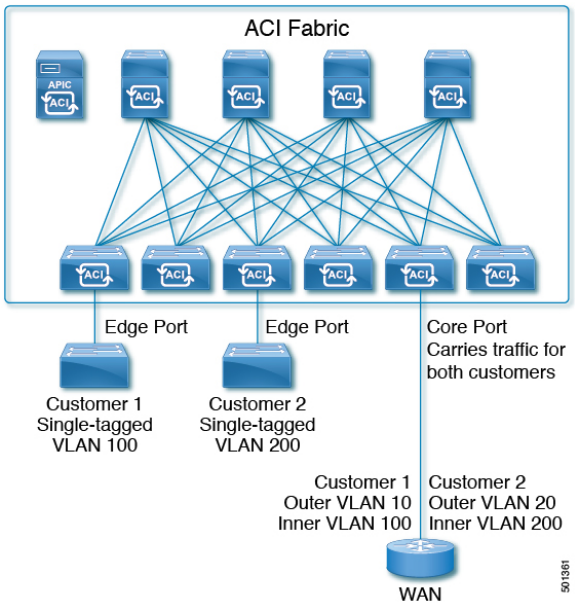
802.1Q Tunnels

This chapter contains the following sections:

About ACI 802.1Q Tunnels

Figure 1: 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 1: 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>
```