Managing Layer 4 to Layer 7 Services

About Layer 4 to Layer 7 Services

About Application-Centric Infrastructure Layer 4 to Layer 7 Services

Although VLAN and virtual routing and forwarding (VRF) stitching is supported by traditional service insertion models, the Application Policy Infrastructure Controller (APIC) can automate service insertion while acting as a central point of policy control. The APIC policies manage both the network fabric and services appliances. The APIC can configure the network automatically so that traffic flows through the services. The APIC can also automatically configure the service according to the application's requirements, which allows organizations to automate service insertion and eliminate the challenge of managing the complex techniques of traditional service insertion.

Before you begin, the following APIC objects must be configured:

- The tenant that will provide/consume the Layer 4 to Layer 7 services
- A Layer 3 outside network for the tenant
- At least one bridge domain
- An application profile
- A physical domain or a VMM domain

For a VMM domain, configure VMM domain credentials and configure a vCenter/vShield controller profile.

- A VLAN pool with an encapsulation block range
- At least one contract
- At least one EPG

You must perform the following tasks to deploy Layer 4 to Layer 7 services:

1. Import a **Device Package** .

   Only the provider administrator can import the device package.
2. Register the device and the logical interfaces.

   This task also registers concrete devices and concrete interfaces, and configures concrete device parameters.
3. Create a **Logical Device**.
4. Configure device parameters.
5. Optional. If you are configuring an ASA Firewall service, enable trunking on the device.
6. Configure a **Device Selection Policy**.
7. Configure a **Service Graph Template**.
   a. Select the default service graph template parameters from an application profile.
   b. Configure additional service graph template parameters, if needed.
8. Attach the service graph template to a contract.
9. Configure additional configuration parameters, if needed.

For more information about deploying Layer 4 to Layer 7 services, see the *Cisco APIC Layer 4 to Layer 7 Services Deployment Guide*.

Access for Managing Layer 4 to Layer 7 Services

Configure In-Band Connectivity to Devices Using Tenant's VRF Using the REST API

The following is an example of using REST APIs to configure in-band connectivity to devices using tenant's VRF:

1. Define the EPG that is to be used for management.

Note Ensure to open up the ports to the domain mappings using the appropriate selectors configuration.

In the following, the EPG "services" is used for the management of the Services Devices/VMs subnet that is used for Tenant vrf devicemanagement (3.3.3.0/24).

```
<polUni>
  <fvTenant name="tenant1">
    <fvCtx name="mgmt_ctx1"/>
    <vnsCtrlrMgmtPol ctxDn="uni/tn-tenant1/ctx-mgmt_ctx1">
      <vnsRsMgmtAddr tDn="uni/tn-tenant1/ap-services/epg-ifc/CtrlrAddrInst-ifc"/>
    </vnsCtrlrMgmtPol>
    <fvBD name="mgmt_ServicesMgmtBD">
      <fvRsCtx tnFvCtxName="mgmt_ctx1"/>
      <fvSubnet ip="3.3.3.3/24"/>
    </fvBD>
    <fvAp name="services">
      <fvAEPg name="ifc">
        <fvRsBd tnFvBDName="mgmt_ServicesMgmtBD"/>
        <vnsAddrInst name="ifc">
          <fvnsUcastAddrBlk from="3.3.3.100/24" to="3.3.3.200/24"/>
        </vnsAddrInst>
        <fvRsDomAtt tDn="uni/vmmp-VMware/dom-mininet"/>
      </fvAEPg>
    </fvAp>
  </fvTenant>
</polUni>
```

2. Associate the EPG to the LDevVip.

```
<polUni>
    <fvTenant name="tenant1">

        <vnsLDevVip name="ADCCluster1"
                    funcType="GoTo" devtype="VIRTUAL">
            <vnsRsMDevAtt tDn="uni/infra/mDev-Citrix-NetScaler-10.5"/>
            <vnsRsALDevToDomP tDn="uni/vmmp-VMware/dom-mininet"/>
            <vnsRsDevEpg tDn="uni/tn-tenant1/ap-services/epg-ifc"/>

            <vnsCMgmt name="devMgmt"
                      host="3.3.3.180"
                      port="80"/>

            <vnsCCred name="username"
                      value="nsroot"/>

            <vnsCCredSecret name="password"
                      value="nsroot"/>
        </vnsLDevVip>
    </fvTenant>
</polUni>
```

Configuring In-Band Connectivity to Devices Using Management Tenant VRF Using the REST API

The following is an example of using REST APIs to configure in-band connectivity to devices using management tenant VRF:

1. Create an EPG l4l7MgmtEpg in tenant management.

Note l4l7MgmtEpg is a part of bd access which is under inb context in tn-mgmt.

contract1 is the contract between the tn-mgmt l4l7MgmtEpg and tn-mgmt inb default EPG.

```
<polUni>
    <fvTenant dn="uni/tn-mgmt">
        <fvAp name="services">
            <fvAEPg name="l4l7MgmtEpg">
              <fvRsBd tnFvBDName="access" />
                <fvRsDomAtt tDn="uni/vmmp-VMware/dom-mininet" />
             <fvRsCons tnVzBrCPName='contract1'>
              </fvRsCons>
            </fvAEPg>
        </fvAp>
        <fvBD name="access">
            <fvSubnet ip="3.3.3.3/24" />
            <fvRsCtx tnFvCtxName="inb"/>
        </fvBD>
        <vzFilter name='all'>
            <vzEntry name='all' ></vzEntry>
        </vzFilter>
        <vzBrCP name="contract1" scope="tenant">
            <vzSubj name='subj1'>
                <vzInTerm>
                    <vzRsFiltAtt tnVzFilterName="all" />
                </vzInTerm>
                <vzOutTerm>
                    <vzRsFiltAtt tnVzFilterName="all" />
                </vzOutTerm>
            </vzSubj>
        </vzBrCP>
    </fvTenant>
</polUni>
```

2. Ensure that the Service Device/VM has the mgmt IP address in the subnet 3.3.3.0/24.

   This is the same subnet that tn-mgmt access BD has been configured with. (See configuration in earlier step.)

3. Add the following to the LDevVip:

Note This points to the EPG that was created in the earlier step

<vnsRsDevEpg tDn="uni/tn-mgmt/ap-services/epg-l4l7MgmtEpg"/>.

```
<polUni>
    <fvTenant name="mgmt">

        <vnsLDevVip name="ADCCluster1"
                    funcType="GoTo" devtype="VIRTUAL">
            <vnsRsMDevAtt tDn="uni/infra/mDev-Citrix-NetScaler-10.5"/>
            <vnsRsALDevToDomP tDn="uni/vmmp-VMware/dom-mininet"/>
            <vnsRsDevEpg tDn="uni/tn-mgmt/ap-services/epg-l4l7MgmtEpg"/>

            <vnsCMgmt name="devMgmt"
```

```
                        host="3.3.3.180"
                        port="80"/>

                <vnsCCred name="username"
                        value="nsroot"/>

                <vnsCCredSecret name="password"
                                value="nsroot"/>

            </vnsLDevVip>

        </fvTenant>
</polUni>
```

4. Add the route in service Device/VM to point to the IFC inband gateway.

   For example, on the route on netScaler, add route 3.0.0.0 255.255.255.0 3.3.3.3, where 3.0.0.0/24 is the IFC inband subnet and 3.3.3.3 is the SVI IP for l4l7MgmtEpg.

5. Verify the following:
   - The route table on IFC has an entry for ifc inband IP.
   - The IFC can ping the l4l7MgmtEpg gateway on the leaf.
   - The service node can ping the l4l7MgmtEpg SVI gateway and IFC inb SVI Ip.

Device Packages

About the Device Package

The Application Policy Infrastructure Controller (APIC) requires a device package to configure and monitor service devices. A device package manages a single class of service devices and provides the APIC with information about the device and its capabilities.

For more information about device packages, see the *Cisco APIC Layer 4 to Layer 7 Device Package Development Guide*.

Notes for Installing a Device Package with the REST APIs

- A device package can be installed using an HTTP or HTTPS POST.
- If HTTP is enabled on APIC, the URL for the POST is "http://10.10.10.10/ppi/node/mo/.xml".
- If HTTPS is enabled on APIC, the URL for the POST is "https://10.10.10.10/ppi/node/mo/.xml".
- The message must have a valid session cookie.
- The body of the POST should contain the device package being uploaded. Only one package is allowed in a POST.

Uploading a Device Package File Using the API

To install a service device, you must upload a device package file to APIC. The API command for this operation uses a special form of URI:

{ **http** | **https** } **://** *host* [*:port*] **/ppi /node /mo / .** { **json** | **xml** }

The URI path contains 'ppi' (package programming interface) instead of 'api', and the command is sent as a POST operation with the device package file as the body of the message. The device package file is a zip file.

This example shows an API operation that uploads a device package file:

```
POST https://192.0.20.123/ppi/node/mo/.json
```

For more information about installing L4-L7 service device packages, see *Cisco APIC Layer 4 to Layer 7 Services Deployment Guide*.

Installing a Device Package Using the REST API

You can install a device package using an HTTP or HTTPS POST.

Install the device package.

- If HTTP is enabled on the Application Policy Infrastructure Controller (APIC) , the URL for the POST is as follows:

  ```
  http://10.10.10.10/ppi/node/mo/.xml
  ```

- If HTTPS is enabled on the APIC, the URL for the POST is as follows:

  ```
  https://10.10.10.10/ppi/node/mo/.xml
  ```

The message must have a valid session cookie.

The body of the POST should contain the device package being uploaded. Only one package is allowed in a POST.

Using an Imported Device with the REST APIs

The following REST API uses an imported device:

```
<polUni>
  <fvTenant dn="uni/tn-tenant1" name="tenant1">
    <vnsLDevIf ldev="uni/tn-mgmt/lDevVip-ADCCluster1"/>
    <vnsLDevCtx ctrctNameOrLbl="any" graphNameOrLbl="any" nodeNameOrLbl="any">
      <vnsRsLDevCtxToLDev tDn="uni/tn-tenant1/lDevIf-[uni/tn-mgmt/lDevVip-ADCCluster1]"/>
      <vnsLIfCtx connNameOrLbl="inside">
        <vnsRsLIfCtxToLIf
tDn="uni/tn-tenant1/lDevIf-[uni/tn-mgmt/lDevVip-ADCCluster1]/lDevIfLIf-inside"/>
        <fvSubnet ip="10.10.10.10/24"/>
        <vnsRsLIfCtxToBD tDn="uni/tn-tenant1/BD-tenant1BD1"/>
      </vnsLIfCtx>
      <vnsLIfCtx connNameOrLbl="outside">
        <vnsRsLIfCtxToLIf
tDn="uni/tn-tenant1/lDevIf-[uni/tn-mgmt/lDevVip-ADCCluster1]/lDevIfLIf-outside"/>
        <fvSubnet ip="70.70.70.70/24"/>
```

```
            <vnsRsLIfCtxToBD tDn="uni/tn-tenant1/BD-tenant1BD4"/>
          </vnsLIfCtx>
        </vnsLDevCtx>
      </fvTenant>
    </polUni>
```

Trunking

About Trunking

You can enable trunking for a Layer 4 to Layer 7 virtual ASA device, which uses trunk port groups to aggregate the traffic of endpoint groups. Without trunking, a virtual service device can have only 1 VLAN per interface and up to 10 service graphs. With trunking enabled, the virtual service device can have an unlimited number of service graphs.

For more information about trunk port groups, see the *Cisco ACI Virtualization Guide*.

Trunking is supported only on a virtual ASA device. The ASA device package must be version 1.2.7.8 or later.

Enabling Trunking on a Layer 4 to Layer 7 Virtual ASA device Using the REST APIs

The following procedure provides an example of enabling trunking on a Layer 4 to Layer 7 virtual ASA device using the REST APIs.

Before you begin

- You must have configured a Layer 4 to Layer 7 virtual ASA device.

Enable trunking on the Layer 4 to Layer 7 device named `InsiemeCluster`:

```
<polUni>
    <fvTenant name="tenant1">
        <vnsLDevVip name="InsiemeCluster" devtype="VIRTUAL" trunking="yes">
            ...
            ...
        </vnsLDevVip>
    </fvTenant>
</polUni>
```

Device Selection Policies

About Device Selection Policies

A device can be selected based on a contract name, a graph name, or the function node name inside the graph. After you create a device, you can create a device context, which provides a selection criteria policy for a device.

A device selection policy (also known as a device context) specifies the policy for selecting a device for a service graph template. This allows an administrator to have multiple device and then be able to use them for different service graph templates. For example, an administrator can have a device that has high-performance ADC appliances and another device that has lower-performance ADC appliances. Using two different device selection policies, one for the high-performance ADC device and the other for the low-performance ADC device, the administrator can select the high-performance ADC device for the applications that require higher performance and select the low-performance ADC devices for the applications that require lower performance.

Creating a Device Selection Policy Using the REST API

The following REST API creates a device selection policy:

```
<polUni>
    <fvTenant dn="uni/tn-acme" name="acme">
        <vnsLDevCtx ctrctNameOrLbl="webCtrct" graphNameOrLbl="G1" nodeNameOrLbl="Node1">
            <vnsRsLDevCtxToLDev tDn="uni/tn-acme/lDevVip-ADCCluster1"/>

             <!-- The connector name C4, C5, etc.. should match the
                  Function connector name used in the service graph template -->

             <vnsLIfCtx connNameOrLbl="C4">
                <vnsRsLIfCtxToLIf tDn="uni/tn-acme/lDevVip-ADCCluster1/LIf-ext"/>
            </vnsLIfCtx>
            <vnsLIfCtx connNameOrLbl="C5">
                <vnsRsLIfCtxToLIf tDn="uni/tn-acme/lDevVip-ADCCluster1/LIf-int"/>
            </vnsLIfCtx>
        </vnsLDevCtx>
    </fvTenant>
</polUni>
```

Adding a Logical Interface in a Device Using the REST APIs

The following REST API adds a logical interface in a device:

```
<polUni>
    <fvTenant dn="uni/tn-acme" name="acme">
        <vnsLDevVip name="ADCCluster1">

            <!-- The LIF name defined here (such as e.g., ext, or int) should match the
                 vnsRsLIfCtxToLIf 'tDn' defined in LifCtx  -->

            <vnsLIf name="ext">

                <vnsRsMetaIf tDn="uni/infra/mDev-Acme-ADC-1.0/mIfLbl-outside"/>
                <vnsRsCIfAtt tDn="uni/tn-acme/lDevVip-ADCCluster1/cDev-ADC1/cIf-ext"/>
            </vnsLIf>
            <vnsLIf name="int">
```

```
                <vnsRsMetaIf tDn="uni/infra/mDev-Acme-ADC-1.0/mIfLbl-inside"/>
                <vnsRsCIfAtt tDn="uni/tn-acme/lDevVip-ADCCluster1/cDev-ADC1/cIf-int"/>
            </vnsLIf>
        </vnsLDevVip>
    </fvTenant>
</polUni>
```

Policy Based Redirect and Service Nodes Tracking

Policy-Based Redirect and Tracking Service Nodes

Beginning with the Cisco Application Policy Infrastructure Controller (APIC) 2.2(3) and 3.1(1) releases (but, excluding the 3.0 releases), the policy-based redirect feature (PBR) supports the ability to track service nodes. Tracking enables you to prevent redirection of traffic to a service node that is down. If a service node (PBR destination) is down, the PBR hashing can begin selecting an available PBR destination in a policy. This feature requires Cisco Nexus 9300-EX, -FX, or later platform leaf switches.

Service nodes can support dual IP address stacking. Therefore, this feature has the capability to track both IPv4 and IPv6 addresses at the same time. When both IPv4 and IPv6 addresses are "up," the PBR destination is marked as "up."

Switches internally use the Cisco IP SLA monitoring feature to support PBR tracking. The tracking feature marks a redirect destination node as "down" if the service node is not reachable. The tracking feature marks a redirect destination as node "up" if the service node resumes connectivity. When a service node is marked as "down," it will not be used to send or hash the traffic. Instead, the traffic will be sent or hashed to a different service node in the cluster of redirection destination nodes.

To avoid black holing of the traffic in one direction, you can associate a service node's ingress and egress redirect destination nodes with a redirection health policy. Doing so ensures that if either an ingress or egress redirection destination node is down, the other redirection destination node will also be marked as "down." Hence, both ingress and egress traffic gets hashed to a different service node in the cluster of the redirect destination nodes.

You can use the following protocols for tracking:

- ICMP (for Layer 3 PBR)
- TCP (for Layer 3 PBR)
- L2ping (for Layer 1/2 PBR)

Policy-Based Redirect and Threshold Settings for Tracking Service Nodes

The following threshold settings are available when configuring a policy-based redirect (PBR) policy for tracking service nodes:

- Threshold enabled or disabled: When the threshold is enabled, you an specify the minimum and maximum threshold percentages. Threshold enabled is required when you want to disable the redirect destination group completely and prevent any redirection. When there is no redirection, the traffic is directly sent between the consumer and the provider.
- Minimum threshold: The minimum threshold percentage specified. If the traffic goes below the minimum percentage, the packet is permitted instead of being redirected. The default value is 0.

- Maximum threshold: The maximum threshold percentage specified. Once the minimum threshold is reached, to get back to operational state, the maximum percentage must first be reached. The default value is 0.

Let us assume as an example that there are three redirect destinations in a policy. The minimum threshold is specified at 70% and the maximum threshold is specified at 80%. If one of the three redirect destination policies goes down, the percentage of availability goes down by one of three (or 33%), which is less than the minimum threshold. As a result, the minimum threshold percentage of the redirect destination group is brought down and traffic begins to get permitted instead of being redirected. Continuing with the same example, if the maximum threshold is 80%, to bring the redirect policy destination group back to the operational state, a percentage greater than the maximum threshold percentage must be reached.

Guidelines and Limitations for Policy-Based Redirect With Tracking Service Nodes

Follow these guidelines and limitations when using policy-based redirect (PBR) tracking with service nodes:

- Beginning in release 4.0(1), remote leaf switch configurations support PBR tracking, but only if system-level global GIPo is enabled. See *Configuring Global GIPo for Remote Leaf Using the GUI*.
- Beginning in release 4.0(1), remote leaf switch configurations support PBR resilient hashing.
- A Cisco ACI Multi-Pod fabric setup is supported.
- A Cisco ACI Multi-Site setup is not supported.
- An L3Out is supported for the consumer and provider EPGs.
- TCP or ICMP protocol types are used to track the redirect destination nodes.
- PBR supports up to 100 trackable IP addresses in leaf switches and 200 trackable IP addresses in the Cisco Application Centric Infrastructure (ACI) fabric.
- PBR supports up to 1,000 service graph instances per Cisco ACI fabric.
- PBR supports up to 100 service graph instances per device.
- You can configure up to 40 service nodes per PBR policy.
- You can configure up to 3 service nodes per service chain.
- Shared services are supported with PBR tracking.
- The following threshold down actions are supported:
  - deny action
  - permit action
- If multiple PBR policies have the same PBR destination IP address in the same VRF instance, the policies must use the same IP SLA policy and health group for the PBR destination.

Configuring PBR to Support Tracking Service Nodes Using the REST API

Configure PBR to support tracking service nodes.

Example:

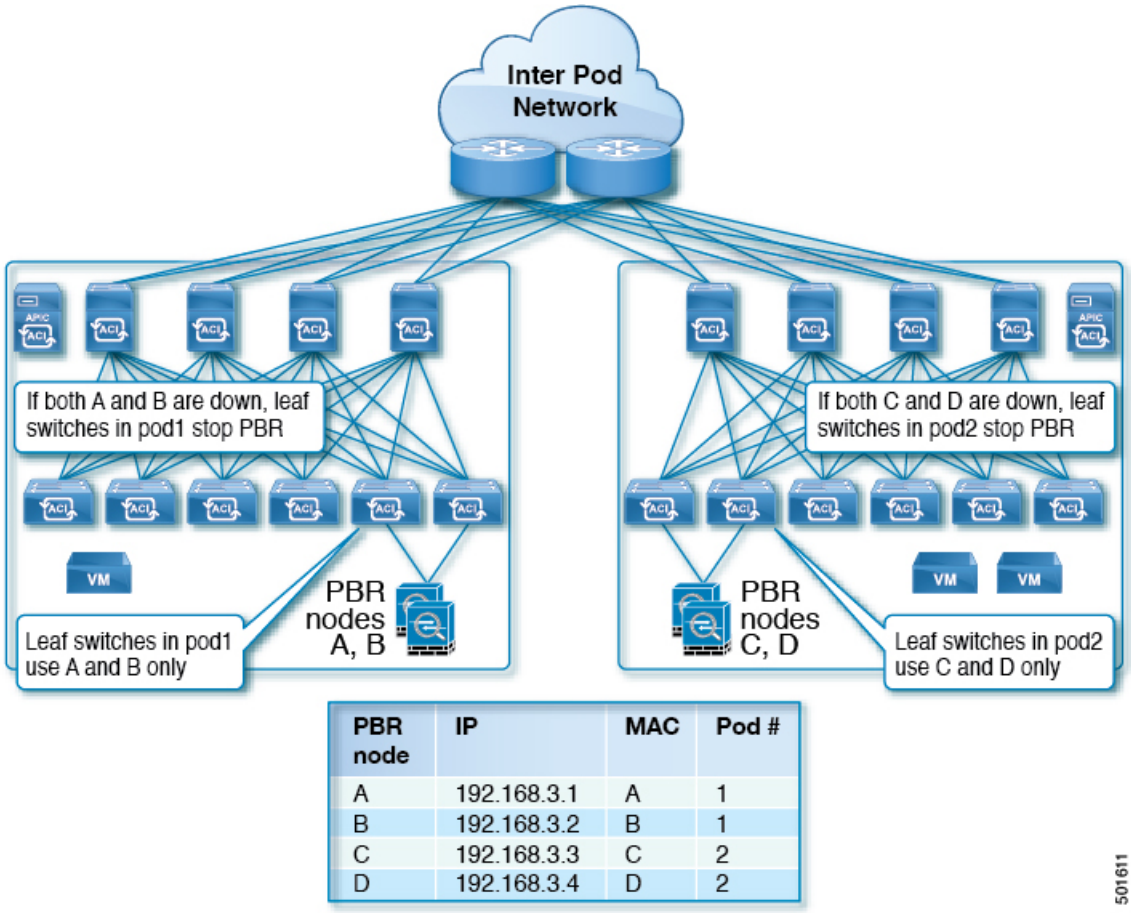
```
<polUni>
    <fvTenant name="t1" >
    <fvIPSLAMonitoringPol name="tcp_Freq60_Pol1" slaType="tcp" slaFrequency="60" slaPort="2222" />
    <vnsSvcCont>
        <vnsRedirectHealthGroup name="fwService1"/>
          <vnsSvcRedirectPol name="fwExt" hashingAlgorithm="sip" thresholdEnable="yes"
            minThresholdPercent="20" maxThresholdPercent="80">
            <vnsRedirectDest ip="40.40.40.100" mac="00:00:00:00:00:01">
               <vnsRsRedirectHealthGroup tDn="uni/tn-t1/svcCont/redirectHealthGroup-fwService1"/>
            </vnsRedirectDest>
            <vnsRsIPSLAMonitoringPol tDn="uni/tn-t1/ipslaMonitoringPol-tcp_Freq60_Pol1"/>
          </vnsSvcRedirectPol>
         <vnsSvcRedirectPol name="fwInt" hashingAlgorithm="sip" thresholdEnable="yes"
           minThresholdPercent="20" maxThresholdPercent="80">
             <vnsRedirectDest ip="30.30.30.100" mac="00:00:00:00:00:02">
                <vnsRsRedirectHealthGroup tDn="uni/tn-t1/svcCont/redirectHealthGroup-fwService1"/>
             </vnsRedirectDest>
             <vnsRsIPSLAMonitoringPol tDn="uni/tn-t1/ipslaMonitoringPol-tcp_Freq60_Pol1"/>
          </vnsSvcRedirectPol>
        </vnsSvcCont>
    </fvTenant>
</polUni>
```

About Location-Aware Policy Based Redirect

Location-Aware Policy Based Redirect (PBR) is now supported. This feature is useful in a multipod configuration scenario. Now there is pod-awareness support, and you can specify the preferred local PBR node. When you enable location-aware redirection, and Pod IDs are specified, all the redirect destinations in the Layer 4-Layer 7 PBR policy will have pod awareness. The redirect destination is programmed only in the leaf switches located in a specific pod.

The following image displays an example with two pods. PBR nodes A and B are in Pod 1 and PBR nodes C and D are in Pod 2. When you enable the location-aware PBR configuration, the leaf switches in Pod 1 prefer to use PBR nodes A and B, and the leaf switches in Pod 2 use PBR nodes in C and D. If PBR nodes A and B in Pod 1 are down, then the leaf switches in Pod 1 will start to use PBR nodes C and D. Similarly, if PBR nodes C and D in Pod 2 are down, the leaf switches in Pod 2 will start to use PBR nodes A and B.

Figure 1: An Example of Location Aware PBR Configuration with Two Pods



PBR node	IP	MAC	Pod #
A	192.168.3.1	A	1
B	192.168.3.2	B	1
C	192.168.3.3	C	2
D	192.168.3.4	D	2

Guidelines for Location-Aware PBR

Follow these guidelines when using location-aware PBR:

- The Cisco Nexus 9300 (except Cisco Nexus 9300–EX and 9300–FX) platform switches do not support the location-aware PBR feature.
- Use location-aware PBR for north-south firewall integration with GOLF host advertisement.

  Use location-aware PBR for a contract that is enforced on the same leaf nodes for incoming and returning traffic, such as an intra-VRF contract for external-EPG-to-EPG and an inter-VRF contract for EPG-to-EPG traffic. Otherwise, there can be a loss of traffic symmetry.
- If multiple PBR policies have the same PBR destination IP address in the same VRF, then all of the policies must either have Pod ID aware redirection enabled or Pod ID aware redirection disabled. The same (VRF, IP address) pair cannot be used in Pod ID aware redirection enabled and Pod ID aware redirection disabled policies at the same time. For example, the following configuration is not supported:
  - PBR-policy1 has PBR destination 192.168.1.1 in VRF A, Pod ID aware redirection enabled, and 192.168.1.1 is set to POD 1.
  - PBR-policy2 has PBR destination 192.168.1.1 in VRF A and Pod ID aware redirection disabled.

Configuring Location-Aware PBR Using the REST API

You must configure two items to enable location-aware PBR and to program redirect destinations in the leaf switches located in the specific pods. The attributes that are configured to enable location-aware PBR in the following example are: programLocalPodOnly and podId.

Configure location-aware PBR.

Example:

```
<polUni>
    <fvTenant name="coke" >
    <fvIPSLAMonitoringPol name="icmp_Freq60_Pol1" slaType="icmp" slaFrequency="60"/>
    <vnsSvcCont>
        <vnsRedirectHealthGroup name="fwService1"/>
          <vnsSvcRedirectPol name="fwExt" hashingAlgorithm="sip" thresholdEnable="yes"
minThresholdPercent="20" maxThresholdPercent="80" programLocalPodOnly="yes">
            <vnsRedirectDest ip="40.40.40.100" mac="00:00:00:00:00:01" podId="2">
               <vnsRsRedirectHealthGroup tDn="uni/tn-coke/svcCont/redirectHealthGroup-fwService1"/>
            </vnsRedirectDest>
            <vnsRsIPSLAMonitoringPol tDn="uni/tn-coke/ipslaMonitoringPol-icmp_Freq60_Pol1"/>
          </vnsSvcRedirectPol>
          <vnsSvcRedirectPol name="fwInt" hashingAlgorithm="dip" thresholdEnable="yes"
minThresholdPercent="20" maxThresholdPercent="80">
             <vnsRedirectDest ip="30.30.30.100" mac="00:00:00:00:00:02">
                <vnsRsRedirectHealthGroup tDn="uni/tn-coke/svcCont/redirectHealthGroup-fwService1"/>

             </vnsRedirectDest>
             <vnsRsIPSLAMonitoringPol tDn="uni/tn-coke/ipslaMonitoringPol-icmp_Freq60_Pol1"/>
          </vnsSvcRedirectPol>
        </vnsSvcCont>
    </fvTenant>
</polUni>
```

Service Graph Templates

About Service Graph Templates

The Cisco Application Centric Infrastructure (ACI) allows you to define a sequence of meta-devices, such a firewall of a certain type followed by a load balancer of a certain make and version. This is called an service graph template, also known as an abstract graph. When a service graph template is referenced by a contract, the service graph template is instantiated by mapping it to concrete devices, such as the firewall and load balancers that are present in the fabric. The mapping happens with the concept of a *context*. The *device context* is the mapping configuration that allows Cisco ACI to identify which firewalls and which load balancers can be mapped to the service graph template. Another key concept is the *logical device*, which represents the cluster of concrete devices. The rendering of the service graph template is based on identifying the suitable logical devices that can be inserted in the path that is defined by a contract.

Cisco ACI treats services as an integral part of an application. Any services that are required are treated as a service graph that is instantiated on the Cisco ACI fabric from the Cisco Application Policy Infrastructure Controller (APIC). Users define the service for the application, while service graph templates identify the set

of network or service functions that are needed by the application. Once the graph is configured in the Cisco APIC, the Cisco APIC automatically configures the services according to the service function requirements that are specified in the service graph template. The Cisco APIC also automatically configures the network according to the needs of the service function that is specified in the service graph template, which does not require any change in the service device.

Configuring a Service Graph Template Using the REST APIs

You can configure a service graph template using the following REST API:

```
<polUni>
  <fvTenant name="acme">
    <vnsAbsGraph name="G1">
      <vnsAbsTermNodeCon name="Input1">
        <vnsAbsTermConn name="C1">
            </vnsAbsTermConn>
      </vnsAbsTermNodeCon>
      <vnsAbsNode name="Node" funcType="GoTo">
        <vnsRsDefaultScopeToTerm
          tDn="uni/tn-acme/AbsGraph-G1/AbsTermNodeProv-Output1/outtmnl"/>
        <vnsAbsFuncConn name="inside">
          <vnsRsMConnAtt
            tDn="uni/infra/mDev-Insieme-Generic-1.0/mFunc-SubnetFunc/mConn-external"/>
        </vnsAbsFuncConn>
        <vnsAbsFuncConn name="outside">
          <vnsRsMConnAtt
            tDn="uni/infra/mDev-Insieme-Generic-1.0/mFunc-SubnetFunc/mConn-internal"/>
        </vnsAbsFuncConn>
        <vnsAbsDevCfg>
          <vnsAbsFolder key="oneFolder" name="f1">
            <vnsAbsParam key="oneParam" name="p1" value="v1"/>
          </vnsAbsFolder>
        </vnsAbsDevCfg>
        <vnsAbsFuncCfg>
          <vnsAbsFolder key="folder" name="folder1" devCtxLbl="C1">
            <vnsAbsParam key="param" name="param" value="value"/>
          </vnsAbsFolder>
          <vnsAbsFolder key="folder" name="folder2" devCtxLbl="C2">
            <vnsAbsParam key="param" name="param" value="value"/>
          </vnsAbsFolder>
        </vnsAbsFuncCfg>
        <vnsRsNodeToMFunc tDn="uni/infra/mDev-Insieme-Generic-1.0/mFunc-SubnetFunc"/>
      </vnsAbsNode>
      <vnsAbsTermNodeProv name="Output1">
        <vnsAbsTermConn name="C6">
            </vnsAbsTermConn>
      </vnsAbsTermNodeProv>
      <vnsAbsConnection name="CON1">
        <vnsRsAbsConnectionConns
          tDn="uni/tn-acme/AbsGraph-G1/AbsTermNodeCon-Input1/AbsTConn"/>
       <vnsRsAbsConnectionConns tDn="uni/tn-acme/AbsGraph-G1/AbsNode-Node/AbsFConn-inside"/>

      </vnsAbsConnection>
      <vnsAbsConnection name="CON3">
       <vnsRsAbsConnectionConns tDn="uni/tn-acme/AbsGraph-G1/AbsNode-Node/AbsFConn-outside"/>

        <vnsRsAbsConnectionConns
          tDn="uni/tn-acme/AbsGraph-G1/AbsTermNodeProv-Output1/AbsTConn"/>
      </vnsAbsConnection>
    </vnsAbsGraph>
  </fvTenant>
</polUni>
```

Creating a Security Policy Using the REST APIs

You can create a security policy using the following REST API:

```
<polUni>
    <fvTenant dn="uni/tn-acme" name="acme">
        <vzFilter name="HttpIn">
            <vzEntry name="e1" prot="6" dToPort="80"/>
        </vzFilter>
        <vzBrCP name="webCtrct">
            <vzSubj name="http">
                <vzRsSubjFiltAtt tnVzFilterName="HttpIn"/>
            </vzSubj>
        </vzBrCP>
    </fvTenant>
</polUni>
```

Layer 4 to Layer 7 Parameters

About Modifying the Configuration Parameters of a Deployed Service Graph

When you first deploy a service graph, the configuration parameters or functions for the service graph must be defined before you can successfully deploy the service graph. These configuration parameters or functions include device network configurations, such as IP addresses, route prefix, and next hop information, as well as the services configuration, such as the IP access list for a firewall or server load balancing configuration for a load balancer.

You must modify the service graph function as part of the day-to-day operation of the Application Policy Infrastructure Controller (APIC). You can modify a service graph's configuration parameters and functions by using the GUI or CLI of the APIC. Modifying functions of a service device through the APIC does not require changes on a service device.

Example XML POST for an Application EPG With Configuration Parameters

The following XML example shows configuration parameters inside of the device package:

```
<fvAEPg dn="uni/tn-acme/ap-myApp/epg-app"  name="app">

   <vnsFolderInst ctrctNameOrLbl="any" graphNameOrLbl="any" nodeNameOrLbl="any" key="Monitor"

      name="monitor1">
       <vnsRsFolderInstToMFolder tDn="uni/infra/mDev-Acme-ADC-1.0/mDevCfg/mFolder-Monitor"/>

        <vnsParamInst name="weight" key="weight" value="10"/>
    </vnsFolderInst>

   <vnsFolderInst ctrctNameOrLbl="any" graphNameOrLbl="any" nodeNameOrLbl="any" key="Service"

      name="Service1">
        <vnsParamInst name="servicename" key="servicename" value="crpvgrtst02-8010"/>
        <vnsParamInst name="servicetype" key="servicetype" value="TCP"/>
        <vnsParamInst name="servername" key="servername" value="s192.168.100.100"/>
       <vnsParamInst name="serveripaddress" key="serveripaddress" value="192.168.100.100"/>

        <vnsParamInst name="serviceport" key="serviceport" value="8080"/>
        <vnsParamInst name="svrtimeout" key="svrtimeout" value="9000" />
```

```
     <vnsParamInst name="clttimeout" key="clttimeout" value="9000" />
     <vnsParamInst name="usip" key="usip" value="NO" />
     <vnsParamInst name="useproxyport" key="useproxyport" value="" />
     <vnsParamInst name="cip" key="cip" value="ENABLED" />
     <vnsParamInst name="cka" key="cka" value="NO" />
     <vnsParamInst name="sp" key="sp" value="OFF" />
     <vnsParamInst name="cmp" key="cmp" value="NO" />
     <vnsParamInst name="maxclient" key="maxclient" value="0" />
     <vnsParamInst name="maxreq" key="maxreq" value="0" />
     <vnsParamInst name="tcpb" key="tcpb" value="NO" />
     <vnsCfgRelInst name="MonitorConfig" key="MonitorConfig" targetName="monitor1"/>
 </vnsFolderInst>

<vnsFolderInst ctrctNameOrLbl="any" graphNameOrLbl="G2" nodeNameOrLbl="any" key="Network"

   name="Network">
    <vnsFolderInst ctrctNameOrLbl="any" graphNameOrLbl="G2" nodeNameOrLbl="any" key="vip"

       name="vip">
         <vnsParamInst name="vipaddress1" key="vipaddress" value="10.10.10.200"/>
     </vnsFolderInst>
     <vnsFolderInst ctrctNameOrLbl="any" graphNameOrLbl="G2" nodeNameOrLbl="any"
       devCtxLbl="C1" key="snip" name="snip1">
         <vnsParamInst name="snipaddress" key="snipaddress" value="192.168.1.200"/>
     </vnsFolderInst>
     <vnsFolderInst ctrctNameOrLbl="any" graphNameOrLbl="G2" nodeNameOrLbl="any"
       devCtxLbl="C2" key="snip" name="snip2">
<vnsFolderInst ctrctNameOrLbl="any" graphNameOrLbl="G1" nodeNameOrLbl="any" key="Network"

   name="Network">
    <vnsFolderInst ctrctNameOrLbl="any" graphNameOrLbl="G1" nodeNameOrLbl="any" key="vip"

       name="vip">
         <vnsParamInst name="vipaddress1" key="vipaddress" value="10.10.10.100"/>
     </vnsFolderInst>
     <vnsFolderInst ctrctNameOrLbl="any" graphNameOrLbl="G1" nodeNameOrLbl="any"
       devCtxLbl="C1" key="snip" name="snip1">
         <vnsParamInst name="snipaddress" key="snipaddress" value="192.168.1.100"/>
     </vnsFolderInst>
     <vnsFolderInst ctrctNameOrLbl="any" graphNameOrLbl="G1" nodeNameOrLbl="any"
       devCtxLbl="C2" key="snip" name="snip2">
         <vnsParamInst name="snipaddress" key="snipaddress" value="192.168.1.101"/>
     </vnsFolderInst>
     <vnsFolderInst ctrctNameOrLbl="any" graphNameOrLbl="G1" nodeNameOrLbl="any"
       devCtxLbl="C3" key="snip" name="snip3">
         <vnsParamInst name="snipaddress" key="snipaddress" value="192.168.1.102"/>
     </vnsFolderInst>
 </vnsFolderInst>

 <!-- SLB Configuration -->
<vnsFolderInst ctrctNameOrLbl="any" graphNameOrLbl="any" nodeNameOrLbl="any" key="VServer"

   name="VServer">
     <!-- Virtual Server Configuration -->
     <vnsParamInst name="port" key="port" value="8010"/>
     <vnsParamInst name="vip" key="vip" value="10.10.10.100"/>
     <vnsParamInst name="vservername" key="vservername" value="crpvgrtst02-vip-8010"/>
     <vnsParamInst name="servicename" key="servicename" value="crpvgrtst02-8010"/>
     <vnsFolderInst ctrctNameOrLbl="any" graphNameOrLbl="any" nodeNameOrLbl="any"
       key="VServerGlobalConfig" name="VServerGlobalConfig">
        <vnsCfgRelInst name="ServiceConfig" key="ServiceConfig" targetName="Service1"/>

         <vnsCfgRelInst name="VipConfig" key="VipConfig" targetName="Network/vip"/>
     </vnsFolderInst>
```

```
    </vnsFolderInst>
</fvAEPg>
```

Example XML of Configuration Parameters Inside the Device Package

The following XML example shows configuration parameters inside of the device package:

```
<vnsMFolder key="VServer" scopedBy="epg">
    <vnsRsConnector tDn="uni/infra/mDev-Acme-ADC-1.0/mFunc-SLB/mConn-external"/>
    <vnsMParam key="vservername" description="Name of VServer" mandatory="true"/>
    <vnsMParam key="vip" description="Virtual IP"/>
    <vnsMParam key="subnet" description="Subnet IP"/>
    <vnsMParam key="port" description="Port for Virtual server"/>
    <vnsMParam key="persistencetype" description="persistencetype"/>
    <vnsMParam key="servicename" description="Service bound to this vServer"/>
    <vnsMParam key="servicetype" description="Service bound to this vServer"/>
    <vnsMParam key="clttimeout" description="Client timeout"/>
    <vnsMFolder key="VServerGlobalConfig"
      description="This references the global configuration">
        <vnsMRel key="ServiceConfig">
            <vnsRsTarget tDn="uni/infra/mDev-Acme-ADC-1.0/mDevCfg/mFolder-Service"/>
        </vnsMRel>
        <vnsMRel key="ServerConfig">
            <vnsRsTarget tDn="uni/infra/mDev-Acme-ADC-1.0/mDevCfg/mFolder-Server"/>
        </vnsMRel>
        <vnsMRel key="VipConfig">
            <vnsRsTarget
              tDn="uni/infra/mDev-Acme-ADC-1.0/mDevCfg/mFolder-Network/mFolder-vip"/>
            <vnsRsConnector tDn="uni/infra/mDev-Acme-ADC-1.0/mFunc-SLB/mConn-external"/>
        </vnsMRel>
    </vnsMFolder>
</vnsMFolder>
```

Example XML POST for an Abstract Function Node With Configuration Parameters

The following XML POST example shows an abstract function node with configuration parameters:

```
<vnsAbsNode name = "SLB" funcType="GoTo" >
   <vnsRsDefaultScopeToTerm tDn="uni/tn-tenant1/AbsGraph-G3/AbsTermNode-Output1/outtmnl"/>

    <vnsAbsFuncConn name = "C4" direction = "input">
        <vnsRsMConnAtt tDn="uni/infra/mDev-Acme-ADC-1.0/mFunc-SLB/mConn-external" />
    </vnsAbsFuncConn>
    <vnsAbsFuncConn name = "C5" direction = "output">
        <vnsRsMConnAtt tDn="uni/infra/mDev-Acme-ADC-1.0/mFunc-SLB/mConn-internal" />
    </vnsAbsFuncConn>

    <vnsAbsDevCfg>
        <vnsAbsFolder key="Network" name="Network" scopedBy="epg">
            <!-- Following scopes this folder to input terminal or Src Epg -->
          <vnsRsScopeToTerm tDn="uni/tn-tenant1/AbsGraph-G3/AbsTermNode-Output1/outtmnl"/>


            <!-- VIP address -->
            <vnsAbsFolder key="vip" name="vip" scopedBy="epg">
                <vnsAbsParam name="vipaddress" key="vipaddress" value=""/>
            </vnsAbsFolder>

            <!-- SNIP address -->
```

```
            <vnsAbsFolder key="snip" name="snip" scopedBy="epg">
                <vnsAbsParam name="snipaddress" key="snipaddress" value=""/>
            </vnsAbsFolder>

        </vnsAbsFolder>

        <vnsAbsFolder key="Service" name="Service" scopedBy="epg" cardinality="n">
          <vnsRsScopeToTerm tDn="uni/tn-tenant1/AbsGraph-G3/AbsTermNode-Output1/outtmnl"/>

            <vnsAbsParam name="servicename" key="servicename" value=""/>
            <vnsAbsParam name="servername" key="servername" value=""/>
            <vnsAbsParam name="serveripaddress" key="serveripaddress" value=""/>
        </vnsAbsFolder>
    </vnsAbsDevCfg>

    <vnsAbsFuncCfg>
        <vnsAbsFolder key="VServer" name="VServer" scopedBy="epg">
          <vnsRsScopeToTerm tDn="uni/tn-tenant1/AbsGraph-G3/AbsTermNode-Output1/outtmnl"/>

            <!-- Virtual Server Configuration -->
            <vnsAbsParam name="vip" key="vip" value=""/>
            <vnsAbsParam name="vservername" key="vservername" value=""/>
            <vnsAbsParam name="servicename" key="servicename"/>
            <vnsRsCfgToConn tDn="uni/tn-tenant1/AbsGraph-G3/AbsNode-Node2/AbsFConn-C4" />
        </vnsAbsFolder>
    </vnsAbsFuncCfg>
    <vnsRsNodeToMFunc tDn="uni/infra/mDev-Acme-ADC-1.0/mFunc-SLB"/>
</vnsAbsNode>
```

Example XML POST for an Abstract Function Profile With Configuration Parameters

The following XML POST example shows an abstract function profile with configuration parameters:

```
<vnsAbsFuncProfContr name = "NP">
    <vnsAbsFuncProfGrp name = "Grp1">
        <vnsAbsFuncProf name = "P1">
            <vnsRsProfToMFunc tDn="uni/infra/mDev-Acme-ADC-1.0/mFunc-SLB"/>
            <vnsAbsDevCfg name="D1">
                <vnsAbsFolder key="Service" name="Service-Default" cardinality="n">
                    <vnsAbsParam name="servicetype" key="servicetype" value="TCP"/>
                    <vnsAbsParam name="serviceport" key="serviceport" value="80"/>
                    <vnsAbsParam name="maxclient" key="maxclient" value="1000"/>
                    <vnsAbsParam name="maxreq" key="maxreq" value="100"/>
                    <vnsAbsParam name="cip" key="cip" value="enable"/>
                    <vnsAbsParam name="usip" key="usip" value="enable"/>
                    <vnsAbsParam name="sp" key="sp" value=""/>
                    <vnsAbsParam name="svrtimeout" key="svrtimeout" value="60"/>
                    <vnsAbsParam name="clttimeout" key="clttimeout" value="60"/>
                    <vnsAbsParam name="cka" key="cka" value="NO"/>
                    <vnsAbsParam name="tcpb" key="tcpb" value="NO"/>
                    <vnsAbsParam name="cmp" key="cmp" value="NO"/>
                </vnsAbsFolder>
            </vnsAbsDevCfg>
            <vnsAbsFuncCfg name="SLB">
                <vnsAbsFolder key="VServer" name="VServer-Default">
                    <vnsAbsParam name="port" key="port" value="80"/>
                    <vnsAbsParam name="persistencetype" key="persistencetype"
                      value="cookie"/>
                    <vnsAbsParam name="clttimeout" key="clttimeout" value="100"/>
                    <vnsAbsParam name="servicetype" key="servicetype" value="TCP"/>
                    <vnsAbsParam name="servicename" key="servicename"/>
```

```
                </vnsAbsFolder>
            </vnsAbsFuncCfg>
        </vnsAbsFuncProf>
    </vnsAbsFuncProfGrp>
</vnsAbsFuncProfContr>
```

Copy Services

About Copy Services

Unlike Switched Port Analyzer (SPAN), which duplicates all traffic, the Cisco Application Centric Infrastructure (ACI) copy services feature enables selectively copying portions of the traffic between endpoint groups, according to the specifications of the contract. Broadcast, unknown unicast and multicast (BUM), and control plan traffic not covered by the contract are not copied. In contrast, SPAN copies everything out of endpoint groups, access ports, or uplink ports. Unlike SPAN, copy services do not add headers to the copied traffic. Copy service traffic is managed internally in the switch to minimize impact on normal traffic forwarding.

For more information about deploying Layer 4 to Layer 7 services, see the *Cisco APIC Layer 4 to Layer 7 Services Deployment Guide*.

Configuring Copy Services Using the REST API

A copy device is used as part of the copy services feature to create a copy node. A copy node specifies at which point of the data flow between endpoint groups to copy traffic.

This procedure provides examples of using the REST API to configure copy services.

Note When you configure a copy device, the context aware parameter is not used. The context aware parameter has a default value of `single context`, which can be ignored.

Before you begin

You must have configured a tenant.

Step 1 Create a copy device.

Example:

```
<vnsLDevVip contextAware="single-Context" devtype="PHYSICAL" funcType="None" isCopy="yes"
  managed="no" mode="legacy-Mode" name="copy0" packageModel="" svcType="COPY" trunking="no">
    <vnsRsALDevToPhysDomP tDn="uni/phys-phys_scale_copy"/>
    <vnsCDev devCtxLbl="" name="copy_Dyn_Device_0" vcenterName="" vmName="">
        <vnsCIf name="int1" vnicName="">
            <vnsRsCIfPathAtt tDn="topology/pod-1/paths-104/pathep-[eth1/15]"/>
        </vnsCIf>
        <vnsCIf name="int2" vnicName="">
            <vnsRsCIfPathAtt tDn="topology/pod-1/paths-105/pathep-[eth1/15]"/>
        </vnsCIf>
    </vnsCDev>
    <vnsLIf encap="vlan-3540" name="TAP">
        <vnsRsCIfAttN tDn="uni/tn-t22/lDevVip-copy0/cDev-copy_Dyn_Device_0/cIf-[int2]"/>
```

```
        <vnsRsCIfAttN tDn="uni/tn-t22/lDevVip-copy0/cDev-copy_Dyn_Device_0/cIf-[int1]"/>
    </vnsLIf>
</vnsLDevVip>
```

Step 2 Create a logical device context (also known as a device selection policy).

Example:

```
<vnsLDevCtx ctrctNameOrLbl="c0" descr="" graphNameOrLbl="g0" name="" nodeNameOrLbl="CP1">
    <vnsRsLDevCtxToLDev tDn="uni/tn-t22/lDevVip-copy0"/>
    <vnsLIfCtx connNameOrLbl="copy" descr="" name="">
        <vnsRsLIfCtxToLIf tDn="uni/tn-t22/lDevVip-copy0/lIf-TAP"/>
    </vnsLIfCtx>
</vnsLDevCtx>
```

Step 3 Create and apply the copy graph template.

Example:

```
<vnsAbsGraph descr="" name="g0" ownerKey="" ownerTag="" uiTemplateType="UNSPECIFIED">
    <vnsAbsTermNodeCon descr="" name="T1" ownerKey="" ownerTag="">
        <vnsAbsTermConn attNotify="no" descr="" name="1" ownerKey="" ownerTag=""/>
        <vnsInTerm descr="" name=""/>
        <vnsOutTerm descr="" name=""/>
    </vnsAbsTermNodeCon>
    <vnsAbsTermNodeProv descr="" name="T2" ownerKey="" ownerTag="">
        <vnsAbsTermConn attNotify="no" descr="" name="1" ownerKey="" ownerTag=""/>
        <vnsInTerm descr="" name=""/>
        <vnsOutTerm descr="" name=""/>
    </vnsAbsTermNodeProv>
    <vnsAbsConnection adjType="L2" connDir="provider" connType="external" descr="" name="C1"
      ownerKey="" ownerTag="" unicastRoute="yes">
        <vnsRsAbsConnectionConns tDn="uni/tn-t22/AbsGraph-g0/AbsTermNodeCon-T1/AbsTConn"/>
        <vnsRsAbsConnectionConns tDn="uni/tn-t22/AbsGraph-g0/AbsTermNodeProv-T2/AbsTConn"/>
        <vnsRsAbsCopyConnection tDn="uni/tn-t22/AbsGraph-g0/AbsNode-CP1/AbsFConn-copy"/>
    </vnsAbsConnection>
    <vnsAbsNode descr="" funcTemplateType="OTHER" funcType="None" isCopy="yes" managed="no"
      name="CP1" ownerKey="" ownerTag="" routingMode="unspecified" sequenceNumber="0"
      shareEncap="no">
        <vnsAbsFuncConn attNotify="no" descr="" name="copy" ownerKey="" ownerTag=""/>
        <vnsRsNodeToLDev tDn="uni/tn-t22/lDevVip-copy0"/>
    </vnsAbsNode>
</vnsAbsGraph>
```

Step 4 Define the relation to the copy graph in the contract that is associated with the endpoint groups.

Example:

```
<vzBrCP descr="" name="c0" ownerKey="" ownerTag="" prio="unspecified" scope="tenant"
  targetDscp="unspecified">
    <vzSubj consMatchT="AtleastOne" descr="" name="Subject" prio="unspecified"
      provMatchT="AtleastOne" revFltPorts="yes" targetDscp="unspecified">
        <vzRsSubjFiltAtt directives="" tnVzFilterName="default"/>
        <vzRsSubjGraphAtt directives="" tnVnsAbsGraphName="g0"/>
    </vzSubj>
</vzBrCP>
```

Step 5 Attach the contract to the endpoint group.

Example:

```
<fvAEPg name="epg2860">
    <fvRsCons tnVzBrCPName="c0"/>
    <fvRsBd tnFvBDName="bd0"/>
    <fvRsDomAtt tDn="uni/phys-phys_scale_SB"/>
    <fvRsPathAtt tDn="topology/pod-1/paths-104/pathep-[PC_int2_g1]" encap="vlan-2860"
```

```
      instrImedcy="immediate"/>
</fvAEPg>
<fvAEPg name="epg2861">
    <fvRsProv tnVzBrCPName="c0"/>
    <fvRsBd tnFvBDName="bd0"/>
    <fvRsDomAtt tDn="uni/phys-phys_scale_SB"/>
    <fvRsPathAtt tDn="topology/pod-1/paths-105/pathep-[PC_policy]" encap="vlan-2861"
      instrImedcy="immediate"/>
</fvAEPg>
```

Developing Automation

About the REST APIs

Automation relies on the Application Policy Infrastructure Controller (APIC) northbound Representational State Transfer (REST) APIs. Anything that can be done through the Cisco APIC GUI can also be done using XML-based REST POSTs using the northbound APIs. For example, you can monitor events through those APIs, dynamically enable EPGs, and add policies.

You can also use the northbound REST APIs to monitor for notifications that a device has been brought onboard, and to monitor faults. In both cases, you can monitor events that trigger specific actions. For example, if you see faults that occur on a specific application tier and determine that there is a loss of connectivity and a leaf node is going down, you can trigger an action to redeploy those applications somewhere else. If you have certain contracts on which you detect packet drops occurring, you could enable some copies of those contracts on the particular application. You can also use a statistics monitoring policy, where you monitor certain counters because of issues that have been reported.

For information on how to construct the XML files submitted to the Cisco APIC northbound API, see *Cisco APIC Layer 4 to Layer 7 Device Package Development Guide*.

The following Python APIs, defined in the *Cisco APIC Management Information Model Reference* can be used to submit REST POST calls using the northbound API:

- `vns:LDevVip`: Upload a device cluster
- `vns:CDev`: Upload a device
- `vns:LIf`: Create logical interfaces
- `vns:AbsGraph`: Create a graph
- `vz:BrCP`: Attach a graph to a contract

Examples of Automating Using the REST APIs

This section contains examples of using the REST APIs to automate tasks.

The following REST request creates a tenant with a broadcast domain, a Layer 3 network, application endpoint groups, and an application profile:

```
<polUni>
    <fvTenant dn="uni/tn-acme" name="acme">
```

```
        <!—L3 Network-->
        <fvCtx name="MyNetwork"/>

        <!-- Bridge Domain for MySrvr EPG -->
        <fvBD name="MySrvrBD">
            <fvRsCtx tnFvCtxName="MyNetwork"/>
            <fvSubnet ip="10.10.10.10/24">
            </fvSubnet>
        </fvBD>

        <!-- Bridge Domain for MyClnt EPG -->
        <fvBD name="MyClntBD">
            <fvRsCtx tnFvCtxName="MyNetwork"/>
            <fvSubnet ip="20.20.20.20/24">
            </fvSubnet>
        </fvBD>

        <fvAp dn="uni/tn-acme/ap-MyAP" name="MyAP">

            <fvAEPg dn="uni/tn-acme/ap-MyAP/epg-MyClnt" name="MyClnt">
                <fvRsBd tnFvBDName="MySrvrBD"/>
                <fvRsDomAtt tDn="uni/vmmp-Vendor1/dom-MyVMs"/>
                <fvRsProv tnVzBrCPName="webCtrct"> </fvRsProv>
                <fvRsPathAtt tDn="topology/pod-1/paths-17/pathep-[eth1/21]"
                  encap="vlan-202"/>
                <fvRsPathAtt tDn="topology/pod-1/paths-18/pathep-[eth1/21]"
                  encap="vlan-202"/>
           </fvAEPg>

            <fvAEPg dn="uni/tn-acme/ap-MyAP/epg-MySRVR" name="MySRVR">
                <fvRsBd tnFvBDName="MyClntBD"/>
                <fvRsDomAtt tDn="uni/vmmp-Vendor1/dom-MyVMs"/>
                <fvRsCons tnVzBrCPName="webCtrct"> </fvRsCons>
                <fvRsPathAtt tDn="topology/pod-1/paths-17/pathep-[eth1/21]"
                  encap="vlan-203"/>
                <fvRsPathAtt tDn="topology/pod-1/paths-18/pathep-[eth1/21]"
                  encap="vlan-203"/>
            </fvAEPg>
        </fvAp>
    </fvTenant>
</polUni>
```

The following REST request creates a VLAN namespace:

```
<polUni>
    <infraInfra>
        <fvnsVlanInstP name="MyNS" allocMode="dynamic">
            <fvnsEncapBlk name="encap" from="vlan-201" to="vlan-300"/>
        </fvnsVlanInstP>
    </infraInfra>
</polUni>
```

The following REST request creates a VMM domain:

```
<polUni>
    <vmmProvP vendor="Vendor1">
        <vmmDomP name="MyVMs">
            <infraRsVlanNs tDn="uni/infra/vlanns-MyNS-dynamic"/>
            <vmmUsrAccP name="admin" usr="administrator" pwd="in$1eme"/>
            <vmmCtrlrP name="vcenter1" hostOrIp="192.168.64.186">
                <vmmRsAcc tDn="uni/vmmp-Vendor1/dom-MyVMs/usracc-admin"/>
            </vmmCtrlrP>
        </vmmDomP>
    </vmmProvP>
</polUni>
```

The following REST request creates a physical domain:

```
<polUni>
    <physDomP name="phys">
        <infraRsVlanNs tDn="uni/infra/vlanns-MyNS-dynamic"/>
    </physDomP>
</polUni>
```

The following REST request creates a managed device cluster:

```
<polUni>
    <fvTenant dn="uni/tn-acme" name="acme">
        <vnsLDevVip name="ADCCluster1" contextAware=1>
            <vnsRsMDevAtt tDn="uni/infra/mDev-Acme-ADC-1.0"/>
            <vnsRsDevEpg tDn="uni/tn-acme/ap-services/epg-ifc"/>
            <vnsRsALDevToPhysDomP tDn="uni/phys-phys"/>
            <vnsCMgmt name="devMgmt" host="42.42.42.100" port="80"/>
            <vnsCCred name="username"value="admin"/>
            <vnsCCredSecret name="password" value="admin"/>
        </vnsLDevVip>
    </fvTenant>
</polUni>
```

The following REST request creates an unmanaged device cluster:

```
<polUni>
    <fvTenant name="HA_Tenant1">
        <vnsLDevVip name="ADCCluster1" devtype="VIRTUAL" managed="no">
            <vnsRsALDevToDomP tDn="uni/vmmp-VMware/dom-mininet"/>
        </vnsLDevVip>
    </fvTenant>
</polUni>
```

The following REST request creates a device cluster context:

```
<polUni>
    <fvTenant dn="uni/tn-acme" name="acme">
        <vnsLDevCtx ctrctNameOrLbl="webCtrct" graphNameOrLbl="G1" nodeNameOrLbl="Node1">
            <vnsRsLDevCtxToLDev tDn="uni/tn-acme/lDevVip-ADCCluster1"/>
            <vnsLIfCtx connNameOrLbl="ssl-inside">
                <vnsRsLIfCtxToLIf tDn="uni/tn-acme/lDevVip-ADCCluster1/lIf-int"/>
            </vnsLIfCtx>
            <vnsLIfCtx connNameOrLbl="any">
                <vnsRsLIfCtxToLIf tDn="uni/tn-acme/lDevVip-ADCCluster1/lIf-ext"/>
            </vnsLIfCtx>
        </vnsLDevCtx>
    </fvTenant>
</polUni>
```

The following REST request creates a device cluster context used in route peering:

```
<polUni>
    <fvTenant dn="uni/tn-coke{{tenantId}}" name="coke{{tenantId}}">
        <vnsRtrCfg name="Dev1Ctx1" rtrId="180.0.0.12"/>
            <vnsLDevCtx ctrctNameOrLbl="webCtrct1" graphNameOrLbl="WebGraph"
              nodeNameOrLbl="FW">
                <vnsRsLDevCtxToLDev tDn="uni/tn-tenant1/lDevVip-Firewall"/>
                <vnsRsLDevCtxToRtrCfg tnVnsRtrCfgName="FwRtrCfg"/>
                <vnsLIfCtx connNameOrLbl="internal">
                  <vnsRsLIfCtxToInstP tDn="uni/tn-tenant1/out-OspfInternal/instP-IntInstP"

                      status="created,modified"/>
                   <vnsRsLIfCtxToLIf tDn="uni/tn-tenant1/lDevVip-Firewall/lIf-internal"/>
                </vnsLIfCtx>
                <vnsLIfCtx connNameOrLbl="external">
                   <vnsRsLIfCtxToInstP tDn="uni/tn-common/out-OspfExternal/instP-ExtInstP"
```

```
                    status="created,modified"/>
                <vnsRsLIfCtxToLIf tDn="uni/tn-tenant1/lDevVip-Firewall/lIf-external"/>
            </vnsLIfCtx>
        </vnsLDevCtx>
    </fvTenant>
</polUni>
```

Note For information about configuring external connectivity for tenants (a Layer 3 outside), see the *Cisco APIC Basic Configuration Guide*.

The following REST request adds a logical interface in a device cluster:

```
<polUni>
    <fvTenant dn="uni/tn-acme" name="acme">
        <vnsLDevVip name="ADCCluster1">
            <vnsLIf name="C5">
                <vnsRsMetaIf tDn="uni/infra/mDev-Acme-ADC-1.0/mIfLbl-outside"/>
                <vnsRsCIfAtt tDn="uni/tn-acme/lDevVip-ADCCluster1/cDev-ADC1/cIf-int"/>
            </vnsLIf>
            <vnsLIf name="C4">
                <vnsRsMetaIf tDn="uni/infra/mDev-Acme-ADC-1.0/mIfLbl-inside"/>
                <vnsRsCIfAtt tDn="uni/tn-acme/lDevVip-ADCCluster1/cDev-ADC1/cIf-ext"/>
            </vnsLIf>
        </vnsLDevVip>
    </fvTenant>
</polUni>
```

The following REST request adds a concrete device in a physical device cluster:

```
<polUni>
    <fvTenant dn="uni/tn-acme" name="acme">
        <vnsLDevVip name="ADCCluster1">
            <vnsCDev name="ADC1" devCtxLbl="C1">
                <vnsCIf name="int">
                    <vnsRsCIfPathAtt tDn="topology/pod-1/paths-17/pathep-[eth1/22]"/>
                </vnsCIf>
                <vnsCIf name="ext">
                    <vnsRsCIfPathAtt tDn="topology/pod-1/paths-17/pathep-[eth1/21]"/>
                </vnsCIf>
                <vnsCIf name="mgmt">
                    <vnsRsCIfPathAtt tDn="topology/pod-1/paths-17/pathep-[eth1/20]"/>
                </vnsCIf>
                <vnsCMgmt name="devMgmt" host="172.30.30.100" port="80"/>
                <vnsCCred name="username" value="admin"/>
                <vnsCCredSecret name="password" value="admin"/>
            </vnsCDev>
            <vnsCDev name="ADC2" devCtxLbl="C2">
                <vnsCIf name="int">
                    <vnsRsCIfPathAtt tDn="topology/pod-1/paths-17/pathep-[eth1/23]"/>
                </vnsCIf>
                <vnsCIf name="ext">
                    <vnsRsCIfPathAtt tDn="topology/pod-1/paths-17/pathep-[eth1/24]"/>
                </vnsCIf>
                <vnsCIf name="mgmt">
                    <vnsRsCIfPathAtt tDn="topology/pod-1/paths-17/pathep-[eth1/30]"/>
                </vnsCIf>
                <vnsCMgmt name="devMgmt" host="172.30.30.200" port="80"/>
                <vnsCCred name="username" value="admin"/>
                <vnsCCredSecret name="password" value="admin"/>
            </vnsCDev>
        </vnsLDevVip>
```

```
    </fvTenant>
</polUni>
```

The following REST request adds a concrete device in a virtual device cluster:

```
<polUni>
    <fvTenant dn="uni/tn-coke5" name="coke5">
        <vnsLDevVip name="Firewall5" devtype="VIRTUAL">
            <vnsCDev name="ASA5" vcenterName="vcenter1" vmName="ifav16-ASAv-scale-05">
                <vnsCIf name="Gig0/0" vnicName="Network adapter 2"/>
                <vnsCIf name="Gig0/1" vnicName="Network adapter 3"/>
                <vnsCIf name="Gig0/2" vnicName="Network adapter 4"/>
                <vnsCIf name="Gig0/3" vnicName="Network adapter 5"/>
                <vnsCIf name="Gig0/4" vnicName="Network adapter 6"/>
                <vnsCIf name="Gig0/5" vnicName="Network adapter 7"/>
                <vnsCIf name="Gig0/6" vnicName="Network adapter 8"/>
                <vnsCIf name="Gig0/7" vnicName="Network adapter 9"/>
                <vnsCMgmt name="devMgmt" host="3.5.3.170" port="443"/>
                <vnsCCred name="username" value="admin"/>
                <vnsCCredSecret name="password" value="insieme"/>
            </vnsCDev>
        </vnsLDevVip>
    </fvTenant>
</polUni>
```

The following REST request creates a service graph in managed mode:

```
<polUni>
    <fvTenant name="acme">
        <vnsAbsGraph name = "G1">

        <vnsAbsTermNode name = "Input1">
            <vnsAbsTermConn name = "C1" direction = "output">
            </vnsAbsTermConn>
        </vnsAbsTermNode>

        <!-- Node1 Provides SLB functionality -->
        <vnsAbsNode name = "Node1" funcType="GoTo" >
            <vnsRsDefaultScopeToTerm
                tDn="uni/tn-acme/AbsGraph-G1/AbsTermNode-Output1/outtmnl"/>

            <vnsAbsFuncConn name = "C4" direction = "input">
               <vnsRsMConnAtt tDn="uni/infra/mDev-Acme-ADC-1.0/mFunc-SLB/mConn-external"/>

                <vnsRsConnToLIf tDn="uni/tn-acme/lDevVip-ADCCluster1/lIf-C4"/>
            </vnsAbsFuncConn>

            <vnsAbsFuncConn name = "C5" direction = "output">
               <vnsRsMConnAtt tDn="uni/infra/mDev-Acme-ADC-1.0/mFunc-SLB/mConn-internal"/>

                <vnsRsConnToLIf tDn="uni/tn-acme/lDevVip-ADCCluster1/lIf-C5"/>
            </vnsAbsFuncConn>

            <vnsRsNodeToMFunc tDn="uni/infra/mDev-Acme-ADC-1.0/mFunc-SLB"/>
        </vnsAbsNode>

        <vnsAbsTermNode name = "Output1">
            <vnsAbsTermConn name = "C6" direction = "input">
            </vnsAbsTermConn>
        </vnsAbsTermNode>

        <vnsAbsConnection name = "CON1">
            <vnsRsAbsConnectionConns
                tDn="uni/tn-acme/AbsGraph-G1/AbsTermNode-Input1/AbsTConn"/>
            <vnsRsAbsConnectionConns
```

```
                tDn="uni/tn-acme/AbsGraph-G1/AbsNode-Node1/AbsFConn-C4"/>
        </vnsAbsConnection>

        <vnsAbsConnection name = "CON3">
            <vnsRsAbsConnectionConns
                tDn="uni/tn-acme/AbsGraph-G1/AbsNode-Node1/AbsFConn-C5"/>
            <vnsRsAbsConnectionConns
                tDn="uni/tn-acme/AbsGraph-G1/AbsTermNode-Output1/AbsTConn"/>
        </vnsAbsConnection>
    </vnsAbsGraph>
  </fvTenant>
</polUni>
```

The following REST request creates a service graph in unmanaged mode:

```
<polUni>
    <fvTenant name="HA_Tenant1">
        <vnsAbsGraph name="g1">

            <vnsAbsTermNodeProv name="Input1">
                <vnsAbsTermConn name="C1">
                </vnsAbsTermConn>
            </vnsAbsTermNodeProv>

            <!-- Node1 Provides LoadBalancing functionality -->
            <vnsAbsNode name="Node1" managed="no">
                <vnsRsDefaultScopeToTerm
                  tDn="uni/tn-HA_Tenant1/AbsGraph-g1/AbsTermNodeProv-Input1/outtmnl"/>
                <vnsAbsFuncConn name="outside" attNotify="true">
                </vnsAbsFuncConn>
                <vnsAbsFuncConn name="inside" attNotify="true">
                </vnsAbsFuncConn>
            </vnsAbsNode>

            <vnsAbsTermNodeCon name="Output1">
                <vnsAbsTermConn name="C6">
                </vnsAbsTermConn>
            </vnsAbsTermNodeCon>

            <vnsAbsConnection name="CON2" adjType="L3" unicastRoute="yes">
                <vnsRsAbsConnectionConns
                  tDn="uni/tn-HA_Tenant1/AbsGraph-g1/AbsTermNodeCon-Output1/AbsTConn"/>
                <vnsRsAbsConnectionConns
                  tDn="uni/tn-HA_Tenant1/AbsGraph-g1/AbsNode-Node1/AbsFConn-outside"/>
            </vnsAbsConnection>

            <vnsAbsConnection name="CON1" adjType="L2" unicastRoute="no">
                <vnsRsAbsConnectionConns
                  tDn="uni/tn-HA_Tenant1/AbsGraph-g1/AbsNode-Node1/AbsFConn-inside"/>
                <vnsRsAbsConnectionConns
                  tDn="uni/tn-HA_Tenant1/AbsGraph-g1/AbsTermNodeProv-Input1/AbsTConn"/>
            </vnsAbsConnection>

        </vnsAbsGraph>
    </fvTenant>
</polUni>
```

The following REST request creates a filter and a security policy (contract):

```
<polUni>
    <fvTenant dn="uni/tn-acme" name="acme">
        <vzFilter name="HttpIn">
            <vzEntry name="e1" prot="6" dToPort="80"/>
        </vzFilter>
```

```
        <vzBrCP name="webCtrct">
            <vzSubj name="http">
                <vzRsSubjFiltAtt tnVzFilterName="HttpIn"/>
            </vzSubj>
        </vzBrCP>
    </fvTenant>
</polUni>
```

The following REST request provides graph configuration parameters from an application EPG:

```
<polUni>
   <fvTenant dn="uni/tn-acme" name="acme">

      <!-- Application Profile -->
      <fvAp dn="uni/tn-acme/ap-MyAP" name="MyAP">

          <!-- EPG 1  -->
          <fvAEPg dn="uni/tn-acme/ap-MyAP/epg-MyClnt" name="MyClnt">
              <fvRsBd tnFvBDName="MyClntBD"/>
              <fvRsDomAtt tDn="uni/vmmp-Vendor1/dom-MyVMs"/>
              <fvRsProv tnVzBrCPName="webCtrct">
              </fvRsProv>
            <fvRsPathAtt tDn="topology/pod-1/paths-17/pathep-[eth1/20]" encap="vlan-201"/>

              <fvSubnet name="SrcSubnet" ip="192.168.10.1/24"/>
          </fvAEPg>

          <!-- EPG 2 -->
          <fvAEPg dn="uni/tn-acme/ap-MyAP/epg-MySRVR"  name="MySRVR">
             <fvRsBd tnFvBDName="MyClntBD"/>
             <fvRsDomAtt tDn="uni/vmmp-Vendor1/dom-MyVMs"/>
             <fvRsCons tnVzBrCPName="webCtrct">
             </fvRsCons>

             <vnsFolderInst ctrctNameOrLbl="any" graphNameOrLbl="any" nodeNameOrLbl="any"
               key="Monitor" name="monitor1">
                 <vnsParamInst name="weight" key="weight" value="10"/>
             </vnsFolderInst>

             <vnsFolderInst ctrctNameOrLbl="any" graphNameOrLbl="any" nodeNameOrLbl="any"
               key="Service" name="Service1">
                 <vnsParamInst name="servicename" key="servicename"
                   value="crpvgrtst02-8010"/>
                 <vnsParamInst name="servicetype" key="servicetype" value="TCP"/>
                 <vnsParamInst name="servername" key="servername"
                   value="s192.168.100.100"/>
                 <vnsParamInst name="serveripaddress" key="serveripaddress"
                   value="192.168.100.100"/>
                 <vnsParamInst name="serviceport" key="serviceport" value="8080"/>
                 <vnsParamInst name="svrtimeout" key="svrtimeout" value="9000"/>
                 <vnsParamInst name="clttimeout" key="clttimeout" value="9000"/>
                 <vnsParamInst name="usip" key="usip" value="NO"/>
                 <vnsParamInst name="useproxyport" key="useproxyport" value=""/>
                 <vnsParamInst name="cip" key="cip" value="ENABLED"/>
                 <vnsParamInst name="cka" key="cka" value="NO"/>
                 <vnsParamInst name="sp" key="sp" value="OFF"/>
                 <vnsParamInst name="cmp" key="cmp" value="NO"/>
                 <vnsParamInst name="maxclient" key="maxclient" value="0"/>
                 <vnsParamInst name="maxreq" key="maxreq" value="0"/>
                 <vnsParamInst name="tcpb" key="tcpb" value="NO"/>
                 <vnsCfgRelInst name="MonitorConfig" key="MonitorConfig"
                   targetName="monitor1"/>
              </vnsFolderInst>

              <vnsFolderInst ctrctNameOrLbl="any" graphNameOrLbl="any"
```

```
                  nodeNameOrLbl="any" key="Network" name="Network">
                   <vnsFolderInst ctrctNameOrLbl="any" graphNameOrLbl="any"
                     nodeNameOrLbl="any" key="vip" name="vip">
                       <vnsParamInst name="vipaddress1" key="vipaddress"
                         value="10.10.10.100"/>
                   </vnsFolderInst>
                   <vnsFolderInst ctrctNameOrLbl="any" graphNameOrLbl="any"
                     nodeNameOrLbl="any" devCtxLbl="C1" key="snip" name="snip1">
                       <vnsParamInst name="snipaddress" key="snipaddress"
                         value="192.168.1.100"/>
                   </vnsFolderInst>
                   <vnsFolderInst ctrctNameOrLbl="any" graphNameOrLbl="any"
                     nodeNameOrLbl="any" devCtxLbl="C2" key="snip" name="snip2">
                       <vnsParamInst name="snipaddress" key="snipaddress"
                         value="192.168.1.101"/>
                   </vnsFolderInst>
                   <vnsFolderInst ctrctNameOrLbl="any" graphNameOrLbl="any"
                     nodeNameOrLbl="any" devCtxLbl="C3" key="snip" name="snip3">
                       <vnsParamInst name="snipaddress" key="snipaddress"
                         value="192.168.1.102"/>
                   </vnsFolderInst>
                 </vnsFolderInst>

                 <!-- SLB Configuration -->
                 <vnsFolderInst ctrctNameOrLbl="any" graphNameOrLbl="any"
                  nodeNameOrLbl="any" key="VServer" name="VServer">
                   <!-- Virtual Server Configuration -->
                   <vnsParamInst name="port" key="port" value="8010"/>
                   <vnsParamInst name="vip" key="vip" value="10.10.10.100"/>
                   <vnsParamInst name="vservername" key="vservername"
                     value="crpvgrtst02-vip-8010"/>
                   <vnsParamInst name="servicename" key="servicename"
                     value="crpvgrtst02-8010"/>
                   <vnsParamInst name="servicetype" key="servicetype" value="TCP"/>
                   <vnsFolderInst ctrctNameOrLbl="any" graphNameOrLbl="any"
                    nodeNameOrLbl="any" key="VServerGlobalConfig" name="VServerGlobalConfig">

                       <vnsCfgRelInst name="ServiceConfig" key="ServiceConfig"
                         targetName="Service1"/>
                       <vnsCfgRelInst name="VipConfig" key="VipConfig"
                         targetName="Network/vip"/>
                   </vnsFolderInst>
                 </vnsFolderInst>
           </fvAEPg>
       </fvAp>
    </fvTenant>
</polUni>
```

The following REST request attaches a service graph to a contract:

```
<polUni>
    <fvTenant name="acme">
        <vzBrCP name="webCtrct">
            <vzSubj name="http">
                <vzRsSubjGraphAtt graphName="G1" termNodeName="Input1"/>
            </vzSubj>
        </vzBrCP>
    </fvTenant>
</polUni>
```

Example: Configuring Layer 4 to Layer 7 Services (Firewall)

Example: Configuring Layer 4 to Layer 7 Services Using the REST API

This topic shows the steps for configuring Layer 4 to Layer 7 services (ASA Firewall) using the REST API.

Before you begin

- Create the tenant to use the Layer 4 to Layer 7 services, with a Layer 3 outside network and bridge domains.
- Create application profiles.
- Configure a physical or VMM domain.
- Import and register the device packages and configure parameters for them.

Step 1 Create a Layer 4 to Layer 7 **ASAv** device package model, using XML such as the following example:

Example:

```
<vnsLDevVip trunking="no" svcType="FW"
packageModel="ASAv" name="ASAv" mode="legacy-Mode"
managed="yes" isCopy="no" funcType="GoTo"
dn="uni/tn-Tenant-test/lDevVip-ASAv" devtype="VIRTUAL"
contextAware="single-Context">

<vnsCCred name="username" value="admin"/>
<vnsRsMDevAtt tDn="uni/infra/mDev-CISCO-ASA-1.2"/>
<vnsCCredSecret name="password"/>
<vnsCMgmt name="" port="443" host="172.31.184.249"/>
<vnsRsALDevToDomP tDn="uni/vmmp-VMware/dom-ACI_vDS"/>
<vnsCDev name="Device1" vmName="ASAv-L3" vcenterName="vcenter" devCtxLbl="">
<vnsCCred name="username" value="admin"/>
<vnsCCredSecret name="password"/>
<vnsCMgmt name="" port="443" host="172.31.184.249"/>
<vnsCIf name="GigabitEthernet0/1" vnicName="Network adapter 3"/>
<vnsCIf name="GigabitEthernet0/0" vnicName="Network adapter 2"/>
<vnsRsCDevToCtrlrP tDn="uni/vmmp-VMware/dom-ACI_vDS/ctrlr-vcenter"/>
</vnsCDev>

<vnsLIf name="provider" encap="unknown">
<vnsRsMetaIf tDn="uni/infra/mDev-CISCO-ASA-1.2/mIfLbl-internal" isConAndProv="no"/>
<vnsRsCIfAttN tDn="uni/tn-Tenant-test/lDevVip-ASAv/cDev-Device1/cIf-[GigabitEthernet0/1]"/>
</vnsLIf>

<vnsLIf name="consumer" encap="unknown">
<vnsRsMetaIf tDn="uni/infra/mDev-CISCO-ASA-1.2/mIfLbl-external" isConAndProv="no"/>
<vnsRsCIfAttN tDn="uni/tn-Tenant-test/lDevVip-ASAv/cDev-Device1/cIf-[GigabitEthernet0/0]"/>
</vnsLIf>
</vnsLDevVip>
```

Step 2 Configure a Layer 4 to Layer 7 **FW-Graph** using XML such as the following example:

Example:

```
<vnsAbsGraph uiTemplateType="UNSPECIFIED" ownerTag="" ownerKey="" name="FW-Graph"
dn="uni/tn-Tenant-test/AbsGraph-FW-Graph" descr="">

<vnsAbsTermNodeCon ownerTag="" ownerKey="" name="T1" descr="">
<vnsAbsTermConn ownerTag="" ownerKey="" name="1" descr="" attNotify="no"/>
<vnsInTerm name="" descr=""/>
<vnsOutTerm name="" descr=""/>
</vnsAbsTermNodeCon>

<vnsAbsTermNodeProv ownerTag="" ownerKey="" name="T2" descr="">
<vnsAbsTermConn ownerTag="" ownerKey="" name="1" descr="" attNotify="no"/>
<vnsInTerm name="" descr=""/>
<vnsOutTerm name="" descr=""/>
</vnsAbsTermNodeProv>

<vnsAbsConnection ownerTag="" ownerKey="" name="C1" descr="" unicastRoute="yes"
directConnect="no" connType="external" connDir="provider" adjType="L2">
<vnsRsAbsConnectionConns tDn="uni/tn-Tenant-test/AbsGraph-FW-Graph/AbsNode-N1/AbsFConn-consumer"/>
<vnsRsAbsConnectionConns tDn="uni/tn-Tenant-test/AbsGraph-FW-Graph/AbsTermNodeCon-T1/AbsTConn"/>
</vnsAbsConnection>

<vnsAbsConnection ownerTag="" ownerKey="" name="C2" descr="" unicastRoute="yes"
directConnect="no" connType="external" connDir="provider" adjType="L2">
<vnsRsAbsConnectionConns tDn="uni/tn-Tenant-test/AbsGraph-FW-Graph/AbsNode-N1/AbsFConn-provider"/>
<vnsRsAbsConnectionConns tDn="uni/tn-Tenant-test/AbsGraph-FW-Graph/AbsTermNodeProv-T2/AbsTConn"/>
</vnsAbsConnection>

<vnsAbsNode ownerTag="" ownerKey="" name="N1" descr="" shareEncap="no" sequenceNumber="0"
routingMode="unspecified" managed="yes" isCopy="no" funcType="GoTo" funcTemplateType="FW_ROUTED">
<vnsAbsFuncConn ownerTag="" ownerKey="" name="consumer" descr="" attNotify="no">
<vnsRsMConnAtt tDn="uni/infra/mDev-CISCO-ASA-1.2/mFunc-Firewall/mConn-external"/>
</vnsAbsFuncConn>

<vnsAbsFuncConn ownerTag="" ownerKey="" name="provider" descr="" attNotify="no">
<vnsRsMConnAtt tDn="uni/infra/mDev-CISCO-ASA-1.2/mFunc-Firewall/mConn-internal"/>
</vnsAbsFuncConn>
<vnsRsNodeToAbsFuncProf
tDn="uni/infra/mDev-CISCO-ASA-1.2/absFuncProfContr/absFuncProfGrp-WebServiceProfileGroup/absFuncProf-WebPolicyForRoutedMode"/>
<vnsRsNodeToLDev tDn="uni/tn-Tenant-test/lDevVip-ASAv"/>
<vnsRsNodeToMFunc tDn="uni/infra/mDev-CISCO-ASA-1.2/mFunc-Firewall"/>
</vnsAbsNode>
</vnsAbsGraph>
```

Step 3 Create a device selection policy, using XML such as the following example:

Example:

```
<vnsLDevCtx nodeNameOrLbl="N1" name="" graphNameOrLbl="FW-Graph"
dn="uni/tn-Tenant-test/ldevCtx-c-Client-to-Web-g-FW-Graph-n-N1" descr="" ctrctNameOrLbl="Client-to-Web">
<vnsRsLDevCtxToLDev tDn="uni/tn-Tenant-test/lDevVip-ASAv"/>

<vnsLIfCtx name="" descr="" connNameOrLbl="provider">
<vnsRsLIfCtxToBD tDn="uni/tn-Tenant-test/BD-BD2"/>
<vnsRsLIfCtxToLIf tDn="uni/tn-Tenant-test/lDevVip-ASAv/lIf-provider"/>
</vnsLIfCtx>

<vnsLIfCtx name="" descr="" connNameOrLbl="consumer">
<vnsRsLIfCtxToBD tDn="uni/tn-Tenant-test/BD-BD1"/>
<vnsRsLIfCtxToLIf tDn="uni/tn-Tenant-test/lDevVip-ASAv/lIf-consumer"/>
</vnsLIfCtx>
</vnsLDevCtx>
```

Step 4 Configure a contract, associated with the **FW-Graph** service graph template, using XML such as the following example:

Example:

```
<vzBrCP targetDscp="unspecified" scope="tenant" prio="unspecified" ownerTag=""
ownerKey="" name="Client-to-Web" dn="uni/tn-Tenant-test/brc-Client-to-Web" descr="">

<vzSubj targetDscp="unspecified" prio="unspecified" name="Subject" descr=""
revFltPorts="yes" provMatchT="AtleastOne" consMatchT="AtleastOne"
<vzRsSubjFiltAtt tnVzFilterName="default" directives=""/>
<vzRsSubjGraphAtt directives="" tnVnsAbsGraphName="FW-Graph"/>
</vzSubj>
</vzBrCP>
```

Step 5 Create the **Client** EPG, using XML such as the following example:

Example:

```
<fvAEPg prio="unspecified" pcEnfPref="unenforced" name="Client"
matchT="AtleastOne" isAttrBasedEPg="no" fwdCtrl="" dn="uni/tn-Tenant-test/ap-ANP/epg-Client" descr="">
<fvRsCons prio="unspecified" tnVzBrCPName="Client-to-Web"/>
<fvRsDomAtt tDn="uni/vmmp-VMware/dom-ACI_vDS" resImedcy="lazy" primaryEncap="unknown"
instrImedcy="lazy" encap="unknown" delimiter="" classPref="encap"/>
<fvRsBd tnFvBDName="BD1"/>
<fvRsCustQosPol tnQosCustomPolName=""/>
</fvAEPg>
```

Step 6 Create the **Web** EPG, using XML such as the following example:

Example:

```
-<fvAEPg prio="unspecified" pcEnfPref="unenforced" name="Web" matchT="AtleastOne"
isAttrBasedEPg="no" fwdCtrl="" dn="uni/tn-Tenant-test/ap-ANP/epg-Web" descr="">
<fvRsDomAtt tDn="uni/vmmp-VMware/dom-ACI_vDS" resImedcy="lazy" primaryEncap="unknown"
instrImedcy="lazy" encap="unknown" delimiter="" classPref="encap"/>
<fvRsBd tnFvBDName="BD2"/>
<fvRsCustQosPol tnQosCustomPolName=""/>

<vnsFolderInst name="internalIf" scopedBy="epg" nodeNameOrLbl="N1" locked="no" key="Interface"
graphNameOrLbl="FW-Graph" devCtxLbl="" ctrctNameOrLbl="Client-to-Web" cardinality="unspecified">

<vnsFolderInst name="internalIfCfg" scopedBy="epg" nodeNameOrLbl="N1" locked="no" key="InterfaceConfig"

graphNameOrLbl="FW-Graph" devCtxLbl="" ctrctNameOrLbl="Client-to-Web" cardinality="unspecified">
<vnsParamInst name="internal_security_level" locked="no" key="security_level" cardinality="unspecified"

value="100" validation="" mandatory="no"/>
</vnsFolderInst>
</vnsFolderInst>

<vnsFolderInst name="externalIf" scopedBy="epg" nodeNameOrLbl="N1" locked="no" key="Interface"
graphNameOrLbl="FW-Graph" devCtxLbl="" ctrctNameOrLbl="Client-to-Web" cardinality="unspecified">

<vnsFolderInst name="ExtAccessGroup" scopedBy="epg" nodeNameOrLbl="N1" locked="no" key="AccessGroup"

graphNameOrLbl="FW-Graph" devCtxLbl="" ctrctNameOrLbl="Client-to-Web" cardinality="unspecified">
<vnsCfgRelInst name="name" locked="no" key="inbound_access_list_name" cardinality="unspecified"
mandatory="no"
targetName="access-list-inbound"/>
</vnsFolderInst>

<vnsFolderInst name="externalIfCfg" scopedBy="epg" nodeNameOrLbl="N1" locked="no" key="InterfaceConfig"

graphNameOrLbl="FW-Graph" devCtxLbl="" ctrctNameOrLbl="Client-to-Web" cardinality="unspecified">

<vnsParamInst name="external_security_level" locked="no" key="security_level"
cardinality="unspecified" value="50" validation="" mandatory="no"/>
</vnsFolderInst>
</vnsFolderInst>
```

```
<vnsFolderInst name="IntConfig" scopedBy="epg" nodeNameOrLbl="N1" locked="no"
key="InIntfConfigRelFolder"
graphNameOrLbl="FW-Graph" devCtxLbl="" ctrctNameOrLbl="Client-to-Web" cardinality="unspecified">
<vnsCfgRelInst name="InConfigrel" locked="no" key="InIntfConfigRel"
cardinality="unspecified" mandatory="no" targetName="internalIf"/>
</vnsFolderInst>

<vnsFolderInst name="ExtConfig" scopedBy="epg" nodeNameOrLbl="N1" locked="no"
key="ExIntfConfigRelFolder"
graphNameOrLbl="FW-Graph" devCtxLbl="" ctrctNameOrLbl="Client-to-Web" cardinality="unspecified">
<vnsCfgRelInst name="ExtConfigrel" locked="no" key="ExIntfConfigRel" cardinality="unspecified"
mandatory="no"
targetName="externalIf"/>
</vnsFolderInst>

<vnsFolderInst name="access-list-inbound" scopedBy="epg" nodeNameOrLbl="N1" locked="no" key="AccessList"

graphNameOrLbl="FW-Graph" devCtxLbl="" ctrctNameOrLbl="Client-to-Web" cardinality="unspecified">
<vnsFolderInst name="permit-https" scopedBy="epg" nodeNameOrLbl="N1" locked="no"
key="AccessControlEntry"
graphNameOrLbl="FW-Graph" devCtxLbl="" ctrctNameOrLbl="Client-to-Web" cardinality="unspecified">
<vnsParamInst name="action-permit" locked="no" key="action" cardinality="unspecified" value="permit"

validation="" mandatory="no"/>
<vnsParamInst name="order1" locked="no" key="order" cardinality="unspecified" value="10" validation=""
 mandatory="no"/>

<vnsFolderInst name="dest-service" scopedBy="epg" nodeNameOrLbl="N1" locked="no"
key="destination_service"
graphNameOrLbl="FW-Graph" devCtxLbl="" ctrctNameOrLbl="Client-to-Web" cardinality="unspecified">
<vnsParamInst name="op" locked="no" key="operator" cardinality="unspecified" value="eq" validation=""
 mandatory="no"/>
<vnsParamInst name="port" locked="no" key="low_port" cardinality="unspecified" value="https"
validation="" mandatory="no"/>
</vnsFolderInst>

<vnsFolderInst name="dest-address" scopedBy="epg" nodeNameOrLbl="N1" locked="no"
key="destination_address"
graphNameOrLbl="FW-Graph" devCtxLbl="" ctrctNameOrLbl="Client-to-Web" cardinality="unspecified">
<vnsParamInst name="any" locked="no" key="any" cardinality="unspecified" value="any" validation=""
mandatory="no"/>
</vnsFolderInst>

<vnsFolderInst name="src-address" scopedBy="epg" nodeNameOrLbl="N1" locked="no" key="source_address"

graphNameOrLbl="FW-Graph" devCtxLbl="" ctrctNameOrLbl="Client-to-Web" cardinality="unspecified">
<vnsParamInst name="any" locked="no" key="any" cardinality="unspecified" value="any" validation=""
mandatory="no"/>
</vnsFolderInst>

<vnsFolderInst name="tcp" scopedBy="epg" nodeNameOrLbl="N1" locked="no" key="protocol"
graphNameOrLbl="FW-Graph" devCtxLbl="" ctrctNameOrLbl="Client-to-Web" cardinality="unspecified">
<vnsParamInst name="tcp" locked="no" key="name_number" cardinality="unspecified" value="tcp"
validation="" mandatory="no"/>
</vnsFolderInst>
</vnsFolderInst>

<vnsFolderInst name="permit-http" scopedBy="epg" nodeNameOrLbl="N1" locked="no" key="AccessControlEntry"

graphNameOrLbl="FW-Graph" devCtxLbl="" ctrctNameOrLbl="Client-to-Web" cardinality="unspecified">
<vnsParamInst name="action-permit" locked="no" key="action" cardinality="unspecified" value="permit"

validation="" mandatory="no"/>
```

```
<vnsParamInst name="order1" locked="no" key="order" cardinality="unspecified" value="10" validation=""
 mandatory="no"/>

<vnsFolderInst name="dest-service" scopedBy="epg" nodeNameOrLbl="N1" locked="no"
key="destination_service"
graphNameOrLbl="FW-Graph" devCtxLbl="" ctrctNameOrLbl="Client-to-Web" cardinality="unspecified">
<vnsParamInst name="op" locked="no" key="operator" cardinality="unspecified" value="eq" validation=""
 mandatory="no"/>
<vnsParamInst name="port" locked="no" key="low_port" cardinality="unspecified" value="http"
validation="" mandatory="no"/>
</vnsFolderInst>

<vnsFolderInst name="dest-address" scopedBy="epg" nodeNameOrLbl="N1" locked="no"
key="destination_address"
graphNameOrLbl="FW-Graph" devCtxLbl="" ctrctNameOrLbl="Client-to-Web" cardinality="unspecified">
<vnsParamInst name="any" locked="no" key="any" cardinality="unspecified" value="any" validation=""
mandatory="no"/>
</vnsFolderInst>

<vnsFolderInst name="src-address" scopedBy="epg" nodeNameOrLbl="N1" locked="no" key="source_address"

graphNameOrLbl="FW-Graph" devCtxLbl="" ctrctNameOrLbl="Client-to-Web" cardinality="unspecified">
<vnsParamInst name="any" locked="no" key="any" cardinality="unspecified" value="any" validation=""
mandatory="no"/>
</vnsFolderInst>

<vnsFolderInst name="tcp" scopedBy="epg" nodeNameOrLbl="N1" locked="no" key="protocol"
graphNameOrLbl="FW-Graph" devCtxLbl="" ctrctNameOrLbl="Client-to-Web" cardinality="unspecified">

<vnsParamInst name="tcp" locked="no" key="name_number" cardinality="unspecified"
value="tcp" validation="" mandatory="no"/>
</vnsFolderInst>
</vnsFolderInst>
</vnsFolderInst>

<fvRsProv prio="unspecified" matchT="AtleastOne" tnVzBrCPName="Client-to-Web"/>
</fvAEPg>
```

Example

To configure the entire Layer 4 to Layer 7 ASAv firewall services for a tenant, use XML such as the following example;

```
<fvTenant ownerTag="" ownerKey="" name="Tenant-test" dn="uni/tn-Tenant-test" descr="">

<vnsLDevCtx name="" descr="" nodeNameOrLbl="N1" graphNameOrLbl="FW-Graph"
ctrctNameOrLbl="Client-to-Web"><vnsRsLDevCtxToLDev tDn="uni/tn-Tenant-test/lDevVip-ASAv"/>

<vnsLIfCtx name="" descr="" connNameOrLbl="provider">
<vnsRsLIfCtxToBD tDn="uni/tn-Tenant-test/BD-BD2"/>
<vnsRsLIfCtxToLIf tDn="uni/tn-Tenant-test/lDevVip-ASAv/lIf-provider"/>
</vnsLIfCtx>

<vnsLIfCtx name="" descr="" connNameOrLbl="consumer">
<vnsRsLIfCtxToBD tDn="uni/tn-Tenant-test/BD-BD1"/>
<vnsRsLIfCtxToLIf tDn="uni/tn-Tenant-test/lDevVip-ASAv/lIf-consumer"/>
</vnsLIfCtx>
</vnsLDevCtx>
```

```
<vzBrCP ownerTag="" ownerKey="" name="Client-to-Web" descr="" targetDscp="unspecified"
scope="tenant" prio="unspecified">

<vzSubj name="Subject" descr="" targetDscp="unspecified" prio="unspecified" revFltPorts="yes"

provMatchT="AtleastOne" consMatchT="AtleastOne">
<vzRsSubjFiltAtt tnVzFilterName="default" directives=""/>
<vzRsSubjGraphAtt directives="" tnVnsAbsGraphName="FW-Graph"/>
</vzSubj>
</vzBrCP>

<vnsAbsGraph ownerTag="" ownerKey="" name="FW-Graph" descr="" uiTemplateType="UNSPECIFIED">
<vnsAbsTermNodeCon ownerTag="" ownerKey="" name="T1" descr="">
<vnsAbsTermConn ownerTag="" ownerKey="" name="1" descr="" attNotify="no"/>
<vnsInTerm name="" descr=""/>
<vnsOutTerm name="" descr=""/>
</vnsAbsTermNodeCon>

<vnsAbsTermNodeProv ownerTag="" ownerKey="" name="T2" descr="">
<vnsAbsTermConn ownerTag="" ownerKey="" name="1" descr="" attNotify="no"/>
<vnsInTerm name="" descr=""/>
<vnsOutTerm name="" descr=""/>
</vnsAbsTermNodeProv>

<vnsAbsConnection ownerTag="" ownerKey="" name="C1" descr="" unicastRoute="yes"
directConnect="no" connType="external" connDir="provider" adjType="L2">
<vnsRsAbsConnectionConns
tDn="uni/tn-Tenant-test/AbsGraph-FW-Graph/AbsNode-N1/AbsFConn-consumer"/>
<vnsRsAbsConnectionConns
tDn="uni/tn-Tenant-test/AbsGraph-FW-Graph/AbsTermNodeCon-T1/AbsTConn"/>
</vnsAbsConnection>

<vnsAbsConnection ownerTag="" ownerKey="" name="C2" descr="" unicastRoute="yes"
directConnect="no" connType="external" connDir="provider" adjType="L2">
<vnsRsAbsConnectionConns
tDn="uni/tn-Tenant-test/AbsGraph-FW-Graph/AbsNode-N1/AbsFConn-provider"/>
<vnsRsAbsConnectionConns
tDn="uni/tn-Tenant-test/AbsGraph-FW-Graph/AbsTermNodeProv-T2/AbsTConn"/>
</vnsAbsConnection>

<vnsAbsNode ownerTag="" ownerKey="" name="N1" descr="" shareEncap="no" sequenceNumber="0"
routingMode="unspecified" managed="yes" isCopy="no" funcType="GoTo"
funcTemplateType="FW_ROUTED">

<vnsAbsFuncConn ownerTag="" ownerKey="" name="consumer" descr="" attNotify="no">
<vnsRsMConnAtt tDn="uni/infra/mDev-CISCO-ASA-1.2/mFunc-Firewall/mConn-external"/>
</vnsAbsFuncConn>

<vnsAbsFuncConn ownerTag="" ownerKey="" name="provider" descr="" attNotify="no">
<vnsRsMConnAtt tDn="uni/infra/mDev-CISCO-ASA-1.2/mFunc-Firewall/mConn-internal"/>
</vnsAbsFuncConn>
<vnsRsNodeToAbsFuncProf
tDn="uni/infra/mDev-CISCO-ASA-1.2/absFuncProfContr/absFuncProfGrp-WebServiceProfileGroup/absFuncProf-WebPolicyForRoutedMode"/>
<vnsRsNodeToLDev tDn="uni/tn-Tenant-test/lDevVip-ASAv"/>
<vnsRsNodeToMFunc tDn="uni/infra/mDev-CISCO-ASA-1.2/mFunc-Firewall"/>
</vnsAbsNode>
</vnsAbsGraph>

<fvBD ownerTag="" ownerKey="" name="BD1" descr="" unicastRoute="yes" vmac="not-applicable"

unkMcastAct="flood" unkMacUcastAct="proxy" type="regular" multiDstPktAct="bd-flood"
mcastAllow="no"
mac="00:22:BD:F8:19:FF" llAddr="::" limitIpLearnToSubnets="no" ipLearning="yes"
epMoveDetectMode="" arpFlood="no">
```

```
<fvRsBDToNdP tnNdIfPolName=""/>
<fvRsCtx tnFvCtxName="VRF1"/>
<fvRsIgmpsn tnIgmpSnoopPolName=""/>
<fvRsBdToEpRet tnFvEpRetPolName="" resolveAct="resolve"/>
</fvBD>

<fvBD ownerTag="" ownerKey="" name="BD2" descr="" unicastRoute="yes" vmac="not-applicable"

unkMcastAct="flood" unkMacUcastAct="proxy" type="regular" multiDstPktAct="bd-flood"
mcastAllow="no"
mac="00:22:BD:F8:19:FF" llAddr="::" limitIpLearnToSubnets="no" ipLearning="yes"
epMoveDetectMode="" arpFlood="no">
<fvRsBDToNdP tnNdIfPolName=""/>
<fvRsCtx tnFvCtxName="VRF1"/>
<fvRsIgmpsn tnIgmpSnoopPolName=""/>
<fvRsBdToEpRet tnFvEpRetPolName="" resolveAct="resolve"/>
</fvBD>

<fvCtx ownerTag="" ownerKey="" name="VRF1" descr="" pcEnfPref="enforced" pcEnfDir="ingress"

knwMcastAct="permit">
<fvRsBgpCtxPol tnBgpCtxPolName=""/>
<fvRsCtxToExtRouteTagPol tnL3extRouteTagPolName=""/>
<fvRsOspfCtxPol tnOspfCtxPolName=""/>
<vzAny name="" descr="" matchT="AtleastOne"/>
<fvRsCtxToEpRet tnFvEpRetPolName=""/>
</fvCtx>
<vnsSvcCont/>

<fvAp ownerTag="" ownerKey="" name="ANP" descr="" prio="unspecified">

<fvAEPg name="Web" descr="" prio="unspecified" pcEnfPref="unenforced"
matchT="AtleastOne" isAttrBasedEPg="no" fwdCtrl="">
<fvRsDomAtt tDn="uni/vmmp-VMware/dom-ACI_vDS" resImedcy="lazy" primaryEncap="unknown"
instrImedcy="lazy" encap="unknown" delimiter="" classPref="encap"/>
<fvRsBd tnFvBDName="BD2"/>
<fvRsCustQosPol tnQosCustomPolName=""/>

<vnsFolderInst name="internalIf" nodeNameOrLbl="N1" graphNameOrLbl="FW-Graph"
ctrctNameOrLbl="Client-to-Web"scopedBy="epg" locked="no" key="Interface"
devCtxLbl="" cardinality="unspecified">

<vnsFolderInst name="internalIfCfg" nodeNameOrLbl="N1" graphNameOrLbl="FW-Graph"
ctrctNameOrLbl="Client-to-Web" scopedBy="epg" locked="no" key="InterfaceConfig"
devCtxLbl="" cardinality="unspecified">
<vnsParamInst name="internal_security_level" locked="no" key="security_level"
cardinality="unspecified"
value="100" validation="" mandatory="no"/>
</vnsFolderInst>
</vnsFolderInst>

<vnsFolderInst name="externalIf" nodeNameOrLbl="N1" graphNameOrLbl="FW-Graph"
ctrctNameOrLbl="Client-to-Web" scopedBy="epg" locked="no" key="Interface"
devCtxLbl="" cardinality="unspecified">

<vnsFolderInst name="ExtAccessGroup" nodeNameOrLbl="N1" graphNameOrLbl="FW-Graph"
ctrctNameOrLbl="Client-to-Web" scopedBy="epg" locked="no" key="AccessGroup" devCtxLbl=""
cardinality="unspecified">
<vnsCfgRelInst name="name" locked="no" key="inbound_access_list_name"
cardinality="unspecified"
mandatory="no" targetName="access-list-inbound"/>
</vnsFolderInst>

<vnsFolderInst name="externalIfCfg" nodeNameOrLbl="N1" graphNameOrLbl="FW-Graph"
```

```
ctrctNameOrLbl="Client-to-Web" scopedBy="epg" locked="no" key="InterfaceConfig"
devCtxLbl="" cardinality="unspecified">
<vnsParamInst name="external_security_level" locked="no" key="security_level"
cardinality="unspecified" value="50" validation="" mandatory="no"/>
</vnsFolderInst>
</vnsFolderInst>

<vnsFolderInst name="IntConfig" nodeNameOrLbl="N1" graphNameOrLbl="FW-Graph"
ctrctNameOrLbl="Client-to-Web" scopedBy="epg" locked="no" key="InIntfConfigRelFolder"
devCtxLbl="" cardinality="unspecified">
<vnsCfgRelInst name="InConfigrel" locked="no" key="InIntfConfigRel" cardinality="unspecified"

mandatory="no" targetName="internalIf"/>
</vnsFolderInst>

<vnsFolderInst name="ExtConfig" nodeNameOrLbl="N1" graphNameOrLbl="FW-Graph"
ctrctNameOrLbl="Client-to-Web" scopedBy="epg" locked="no" key="ExIntfConfigRelFolder"
devCtxLbl="" cardinality="unspecified">
<vnsCfgRelInst name="ExtConfigrel" locked="no" key="ExIntfConfigRel" cardinality="unspecified"

mandatory="no" targetName="externalIf"/>
</vnsFolderInst>

<vnsFolderInst name="access-list-inbound" nodeNameOrLbl="N1" graphNameOrLbl="FW-Graph"
ctrctNameOrLbl="Client-to-Web" scopedBy="epg" locked="no" key="AccessList" devCtxLbl=""
cardinality="unspecified">

<vnsFolderInst name="permit-https" nodeNameOrLbl="N1" graphNameOrLbl="FW-Graph"
ctrctNameOrLbl="Client-to-Web" scopedBy="epg" locked="no" key="AccessControlEntry"
devCtxLbl="" cardinality="unspecified">
<vnsParamInst name="action-permit" locked="no" key="action" cardinality="unspecified"
value="permit" validation="" mandatory="no"/>
<vnsParamInst name="order1" locked="no" key="order" cardinality="unspecified" value="10"
validation="" mandatory="no"/>

<vnsFolderInst name="dest-service" nodeNameOrLbl="N1" graphNameOrLbl="FW-Graph"
ctrctNameOrLbl="Client-to-Web" scopedBy="epg" locked="no" key="destination_service"
devCtxLbl="" cardinality="unspecified">
<vnsParamInst name="op" locked="no" key="operator" cardinality="unspecified"
value="eq" validation="" mandatory="no"/>
<vnsParamInst name="port" locked="no" key="low_port" cardinality="unspecified"
value="https" validation="" mandatory="no"/>
</vnsFolderInst>
<vnsFolderInst name="dest-address" nodeNameOrLbl="N1" graphNameOrLbl="FW-Graph"
ctrctNameOrLbl="Client-to-Web" scopedBy="epg" locked="no" key="destination_address"
devCtxLbl="" cardinality="unspecified">
<vnsParamInst name="any" locked="no" key="any" cardinality="unspecified" value="any"
validation="" mandatory="no"/>
</vnsFolderInst>

<vnsFolderInst name="src-address" nodeNameOrLbl="N1" graphNameOrLbl="FW-Graph"
ctrctNameOrLbl="Client-to-Web" scopedBy="epg" locked="no" key="source_address"
devCtxLbl="" cardinality="unspecified">
<vnsParamInst name="any" locked="no" key="any" cardinality="unspecified" value="any"
validation="" mandatory="no"/>
</vnsFolderInst>

<vnsFolderInst name="tcp" nodeNameOrLbl="N1" graphNameOrLbl="FW-Graph"
ctrctNameOrLbl="Client-to-Web" scopedBy="epg" locked="no" key="protocol" devCtxLbl=""
cardinality="unspecified">
<vnsParamInst name="tcp" locked="no" key="name_number" cardinality="unspecified"
value="tcp" validation="" mandatory="no"/>
</vnsFolderInst>
</vnsFolderInst>
```

```
<vnsFolderInst name="permit-http" nodeNameOrLbl="N1" graphNameOrLbl="FW-Graph"
ctrctNameOrLbl="Client-to-Web" scopedBy="epg" locked="no" key="AccessControlEntry"
devCtxLbl="" cardinality="unspecified">
<vnsParamInst name="action-permit" locked="no" key="action" cardinality="unspecified"
value="permit" validation="" mandatory="no"/>
<vnsParamInst name="order1" locked="no" key="order" cardinality="unspecified" value="10"
validation="" mandatory="no"/>

<vnsFolderInst name="dest-service" nodeNameOrLbl="N1" graphNameOrLbl="FW-Graph"
ctrctNameOrLbl="Client-to-Web" scopedBy="epg" locked="no" key="destination_service"
devCtxLbl="" cardinality="unspecified">
<vnsParamInst name="op" locked="no" key="operator" cardinality="unspecified" value="eq"
validation="" mandatory="no"/>
<vnsParamInst name="port" locked="no" key="low_port" cardinality="unspecified" value="http"

validation="" mandatory="no"/>
</vnsFolderInst>

<vnsFolderInst name="dest-address" nodeNameOrLbl="N1" graphNameOrLbl="FW-Graph"
ctrctNameOrLbl="Client-to-Web" scopedBy="epg" locked="no" key="destination_address"
devCtxLbl=""
cardinality="unspecified">
<vnsParamInst name="any" locked="no" key="any" cardinality="unspecified" value="any"
validation="" mandatory="no"/>
</vnsFolderInst>

<vnsFolderInst name="src-address" nodeNameOrLbl="N1" graphNameOrLbl="FW-Graph"
ctrctNameOrLbl="Client-to-Web" scopedBy="epg" locked="no" key="source_address" devCtxLbl=""

cardinality="unspecified">
<vnsParamInst name="any" locked="no" key="any" cardinality="unspecified" value="any"
validation="" mandatory="no"/>
</vnsFolderInst>

<vnsFolderInst name="tcp" nodeNameOrLbl="N1" graphNameOrLbl="FW-Graph"
ctrctNameOrLbl="Client-to-Web" scopedBy="epg" locked="no" key="protocol" devCtxLbl=""
cardinality="unspecified">

<vnsParamInst name="tcp" locked="no" key="name_number" cardinality="unspecified" value="tcp"

validation="" mandatory="no"/>
</vnsFolderInst>
</vnsFolderInst>
</vnsFolderInst>
<fvRsProv prio="unspecified" matchT="AtleastOne" tnVzBrCPName="Client-to-Web"/>
</fvAEPg>

<fvAEPg name="Client" descr="" prio="unspecified" pcEnfPref="unenforced" matchT="AtleastOne"

isAttrBasedEPg="no" fwdCtrl="">
<fvRsCons prio="unspecified" tnVzBrCPName="Client-to-Web"/>
<fvRsDomAtt tDn="uni/vmmp-VMware/dom-ACI_vDS" resImedcy="lazy" primaryEncap="unknown"
instrImedcy="lazy" encap="unknown" delimiter="" classPref="encap"/>
<fvRsBd tnFvBDName="BD1"/>
<fvRsCustQosPol tnQosCustomPolName=""/>
</fvAEPg>
</fvAp>
<fvRsTenantMonPol tnMonEPGPolName=""/>

<vnsLDevVip name="ASAv" managed="yes" isCopy="no" funcType="GoTo" trunking="no"
svcType="FW" packageModel="ASAv" mode="legacy-Mode" devtype="VIRTUAL"
contextAware="single-Context">
<vnsCCred name="username" value="admin"/>
```

```
<vnsRsMDevAtt tDn="uni/infra/mDev-CISCO-ASA-1.2"/>
<vnsCCredSecret name="password"/>
<vnsCMgmt name="" port="443" host="172.31.184.249"/>
<vnsRsALDevToDomP tDn="uni/vmmp-VMware/dom-ACI_vDS"/>

<vnsCDev name="Device1" devCtxLbl="" vmName="ASAv-L3" vcenterName="vcenter">
<vnsCCred name="username" value="admin"/>
<vnsCCredSecret name="password"/>
<vnsCMgmt name="" port="443" host="172.31.184.249"/>
<vnsCIf name="GigabitEthernet0/1" vnicName="Network adapter 3"/>
<vnsCIf name="GigabitEthernet0/0" vnicName="Network adapter 2"/>
<vnsRsCDevToCtrlrP tDn="uni/vmmp-VMware/dom-ACI_vDS/ctrlr-vcenter"/>
</vnsCDev>

<vnsLIf name="provider" encap="unknown">
<vnsRsMetaIf tDn="uni/infra/mDev-CISCO-ASA-1.2/mIfLbl-internal" isConAndProv="no"/>
<vnsRsCIfAttN tDn="uni/tn-Tenant-test/lDevVip-ASAv/cDev-Device1/cIf-[GigabitEthernet0/1]"/>
</vnsLIf>

<vnsLIf name="consumer" encap="unknown">
<vnsRsMetaIf tDn="uni/infra/mDev-CISCO-ASA-1.2/mIfLbl-external" isConAndProv="no"/>
<vnsRsCIfAttN tDn="uni/tn-Tenant-test/lDevVip-ASAv/cDev-Device1/cIf-[GigabitEthernet0/0]"/>
</vnsLIf>
</vnsLDevVip>
</fvTenant>
```

Example: Configuring Layer 4 to Layer 7 Route Peering

Configuring Layer 4 to Layer 7 Route Peering With the REST API

These `l3extOut` policies specify the OSPF configurations needed to enable OSPF on the fabric leaf and are very similar to the `l3extOut` policies used for external communication.

The `l3extOut` policies also specify the prefix-based EPGs that control which routes are distributed in/out of the fabric. The `scope=import` attribute controls two things: which endpoint prefixes are learned; and directs the external L4-L7 device to advertise this route. The `scope=export` attribute specifies that the fabric has to advertise this route to the L4-L7 device.

Two sample `l3extOut` policies are shown below: **`OspfInternal`** deployed on **`eth1/23`**, and **`OspfExternal`** deployed on **`eth1/25`**.

Before you begin

Create one or more l3extOut external network connections and deploy them on the fabric leaf nodes where the service device is connected.

Step 1 To configure OspfInternal on eth1/23, send a post with XML similar to the following example:

Example:

```
<?xml version="1.0" encoding="UTF-8?>
<!-- /api/policymgr/mo.xml -->
<polUni>
   <fvTenant name="coke{{tenantId}}">
      {% if status is not defined %}
         {% set status = "created,modified" %}
```

```
      {% endif %}

      <l3extOut name="OspfInternal" status="{{status}}">

         <l3extLNodeP name="bLeaf-101">
             <l3extRsNodeL3OutAtt tDn="topology/pod-1/node-101" rtrId="180.0.0.11"/>
                <l3extLifP name='portIf''>
                   <l3extRsPathL3OutAtt tDn="topology/pod-1/paths-101/pathep-[eth1/23]"
                       ifInstT='ext-svi' encap='vlan-3844' addr="30.30.30.100/28" mtu='1500'/>
                       <!-- <ospfIfP authKey="tecom" authType="md5" authKeyId='1'> -->
                     <ospfIfP>
                         <ospfRsIfPol tnOspfIfPolName='ospfIfPol'/>
                     </ospfIfP>
                </l3extLIfP>
          </l3extLNodeP>

         <ospfExtP areaId='111' areaType='nssa' areaCtrl='redistribute'/>

          <l3extInstP name="OspfInternalInstP">
              <l3extSubnet ip="30.30.30.100/28" scope="import"/>
               <l3extSubnet ip="20.20.20.0/24" scope="import"/>
               <l3extSubnet ip="10.10.10.0/24" scope="export"/>
           </l3extInstP>

           <l3extRsEctx tnFvCtxName="cokectx1"/>

         </l3extOut>

      <ospfIfPol name="ospfIfPol" nwT='bcast' xmitDelay='1'
         helloIntvl='10' deadIntvl='40' status="created,modified"/>
      </fvTenant>
</polUni>
```

Step 2 To configure OspfExternal on eth1/25, send a post with XML similar to the following example:

Example:

```
<?xml version="1.0" encoding="UTF-8?>
<!-- /api/policymgr/mo.xml -->
   <polUni>
      <fvTenant name="common">
         <fvCtx name="commonctx"/>

         {% if status is not defined %}
            {% set status="created,modified" %}
         {% endif %}

        <l3extOut name=OspfExternal" status="{{status}}">
           <l3extLNodeP name="bLeaf-101">
               <l3extRsNodeL3OutAtt tDn="topology/pod-1/node-101" rtrId="180.0.0.8/28"/>
               <l3extLIfP name='portIf'>
               {% if intfType is not defined %}
                  {% set intfType="ext-svi" %}
               {% endif %}
               <l3extRsPathL3OutAtt tDn="topology/pod-1/paths-101/pathep-[eth1/25]"
                 ifInstT='ext-svi' encap='vlan-3843' addr="40.40.40.100/28" mtu='1500'/>
                <!-- ospfIfP authKey="tecom" authType="md5" authKeyId='1'> -->
                <ospfIfP>
                   <ospfRsIfPol tnOspfIfPolName='ospfIfPol'/>
                </ospfIfP>
           </l3extIfP>
       </l3extLNodeP>
```

```
        <ospfExtP areaId='111' areaType='nssa' areaCtrl='redistribute'/>

        <l3extInstP name="OspfExternalInstP">
           <l3extSubnet ip="40.40.40.100/28" scope="import"/>
           <l3extSubnet ip="10.10.10.0/24" scope="import"/>
           <l3extSubnet ip="20.20.20.0/24" scope="export"/>
        </l3extInstP>

         <l3extRsEctx tnfvCtxName="commonctx"/>

       </l3extOut>

       <ospfIfPol name="ospfIfPol" nwT='bcast' xmitDelay='1' helloIntvl='10' deadIntvl='40'
status="created,modified"/>
       </fvTenant>
    </polUni>
```

The `l3extInstP` object specifies that prefixes 40.40.40.100/28 and 10.10.10.0/24 are to be used for prefix based endpoint association and indicate that the L4-L7 device should advertise these routes.

The `l3extRsPathL3OutAtt` object specifies where each L3extOut is deployed.

Note For route peering to work, the `l3extRsPathL3OutAtt` must match the `RsCIfPathAtt` where the L4-L7 logical device cluster is connected.

Specifying an l3extOut Policy for Layer 4 to L7 Route Peering

A specific `l3extOut` policy can be used for a logical device cluster using its selection policy `vnsLIfCtx`. The `vnsRsLIfCtxToInstP` points the `LIfCtx` to the appropriate **`OspfInternal`** and **`OspfExternal`** `l3extInstP` EPGs. To configure an L3extOut policy used for Layer 4 to Layer 7 Route Peering, send a post with XML such as the following example:

```
<vnsLDevCtx ctrctNameOrLbl="webCtrct{{graphId}}" graphNameOrLbl="WebGraph" nodeNameOrLbl="FW">

    <vnsRsLDevCtxToLDev tDn="uni/tn-solar{{tenantId}}/lDevVip-Firewall"/>
    <vnsLIfCtx connNameOrLbl="internal">
        {% if L3ExtOutInternal is not defined %}
        <fvSubnet ip="10.10.10.10/24"/>
        {% endif %}
        <vnsRsLIfCtxToBD tDn="uni/tn-solar{{tenantId}}/BD-solarBD1"/>
        <vnsRsLIfCtxToLIf tDn="uni/tn-solar{{tenantId}}/lDevVip-Firewall/lIf-internal"/>
        {% if L3ExtOutInternal is defined %}
        <vnsRsLIfCtxToInstP
tDn="uni/tn-solar{{tenantId}}/out-OspfInternal/instP-OspfInternalInstP"
status={{L3ExtOutInternal}}"/>
        {% endif %}
    </vnsLIfCtx>
    <vnsLIfCtx connNameOrLbl="external">
        {% if L3ExtOutExternal is not defined %}
        <fvSubnet ip="40.40.40.40/24"/>
        {% endif %}
        <vnsRsLIfCtxToBD tDn="uni/tn-solar{{tenantId}}/BD-solarBD4"/>
        <vnsRsLIfCtxToLIf tDn="uni/tn-solar{{tenantId}}/lDevVip-Firewall/lIf-external"/>
        {% if L3ExtOutExternal is defined %}
        <vnsRsLIfCtxToInstP
tDn="uni/tn-solar{{tenantId}}/out-OspfExternal/instP-OspfExternalInstP"
status={{L3ExtOutExternal}}"/>
```

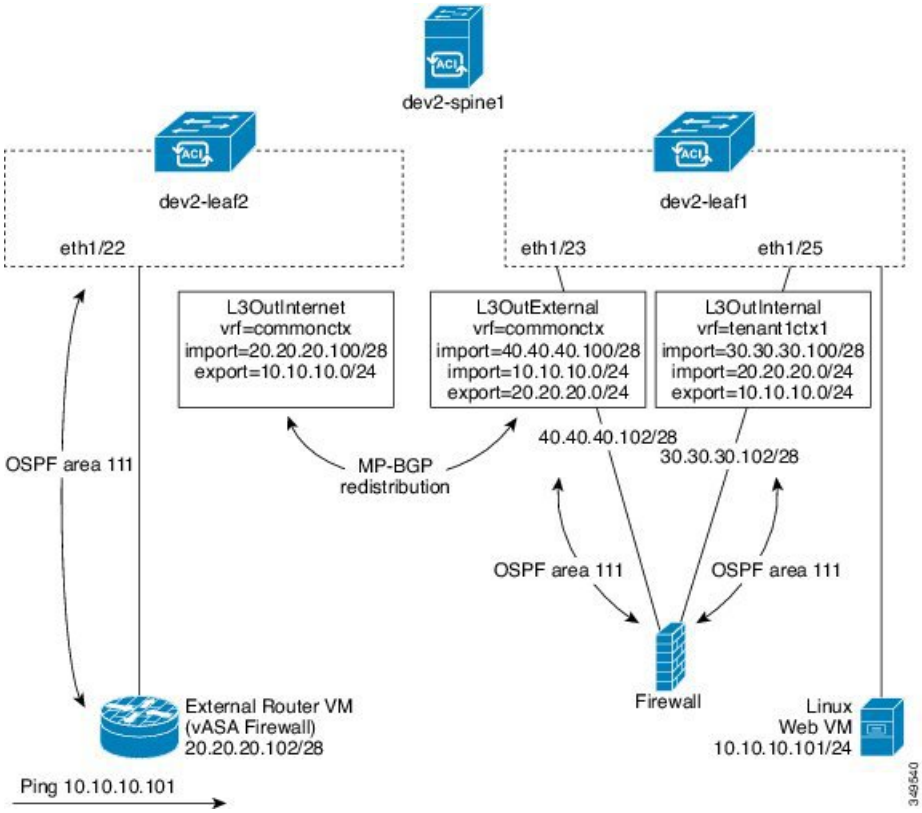
```
        {% endif %}
    </vnsLIfCtx>
</vnsLDevCtx>
```

The associated concrete device needs to have a `vnsRsCIfPathAtt` that deploys it to the same fabric leaf as shown in the following example:

```
<vnsCDev name="ASA">
    <vnsRsLDevCtxToLDev tDn="uni/tn-solar{{tenantId}}/lDevVip-Firewall"/>
    <vnsCIf name="Gig0/0">
        <vnsRsCIfPathAtt tDn="topology/pod-1/paths-101/pathep-[eht1/23]"/>
    </vnsCIf>
    <vnsCIf name="Gig0/1">
        <vnsRsCIfPathAtt tDn="topology/pod-1/paths-101/pathep-[eht1/25]"/>
    </vnsCIf>
    <vnsCMgmt name="devMgmt" host="{{asaIp}}" port="443" />
    <vnsCCred name="username" value="admin" />
    <vnsCCredSecret name="password" value="insieme" />
</vnsCDev>
```

The following figure shows how route peering works end-to-end.

Figure 2: Sample Deployment



In this 2-leaf, 1-spine topology, the linux web server is at IP 10.10.10.101/24 and is hosted on an ESX server connected to dev2-leaf1. A service graph is deployed consisting of a two-arm firewall that is also connected to dev2-leaf1. The service graph is associated with a contract that binds an external `l3extOut` **`L3OutInternet`** with the provider EPG (Web VM). Two internal `l3extOut` policies, an **`L3OutExternal`**, and an **`L3OutInteral`** are also deployed to the leaf ports where the service device is connected.