, send a post with XML such as the following example. The example creates a traffic map to pin server port vFC 1/47 to uplink port FC 1/7:

Example:

```
https://apic-ip-address/api/mo/uni/tn-tenant1.xml

<fvTenant name="tenant1">
  <fvAp name="app1">
    <fvAEPg name="epg1">
      <fvRsFcPathAtt tDn="topology/pod-1/paths-104/pathep-[eth1/47]" vsan="vsan-50"
vsanMode="native">
        <fcPinningLbl name="label1"/>
      </fvRsFcPathAtt>
    </fvAEPg>
  </fvAp>
</fvTenant>

https://apic-ip-address/api/mo/uni/tn-vfc_t1.xml

<fvTenant name="tenant1">
  <fcPinningP name="label1">
    <fcRsPinToPath tDn="topology/pod-1/paths-104/pathep-[fc1/7]"/>
  </fcPinningP>
</fvTenant>
```

Note If traffic map pinning is configured for the first time, the server host port must be shut before configuring the first traffic map.

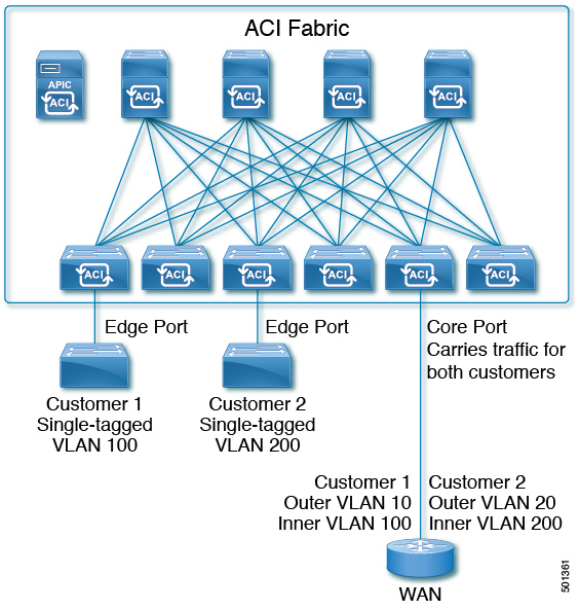
CHAPTER 10

802.1Q Tunnels

This chapter contains the following sections:

About ACI 802.1Q Tunnels

Figure 33: ACI 802.1Q Tunnels



With Cisco ACI and Cisco APIC Release 2.2(1x) and higher, you can configure 802.1Q tunnels on edge (tunnel) ports to enable point-to-multi-point tunneling of Ethernet frames in the fabric, with Quality of Service (QoS) priority settings. A **Dot1q Tunnel** transports untagged, 802.1Q tagged, and 802.1ad double-tagged frames as-is across the fabric. Each tunnel carries the traffic from a single customer and is associated with a single bridge domain. ACI front panel ports can be part of a **Dot1q Tunnel**. Layer 2 switching is done based on Destination MAC (DMAC) and regular MAC learning is done in the tunnel. Edge-port **Dot1q Tunnels**

are supported on second-generation (and later) Cisco Nexus 9000 series switches with "EX" on the end of the switch model name.

With Cisco ACI and Cisco APIC Release 2.3(x) and higher, you can also configure multiple 802.1Q tunnels on the same core port to carry double-tagged traffic from multiple customers, each distinguished with an access encapsulation configured for each 802.1Q tunnel. You can also disable MAC Address Learning on 802.1Q tunnels. Both edge ports and core ports can belong to an 802.1Q tunnel with access encapsulation and disabled MAC Address Learning. Both edge ports and core ports in **Dot1q Tunnels** are supported on third-generation Cisco Nexus 9000 series switches with "FX" and "FX2" on the end of the switch model name.

Terms used in this document may be different in the **Cisco Nexus 9000 Series** documents.

Table 16: 802.1Q Tunnel Terminology

ACI Documents	Cisco Nexus 9000 Series Documents
Edge Port	Tunnel Port
Core Port	Trunk Port

The following guidelines and restrictions apply:

- Layer 2 tunneling of VTP, CDP, LACP, LLDP, and STP protocols is supported with the following restrictions:
  - Link Aggregation Control Protocol (LACP) tunneling functions as expected only with point-to-point tunnels using individual leaf interfaces. It is not supported on port-channels (PCs) or virtual port-channels (vPCs).
  - CDP and LLDP tunneling with PCs or vPCs is not deterministic; it depends on the link it chooses as the traffic destination.
  - To use VTP for Layer 2 protocol tunneling, CDP must be enabled on the tunnel.
  - STP is not supported in an 802.1Q tunnel bridge domain when Layer 2 protocol tunneling is enabled and the bridge domain is deployed on Dot1q Tunnel core ports.
  - ACI leaf switches react to STP TCN packets by flushing the end points in the tunnel bridge domain and flooding them in the bridge domain.
  - CDP and LLDP tunneling with more than two interfaces flood packets on all interfaces.
  - With Cisco APIC Release 2.3(x) or higher, the destination MAC address of Layer 2 protocol packets tunneled from edge to core ports is rewritten as 01-00-0c-cd-cd-d0 and the destination MAC address of Layer 2 protocol packets tunneled from core to edge ports is rewritten with the standard default MAC address for the protocol.
- If a PC or vPC is the only interface in a **Dot1q Tunnel** and it is **deleted** and reconfigured, remove the association of the PC/VPC to the **Dot1q Tunnel** and reconfigure it.
- For 802.1Q tunnels deployed on switches that have EX in the product ID, Ethertype combinations of 0x8100+0x8100, 0x8100+0x88a8, 0x88a8+0x8100, and 0x88a8+0x88a8 for the first two VLAN tags are not supported.

  If the tunnels are deployed on a combination of EX and FX or later switches, then this restriction still applies.

If the tunnels are deployed only on switches that have FX or later in the product ID, then this restriction does not apply.

- For core ports, the Ethertypes for double-tagged frames must be 0x8100 followed by 0x8100.
- You can include multiple edge ports and core ports (even across leaf switches) in a **Dot1q Tunnel**.
- An edge port may only be part of one tunnel, but a core port can belong to multiple Dot1q tunnels.
- With Cisco APIC Release 2.3(x) and higher, regular EPGs can be deployed on core ports that are used in 802.1Q tunnels.
- L3Outs are not supported on interfaces enabled for **Dot1q Tunnels**.
- FEX interfaces are not supported as members of a **Dot1q Tunnel**.
- Interfaces configured as breakout ports do not support 802.1Q tunnels.
- Interface-level statistics are supported for interfaces in **Dot1q Tunnels**, but statistics at the tunnel level are not supported.

Configuring 802.1Q Tunnels Using the GUI

Configuring 802.1Q Tunnel Interfaces Using the APIC GUI

Configure the interfaces that will use the tunnel, with the following steps:

Before you begin

Create the tenant that will be using the tunnel.

Procedure

Step 1 On the menu bar, click **Fabric** > **Access Policies**.

Step 2 On the Navigation bar, click **Policies** > **Interface** > **L2 Interface**.

Step 3 Right-click **L2 Interface**, select **Create L2 Interface Policy**, and perform the following actions:

a) In the **Name** field, type a name for the Layer 2 Interface policy.

b) Optional. Add a description of the policy. We recommended that you describe the purpose for the L2 Interface Policy.

c) To create an interface policy that enables an interface to be used as an edge port in a **Dot1q Tunnel**, in the **QinQ** field, click **edgePort**.

d) To create an interface policy that enables an interface to be used as a core port in **Dot1q Tunnels**, in the **QinQ** field, click **corePort**.

Step 4 Apply the L2 Interface policy to a Policy Group with the following steps:

a) Click on **Fabric** > **Access Policies** > **Interfaces** > **Leaf Interfaces** and expand **Policy Groups**.

b) Right-click **Leaf Access Port**, **PC Interface**, or **VPC Interface** and choose one of the following, depending on the type of interface you are configuring for the tunnel.

- **Create Leaf Access Port Policy Group**

- **Create PC Policy Group**
- **Create VPC Policy Group**

c) In the resulting dialog box, perform the following actions:

- In the **Name** field, type a name for the policy group.

  Optional. Add a description of the policy group. We recommend that you describe the purpose of the policy group.

- In the **L2 Interface Policy** field, click on the down-arrow and choose the L2 Interface Policy that you previously created.
- If you are tunneling the CDP Layer 2 Tunneled Protocol, click on the **CDP Policy** down-arrow, and in the policy dialog box add a name for the policy, disable the Admin State and click **Submit.**.
- If you are tunneling the LLDP Layer 2 Tunneled Protocol, click on the **LLDP Policy** down-arrow, and in the policy dialog box add a name for the policy, disable the Transmit State and click **Submit.**
- Click **Submit**.

Step 5 Create a Leaf Interface Profile with the following steps:

a) Click on **Fabric** > **Access Policies** > **Interfaces** > **Leaf Interfaces** > **Profiles**.

b) Right-click on **Profiles**, select **Create Leaf Interface Profile**, and perform the following steps:

- In the **Name** field, type a name for the **Leaf Interface Profile**.

  Optional. Add a description.

- In the **Interface Selectors** field, click the **+**, and enter the following information:
  - In the **Name** field, type a name for the interface selector.

    Optional. Add a description.

  - In the **Interface IDs** field, enter the **Dot1q Tunnel** interface or multiple interfaces to be included in the tunnel.
  - In the **Interface Policy Group** field, click on the down arrow and select the interface policy group that you previously created .

Step 6 To create a static binding of the tunnel configuration to a port, click on **Tenant** > **Networking** > **Dot1Q Tunnels**. Expand **Dot1Q Tunnels** and click on the **Dot1Q Tunnels** ***policy_name*** perviously created and perform the following actions:

a) Expand the **Static Bindings** table to open **Create Static Binding** dialog box.

b) In the **Port** field, select the type of port.

c) In the **Node** field, select a node from the drop-down.

d) In the **Path** field, select the interface path from the drop-down and click **Submit**.

Configuring 802.1Q Tunnels Using the NX-OS Style CLI

Configuring 802.1Q Tunnels Using the NX-OS Style CLI

Note You can use ports, port-channels, or virtual port channels for interfaces included in a **Dot1q Tunnel**. Detailed steps are included for configuring ports. See the examples below for the commands to configure edge and core port-channels and virtual port channels.

Create a **Dot1q Tunnel** and configure the interfaces for use in the tunnel using the NX-OS Style CLI, with the following steps:

Note **Dot1q Tunnels** must include 2 or more interfaces. Repeat the steps (or configure two interfaces together), to mark each interface for use in a **Dot1q Tunnel**. In this example, two interfaces are configured as edge-switch ports, used by a single customer.

Use the following steps to configure a **Dot1q Tunnel** using the NX-OS style CLI:

1. Configure at least two interfaces for use in the tunnel.
2. Create a **Dot1q Tunnel**.
3. Associate all the interfaces with the tunnel.

Before you begin

Configure the tenant that will use the **Dot1q Tunnel**.

Procedure

	Command or Action	Purpose
Step 1	**configure** **Example:** `apic1# configure`	Enters configuration mode.
Step 2	Configure two interfaces for use in an 802.1Q tunnel, with the following steps:	
Step 3	**leaf** *ID* **Example:** `apic1(config)# leaf 101`	Identifies the leaf where the interfaces of the **Dot1q Tunnel** will be located.
Step 4	**interface ethernet** *slot/port* **Example:**	Identifies the interface or interfaces to be marked as ports in a tunnel.

	Command or Action	Purpose
	`apic1(config-leaf)# interface ethernet 1/13-14`	

Step 5	**switchport mode dot1q-tunnel** {**edgePort** | **corePort**}
	Example:
	`apic1(config-leaf-if)# switchport mode dot1q-tunnel edgePort`
	`apic1(config-leaf-if)# exit`
	`apic1(config-leaf)# exit`
	`apic1(config)# exit`
		Marks the interfaces for use in an 802.1Q tunnel, and then leaves the configuration mode.
		The example shows configuring some interfaces for edge port use. Repeat steps 3 to 5 to configure more interfaces for the tunnel.

Step 6	Create an 802.1Q tunnel with the following steps:	
Step 7	**leaf** *ID* **Example:** `apic1(config)# leaf 101`	Returns to the leaf where the interfaces are located.
Step 8	**interface ethernet** *slot/port* **Example:** `apic1(config-leaf)# interface ethernet 1/13-14`	Returns to the interfaces included in the tunnel.

Step 9	**switchport tenant** *tenant-name* **dot1q-tunnel** *tunnel-name*
	Example:
	`apic1(config-leaf-if)# switchport tenant tenant64 dot1q-tunnel vrf64_edgetunnel`
	`apic1(config-leaf-if)# exit`
		Associates the interfaces to the tunnel and exits the configuration mode.

Step 10	Repeat steps 7 to 10 to associate other interfaces with the tunnel.	

Example: Configuring an 802.1Q Tunnel Using Ports with the NX-OS Style CLI

The example marks two ports as edge port interfaces to be used in a **Dot1q Tunnel**, marks two more ports to be used as core port interfaces, creates the tunnel, and associates the ports with the tunnel.

```
apic1# configure
apic1(config)# leaf 101
apic1(config-leaf)# interface ethernet 1/13-14
apic1(config-leaf-if)# switchport mode dot1q-tunnel edgePort
apic1(config-leaf-if)# exit
apic1(config-leaf)# exit
apic1(config)leaf 102
apic1(config-leaf)# interface ethernet 1/10, 1/21
apic1(config-leaf-if)# switchport mode dot1q-tunnel corePort
```

```
apic1(config-leaf-if)# exit
apic1(config-leaf)# exit

apic1(config)# tenant tenant64
apic1(config-tenant)# dot1q-tunnel vrf64_tunnel
apic1(config-tenant-tunnel)# l2protocol-tunnel cdp
apic1(config-tenant-tunnel)# l2protocol-tunnel lldp

apic1(config-tenant-tunnel)#  access-encap 200


apic1(config-tenant-tunnel)# mac-learning disable

apic1(config-tenant-tunnel)# exit
apic1(config-tenant)# exit
apic1(config)# leaf 101
apic1(config-leaf)# interface ethernet 1/13-14
apic1(config-leaf-if)# switchport tenant tenant64 dot1q-tunnel vrf64_tunnel
apic1(config-leaf-if)# exit
apic1(config-leaf)# exit
apic1(config)# leaf 102
apic1(config-leaf)# interface ethernet 1/10, 1/21
apic1(config-leaf-if)# switchport tenant tenant64 dot1q-tunnel vrf64_tunnel
apic1(config-leaf-if)# exit
apic1(config-leaf)# exit
```

Example: Configuring an 802.1Q Tunnel Using Port-Channels with the NX-OS Style CLI

The example marks two port-channels as edge-port 802.1Q interfaces, marks two more port-channels as core-port 802.1Q interfaces, creates a **Dot1q Tunnel**, and associates the port-channels with the tunnel.

```
apic1# configure
apic1(config)# tenant tenant64
apic1(config-tenant)# dot1q-tunnel vrf64_tunnel
apic1(config-tenant-tunnel)# l2protocol-tunnel cdp
apic1(config-tenant-tunnel)# l2protocol-tunnel lldp

apic1(config-tenant-tunnel)#  access-encap 200


apic1(config-tenant-tunnel)# mac-learning disable

apic1(config-tenant-tunnel)# exit
apic1(config-tenant)# exit
apic1(config)# leaf 101
apic1(config-leaf)# interface port-channel pc1
apic1(config-leaf-if)# exit
apic1(config-leaf)# interface ethernet 1/2-3
apic1(config-leaf-if)# channel-group pc1
apic1(config-leaf-if)# exit
apic1(config-leaf)# interface port-channel pc1
apic1(config-leaf-if)# switchport mode dot1q-tunnel edgePort
apic1(config-leaf-if)# switchport tenant tenant64 dot1q-tunnel vrf64_tunnel
apic1(config-tenant-tunnel)# exit
apic1(config-tenant)# exit
```

```
apic1(config)# leaf 102
apic1(config-leaf)# interface port-channel pc2
apic1(config-leaf-if)# exit
apic1(config-leaf)# interface ethernet 1/4-5
apic1(config-leaf-if)# channel-group pc2
apic1(config-leaf-if)# exit
apic1(config-leaf)# interface port-channel pc2
apic1(config-leaf-if)# switchport mode dot1q-tunnel corePort
apic1(config-leaf-if)# switchport tenant tenant64 dot1q-tunnel vrf64_tunnel
```

Example: Configuring an 802.1Q Tunnel Using Virtual Port-Channels with the NX-OS Style CLI

The example marks two virtual port-channels (vPCs) as edge-port 802.1Q interfaces for the**Dot1q Tunnel**, marks two more vPCs as core-port interfaces for the tunnel, creates the tunnel, and associates the virtual port-channels with the tunnel.

```
apic1# configure
apic1(config)# vpc domain explicit 1 leaf 101 102
apic1(config)# vpc context leaf 101 102
apic1(config-vpc)# interface vpc vpc1
apic1(config-vpc-if)# switchport mode dot1q-tunnel edgePort
apic1(config-vpc-if)# exit
apic1(config-vpc)# exit
apic1(config)# vpc domain explicit 1 leaf 103 104
apic1(config)# vpc context leaf 103 104
apic1(config-vpc)# interface vpc vpc2
apic1(config-vpc-if)# switchport mode dot1q-tunnel corePort
apic1(config-vpc-if)# exit
apic1(config-vpc)# exit
apic1(config)# tenant tenant64
apic1(config-tenant)# dot1q-tunnel vrf64_tunnel
apic1(config-tenant-tunnel)# l2protocol-tunnel cdp
apic1(config-tenant-tunnel)# l2protocol-tunnel lldp

apic1(config-tenant-tunnel)#  access-encap 200


apic1(config-tenant-tunnel)# mac-learning disable

apic1(config-tenant-tunnel)# exit
apic1(config-tenant)# exit
apic1(config)# leaf 103
apic1(config-leaf)# interface ethernet 1/6
apic1(config-leaf-if)# channel-group vpc1 vpc
apic1(config-leaf-if)# exit
apic1(config-leaf)# exit
apic1(config)# leaf 104
apic1(config-leaf)# interface ethernet 1/6
apic1(config-leaf-if)# channel-group vpc1 vpc
apic1(config-leaf-if)# exit
apic1(config-leaf)# exit
apic1(config-vpc)# interface vpc vpc1
apic1(config-vpc-if)# switchport tenant tenant64 dot1q-tunnel vrf64_tunnel
apic1(config-vpc-if)# exit
```

Configuring 802.1Q Tunnels Using the REST API

Configuring 802.1Q Tunnels With Ports Using the REST API

Create a **Dot1q Tunnel**, using ports, and configure the interfaces for it with steps such as the following examples.

Before you begin

Configure the tenant that will use the **Dot1q Tunnel**.

Procedure

Step 1 Create a **Dot1q Tunnel** using the REST API with XML such as the following example.

The example configures the tunnel with the LLDP Layer 2 tunneling protocol, adds the access encapsulation VLAN, and disables MAC learning in the tunnel.

Example:

```
<fvTnlEPg name="VRF64_dot1q_tunnel" qiqL2ProtTunMask="lldp" accEncap="vlan-10"
fwdCtrl="mac-learn-disable"  >
    <fvRsTnlpathAtt  tDn="topology/pod-1/paths-101/pathep-[eth1/13]"/>
</fvTnlEPg>
```

Step 2 Configure a Layer 2 Interface policy with static binding with XML such as the following example.

The example configures a Layer 2 interface policy for edge-switch ports. To configure a policy for core-switch ports, use `corePort` instead of `edgePort` in the `l2IfPol` MO.

Example:

```
<l2IfPol name="VRF64_L2_int_pol" qinq="edgePort"  />
```

Step 3 Apply the Layer 2 Interface policy to a Leaf Access Port Policy Group with XML such as the following example.

Example:

```
<infraAccPortGrp name="VRF64_L2_Port_Pol_Group" >
    <infraRsL2IfPol tnL2IfPolName="VRF64_L2_int_pol"/>
</infraAccPortGrp>
```

Step 4 Configure a Leaf Profile with an Interface Selector with XML such as the following example:

Example:

```
<infraAccPortP name="VRF64_dot1q_leaf_profile" >
    <infraHPortS name="vrf64_access_port_selector" type="range">
        <infraPortBlk name="block2" toPort="15" toCard="1" fromPort="13" fromCard="1"/>
        <infraRsAccBaseGrp tDn="uni/infra/funcprof/accportgrp-VRF64_L2_Port_Pol_Group" />
    </infraHPortS>
</infraAccPortP>
```

Example

The following example shows the port configuration for edge ports in two posts.

XML with Post 1:

```
<polUni>
  <infraInfra>
    <l2IfPol name="testL2IfPol" qinq="edgePort"/>
      <infraNodeP name="Node_101_phys">
      <infraLeafS name="phys101" type="range">
        <infraNodeBlk name="test" from_="101" to_="101"/>
      </infraLeafS>
      <infraRsAccPortP tDn="uni/infra/accportprof-phys21"/>
    </infraNodeP>
    <infraAccPortP name="phys21">
      <infraHPortS name="physHPortS" type="range">
        <infraPortBlk name="phys21" fromCard="1" toCard="1" fromPort="21" toPort="21"/>
        <infraRsAccBaseGrp tDn="uni/infra/funcprof/accportgrp-21"/>
      </infraHPortS>
    </infraAccPortP>
    <infraFuncP>
      <infraAccPortGrp name="21">
        <infraRsL2IfPol tnL2IfPolName="testL2IfPol"/>
        <infraRsAttEntP tDn="uni/infra/attentp-AttEntityProf1701"/>
      </infraAccPortGrp>
    </infraFuncP>
    <l2IfPol name='testL2IfPol' qinq='edgePort'/>
    <infraAttEntityP name="AttEntityProf1701">
      <infraRsDomP tDn="uni/phys-dom1701"/>
    </infraAttEntityP>
  </infraInfra>
</polUni>
```

XML with Post 2:

```
<polUni>
  <fvTenant dn="uni/tn-Coke" name="Coke">
    <fvTnlEPg name="WEB5" qiqL2ProtTunMask="lldp" accEncap="vlan-10"
fwdCtrl="mac-learn-disable" >
        <fvRsTnlpathAtt tDn="topology/pod-1/paths-101/pathep-[eth1/21]"/>
      </fvTnlEPg>
  </fvTenant>
</polUni>
```

Configuring 802.1Q Tunnels With PCs Using the REST API

Create a **Dot1q Tunnel**, using PCs, and configure the interfaces for it with steps such as the following examples.

Before you begin

Configure the tenant that will use the **Dot1q Tunnel**.

Procedure

Step 1 Create a **Dot1q Tunnel** using the REST API with XML such as the following example.

The example configures the tunnel with the LLDP Layer 2 tunneling protocol, adds the access encapsulation VLAN, and disables MAC learning in the tunnel.

Example:

```
   <fvTnlEPg name="WEB" qiqL2ProtTunMask=lldp accEncap="vlan-10" fwdCtrl="mac-learn-disable"
 >
        <fvRsTnlpathAtt tDn="topology/pod-1/paths-101/pathep-[po2]"/>
      </fvTnlEPg>
```

Step 2 Configure a Layer 2 Interface policy with static binding with XML such as the following example.

The example configures a Layer 2 interface policy for edge-switch ports. To configure a Layer 2 interface policy for core-switch ports, use `corePort` instead of `edgePort` in the `l2IfPol` MO.

Example:

```
<l2IfPol name="testL2IfPol" qinq="edgePort"/>
```

Step 3 Apply the Layer 2 Interface policy to a PC Interface Policy Group with XML such as the following:

Example:

```
<infraAccBndlGrp name="po2" lagT="link">
   <infraRsL2IfPol tnL2IfPolName="testL2IfPol"/>
    </infraAccBndlGrp>
```

Step 4 Configure a Leaf Profile with an Interface Selector with XML such as the following:

Example:

```
<infraAccPortP name="PC">
   <infraHPortS name="allow" type="range">
    <infraPortBlk name="block2" fromCard="1" toCard="1" fromPort="10" toPort="11" />
    <infraRsAccBaseGrp tDn="uni/infra/funcprof/accbundle-po2"/>
   </infraHPortS>
  </infraAccPortP>
```

Example

The following example shows the PC configuration in two posts.

This example configures the PC ports as edge ports. To configure them as core ports, use `corePort` instead of `edgePort` in the `l2IfPol` MO, in Post 1.

XML with Post 1:

```
<infraInfra dn="uni/infra">
    <infraNodeP name="bLeaf3">
        <infraLeafS name="leafs3" type="range">
            <infraNodeBlk name="nblk3" from_="101" to_="101">
            </infraNodeBlk>
        </infraLeafS>
        <infraRsAccPortP tDn="uni/infra/accportprof-shipping3"/>
    </infraNodeP>
    <infraAccPortP name="shipping3">
    <infraHPortS name="pselc3" type="range">
        <infraPortBlk name="blk3" fromCard="1" toCard="1" fromPort="24" toPort="25"/>

        <infraRsAccBaseGrp tDn="uni/infra/funcprof/accbundle-accountingLag3" />
    </infraHPortS>
  </infraAccPortP>
```

```
<infraFuncP>
   <infraAccBndlGrp name="accountingLag3" lagT='link'>
    <infraRsAttEntP tDn="uni/infra/attentp-default"/>
        <infraRsLacpPol tnLacpLagPolName='accountingLacp3'/>
        <infraRsL2IfPol tnL2IfPolName="testL2IfPol3"/>
   </infraAccBndlGrp>
</infraFuncP>
<lacpLagPol name='accountingLacp3' ctrl='15' descr='accounting' maxLinks='14' minLinks='1'
 mode='active' />
<l2IfPol name='testL2IfPol3' qinq='edgePort'/>
<infraAttEntityP name="default">
 </infraAttEntityP>
</infraInfra>
```

XML with Post 2:

```
<polUni>
  <fvTenant dn="uni/tn-Coke" name="Coke">
    <!-- bridge domain -->
    <fvTnlEPg name="WEB6" qiqL2ProtTunMask="lldp" accEncap="vlan-10"
fwdCtrl="mac-learn-disable" >
        <fvRsTnlpathAtt tDn="topology/pod-1/paths-101/pathep-[accountingLag1]"/>
      </fvTnlEPg>
  </fvTenant>
</polUni>
```

Configuring 802.1 Q Tunnels With vPCs Using the REST API

Create a **Dot1q Tunnel**, using vPCs, and configure the interfaces for it with steps such as the following examples.

Before you begin

Configure the tenant that will use the **Dot1q Tunnel**.

Procedure

Step 1 Create an 802.1Q tunnel using the REST API with XML such as the following example.

The example configures the tunnel with a Layer 2 tunneling protocol, adds the access encapsulation VLAN, and disables MAC learning in the tunnel.

Example:

```
<fvTnlEPg name="WEB" qiqL2ProtTunMask=lldp accEncap="vlan-10" fwdCtrl="mac-learn-disable"
>
        <fvRsTnlpathAtt tDn="topology/pod-1/protpaths-101-102/pathep-[po4]" />
      </fvTnlEPg>
```

Step 2 Configure a Layer 2 interface policy with static binding with XML such as the following example.

The example configures a Layer 2 interface policy for edge-switch ports. To configure a Layer 2 interface policy for core-switch ports, use the `qinq="corePort"` port type.

Example:

```
<l2IfPol name="testL2IfPol" qinq="edgePort"/>
```

Step 3 Apply the Layer 2 Interface policy to a VPC Interface Policy Group with XML such as the following:

Example:

```
<infraAccBndlGrp name="po4" lagT="node">
   <infraRsL2IfPol tnL2IfPolName="testL2IfPol"/>
   </infraAccBndlGrp>
```

Step 4 Configure a Leaf Profile with an Interface Selector with XML such as the following:

Example:

```
  <infraAccPortP name="VPC">
   <infraHPortS name="allow" type="range">
    <infraPortBlk name="block2" fromCard="1" toCard="1" fromPort="10" toPort="11" />
    <infraRsAccBaseGrp tDn="uni/infra/funcprof/accbundle-po4"/>
   </infraHPortS>
  </infraAccPortP>
```

Example

The following example shows the vPC configuration in three posts.

This example configures the vPC ports as edge ports. To configure them as core ports, use `corePort` instead of `edgePort` in the `l2IfPol` MO, in Post 2

XML with Post 1:

```
<polUni>
    <fabricInst>
        <fabricProtPol pairT="explicit">
            <fabricExplicitGEp name="101-102-vpc1" id="30">
                <fabricNodePEp id="101"/>
                <fabricNodePEp id="102"/>
            </fabricExplicitGEp>
        </fabricProtPol>
    </fabricInst>
</polUni>
```

XML with Post 2:

```
<infraInfra dn="uni/infra">
    <infraNodeP name="bLeaf1">
        <infraLeafS name="leafs" type="range">
            <infraNodeBlk name="nblk" from_="101" to_="101">
            </infraNodeBlk>
        </infraLeafS>
        <infraRsAccPortP tDn="uni/infra/accportprof-shipping1"/>
    </infraNodeP>

    <infraNodeP name="bLeaf2">
        <infraLeafS name="leafs" type="range">
            <infraNodeBlk name="nblk" from_="102" to_="102">
            </infraNodeBlk>
        </infraLeafS>
        <infraRsAccPortP tDn="uni/infra/accportprof-shipping2"/>
    </infraNodeP>

  <infraAccPortP name="shipping1">
    <infraHPortS name="pselc" type="range">
        <infraPortBlk name="blk" fromCard="1" toCard="1" fromPort="4" toPort="4"/>
        <infraRsAccBaseGrp tDn="uni/infra/funcprof/accbundle-accountingLag1" />
    </infraHPortS>
  </infraAccPortP>
```

```
  <infraAccPortP name="shipping2">
    <infraHPortS name="pselc" type="range">
        <infraPortBlk name="blk" fromCard="1" toCard="1" fromPort="2" toPort="2"/>
        <infraRsAccBaseGrp tDn="uni/infra/funcprof/accbundle-accountingLag2" />
    </infraHPortS>
  </infraAccPortP>

<infraFuncP>
   <infraAccBndlGrp name="accountingLag1" lagT='node'>
    <infraRsAttEntP tDn="uni/infra/attentp-default"/>
        <infraRsLacpPol tnLacpLagPolName='accountingLacp1'/>
        <infraRsL2IfPol tnL2IfPolName="testL2IfPol"/>
   </infraAccBndlGrp>
   <infraAccBndlGrp name="accountingLag2" lagT='node'>
    <infraRsAttEntP tDn="uni/infra/attentp-default"/>
         <infraRsLacpPol tnLacpLagPolName='accountingLacp1'/>
         <infraRsL2IfPol tnL2IfPolName="testL2IfPol"/>
   </infraAccBndlGrp>
</infraFuncP>
<lacpLagPol name='accountingLacp1' ctrl='15' descr='accounting' maxLinks='14' minLinks='1'
 mode='active' />
<l2IfPol name='testL2IfPol' qinq='edgePort'/>

 <infraAttEntityP name="default">
 </infraAttEntityP>
</infraInfra>
```

XML with Post 3:

```
<polUni>
  <fvTenant dn="uni/tn-Coke" name="Coke">
    <!-- bridge domain -->
    <fvTnlEPg name="WEB6" qiqL2ProtTunMask="lldp" accEncap="vlan-10"
fwdCtrl="mac-learn-disable" >
        <fvRsTnlpathAtt tDn="topology/pod-1/protpaths-101-102/pathep-[accountingLag2]"/>
      </fvTnlEPg>
  </fvTenant>
</polUni>
```

CHAPTER 11

Q-in-Q Encapsulation Mapping for EPGs

Q-in-Q Encapsulation Mapping for EPGs

Using Cisco Application Policy Infrastructure Controller (APIC), you can map double-tagged VLAN traffic ingressing on a regular interface, PC, or vPC to an EPG. When this feature is enabled, when double-tagged traffic enters the network for an EPG, both tags are processed individually in the fabric and restored to double-tags when egressing the Cisco Application Centric Infrastructure (ACI) switch. Ingressing single-tagged and untagged traffic is dropped.

The following guidelines and limitations apply:

- This feature is only supported on Cisco Nexus 9300-FX platform switches.
- Both the outer and inner tag must be of EtherType 0x8100.
- MAC learning and routing are based on the EPG port, sclass, and VRF instance, not on the access encapsulations.
- QoS priority settings are supported, derived from the outer tag on ingress, and rewritten to both tags on egress.
- EPGs can simultaneously be associated with other interfaces on a leaf switch, that are configured for single-tagged VLANs.
- Service graphs are supported for provider and consumer EPGs that are mapped to Q-in-Q encapsulated interfaces. You can insert service graphs, as long as the ingress and egress traffic on the service nodes is in single-tagged encapsulated frames.
- When vPC ports are enabled for Q-in-Q encapsulation mode, VLAN consistency checks are not performed.

The following features and options are not supported with this feature:

- Per-port VLAN feature
- FEX connections

- Mixed mode

  For example, an interface in Q-in-Q encapsulation mode can have a static path binding to an EPG with double-tagged encapsulation only, not with regular VLAN encapsulation.
- STP and the "Flood in Encapsulation" option
- Untagged and 802.1p mode
- Multi-pod and Multi-Site
- Legacy bridge domain
- L2Out and L3Out connections
- VMM integration
- Changing a port mode from routed to Q-in-Q encapsulation mode
- Per-VLAN mis-cabling protocol on ports in Q-in-Q encapsulation mode

Configuring Q-in-Q Encapsulation Mapping for EPGs Using the GUI

Enabling Q-in-Q Encapsulation on Specific Leaf Switch Interfaces Using the GUI

Leaf switch ports, PCs, or vPCs are enabled for Q-in-Q encapsulation mode in the **Interface** tab of one of the following locations in the APIC GUI.

- **Fabric** > **Inventory** > **Topology**
- **Fabric** > **Inventory** > **Pod**
- **Fabric** > **Inventory** > **Pod** > ***leaf-name***

Configure vPCs on the **Topology** or **Pod Interface** tab.

Before you begin

The tenant, application profile, and the application EPG that will be mapped with an interface configured for Q-in-Q mode should be created.

Procedure

Step 1 On the menu bar, choose **Fabric** > **Inventory** and click **Topology**, **Pod**, or expand **Pod** and choose a leaf.

Step 2 On the **Topology** or **Pod** panel **Interface** tab.

Step 3 Click the **Operation**/**Configuration** toggle-button to display the configuration panel.

Step 4 Click + to add diagrams of leaf switches, choose one or more switches, and click **Add Selected**.

On the ***leaf-name*** panel **Interface** tab, a diagram of the switch appears automatically, after you click the **Operation**/**Configuration**toggle-button.

Step 5 Click the interfaces to be enabled for Q-in-Q encapsulation mode.

Step 6 To configure a port, perform the following steps:

a) Click **L2** on the upper left.
b) On the L2 tab, on the **L2 QinQ State** field, click **Double Q Tag Port** and click **Submit**

Step 7 To configure a PC, perform the following steps:

a) Click **PC** on the upper left.
b) On the **Physical Interface** tab, enter the **Policy Group Name**.
c) On the L2 tab, on the **L2 QinQ State** field, click **Double Q Tag Port** and click **Submit**

Step 8 To configure a vPC, perform the following steps:

a) On two leaf switch diagrams, click the interfaces for the two legs of the VPC.
b) Click **vPC**.
c) On the **Physical Interface** tab, enter the **Logical Pair ID** (The identifier for the auto-protection group. Each protection group has a unique ID. The ID is a range of 1 to 1000) and the **Policy Group Name**.
d) On the L2 tab, on the **L2 QinQ State** field, click **Double Q Tag Port** and click **Submit**

Enabling Q-in-Q Encapsulation for Leaf Interfaces With Fabric Interface Policies Using the GUI

Enable leaf interfaces, PCs, and vPCs for Q-in-Q encapsulation, using a leaf interface profile.

Before you begin

The tenant, application profile, and the application EPG that will be mapped with an interface configured for Q-in-Q mode should be created.

Procedure

Step 1 On the menu bar, click **Fabric** > **External Access Policies**.

Step 2 On the Navigation bar, click **Policies** > **Interface** > **L2 Interface**.

Step 3 Right-click **L2 Interface**, select **Create L2 Interface Policy**, and perform the following actions:

a) In the **Name** field, enter a name for the Layer 2 Interface policy.
b) Optional. Add a description of the policy. We recommend that you describe the purpose for the L2 Interface Policy.
c) To create an interface policy that enables Q-in-Q encapsulation, in the **QinQ** field, click **doubleQtagPort**.
d) Click **Submit**.

Step 4 Apply the L2 Interface policy to a Policy Group with the following steps:

a) Click on **Fabric** > **External Access Policies** > **Interfaces** > **Leaf Interfaces**, and expand **Policy Groups**.
b) Right-click **Leaf Access Port**, **PC Interface**, or **vPC Interface** and choose one of the following, depending on the type of interface you are configuring for the tunnel.

- **Create Leaf Access Port Policy Group**
- **Create PC Policy Group**
- **Create vPC Policy Group**

c) In the resulting dialog box, enter the policy group name, choose the L2 Interface policy that you previously created, and click **Submit.**

Step 5 Create a Leaf Interface Profile with the following steps:

a) Click on **Fabric** > **External Access Policies** > **Interface** > **Leaf Interfaces** > **Profiles**.

b) Right-click on **Leaf Profiles**, choose **Create Leaf Interface Policy**, and perform the following steps:

- In the **Name** field, type a name for the **Leaf Interface Profile**.

  Optional. Add a description.

- On the **Interface Selectors** field, click the **+**, and enter the following information:
  - In the **Name** field, type a name for the interface selector.

    Optional. Add a description.
  - Enter the selector name, and optionally, a description.
  - In the Interface IDs field, enter the interface or multiple interfaces to be included in the profile.
  - In the **Interface Policy Group** field, choose the interface policy group that you previously created.

Mapping an EPG to a Q-in-Q Encapsulation-Enabled Interface Using the GUI

You can associate EPGs with Q-in-Q encapsulation-enabled interfaces in one of the following models:

- Deploy a static EPG on specific Q-in-Q encapsulation-enabled interfaces
- Statically link an EPG with a Q-in-Q encapsulation-enabled leaf switch
- Associate an EPG with a Q-in-Q encapsulation-enabled endpoint (with a static MAC address)

All three tasks are performed in the same area of the APIC GUI.

Before you begin

- Create the tenant, application profile, and application EPG that will be mapped with an interface configured for Q-in-Q mode.
- The target interfaces should be configured for Q-in-Q encapsulation.

Procedure

Step 1 In the menu bar, click **Tenants** > ***tenant-name*** .

Step 2 In the Navigation pane, expand **Application Profiles** > > ***application-profile-name*** > **Application EPGs** > ***application-EPG-name*** .

Step 3 To deploy a static EPG on an interface, PC, or vPC that has been enabled for Q-in-Q mode, perform the following steps:

a) Under the application EPG, right- click **Static Ports** and choose **Deploy Static EPG on PC, vPC, or Interface**.
b) Choose the path type, the node, and the path to the Q-in-Q enabled interface.
c) On the **Port Encap (or Secondary VLAN for Micro-Seg)** field, choose **QinQ** and enter the outer and inner VLAN tags for traffic mapped to the EPG.
d) Click **Submit**.

Step 4 To statically link an EPG with a node enabled with Q-in-Q mode, perform the following steps:

a) Under the application EPG, right- click **Static Leafs** and choose **Statically Link With Node**.
b) In the Node field, choose the Q-in-Q-enabled switches from the list.
c) On the Encap field, choose **QinQ** and enter the outer and inner VLAN tags for the EPG.
d) Click **Submit**.

Step 5 To associate an EPG with a static endpoint, perform the following steps:

a) Under the application EPG, right- click **Static EndPoints** and choose **Create Static EndPoint**.
b) Enter the MAC address of the interface.
c) Choose the path type, node, and path to the Q-in-Q encapsulation-enabled interface.
d) Optional. Add IP addresses for the endpoint.
e) On the **Encap** field, choose **QinQ** and enter the outer and inner VLAN tags.
f) Click **Submit**.

Mapping EPGs to Q-in-Q Encapsulated Leaf Interfaces Using the NX-OS Style CLI

Enable an interface for Q-in-Q encapsulation and associate the interface with an EPG.

Before you begin

Create the tenant, application profile, and application EPG that will be mapped with an interface configured for Q-in-Q mode.

Procedure

	Command or Action	Purpose
Step 1	**Configure** **Example:** `apic1# configure`	Enters global configuration mode.
Step 2	**leaf** *number* **Example:**	Specifies the leaf to be configured.

	Command or Action	Purpose
	apic1(config)# leaf 101	
Step 3	**interface ethernet** *slot/port* **Example:** `apic1 (config-leaf)# interface ethernet 1/25`	Specifies the interface to be configured.
Step 4	**switchport mode dot1q-tunnel** doubleQtagPort **Example:** `apic1(config-leaf-if)# switchport mode dot1q-tunnel doubleQtagPort`	Enables an interface for Q-in-Q encapsulation.
Step 5	**switchport trunk qinq outer-vlan** *vlan-number* **inner-vlan** *vlan-number* **tenant** *tenant-name* **application** *application-name* **epg** *epg-name* **Example:** `apic1(config-leaf-if)# switchport trunk qinq outer-vlan 202 inner-vlan 203 tenant tenant64 application AP64 epg EPG64`	Associates the interface with an EPG.

Example

The following example enables Q-in-Q encapsulation (with outer-VLAN ID 202 and inner-VLAN ID 203) on the leaf interface 101/1/25, and associates the interface with EPG64.

```
apic1(config)# leaf 101
apic1(config-leaf)# interface ethernet 1/25
apic1(config-leaf-if)#switchport mode dot1q-tunnel doubleQtagPort
apic1(config-leaf-if)# switchport trunk qinq outer-vlan 202 inner-vlan 203 tenant tenant64
 application AP64 epg EPG64
```

Mapping EPGs to Q-in-Q Encapsulation Enabled Interfaces Using the REST API

Before you begin

Create the tenant, application profile, and application EPG that will be mapped with an interface configured for Q-in-Q mode.

Procedure

Enable an interface for Q-in-Q encapsulation and associate the interface with an EPG, with XML such as the following example:

Example:

```
<polUni>
  <fvTenant dn="uni/tn-tenant64" name="tenant64">
    <fvCtx name="VRF64"/>
    <fvBD name="BD64_1">
      <fvRsCtx tnFvCtxName="VRF64"/>
      <fvSubnet ip="20.0.1.2/24"/>
    </fvBD>
    <fvAp name="AP64">
      <fvAEPg name="WEB7">
        <fvRsBd tnFvBDName="BD64_1"/>
        <fvRsQinqPathAtt tDn="topology/pod-1/paths-101/pathep-[eth1/25]"
encap="qinq-202-203"/>
      </fvAEPg>
    </fvAp>
  </fvTenant>
</polUni>
```

CHAPTER 12

Dynamic Breakout Ports

This chapter contains the following sections:

Configuration of Dynamic Breakout Ports

Breakout cables are suitable for very short links and offer a cost effective way to connect within racks and across adjacent racks.

Breakout enables a 40 Gigabit (Gb) port to be split into four independent and logical 10Gb ports or a 100Gb port to be split into four independent and logical 25Gb ports.

Before you configure breakout ports, connect a 40Gb port to four 10Gb ports or a 100Gb port to four 25Gb ports with one of the following cables:

- Cisco QSFP-4SFP10G
- Cisco QSFP-4SFP25G

The 40Gb to 10Gb dynamic breakout feature is supported on the access facing ports of the following switches:

- N9K-C9332PQ
- N9K-C93180LC-EX
- N9K-C9336C-FX

The 100Gb to 25Gb breakout feature is supported on the access facing ports of the following switches:

- N9K-C93180LC-EX
- N9K-C9336C-FX2
- N9K-C93180YC-FX in the 3.2 and later releases

Observe the following guidelines and restrictions:

- In general, breakout ports and port profiles (ports changed from uplink to downlink) are not supported on the same port.

  However, from Cisco Application Policy Infrastructure Controller (APIC) release 3.2, dynamic breakouts (both 100Gb and 40Gb) are supported on profiled QSFP ports on the N9K-C93180YC-FX switch.
- Fast Link Failover policies are not supported on the same port with the dynamic breakout feature.
- Breakout ports cannot be used for Cisco APIC connectivity.
- Breakout subports can be used in the same way other port types in the policy model are used.
- When a port is enabled for dynamic breakout, other policies (expect monitoring policies) on the parent port are no longer valid.
- When a port is enabled for dynamic breakout, other EPG deployments on the parent port are no longer valid.
- A breakout sub-port can not be further broken out using a breakout policy group.
- If the LACP transmit rate on port channels that have breakout sub-ports need to be changed, then all the port channels that include breakout sub-ports need to use the same LACP transmit rate configuration. You can configure an override policy to set the transmit rate as follows:
  1. Configure/change the default port channel member policy to include Fast Transmit Rate (**Fabric** > **Access Policies** > **Policies** > **Interface** > **Port Channel Member**).
  2. Configure all the PC/vPC interface policy groups to include the above default port channel member policy under the override policy groups (**Fabric** > **Access Policies** > **Interfaces** > **Leaf Interfaces** > **Policy Groups** > **PC/vPC Interface**).

Configuring Dynamic Breakout Ports Using the APIC GUI

Configure a Breakout Leaf Port with an Leaf Interface Profile, associate the profile with a switch, and configure the sub ports with the following steps.

Note You can also configure ports for breakout in the APIC GUI by navigating to **Fabric** > **Inventory**, and clicking **Topology** or **Pod**, or expanding **Pod** and clicking **Leaf**. Then, enable configuration and click the **Interface** tab.

Procedure

Before you begin

- The ACI fabric is installed, APIC controllers are online, and the APIC cluster is formed and healthy.
- An APIC fabric administrator account is available that will enable creating the necessary fabric infrastructure configurations.
- The target leaf switches are registered in the ACI fabric and available.
- The 40GE or 100GE leaf switch ports are connected with Cisco breakout cables to the downlink ports.

Procedure

Step 1 On the menu bar, choose **Fabric** > **Access Policies**.

Step 2 In the Navigation pane, expand **Interfaces** and **Leaf Interfaces** and **Profiles**.

Step 3 Right-click **Profiles** and choose **Create Leaf Interface Profile**.

Step 4 Type the name and optional description, click the + symbol on **Interface Selectors**

Step 5 Perform the following:

a) Type a name (and optional description) for the **Access Port Selector**.
b) In the **Interface IDs** field, type the slot and port for the breakout port.
c) In the **Interface Policy Group** field, click the down arrow and choose **Create Leaf Breakout Port Group**.
d) Type the name (and optional description) for the **Leaf Breakout Port Group**.
e) In the **Breakout Map** field, choose **10g-4x** or **25g-4x**.

For switches supporting breakout, see Configuration of Dynamic Breakout Ports, on page 227.

f) Click **Submit**.

Step 6 To assign a Breakout Port to an EPG, perform the following steps:

On the menu bar, choose **Tenant** > **Application Profiles** > **Application EPG**. Right-click on **Application EPG** to open **Create Application EPG**dialog box, and perform the following steps:

a) Select the **Statically Link with Leaves/Paths** check box to gain access to the **Leaves/Paths** tab in the dialog box.
b) Complete one of the following sets of steps:

Option	Description
If you want to deploy the EPG on...	Then
A node	**a.** Expand the **Leaves** area. **b.** From the **Node** drop-down list, choose a node. **c.** In the **Encap** field, enter the appropriate VLAN. **d.** (Optional) From the **Deployment Immediacy** drop-down list, accept the default **On Demand** or choose **Immediate**. **e.** (Optional) From the Mode drop-down list, accept the default **Trunk** or choose another mode.
A port on the node	**a.** Expand the **Paths** area. **b.** From the **Path** drop-down list, choose the appropriate node and port. **c.** (Optional) In the **Deployment Immediacy** field drop-down list, accept the default **On Demand** or choose **Immediate**. **d.** (Optional) From the Mode drop-down list, accept the default **Trunk** or choose another mode. **e.** In the **Port Encap** field, enter the secondary VLAN to be deployed.

Option	Description
	f. (Optional) In the **Primary Encap** field, enter the primary VLAN to be deployed.

Step 7 To associate the Leaf Interface Profile to a the leaf switch, perform the following steps:

a) Expand **Switches** and **Leaf Switches**, and **Profiles**.
b) Right-click **Profiles** and select **Create Leaf Profiles**.
c) Type the name and optional description of the Leaf Profile.
d) Click the + symbol on the **Leaf Selectors** area.
e) Type the leaf selector name and an optional description.
f) Click the down arrow on the **Blocks** field and choose the switch to be associated with the breakout leaf interface profile.
g) Click the down arrow on the **Policy Group** field and choose **Create Access Switch Policy Group**.
h) Type a name and optional description for the Access Switch Policy Group.
i) Optional. Enable other policies.
j) Click **Submit**.
k) Click **Update**.
l) Click **Next**.
m) In the **Associations Interface Selector Profiles** area, choose the Interface Selector Profile you previously created for the breakout port.
n) Click **Finish**.

Step 8 To verify the breakout port has been split into four sub ports, perform the following steps:

a) On the Menu bar, click **Fabric** > **Inventory**.
b) On the Navigation bar, click the Pod and Leaf where the breakout port is located.
c) Expand **Interfaces** and **Physical Interfaces**.
You should see four ports at the position where the breakout port was configured. For example, if you configured 1/10 as a breakout port, you should see the following:

- **eth1/10/1**
- **eth1/10/2**
- **eth1/10/3**
- **eth1/10/4**

Step 9 To configure the sub ports, perform the following steps:

a) On the Menu bar, click **Fabric** > **Access Policies**.
b) On the Navigation bar, expand **Interfaces**, **Leaf Interfaces**, **Profiles**, and the breakout leaf interface profile you previously created.

You will see a port selector with the breakout cable. Instead of defining a sub port block under already existing port selector, you need to define on a new access port selector.

c) On the Navigtion bar, right click the higher level interface profile and select **Create Access Port Selector**.
d) In the **Name** field, enter the sub port name.
e) In the **Interface IDs** field, enter the IDs for the four sub ports in a format such as `1/10/1-4`.
f) In the **Interface Policy Group** field, select **Create Leaf Access Port Policy Group**.
g) Click **Submit**.

Step 10 To apply the Policy Group to an individual interface that links the AAEP to the port, perform the following steps:

a) In the **Name** field, enter the name for the Leaf Access Port Group Policy.
b) In the **Link Level Policy** field, select **link-level_auto**.
c) In the **CDP Policy** field, select **cdp_enabled**.
d) In the **LLDP Policy** field, select **default**.
e) In the **Attached Entity Profile** field, select the AAEP profile to attach to the policy group.
f) Click **Submit**.

Configuring Dynamic Breakout Ports Using the NX-OS Style CLI

Use the following steps to configure a breakout port, verify the configuration, and configure an EPG on a sub port, using the NX-OS style CLI.

Before you begin

- The ACI fabric is installed, APIC controllers are online, and the APIC cluster is formed and healthy.
- An APIC fabric administrator account is available that will enable creating the necessary fabric infrastructure configurations.
- The target leaf switches are registered in the ACI fabric and available.
- The 40GE or 100GE leaf switch ports are connected with Cisco breakout cables to the downlink ports.

Procedure

	Command or Action	Purpose
Step 1	**configure** **Example:** `apic1# configure`	Enters configuration mode.
Step 2	**leaf** *ID* **Example:** `apic1(config)# leaf 101`	Selects the leaf switch where the breakout port will be located and enters leaf configuration mode.
Step 3	**interface ethernet** *slot/port* **Example:** `apic1(config-leaf)# interface ethernet 1/16`	Identifies the interface to be enabled as a 40 Gigabit Ethernet (GE) breakout port.
Step 4	**breakout 10g-4x | 25g-4x** **Example:** `apic1(config-leaf-if)# breakout 10g-4x`	Enables the selected interface for breakout. **Note** For switch support for the Dynamic Breakout Port feature, see Configuration of Dynamic Breakout Ports, on page 227.

	Command or Action	Purpose

Step 5	**show run**
	Example:
	`apic1(config-leaf-if)# show run`
	`# Command: show running-config leaf 101`
	`interface ethernet 1 / 16`
	`# Time: Fri Dec  2 18:13:39 2016`
	`leaf 101`
	`interface ethernet 1/16`
	`breakout 10g-4x`
	`apic1(config-leaf-if)# exit`
	`apic1(config-leaf)# exit`
		Verifies the configuration by showing the running configuration of the interface and returns to global configuration mode.

Step 6	**tenant** *tenant-name* **Example:** `apic1(config)# tenant tenant64`	Selects or creates the tenant that will consume the breakout ports and enters tenant configuration mode.

Step 7	**vrf context** *vrf-name*
	Example:
	`apic1(config-tenant)# vrf context vrf64`
	`apic1(config-tenant-vrf)# exit`
		Creates or identifies the Virtual Routing and Forwarding (VRF) instance associated with the tenant and exits the configuration mode.

Step 8	**bridge-domain** *bridge-domain-name* **Example:** `apic1(config-tenant)# bridge-domain bd64`	Creates or identifies the bridge-domain associated with the tenant and enters BD configuration mode.

Step 9	**vrf member** *vrf-name*
	Example:
	`apic1(config-tenant-bd)# vrf member vrf64`
	`apic1(config-tenant-bd)# exit`
		Associates the VRF with the bridge-domain and exits the configuration mode.

Step 10	**application** *application-profile-name* **Example:** `apic1(config-tenant)# application app64`	Creates or identifies the application profile associated with the tenant and the EPG.
Step 11	**epg** *epg-name* **Example:** `apic1(config-tenant)# epg epg64`	Creates or identifies the EPG and enters into EPG configuration mode.

Step 12	**bridge-domain member** *bridge-domain-name*
	Example:
	`apic1(config-tenant-app-epg)# bridge-domain member bd64`
	`apic1(config-tenant-app-epg)# exit`
	`apic1(config-tenant-app)# exit`
	`apic1(config-tenant)# exit`
		Associates the EPG with the bridge domain and returns to global configuration mode.
		Configure the sub ports as desired, for example, use the speed command in leaf interface mode to configure a sub port.

	Command or Action	Purpose

Step 13	**leaf** *leaf-name*
	Example:
	`apic1(config)# leaf 1017`
	`apic1(config-leaf)# interface ethernet 1/13`
	`apic1(config-leaf-if)# vlan-domain member dom1`
	`apic1(config-leaf-if)# switchport trunk allowed vlan 20 tenant t1 application AP1 epg EPG1`
	Note The vlan-domain and vlan-domain member commands mentioned in the above example are a pre-requisite for deploying an EPG on a port.
		Associates the EPG with a break-out port.

Step 14	**speed** *interface-speed*
	Example:
	`apic1(config)# leaf 101`
	`apic1(config-leaf)# interface ethernet 1/16/1`
	`apic1(config-leaf-if)# speed 10G`
	`apic1(config-leaf-if)# exit`
		Enters leaf interface mode, sets the speed of an interface, and exits the configuration mode.

Step 15	**show run** **Example:** `apic1(config-leaf)# show run`	After you have configured the sub ports, entering this command in leaf configuration mode displays the sub port details.

The port on leaf 101 at interface 1/16 is confirmed enabled for breakout with sub ports 1/16/1, 1/16/2, 1/16/3, and 1/16/4.

Example

This example configures the port for breakout:

```
apic1# configure
apic1(config)# leaf 101
apic1(config-leaf)# interface ethernet 1/16
apic1(config-leaf-if)# breakout 10g-4x
```

This example configures the EPG for the sub ports.

```
apic1(config)# tenant tenant64
apic1(config-tenant)# vrf context vrf64
apic1(config-tenant-vrf)# exit
apic1(config-tenant)# bridge-domain bd64
apic1(config-tenant-bd)# vrf member vrf64
apic1(config-tenant-bd)# exit
apic1(config-tenant)# application app64
apic1(config-tenant-app)# epg epg64
```

```
apic1(config-tenant-app-epg)# bridge-domain member bd64
apic1(config-tenant-app-epg)# end
```

This example sets the speed for the breakout sub ports to 10G.

```
apic1(config)# leaf 101
apic1(config-leaf)# interface ethernet 1/16/1
apic1(config-leaf-if)# speed 10G
apic1(config-leaf-if)# exit

apic1(config-leaf)# interface ethernet 1/16/2
apic1(config-leaf-if)# speed 10G
apic1(config-leaf-if)# exit
apic1(config-leaf)# interface ethernet 1/16/3
apic1(config-leaf-if)# speed 10G
apic1(config-leaf-if)# exit
apic1(config-leaf)# interface ethernet 1/16/4
apic1(config-leaf-if)# speed 10G
apic1(config-leaf-if)# exit
```

This example shows the four sub ports connected to leaf 101, interface 1/16.

```
apic1#(config-leaf)# show run
# Command: show running-config leaf 101
# Time: Fri Dec  2 00:51:08 2016
  leaf 101
    interface ethernet 1/16/1
      speed 10G
      negotiate auto
      link debounce time 100
      exit
    interface ethernet 1/16/2
      speed 10G
      negotiate auto
      link debounce time 100
      exit
    interface ethernet 1/16/3
      speed 10G
      negotiate auto
      link debounce time 100
      exit
    interface ethernet 1/16/4
      speed 10G
      negotiate auto
      link debounce time 100
      exit
    interface ethernet 1/16
      breakout 10g-4x
      exit
    interface vfc 1/16
```

Configuring Dynamic Breakout Ports Using the REST API

Configure a Breakout Leaf Port with an Leaf Interface Profile, associate the profile with a switch, and configure the sub ports with the following steps.

For switch support for the breakout feature, see Configuration of Dynamic Breakout Ports, on page 227.

Procedure

Before you begin

- The ACI fabric is installed, APIC controllers are online, and the APIC cluster is formed and healthy.
- An APIC fabric administrator account is available that will enable creating the necessary fabric infrastructure configurations.
- The target leaf switches are registered in the ACI fabric and available.
- The 40GE or 100GE leaf switch ports are connected with Cisco breakout cables to the downlink ports.

Procedure

Step 1 Configure a breakout policy group for the breakout port with JSON, such as the following example:

Example:

In this example, we create an interface profile 'brkout44' with the only port 44 underneath its port selector. The port selector is pointed to a breakout policy group 'new-brkoutPol'.

```
{
            "infraAccPortP": {
              "attributes": {
                "dn":"uni/infra/accportprof-brkout44",
                "name":"brkout44",
                "rn":"accportprof-brkout44",
                "status":"created,modified"
                },
                "children":[ {
                     "infraHPortS": {
                        "attributes": {

"dn":"uni/infra/accportprof-brkout44/hports-new-brekoutPol-typ-range",
                          "name":"new-brkoutPol",
                          "rn":"hports-new-brkoutPol-typ-range",
                          "status":"created,modified"
                     },
                     "children":[ {
                        "infraPortBlk": {
                          "attributes": {

"dn":"uni/infra/accportprof-brkout44/hports-new-brkoutPol-typ-range/portblk-block2",
                            "fromPort":"44",
                            "toPort":"44",
                            "name":"block2",
                            "rn":"portblk-block2",
                            "status":"created,modified"
                          },
                          "children":[] }
                   }, {
            "infraRsAccBaseGrp": {
              "attributes":{
                 "tDn":"uni/infra/funcprof/brkoutportgrp-new-brkoutPol",
                 "status":"created,modified"
              },
              "children":[]
             }
           }
         ]
       }
     }
```

```
    ]
  }
}
```

Step 2 Create a new switch profile and associate it with the port profile, previously created, with JSON such as the following example:

Example:

In this example, we create a new switch profile 'leaf1017' with switch 1017 as the only node. We associate this new switch profile with the port profile 'brkout44' created above. After this, the port 44 on switch 1017 will have 4 sub ports.

Example:

```
{
   "infraNodeP": {
     "attributes": {
       "dn":"uni/infra/nprof-leaf1017",
       "name":"leaf1017","rn":"nprof-leaf1017",
       "status":"created,modified"
      },
      "children": [ {
        "infraLeafS": {
          "attributes": {
            "dn":"uni/infra/nprof-leaf1017/leaves-1017-typ-range",
            "type":"range",
            "name":"1017",
            "rn":"leaves-1017-typ-range",
            "status":"created"
      },
      "children": [ {
        "infraNodeBlk": {
          "attributes": {
           "dn":"uni/infra/nprof-leaf1017/leaves-1017-typ-range/nodeblk-102bf7dc60e63f7e",

            "from_":"1017","to_":"1017",
            "name":"102bf7dc60e63f7e",
            "rn":"nodeblk-102bf7dc60e63f7e",
            "status":"created"
           },
           "children": [] }
          }
        ]
      }
    }, {
      "infraRsAccPortP": {
        "attributes": {
          "tDn":"uni/infra/accportprof-brkout44",
          "status":"created,modified"
     },
     "children": [] }
      }
    ]
  }
}
```

Step 3 Configure the subports.

Example:

This example configures subports 1/44/1, 1/44/2, 1/44/3, 1/44/4 on switch 1017, for instance, in the example below, we configure interface 1/44/3. It also creates the `infraSubPortBlk` object instead of the `infraPortBlk` object.

```
{
   "infraAccPortP": {
     "attributes": {
       "dn":"uni/infra/accportprof-brkout44",
       "name":"brkouttest1",
       "rn":"accportprof-brkout44",
       "status":"created,modified"
     },
     "children": [{
       "infraHPortS": {
         "attributes": {
           "dn":"uni/infra/accportprof-brkout44/hports-sel1-typ-range",
           "name":"sel1",
           "rn":"hports-sel1-typ-range",
           "status":"created,modified"
         },
         "children": [{
           "infraSubPortBlk": {
             "attributes": {
             "dn":"uni/infra/accportprof-brkout44/hports-sel1-typ-range/subportblk-block2",

               "fromPort":"44",
               "toPort":"44",
               "fromSubPort":"3",
               "toSubPort":"3",
               "name":"block2",
               "rn":"subportblk-block2",
               "status":"created"
             },
             "children":[]}
             },
           {
             "infraRsAccBaseGrp": {
                 "attributes": {
                   "tDn":"uni/infra/funcprof/accportgrp-p1",
                   "status":"created,modified"
                 },
                 "children":[]}
                 }
               ]
             }
           }
         ]
       }
     }
```

Step 4 Deploy an EPG on a specific port.

Example:

```
<fvTenant name="<tenant_name>" dn="uni/tn-test1" >
    <fvCtx name="<network_name>" pcEnfPref="enforced" knwMcastAct="permit"/>
    <fvBD name="<bridge_domain_name>" unkMcastAct="flood" >
        <fvRsCtx tnFvCtxName="<network_name>"/>
    </fvBD>
    <fvAp name="<application_profile>" >
        <fvAEPg name="<epg_name>" >
            <fvRsPathAtt tDn="topology/pod-1/paths-1017/pathep-[eth1/13]" mode="regular"
instrImedcy="immediate" encap="vlan-20"/>
        </fvAEPg>
    </fvAp>
</fvTenant>
```

CHAPTER 13

Proxy ARP

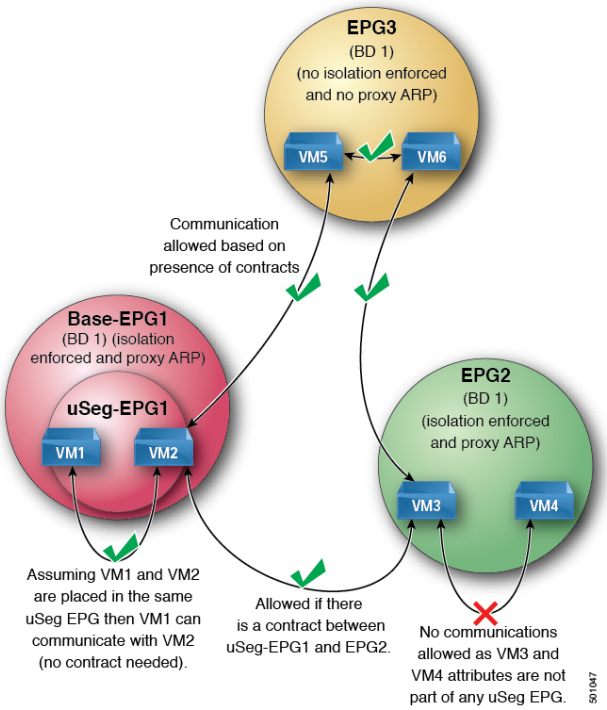
This chapter contains the following sections:

About Proxy ARP

Proxy ARP in Cisco ACI enables endpoints within a network or subnet to communicate with other endpoints without knowing the real MAC address of the endpoints. Proxy ARP is aware of the location of the traffic destination, and offers its own MAC address as the final destination instead.

To enable Proxy ARP, intra-EPG endpoint isolation must be enabled on the EPG see the following figure for details. For more information about intra-EPG isolation and Cisco ACI, see the *Cisco ACI Virtualization Guide*.

Figure 34: Proxy ARP and Cisco APIC



Proxy ARP within the Cisco ACI fabric is different from the traditional proxy ARP. As an example of the communication process, when proxy ARP is enabled on an EPG, if an endpoint A sends an ARP request for endpoint B and if endpoint B is learned within the fabric, then endpoint A will receive a proxy ARP response from the bridge domain (BD) MAC. If endpoint A sends an ARP request for endpoint B, and if endpoint B is not learned within the ACI fabric already, then the fabric will send a proxy ARP request within the BD. Endpoint B will respond to this proxy ARP request back to the fabric. At this point, the fabric does not send a proxy ARP response to endpoint A, but endpoint B is learned within the fabric. If endpoint A sends another ARP request to endpoint B, then the fabric will send a proxy ARP response from the BD MAC.

The following example describes the proxy ARP resolution steps for communication between clients VM1 and VM2:

1. VM1 to VM2 communication is desired.

Figure 35: VM1 to VM2 Communication is Desired.

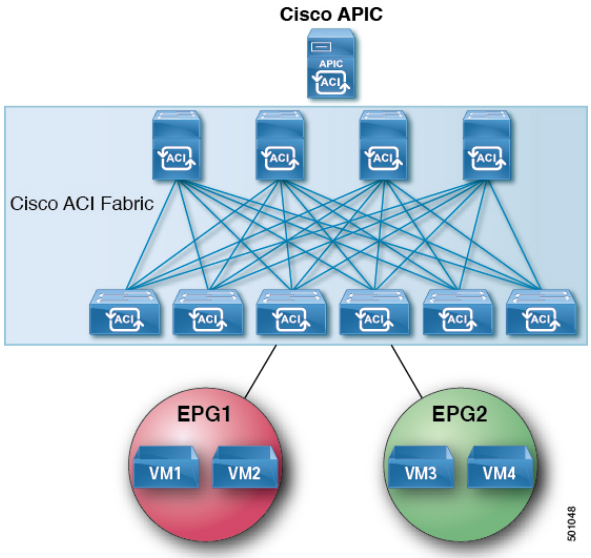


Table 17: ARP Table State

Device	State
VM1	IP = * MAC = *
ACI fabric	IP = * MAC = *
VM2	IP = * MAC = *

2. VM1 sends an ARP request with a broadcast MAC address to VM2.

Figure 36: VM1 sends an ARP Request with a Broadcast MAC address to VM2

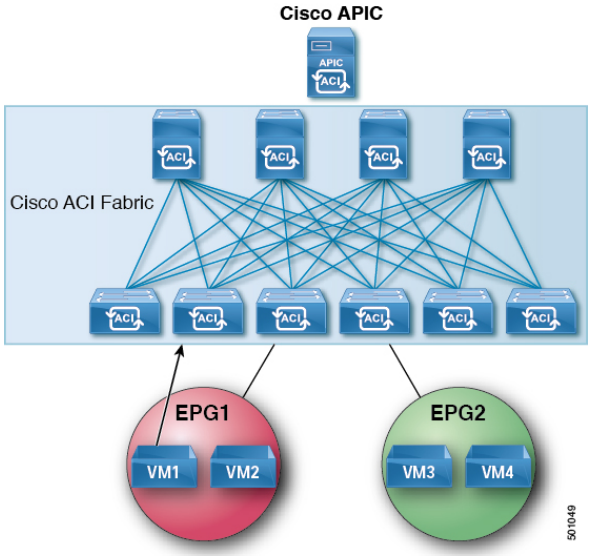


Table 18: ARP Table State

Device	State
VM1	IP = VM2 IP; MAC = ?
ACI fabric	IP = VM1 IP; MAC = VM1 MAC
VM2	IP = * MAC = *

3. The ACI fabric floods the proxy ARP request within the bridge domain (BD).

Figure 37: ACI Fabric Floods the Proxy ARP Request within the BD

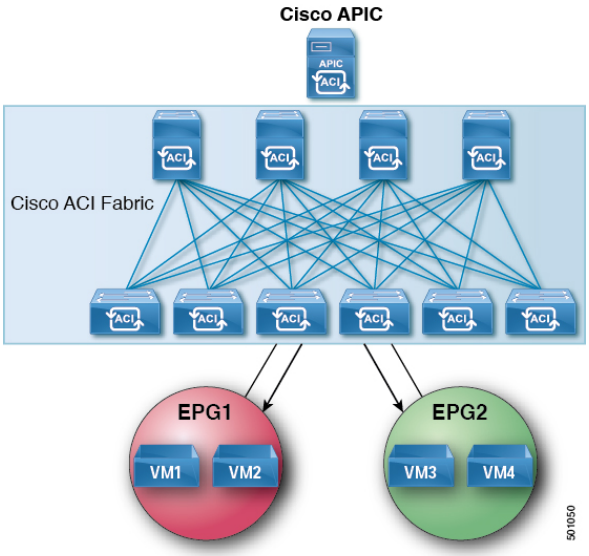


Table 19: ARP Table State

Device	State
VM1	IP = VM2 IP; MAC = ?
ACI fabric	IP = VM1 IP; MAC = VM1 MAC
VM2	IP = VM1 IP; MAC = BD MAC

4. VM2 sends an ARP response to the ACI fabric.

Figure 38: VM2 Sends an ARP Response to the ACI Fabric

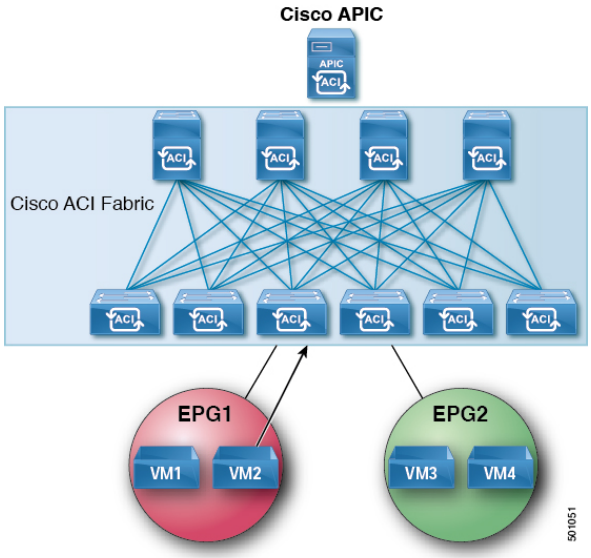


Table 20: ARP Table State

Device	State
VM1	IP = VM2 IP; MAC = ?
ACI fabric	IP = VM1 IP; MAC = VM1 MAC
VM2	IP = VM1 IP; MAC = BD MAC

5. VM2 is learned.

Figure 39: VM2 is Learned

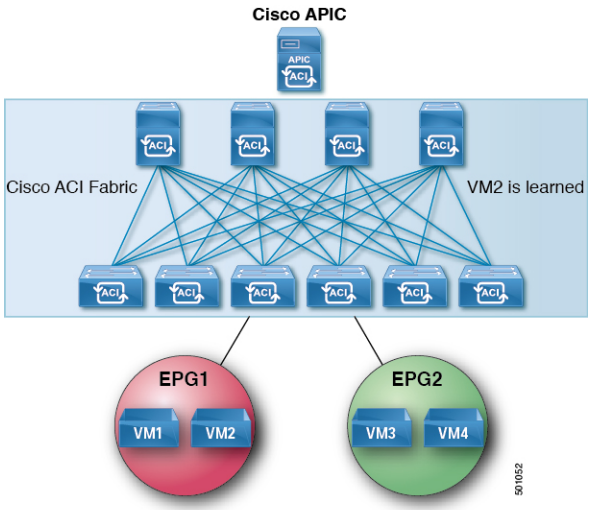


Table 21: ARP Table State

Device	State
VM1	IP = VM2 IP; MAC = ?
ACI fabric	IP = VM1 IP; MAC = VM1 MAC IP = VM2 IP; MAC = VM2 MAC
VM2	IP = VM1 IP; MAC = BD MAC

6. VM1 sends an ARP request with a broadcast MAC address to VM2.

Figure 40: VM1 Sends an ARP Request with a Broadcast MAC Address to VM2

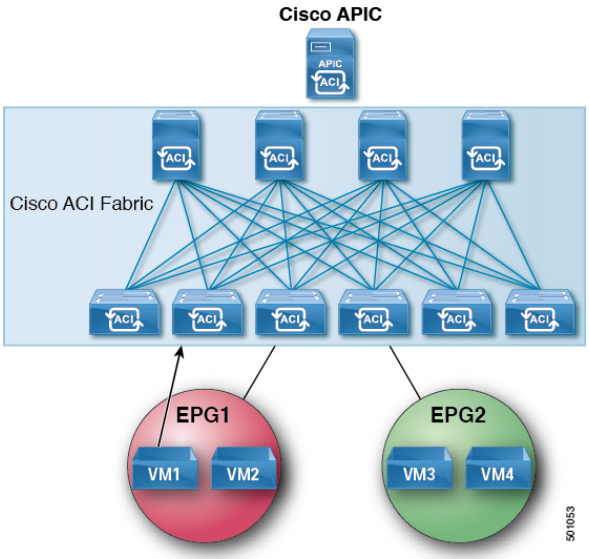


Table 22: ARP Table State

Device	State
VM1	IP = VM2 IP MAC = ?
ACI fabric	IP = VM1 IP; MAC = VM1 MAC IP = VM2 IP; MAC = VM2 MAC
VM2	IP = VM1 IP; MAC = BD MAC

7. The ACI fabric sends a proxy ARP response to VM1.

Figure 41: ACI Fabric Sends a Proxy ARP Response to VM1

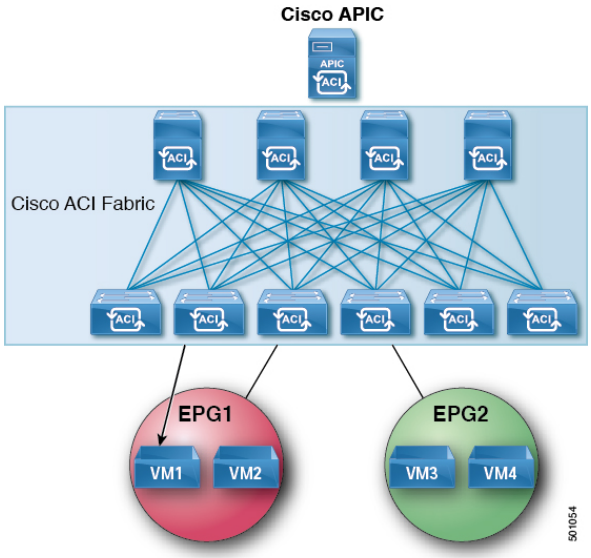


Table 23: ARP Table State

Device	State
VM1	IP = VM2 IP; MAC = BD MAC
ACI fabric	IP = VM1 IP; MAC = VM1 MAC IP = VM2 IP; MAC = VM2 MAC
VM2	IP = VM1 IP; MAC = BD MAC

Guidelines and Limitations

Consider these guidelines and limitations when using Proxy ARP:

- Proxy ARP is supported only on isolated EPGs. If an EPG is not isolated, a fault will be raised. For communication to happen within isolated EPGs with proxy ARP enabled, you must configure uSeg EPGs. For example, within the isolated EPG, there could be multiple VMs with different IP addresses, and you can configure a uSeg EPG with IP attributes matching the IP address range of these VMs.
- ARP requests from isolated endpoints to regular endpoints and from regular endpoints to isolated endpoints do not use proxy ARP. In such cases, endpoints communicate using the real MAC addresses of destination VMs.

Proxy ARP Supported Combinations

The following proxy ARP table provides the supported combinations:

ARP From/To	Regular EPG	Isolated Enforced EPG with Proxy ARP
Regular EPG	ARP	ARP
Isolated Enforced EPG with Proxy ARP	ARP	Proxy ARP

Configuring Proxy ARP Using the Advanced GUI

Before you begin

- The appropriate tenant, VRF, bridge domain, application profile and EPG must be created.
- Intra-EPG isolation must be enabled on the EPG where proxy ARP has to be enabled.

Procedure

Step 1 On the menu bar, click **Tenant** > **Tenant_name**.

Step 2 In the **Navigation** pane, expand the **Tenant_name** > **Application Profiles** > **Application_Profile_name** > **Application EPGs**, right click **Create Application EPG** dialog box to perform the following actions in the **Create Application EPG** dialog box:

a) In the **Name** field, add an EPG name.

Step 3 In the **Intra EPG Isolation** field, choose **Enforced**.
When Intra EPG isolation is enforced, the **Forwarding Control** field becomes available.

Step 4 In the **Forwarding Control** field, check the check box for **proxy-arp**.
This enables proxy-arp.

Step 5 In the **Bridge Domain** field, choose the appropriate bridge domain to associate from the drop-down list.

Step 6 Choose the remaining fields in the dialog box as appropriate, and click **Finish**.

Configuring Proxy ARP Using the Cisco NX-OS Style CLI

Before you begin

- The appropriate tenant, VRF, bridge domain, application profile and EPG must be created.
- Intra-EPG isolation must be enabled on the EPG where proxy ARP has to be enabled.

Procedure

	Command or Action	Purpose
Step 1	**configure** **Example:** `apic1# configure`	Enters configuration mode.
Step 2	**tenant** *tenant-name* **Example:** `apic1(config)# tenant Tenant1`	Enters the tenant configuration mode.
Step 3	**application** *application-profile-name* **Example:** `apic1(config-tenant)# application Tenant1-App`	Creates an application profile and enters the application mode.
Step 4	**epg** *application-profile-EPG-name* **Example:** `apic1(config-tenant-app)# epg Tenant1-epg1`	Creates an EPG and enter the EPG mode.
Step 5	**proxy-arp enable** **Example:** `apic1(config-tenant-app-epg)# proxy-arp enable`	Enables proxy ARP. **Note** You can disable proxy-arp with the **no proxy-arp** command.
Step 6	**exit** **Example:** `apic1(config-tenant-app-epg)# exit`	Returns to application profile mode.
Step 7	**exit** **Example:** `apic1(config-tenant-app)# exit`	Returns to tenant configuration mode.
Step 8	**exit** **Example:** `apic1(config-tenant)# exit`	Returns to global configuration mode.

Examples

This example shows how to configure proxy ARP.

```
apic1# conf t
apic1(config)# tenant Tenant1
apic1(config-tenant)# application Tenant1-App
```

```
apic1(config-tenant-app)# epg Tenant1-epg1
apic1(config-tenant-app-epg)# proxy-arp enable
apic1(config-tenant-app-epg)#
apic1(config-tenant)#
```

Configuring Proxy ARP Using the REST API

Before you begin

- Intra-EPG isolation must be enabled on the EPG where proxy ARP has to be enabled.

Procedure

Configure proxy ARP.

Example:

```
<polUni>
  <fvTenant name="Tenant1" status="">
    <fvCtx name="EngNet"/>
    <!-- bridge domain -->
    <fvBD name="BD1">
        <fvRsCtx tnFvCtxName="EngNet" />
        <fvSubnet ip="1.1.1.1/24"/>
    </fvBD>
    <fvAp name="Tenant1_app">
        <fvAEPg  name="Tenant1_epg" pcEnfPref-"enforced" fwdCtrl="proxy-arp">
            <fvRsBd tnFvBDName="BD1" />
            <fvRsDomAtt tDn="uni/vmmp-VMware/dom-dom9"/>
        </fvAEPg>
    </fvAp>
  </fvTenant>
</polUni>
```

CHAPTER 14

Traffic Storm Control

This chapter contains the following sections:

About Traffic Storm Control

A traffic storm occurs when packets flood the LAN, creating excessive traffic and degrading network performance. You can use traffic storm control policies to prevent disruptions on Layer 2 ports by broadcast, unknown multicast, or unknown unicast traffic storms on physical interfaces.

By default, storm control is not enabled in the ACI fabric. ACI bridge domain (BD) Layer 2 unknown unicast flooding is enabled by default within the BD but can be disabled by an administrator. In that case, a storm control policy only applies to broadcast and unknown multicast traffic. If Layer 2 unknown unicast flooding is enabled in a BD, then a storm control policy applies to Layer 2 unknown unicast flooding in addition to broadcast and unknown multicast traffic.

Traffic storm control (also called traffic suppression) allows you to monitor the levels of incoming broadcast, multicast, and unknown unicast traffic over a one second interval. During this interval, the traffic level, which is expressed either as percentage of the total available bandwidth of the port or as the maximum packets per second allowed on the given port, is compared with the traffic storm control level that you configured. When the ingress traffic reaches the traffic storm control level that is configured on the port, traffic storm control drops the traffic until the interval ends. An administrator can configure a monitoring policy to raise a fault when a storm control threshold is exceeded.

Storm Control Guidelines

Configure traffic storm control levels according to the following guidelines and limitations:

- Typically, a fabric administrator configures storm control in fabric access policies on the following interfaces:
  - A regular trunk interface.

  - A direct port channel on a single leaf switch.
  - A virtual port channel (a port channel on two leaf switches).

- For port channels and virtual port channels, the storm control values (packets per second or percentage) apply to all individual members of the port channel.

Note On switch hardware starting with the APIC 1.3(x) and switch 11.3(x) release, for port channel configurations, the traffic suppression on the aggregated port may be up to two times the configured value. The new hardware ports are internally subdivided into these two groups: slice-0 and slice-1. To check the slicing map, use the `vsh_lc` command `show platform internal hal l2 port gpd` and look for `slice 0` or `slice 1` under the `Sl` column. If port channel members fall on both slice-0 and slice-1, allowed storm control traffic may become twice the configured value because the formula is calculated based on each slice.

- When configuring by percentage of available bandwidth, a value of 100 means no traffic storm control and a value of 0.01 suppresses all traffic.
- Due to hardware limitations and the method by which packets of different sizes are counted, the level percentage is an approximation. Depending on the sizes of the frames that make up the incoming traffic, the actual enforced level might differ from the configured level by several percentage points. Packets-per-second (PPS) values are converted to percentage based on 256 bytes.
- Maximum burst is the maximum accumulation of rate that is allowed when no traffic passes. When traffic starts, all the traffic up to the accumulated rate is allowed in the first interval. In subsequent intervals, traffic is allowed only up to the configured rate. The maximum supported is 65535 KB. If the configured rate exceeds this value, it is capped at this value for both PPS and percentage.
- The maximum burst that can be accumulated is 512 MB.
- On an egress leaf switch in optimized multicast flooding (OMF) mode, traffic storm control will not be applied.
- On an egress leaf switch in non-OMF mode, traffic storm control will be applied.
- On a leaf switch for FEX, traffic storm control is not available on host-facing interfaces.
- Traffic storm control unicast/multicast differentiation is not supported on Cisco Nexus C93128TX, C9396PX, C9396TX, C93120TX, C9332PQ, C9372PX, C9372TX, C9372PX-E, or C9372TX-E switches.
- SNMP traps for traffic storm control are not supported on Cisco Nexus C93128TX, C9396PX, C9396TX, C93120TX, C9332PQ, C9372PX, C9372TX, C9372PX-E, C9372TX-E switches.
- Traffic storm control traps is not supported on Cisco Nexus C93128TX, C9396PX, C9396TX, C93120TX, C9332PQ, C9372PX, C9372TX, C9372PX-E, or C9372TX-E switches.
- Storm Control Action is supported only on physical Ethernet interfaces and port channel interfaces.
- If DPP policer configured for the interface has a value that is lower than storm policer's value, the DPP policer will take the precedence. The lower value that is configured between the DPP policer and storm policer is honored on the configured interface.

- Traffic storm control cannot police multicast traffic in a bridge domain or VRF instance that has PIM enabled.
- When the storm control policer is applied on a port channel interface, the allowed rate may be more than the configured rate. If the member links of the port channel span across multiple slices, then the allowed traffic rate will be equal to the configured rate multiplied by the number of slices across which the member links span.

  The port-to-slice mapping depends on the switch model.

  As an example, assume that there is a port channel that has member links port1, port2, and port3 with a storm policer rate of 10Mbps.

  - If port1, port2, and port3 belong to slice1, then traffic is policed to 10Mbps.
  - If port1 and port2 belong to slice1 and port3 belongs to slice2, then traffic is policed to 20Mbps.
  - If port1 belongs to slice1, port2 belongs to slice2, and port3 belongs to slice3, then traffic is policed to 30Mbps.

Configuring a Traffic Storm Control Policy Using the GUI

Procedure

Step 1 In the menu bar, click **Fabric**.

Step 2 In the submenu bar, click **Access Policies**.

Step 3 In the **Navigation** pane, expand **Policies**.

Step 4 Expand **Interface**.

Step 5 Right-click **Storm Control** and choose **Create Storm Control Interface Policy**.

Step 6 In the **Create Storm Control Interface Policy** dialog box, enter a name for the policy in the **Name** field.

Step 7 In the **Configure Storm Control** field, click the radio button for either **All Types** or **Unicast, Broadcast, Multicast**.

Note Selecting the **Unicast, Broadcast, Multicast** radio button allows you to configure Storm Control on each traffic type separately.

Step 8 In the **Specify Policy In** field, click the radio button for either **Percentage** or **Packets Per Second**.

Step 9 If you chose **Percentage**, perform the following steps:

a) In the **Rate** field, enter a traffic rate percentage.

Enter a number between 0 and 100 that specifies a percentage of the total available bandwidth of the port. When the ingress traffic is either equal to or greater than this level during a one second interval, traffic storm control drops traffic for the remainder of the interval. A value of 100 means no traffic storm control. A value of 0 suppresses all traffic.

b) In the **Max Burst Rate** field, enter a burst traffic rate percentage.

Enter a number between 0 and 100 that specifies a percentage of the total available bandwidth of the port. When the ingress traffic is equal to or greater than, traffic storm control begins to drop traffic.

Note The **Max Burst Rate** should be greater than or equal to the value of **Rate**.

Step 10 If you chose **Packets Per Second**, perform the following steps:

a) In the **Rate** field, enter a traffic rate in packets per second.

During this interval, the traffic level, expressed as packets flowing per second through the port, is compared with the traffic storm control level that you configured. When the ingress traffic is equal to or greater than the traffic storm control level that is configured on the port, traffic storm control drops the traffic until the interval ends.

b) In the **Max Burst Rate** field, enter a burst traffic rate in packets per second.

During this interval, the traffic level, expressed as packets flowing per second through the port, is compared with the burst traffic storm control level that you configured. When the ingress traffic is equal to or greater than the traffic storm control level that is configured on the port, traffic storm control drops the traffic until the interval ends.

Step 11 The policy action can be altered from the default by selecting shutdown in the **Storm Control Action** and adjusting the default in **Storm Control Soak Count** fields.

Note When the **shutdown** action is selected for an interface with the default Soak Instance Count, the packets exceeding the threshold are dropped for 3 seconds and the port is shutdown on the 3rd second.

Step 12 Click **Submit**.

Step 13 Apply the storm control interface policy to an interface port.

a) In the menu bar, click **Fabric**.
b) In the submenu bar, click **Access Policies**.
c) In the **Navigation** pane, expand **Interfaces**.
d) Expand **Leaf Interfaces**.
e) Expand **Policy Groups**.
f) Select **Leaf Policy Groups**.

Note If your APIC version is earlier than 2.x, you select **Policy Groups**.

g) Select the leaf access port policy group, the PC interface policy group, the vPC interface policy group, or the PC/vPC override policy group to which you want to apply the storm control policy.
h) In the **Work** pane, click the drop down for **Storm Control Interface Policy** and select the created **Traffic Storm Control Policy**.
i) Click **Submit**.

Configuring a Traffic Storm Control Policy Using the NX-OS Style CLI

Procedure

	Command or Action	Purpose

Step 1	Enter the following commands to create a PPS policy:
	Example:
	`(config)# template policy-group pg1`
	`(config-pol-grp-if)# storm-control pps 10000 burst-rate 10000`	

Step 2	Enter the following commands to create a percent policy:
	Example:
	`(config)# template policy-group pg2`
	`(config-pol-grp-if)# storm-control level 50 burst-rate 60`	

Step 3	Configure storm control on physical ports, port channels, or virtual port channels:
	Example:
	`[no] storm-control [unicast|multicast|broadcast] level <percentage> [burst-rate <percentage>]`
	`[no] storm-control [unicast|multicast|broadcast] pps <packet-per-second> [burst-rate <packet-per-second>]`
	
	`sd-tb2-ifc1# configure terminal`
	
	`sd-tb2-ifc1(config)# leaf 102`
	
	`sd-tb2-ifc1(config-leaf)# interface ethernet 1/19`
	`sd-tb2-ifc1(config-leaf-if)# storm-control unicast level 35 burst-rate 45`
	`sd-tb2-ifc1(config-leaf-if)# storm-control broadcast level 36 burst-rate 36`
	`sd-tb2-ifc1(config-leaf-if)# storm-control broadcast level 37 burst-rate 38`
	`sd-tb2-ifc1(config-leaf-if)#`
	
	`sd-tb2-ifc1# configure terminal`
	
	`sd-tb2-ifc1(config)# leaf 102`
	
	`sd-tb2-ifc1(config-leaf)# interface`	

	Command or Action	Purpose

	`ethernet 1/19`
	`sd-tb2-ifc1(config-leaf-if)#`
	`storm-control broadcast pps 5000 burst-rate 6000`
	`sd-tb2-ifc1(config-leaf-if)#`
	`storm-control unicast pps 7000 burst-rate 7000`
	`sd-tb2-ifc1(config-leaf-if)#`
	`storm-control unicast pps 8000 burst-rate 10000`
	`sd-tb2-ifc1(config-leaf-if)#`	

Configuring a Traffic Storm Control Policy Using the REST API

To configure a traffic storm control policy, create a `stormctrl:IfPol` object with the desired properties.

To create a policy named MyStormPolicy, send this HTTP POST message:

```
POST https://192.0.20.123/api/mo/uni/infra/stormctrlifp-MyStormPolicy.json
```

In the body of the POST message, Include the following JSON payload structure to specify the policy by percentage of available bandwidth:

```
{"stormctrlIfPol":
    {"attributes":
        {"dn":"uni/infra/stormctrlifp-MyStormPolicy",
        "name":"MyStormPolicy",
        "rate":"75",
        "burstRate":"85",
        "rn":"stormctrlifp-MyStormPolicy",
        "status":"created"
        },
        "children":[]
    }
}
```

In the body of the POST message, Include the following JSON payload structure to specify the policy by packets per second:

```
{"stormctrlIfPol":
    {"attributes":
        {"dn":"uni/infra/stormctrlifp-MyStormPolicy",
        "name":"MyStormPolicy",
        "ratePps":"12000",
        "burstPps":"15000",
        "rn":"stormctrlifp-MyStormPolicy",
        "status":"created"
        },
        "children":[]
    }
}
```

Configuring a Storm Control SNMP Trap

This section describes how to configure a storm control SNMP trap on leaf switches.

You can configure a storm control on SNMP trap using a trap name on the MIB definition. An event on MIB for an interface and when the storm is detected and cleared, a trap is filtered on the same leaf to configure the storm. You can configure the storm in two ways:

- Granular configuration—Sets the type of traffic such as, unicast, multicast and broadcast.
- Non-granular configuration—Sets all types of traffic.

For details on restrictions for triggering the SNMP traps from Cisco ACI when storm control thresholds are met, see Storm Control Guidelines, on page 249. For details on Cisco Nexus switches that are not supported on traffic storm control traps, see the guidelines for Storm Control.

Storm Trap

The storm trap will be triggered whenever there is an event and the storm is active or cleared.

```
cpscEventRev1 NOTIFICATION-TYPE
        OBJECTS   { cpscStatus }
        STATUS     current
        DESCRIPTION
```

The implementation sends this notification when a storm event occurs on an interface with respect to a particular traffic type.

The storm status is updated in the fields: `bcDropIncreased`, `uucDropIncreased`, `mcDropIncreased`, and `dropIncreased` for broadcast, unicast, multicast and non-granular traffic types respectively of `dbgIfStorm` MO. The granular and non-granular configurations use flags to set the storm. When a strom is active the flag is set to 1 and when the storm is cleared the flag is set to 2. The following flags generate the events required for the SNMP trap trigger.

```
cat /mit/sys/phys-\[eth--1\]/dbgIfStorm/summary
```

```
# Interface Storm Drop Couters
bcDropBytes   :0
bcDropIncreased   :2
childAction    :
dn        :sys/phys-[eth/1]/dbgIfStorm
dropBytes      :0
dropIncreased  :2
mcDropBytes    :0
mcDropIncreased :2
modTs           :never
monPoIDn        :uni/infra/moninfra-default
m               :dbgIfStorm
status          :
uucDropBytes    :0
uucDropIncreased :2
```

CHAPTER 15

MACsec

This chapter contains the following sections:

About MACsec

MACsec is an IEEE 802.1AE standards based Layer 2 hop-by-hop encryption that provides data confidentiality and integrity for media access independent protocols.

MACsec, provides MAC-layer encryption over wired networks by using out-of-band methods for encryption keying. The MACsec Key Agreement (MKA) Protocol provides the required session keys and manages the required encryption keys.

The 802.1AE encryption with MKA is supported on all types of links, that is, host facing links (links between network access devices and endpoint devices such as a PC or IP phone), or links connected to other switches or routers.

MACsec encrypts the entire data except for the Source and Destination MAC addresses of an Ethernet packet. The user also has the option to skip encryption up to 50 bytes after the source and destination MAC address.

To provide MACsec services over the WAN or Metro Ethernet, service providers offer Layer 2 transparent services such as E-Line or E-LAN using various transport layer protocols such as Ethernet over Multiprotocol Label Switching (EoMPLS) and L2TPv3.

The packet body in an EAP-over-LAN (EAPOL) Protocol Data Unit (PDU) is referred to as a MACsec Key Agreement PDU (MKPDU). When no MKPDU is received from a participants after 3 hearbeats (each hearbeat is of 2 seconds), peers are deleted from the live peer list. For example, if a client disconnects, the participant on the switch continues to operate MKA until 3 heartbeats have elapsed after the last MKPDU is received from the client.

APIC Fabric MACsec

The APIC will be responsible for the MACsec keychain distribution to all the nodes in a Pod or to particular ports on a node. Below are the supported MACsec keychain and MACsec policy distribution supported by the APIC.

- A single user provided keychain and policy per Pod
- User provided keychain and user provided policy per fabric interface
- Auto generated keychain and user provided policy per Pod

A node can have multiple policies deployed for more than one fabric link. When this happens, the per fabric interface keychain and policy are given preference on the affected interface. The auto generated keychain and associated MACsec policy are then given the least preference.

APIC MACsec supports two security modes. The MACsec **must secure** only allows encrypted traffic on the link while the **should secure** allows both clear and encrypted traffic on the link. Before deploying MACsec in **must secure** mode, the keychain must be deployed on the affected links or the links will go down. For example, a port can turn on MACsec in **must secure** mode before its peer has received its keychain resulting in the link going down. To address this issue the recommendation is to deploy MACsec in **should secure** mode and once all the links are up then change the security mode to **must secure**.

Note Any MACsec interface configuration change will result in packet drops.

MACsec policy definition consists of configuration specific to keychain definition and configuration related to feature functionality. The keychain definition and feature functionality definitions are placed in separate policies. Enabling MACsec per Pod or per interface involves deploying a combination of a keychain policy and MACsec functionality policy.

Note Using internal generated keychains do not require the user to specify a keychain.

APIC Access MACsec

MACsec is used to secure links between leaf switch L3out interfaces and external devices. APIC provides GUI and CLI to allow users to program the MACsec keys and MacSec configuration for the L3Out interfaces on the fabric on a per physical/pc/vpc interface basis. It is the responsibility of the user to make sure that the external peer devices are programmed with the correct MacSec information.

Guidelines and Limitations for MACsec

Configure MACsec according to the following guidelines and limitations:

- MACsec is supported on the following switches:
  - N9K-C93108TC-FX
  - N9K-C93180YC-FX
  - N9K-C93216TC-FX2

  - N9K-C93240YC-FX2
  - N9K-C9332C
  - N9K-C93360YC-FX2
  - N9K-C9336C-FX2
  - N9K-C9348GC-FXP, only with 10G+
  - N9K-C9364C

- MACsec is supported on the following line card:
  - N9K-X9736C-FX

- MACsec is not supported on 10G QSA modules.
- Beginning with Cisco Application Policy Infrastructure Controller (APIC) release 4.0, MACsec is supported on remote leaf switches.
- FEX ports are not supported for MACsec.
- The **must-secure** mode is not supported at the pod level.
- A MACsec policy with the name "default" is not supported.
- Auto key generation is only supported at the pod level for fabric ports.
- Do not clean reboot a node if the fabric ports of that node is running MACsec in **must-secure** mode.
- Adding a new node to a pod or stateless reboot of a node in a pod that is running MACsec, **must-secure** mode requires changing the mode to **should-secure** for the node to join the pod.
- Only initiate an upgrade or downgrade if the fabric links are in the **should-secure** mode. After the upgrade or downgrade has completed, you can change the mode to **must-secure**. Upgrading or downgrading in the **must-secure** mode results in nodes losing connectivity to the fabric. Recovering from connectivity loss requires you to configure in **should-secure** mode the fabric links of the nodes that are visible to the Cisco APIC. If the fabric was downgraded to a version which does not support MACsec, then nodes which are out of fabric will need to be clean rebooted.
- For a PC or vPC interface, MACsec can be deployed using policy groups per PC or vPC interface. Port selectors are used to deploy the policies to a particular set of ports. Therefore, you must create the correct port selector that corresponds to the L3Out interfaces.
- We recommend that you configure MACsec polices with the **should-secure** mode before you export a configuration.
- All of the links on a spine switch are considered to be fabric links. However, if a spine switch link is used for IPN connectivity, then this link will be treated as an access link. This means that a MACsec access policy must be used to deploy MACsec on these links.
- If a remote leaf fabric link is used for IPN connectivity, then this link will be treated as an access link. A MACsec access policy needs to be used to deploy MACsec on these links.
- Improper deployment of **must-secure** mode on remote leaf switch fabric links can result in loss of connectivity to the fabric. Follow the instructions provided in Deploying must-secure mode, on page 260 to prevent such issues.

- MACsec sessions can take up to a minute to form or tear down when a new key is added to an empty keychain or an active key is deleted from a keychain.
- Before reloading a line card or fabric module on a spine switch, all **must-secure** links should be changed to the **should-secure** mode. After the reload completes and the session comes up in the **should-secure** mode, change the mode to **must-secure**.
- When selecting the cipher suite AES 128 or AES 256 without Extended Packet Numbering (XPN), you must explicitly specify the Security Association Key (SAK) expiry time. Leaving the SAK expiry time value at the default ("disabled") can cause interfaces to go out of service randomly.
- A replay window is necessary to support the use of MACsec over provider networks that reorder frames. Frames within the window can be received out of order, but are not replay protected. The default window size is 64. The replay window size can be configured in the range of 0 to 2^{32}-1 if you use the Cisco APIC GUI or CLI. If you use a XPN cipher suite, the maximum replay window size is 2^{30}-1, and if you configure a higher window size, the window size gets restricted to 2^{30}-1. If you change the cipher suite to a non-XPN cipher suite, then there is no restriction and the configured window size is used.
- Link-level flow control (LLFC) and priority flow control (PFC) are not supported with MACsec.

Deploying must-secure mode

Incorrect deployment procedure of a policy that is configured for **must-secure** mode can result in a loss of connectivity. The procedure below should be followed in order to prevent such issues:

- It is necessary to ensure that each link pair has their keychains before enabling MACsec **must-secure** mode. To ensure this, the recommendation is to deploy the policy in **should-secure** mode, and once MACsec sessions are active on the expected links, change the mode to **must-secure**.
- Attempting to replace the keychain on a MACsec policy that is configured to **must-secure** can cause links to go down. The recommended procedure outlined below should be followed in this case:
  - Change MACsec policy that is using the new keychain to **should-secure** mode.
  - Verify that the affected interfaces are using should-secure mode.
  - Update MACsec policy to use new keychain.
  - Verify that relevant interfaces with active MACsec sessions are using the new keychain.
  - Change MACsec policy to **must-secure** mode.
- The following procedure should be followed to disable/remove a MACsec policy deployed in must-secure mode:
  - Change the MACsec policy to **should-secure**.
  - Verify that the affected interfaces are using **should-secure** mode.
  - Disable/remove the MACsec policy.

Keychain Definition

- There should be one key in the keychain with a start time of **now**. If **must-secure** is deployed with a keychain that doesn’t have a key that is immediately active then traffic will be blocked on that link until

the key becomes current and a MACsec session is started. If **should-secure** mode is being used then traffic will be unencrypted until the key becomes current and a MACsec session has started.

- There should be one key in the keychain with an end time of **infinite**. When a keychain expires, then traffic is blocked on affected interfaces which are configured for **must-secure** mode. Interfaces configured for **should-secure** mode transmit unencrypted traffic.
- There should be overlaps in the end time and start time of keys that are used sequentially to ensure the MACsec session stays up when there is a transition between keys.

Configuring MACsec for Fabric Links Using the GUI

Procedure

Step 1 On the menu bar, click **Fabric** > **Fabric Policies** > **Policies** > **MACsec** > **Interfaces**. In the **Navigation** pane, right click on **Interfaces** to open **Create MACsec Fabric Interface Policy** and perform the following actions:

a) In the **Name** field, enter a name for the MACsec Fabric Interface policy.
b) In the **MACsec Parameters** field, either select a previously configured MACsec Parameters policy or create a new one.
c) In the **MACsec Keychain Policy** field, either select a previously configured MACsec Parameters policy or create a new one and click **Submit**.

To create a **MACsec Keychain Policy**, see Configuring MACsec Keychain Policy Using the GUI, on page 262.

Step 2 To apply the **MACsec Fabric Interface Policy** to a Fabric Leaf or Spine Port Policy Group, in the Navigation pane, click **Interfaces** > ***Leaf/Spine*** **Interfaces** > **Policy Groups** > ***Spine/Leaf*** **Port Policy Group_*name*** . In the **Work** pane, select the **MACsec Fabric Interface Policy** just created.

Step 3 To apply the **MACsec Fabric Interface Policy** to a Pod Policy Group, in the Navigation pane, click **Pods** > **Policy Groups** > **Pod Policy Group_*name*** . In the **Work** pane, select the **MACsec Fabric Interface Policy** just created.

Configuring MACsec for Access Links Using the GUI

Procedure

Step 1 On the menu bar, click **Fabric** > **External Access Policies**. In the **Navigation** pane, click on **Policies** > **Interface** > **MACsec** > **Interfaces** and right click on **Interfaces** to open **Create MACsec Fabric Interface Policy** and perform the following actions:

a) In the **Name** field, enter a name for the MACsec Access Interface policy.
b) In the **MACsec Parameters** field, either select a previously configured MACsec Parameters policy or create a new one.

c) In the **MACsec Keychain Policy** field, either select a previously configured MACsec Parameters policy or create a new one and click **Submit**.

To create a **MACsec Keychain Policy**, see Configuring MACsec Keychain Policy Using the GUI, on page 262.

Step 2 To apply the **MACsec Access Interface Policy** to a Fabric Leaf or Spine Port Policy Group, in the Navigation pane, click **Interfaces** > ***Leaf/Spine*** **Interfaces** > **Policy Groups** > ***Spine/Leaf*** **Policy Group_*name*** . In the **Work** pane, select the **MACsec Fabric Interface Policy** just created.

Configuring MACsec Parameters Using the APIC GUI

Procedure

Step 1 On the menu bar, click **Fabric** > **Access Policies**. In the **Navigation** pane, click on **Interface Policies** > **Policies** and right click on **MACsec Policies** to open **Create MACsec Access Parameters Policy** and perform the following actions:

a) In the **Name** field, enter a name for the MACsec Access Parameters policy.
b) In the **Security Policy** field, select a mode for encrypted traffic and click **Submit**.

Note Before deploying MACsec in **Must Secure Mode**, the keychain must be deployed on the affected interface or the interface will go down.

Step 2 To apply the **MACsec Access Parameters Policy** to a Leaf or Spine Port Policy Group, in the Navigation pane, click **Interface Policies** > **Policy Groups** > **Spine/Leaf Policy Group_*name*** . In the **Work** pane, select the **MACsec Access Interface Policy** just created.

Configuring MACsec Keychain Policy Using the GUI

Procedure

Step 1 On the menu bar, click **Fabric** > **Fabric Policies** > **Policies** > **MACsec** > **KeyChains**. In the **Navigation** pane, right click on **KeyChains** to open **Create MACsec Keychain Policy** and perform the following actions:

a) In the **Name** field, enter a name for the MACsec Fabric Interface policy.
b) Expand the **MACsec Key Policy** table to create the Key policy.

Step 2 In the **MACsec Key Policy** dialog box perform the following actions:

a) In the **Name** field, enter a name for the MACsec Key policy.
b) In the **Key Name** field, enter a key name (up to 64 hexadecimal characters).

Note A maximum of 64 keys are supported per keychain.

c) In the **Pre-shared Key** field, enter the pre-shared key information.

Note
- For 128-bit cipher suites only 32 character PSKs are permitted.
- For 256-bit cipher suites only 64 Character PSKs are permitted.

d) In the **Start Time** field, select a date for the key to become valid.
e) In the **End Time** field, select a date for the key to expire. Click **Ok** and **Submit**.

Note When defining multiple keys in a keychain, the keys must be defined with overlapping times in order to assure a smooth transition from the old key to the new key. The endTime of the old key should overlap with the startTime of the new key.

For configuring the Keychain policy through Access Policies, on the menu bar click **Fabric** > **External Access Policies**. In the **Navigation** pane, click on **Policies** > **Interface** > **MACsec** > **MACsec KeyChain Policies** and right click on to open **Create MACsec Keychain Policy** and perform the steps above.

Configuring MACsec Using the NX-OS Style CLI

Procedure

Step 1 Configure MACsec Security Policy for access interfaces

Example:

```
apic1# configure
apic1(config)#   template macsec access security-policy accmacsecpol1
apic1(config-macsec-param)#     cipher-suite gcm-aes-128
apic1(config-macsec-param)#     conf-offset offset-30
apic1(config-macsec-param)#     description 'description for mac sec parameters'
apic1(config-macsec-param)#     key-server-priority 1
apic1(config-macsec-param)#     sak-expiry-time 110
apic1(config-macsec-param)#     security-mode must-secure
aapic1(config-macsec-param)#     window-size 1
apic1(config-macsec-param)#     exit
apic1(config)#
```

Step 2 Configure MACsec key chain for access interface:

PSK can be configured in 2 ways:

Note
- Inline with the **psk-string** command as illustrated in key 12ab below. The PSK is not secure because it is logged and exposed.
- Entered separately in a new command **Enter PSK string** after the **psk-string** command as illustrated in key ab12. The PSK is secured because it is only echoed locally and is not logged.

Example:

```
apic1# configure
apic1(config)#   template macsec access keychain acckeychainpol1
```

```
apic1(config-macsec-keychain)#     description 'macsec key chain kc1'
apic1(config-macsec-keychain)#     key 12ab
apic1(config-macsec-keychain-key)#       life-time start 2017-09-19T12:03:15 end
2017-12-19T12:03:15
apic1(config-macsec-keychain-key)#       psk-string 123456789a223456789a323456789abc
apic1(config-macsec-keychain-key)#       exit
apic1(config-macsec-keychain)#     key ab12
apic1(config-macsec-keychain-key)#       life-time start now end infinite
apic1(config-macsec-keychain-key)#       life-time start now end infinite
apic1(config-macsec-keychain-key)# psk-string
Enter PSK string: 123456789a223456789a323456789abc
apic1(config-macsec-keychain-key)# exit
apic1(config-macsec-keychain)# exit
apic1(config)#
```

Step 3 Configure MACsec interface policy for access interface:

Example:

```
apic1# configure
apic1(config)#   template macsec access interface-policy accmacsecifpol1
apic1(config-macsec-if-policy)#     inherit macsec security-policy accmacsecpol1 keychain
acckeychainpol1
apic1(config-macsec-if-policy)#     exit
apic1(config)#
```

Step 4 Associate MACsec interface policy to access interfaces on leaf (or spine):

Example:

```
apic1# configure
apic1(config)#   template macsec access interface-policy accmacsecifpol1
apic1(config-macsec-if-policy)#     inherit macsec security-policy accmacsecpol1 keychain
acckeychainpol1
apic1(config-macsec-if-policy)#     exit
apic1(config)
```

Step 5 Configure MACsec Security Policy for fabric interfaces:

Example:

```
apic1# configure
apic1(config)#   template macsec fabric security-policy fabmacsecpol1
apic1(config-macsec-param)#     cipher-suite gcm-aes-xpn-128
apic1(config-macsec-param)#     description 'description for mac sec parameters'
apic1(config-macsec-param)#     window-size 1
apic1(config-macsec-param)#     sak-expiry-time 100
apic1(config-macsec-param)#     security-mode must-secure
apic1(config-macsec-param)#     exit
apic1(config)#
```

Step 6 Configure MACsec key chain for fabric interface:

PSK can be configured in 2 ways:

Note

- Inline with the **psk-string** command as illustrated in key 12ab below. The PSK is not secure because it is logged and exposed.
- Entered separately in a new command **Enter PSK string** after the **psk-string** command as illustrated in key ab12. The PSK is secured because it is only echoed locally and is not logged.

Example:

```
apic1# configure
apic1(config)#   template macsec fabric security-policy fabmacsecpol1
apic1(config-macsec-param)#     cipher-suite gcm-aes-xpn-128
apic1(config-macsec-param)#     description 'description for mac sec parameters'
apic1(config-macsec-param)#     window-size 1
apic1(config-macsec-param)#     sak-expiry-time 100
apic1(config-macsec-param)#     security-mode must-secure
apic1(config-macsec-param)#     exit
apic1(config)#   template macsec fabric keychain fabkeychainpol1
apic1(config-macsec-keychain)#     description 'macsec key chain kc1'
apic1(config-macsec-keychain)#     key 12ab
apic1(config-macsec-keychain-key)#       psk-string 123456789a223456789a323456789abc
apic1(config-macsec-keychain-key)#       life-time start 2016-09-19T12:03:15 end
2017-09-19T12:03:15
apic1(config-macsec-keychain-key)#       exit
apic1(config-macsec-keychain)#     key cd78
apic1(config-macsec-keychain-key)# psk-string
Enter PSK string: 123456789a223456789a323456789abc
apic1(config-macsec-keychain-key)# life-time start now end infinite
apic1(config-macsec-keychain-key)# exit
apic1(config-macsec-keychain)# exit
apic1(config)#
```

Step 7 Associate MACsec interface policy to fabric interfaces on leaf (or spine):

Example:

```
apic1# configure
apic1(config)#   leaf 101
apic1(config-leaf)#     fabric-interface ethernet 1/52-53
apic1(config-leaf-if)#       inherit macsec interface-policy fabmacsecifpol2
apic1(config-leaf-if)#       exit
apic1(config-leaf)#
```

Configuring MACsec Using the REST API

Apply a MACsec fabric policy to all Pods in the fabric:

Example:

```
<fabricInst>
 <macsecFabPolCont>
       <macsecFabParamPol name="fabricParam1"  secPolicy="should-secure" replayWindow="120"
 >
        </macsecFabParamPol>
        <macsecKeyChainPol name="fabricKC1">
            <macsecKeyPol name="Key1"
preSharedKey="0102030405060708090A0B0C0D0E0F10010203040506070809 0A0B0C0D0E0F10"
keyName="A1A2A3A0" startTime="now" endTime="infinite"/>
        </macsecKeyChainPol>
    </macsecFabPolCont>

    <macsecFabIfPol name="fabricPodPol1" useAutoKeys="0">
        <macsecRsToParamPol tDn="uni/fabric/macsecpcontfab/fabparamp-fabricParam1"/>
        <macsecRsToKeyChainPol tDn="uni/fabric/macsecpcontfab/keychainp-fabricKC1"/>
    </macsecFabIfPol>

    <fabricFuncP>
  <fabricPodPGrp name = "PodPG1">
   <fabricRsMacsecPol tnMacsecFabIfPolName="fabricPodPol1"/>
```

```
     </fabricPodPGrp>
 </fabricFuncP>

    <fabricPodP name="PodP1">
  <fabricPodS name="pod1" type="ALL">
   <fabricRsPodPGrp tDn="uni/fabric/funcprof/podpgrp-PodPG1"/>
     </fabricPodS>
 </fabricPodP>

</fabricInst>
```

Applying a MACsec access policy on eth1/4 of leaf-101:

Example:

```
<infraInfra>
 <macsecPolCont>
        <macsecParamPol name="accessParam1"  secPolicy="should-secure" replayWindow="120"
>
        </macsecParamPol>
        <macsecKeyChainPol name="accessKC1">
            <macsecKeyPol name="Key1"
preSharedKey="0102030405060708090A0B0C0D0E0F100102030405060708090A0B0C0D0E0F10"
keyName="A1A2A3A0" startTime="now" endTime="infinite"/>
        </macsecKeyChainPol>
    </macsecPolCont>

    <macsecIfPol name="accessPol1">
        <macsecRsToParamPol tDn="uni/infra/macsecpcont/paramp-accessParam1"/>
        <macsecRsToKeyChainPol tDn="uni/infra/macsecpcont/keychainp-accessKC1"/>
    </macsecIfPol>

    <infraFuncP>
  <infraAccPortGrp name = "LeTestPGrp">
   <infraRsMacsecIfPol tnMacsecIfPolName="accessPol1"/>
     </infraAccPortGrp>
 </infraFuncP>

 <infraHPathS name="leaf">
  <infraRsHPathAtt tDn="topology/pod-1/paths-101/pathep-[eth1/4]" />
  <infraRsPathToAccBaseGrp tDn="uni/infra/funcprof/accportgrp-LeTestPGrp" />
 </infraHPathS>

</infraInfra>
```

Applying a MACsec fabric policy on eth1/49 of leaf-101 and eth 5/1 of spine-102:

```
<fabricInst>
 <macsecFabPolCont>
       <macsecFabParamPol name="fabricParam1"  secPolicy="should-secure" replayWindow="120"
 >
        </macsecFabParamPol>
        <macsecKeyChainPol name="fabricKC1">
            <macsecKeyPol name="Key1"
preSharedKey="0102030405060708090A0B0C0D0E0F100102030405060708090A0B0C0D0E0F10"
keyName="A1A2A3A0" startTime="now" endTime="infinite"/>
        </macsecKeyChainPol>
    </macsecFabPolCont>

    <macsecFabIfPol name="fabricPol1" useAutoKeys="0">
        <macsecRsToParamPol tDn="uni/fabric/macsecpcontfab/fabparamp-fabricParam1"/>
        <macsecRsToKeyChainPol tDn="uni/fabric/macsecpcontfab/keychainp-fabricKC1"/>
    </macsecFabIfPol>

    <fabricFuncP>
```

```
 <fabricLePortPGrp name = "LeTestPGrp">
  <fabricRsMacsecFabIfPol tnMacsecFabIfPolName="fabricPol1"/>
    </fabricLePortPGrp>

 <fabricSpPortPGrp name = "SpTestPGrp">
  <fabricRsMacsecFabIfPol tnMacsecFabIfPolName="fabricPol1"/>
    </fabricSpPortPGrp>
</fabricFuncP>

<fabricLFPathS name="leaf">
 <fabricRsLFPathAtt tDn="topology/pod-1/paths-101/pathep-[eth1/49]" />
 <fabricRsPathToLePortPGrp tDn="uni/fabric/funcprof/leportgrp-LeTestPGrp" />
</fabricLFPathS>

<fabricSpPortP name="spine_profile">
 <fabricSFPortS name="spineIf" type="range">
  <fabricPortBlk name="spBlk" fromCard="5" fromPort="1" toCard="5" toPort="1" />
  <fabricRsSpPortPGrp tDn="uni/fabric/funcprof/spportgrp-SpTestPGrp" />
 </fabricSFPortS>
</fabricSpPortP>

<fabricSpineP name="SpNode" >
 <fabricRsSpPortP tDn="uni/fabric/spportp-spine_profile" />
 <fabricSpineS name="spsw" type="range">
  <fabricNodeBlk name="node102" to_="102" from_="102" />
 </fabricSpineS>
</fabricSpineP>
</fabricInst>
```

CHAPTER 16

Fabric Port Tracking

About Fabric Port Tracking

The port tracking feature manages the status of downlink ports on each leaf node based on the status of its fabric ports. Fabric ports are the links between leaf and spine nodes. Links between tier-1 and tier-2 leaf nodes in multi-tier topologies and links between remote leaf nodes (back-to-back links) are also considered to be fabric links.

When this feature is enabled and the number of operational fabric ports on a given leaf node is decreased to the configured threshold or lower, the downlink ports of the leaf node will be brought down so that external devices can switch over to other healthy leaf nodes. When the number of operational fabric ports comes back up to greater than the configured threshold, the downlink ports will be brought back up. At this time, there will be a wait time of the configured delay before the downlink ports are brought up. If the leaf node is part of a vPC peer and does not have any infra ISIS adjacencies--meaning that the node is unable to communicate with the other vPC peer leaf node--when port tracking is triggered such as when all fabric ports went down, the wait time for the vPC downlink ports to come up after the status restoration will be the longer time of either the vPC delay timer or the configured delay in port tracking. Non-vPC downlink ports always follow the delay timer configured in port tracking.

Note Port tracking checks the conditions to bring down the ports or bring up the ports every second on each leaf node.

FEX fabric ports--that is, the network interface (NIF) to connect the FEX and the FEX's parent leaf node--are not impacted by port tracking.

Configuring Fabric Port Tracking Using the GUI

This procedure uses the Cisco Application Policy Infrastructure Controller (APIC) GUI to configure the port track feature.

Procedure

Step 1 In the 3.2(1) release or earlier, navigate to the following location:

a) On the menu bar, choose **Fabric** > **Access Policies**.
b) In the Navigation pane, choose **Policies** > **Global**.
c) Click **Port Tracking**.

Step 2 In the 3.2(2) release, navigate to the following location:

a) On the menu bar, choose **System** > **System Settings**.
b) In the Navigation pane, choose **Port Tracking**.

Step 3 For the **Port tracking state** parameter, choose **on** to enable fabric port tracking.

Step 4 For the **Delay restore timer** parameter, specify the time in seconds.

This parameter determines how long the leaf node will wait before bringing up its downlink ports after the fabric port status and infra ISIS adjacencies are restored.

Step 5 Configure the **Number of active fabric ports that triggers port tracking** parameter.

When the number of operational fabric ports on a leaf node is decreased to the configured number or lower, the leaf node will bring down its downlink ports.

Configuring Fabric Port Tracking Using the REST API

This procedure configures the port track feature using the REST API.

Send a REST API POST similar to the following example:

POST: *apic_ip_address*/api/mo/uni.xml

```
<infraPortTrackPol
    dn="uni/infra/trackEqptFabP-default"     # Fixed DN. Do not change.
    adminSt="on"                             # 'on' to enable, 'off' to disable
    delay="120"                              # The delay timer (sec) to bring up the
                                             #   downlink ports
    minlinks="0"                             # The minimum required number of operational
                                             #   fabric ports
/>
```