Route Control

This chapter contains the following sections:

Route Maps/Profiles with Explicit Prefix Lists

About Route Map/Profile

The route profile is a logical policy that defines an ordered set (rtctrlCtxP) of logical match action rules with associated set action rules. The route profile is the logical abstract of a route map. Multiple route profiles can be merged into a single route map. A route profile can be one of the following types:

- Match Prefix and Routing Policy: Pervasive subnets (fvSubnet) and external subnets (l3extSubnet) are combined with a route profile and merged into a single route map (or route map entry). Match Prefix and Routing Policy is the default value.
- Match Routing Policy Only: The route profile is the only source of information to generate a route map, and it will overwrite other policy attributes.

Note When explicit prefix list is used, the type of the route profile should be set to "match routing policy only".

After the match and set profiles are defined, the route map must be created in the Layer 3 Out. Route maps can be created using one of the following methods:

- Create a "default-export" route map for export route control, and a "default-import" route map for import route control.
- Create other route maps (not named default-export or default-import) and setup the relation from one or more l3extInstPs or subnets under the l3extInstP.
- In either case, match the route map on explicit prefix list by pointing to the rtctrlSubjP within the route map.

In the export and import route map, the set and match rules are grouped together along with the relative sequence across the groups (rtctrlCtxP). Additionally, under each group of match and set statements (rtctrlCtxP) the relation to one or more match profiles are available (rtctrlSubjP).

Any protocol enabled on Layer 3 Out (for example BGP protocol), will use the export and import route map for route filtering.

About Explicit Prefix List Support for Route Maps/Profile

In Cisco APIC, for public bridge domain (BD) subnets and external transit networks, inbound and outbound route controls are provided through an explicit prefix list. Inbound and outbound route control for Layer 3 Out is managed by the route map/profile (rtctrlProfile). The route map/profile policy supports a fully controllable prefix list for Layer 3 Out in the Cisco ACI fabric.

The subnets in the prefix list can represent the bridge domain public subnets or external networks. Explicit prefix list presents an alternate method and can be used instead of the following:

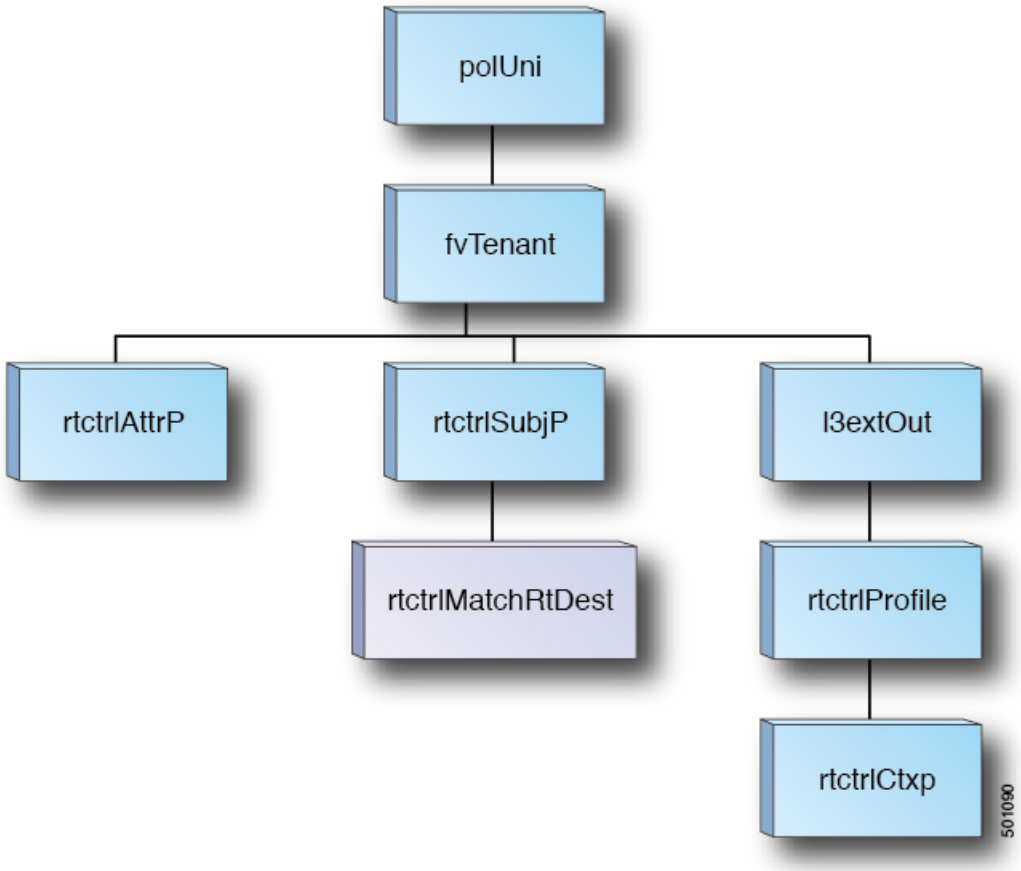
- Advertising BD subnets through BD to Layer 3 Out relation.

Note The subnet in the BD must be marked public for the subnet to be advertised out.

- Specifying a subnet in the l3extInstP with export/import route control for advertising transit and external networks.

Explicit prefix list is defined through a new match type that is called match route destination (rtctrlMatchRtDest). An example usage is provided in the API example that follows.

Figure 1: External Policy Model of API



Additional information about match rules, set rules when using explicit prefix list are as follows:

Match Rules

- Under the tenant (fvTenant), you can create match profiles (rtctrlSubjP) for route map filtering. Each match profile can contain one or more match rules. Match rule supports multiple match types. Prior to Cisco APIC release 2.1(x), match types supported were explicit prefix list and community list.

  Starting with Cisco APIC release 2.1(x), explicit prefix match or match route destination (rtctrlMatchRtDest) is supported.

  Match prefix list (rtctrlMatchRtDest) supports one or more subnets with an optional aggregate flag. Aggregate flags are used for allowing prefix matches with multiple masks starting with the mask mentioned in the configuration till the maximum mask allowed for the address family of the prefix . This is the equivalent of the "le " option in the prefix-list in NX-OS software (example, 10.0.0.0/8 le 32).

  The prefix list can be used for covering the following cases:

  - Allow all (0.0.0.0/0 with aggregate flag, equivalent of 0.0.0.0/0 le 32)
  - One or more of specific prefixes (example: 10.1.1.0/24)
  - One or more of prefixes with aggregate flag (example, equivalent of 10.1.1.0/24 le 32).

Note When a route map with a match prefix “0.0.0.0/0 with aggregate flag” is used under an L3Out EPG in the export direction, the rule is applied only for redistribution from dynamic routing protocols. Therefore, the rule is not applied to the following (in routing protocol such as OSPF or EIGRP):

- Bridge domain (BD) subnets
- Directly connected subnets on the border leaf switch
- Static routes defined on the L3Out

- The explicit prefix match rules can contain one or more subnets, and these subnets can be bridge domain public subnets or external networks. Subnets can also be aggregated up to the maximum subnet mask (/32 for IPv4 and /128 for IPv6).
- When multiple match rules of different types are present (such as match community and explicit prefix match), the match rule is allowed only when the match statements of all individual match types match. This is the equivalent of the AND filter. The explicit prefix match is contained by the subject profile (rtctrlSubjP) and will form a logical AND if other match rules are present under the subject profile.
- Within a given match type (such as match prefix list), at least one of the match rules statement must match. Multiple explicit prefix match (rtctrlMatchRtDest) can be defined under the same subject profile (rtctrlSubjP) which will form a logical OR.

Set Rules

- Set policies must be created to define set rules that are carried with the explicit prefixes such as set community, set tag.

Aggregation Support for Explicit Prefix List

Each prefix (rtctrlMatchRtDest) in the match prefixes list can be aggregated to support multiple subnets matching with one prefix list entry.

Aggregated Prefixes and BD Private Subnets

Although subnets in the explicit prefix list match may match the BD private subnets using aggregated or exact match, private subnets will not be advertised through the routing protocol using the explicit prefix list. The scope of the BD subnet must be set to "public" for the explicit prefix list feature to advertise the BD subnets.

Differences in Behavior for 0.0.0.0/0 with Aggregation

The 0.0.0.0/0 with Aggregate configuration creates an IP prefix-list equivalent to “0.0.0.0/0 le 32”. The 0.0.0.0/0 with Aggregate configuration can be used mainly in two situations:

- “Export Route Control Subnet” with “Aggregate Export” scope in L3Out subnet under the L3Out network (L3Out EPG)
- An explicit prefix-list (Match Prefix rule) assigned to a route map with the name “default-export”

When used with the “Export Route Control Subnet” scope under the L3Out subnet, the route map will only match routes learned from dynamic routing protocols. It will not match BD subnets or directly-connected networks.

When used with the explicit route map configuration, the route map will match all routes, including BD subnets and directly-connected networks.

Consider the following examples to get a better understanding of the expected and unexpected (inconsistent) behavior in the two situations described above.

Scenario 1

For the first scenario, we configure a route map (with a name of `rpm_with_catch_all`) using a configuration post similar to the following:

```
<l3extOut annotation="" descr="" dn="uni/tn-t9/out-L3-out" enforceRtctrl="export"
name="L3-out" nameAlias="" ownerKey="" ownerTag="" targetDscp="unspecified">
    <rtctrlProfile annotation="" descr="" name="rpm_with_catch_all" nameAlias="" ownerKey=""
 ownerTag="" type="combinable">
        <rtctrlCtxP action="permit" annotation="" descr="" name="catch_all" nameAlias=""
order="0">
            <rtctrlScope annotation="" descr="" name="" nameAlias="">
                <rtctrlRsScopeToAttrP annotation="" tnRtctrlAttrPName="set_metric_type"/>
            </rtctrlScope>
        </rtctrlCtxP>
    </rtctrlProfile>
    <ospfExtP annotation="" areaCost="1" areaCtrl="redistribute,summary" areaId="backbone"
 areaType="regular" descr="" multipodInternal="no" nameAlias=""/>
    <l3extRsEctx annotation="" tnFvCtxName="ctx0"/>
   <l3extLNodeP annotation="" configIssues="" descr="" name="leaf" nameAlias="" ownerKey=""
 ownerTag="" tag="yellow-green" targetDscp="unspecified">
       <l3extRsNodeL3OutAtt annotation="" configIssues="" rtrId="20.2.0.2" rtrIdLoopBack="no"
 tDn="topology/pod-1/node-104">
          <l3extLoopBackIfP addr="14.1.1.1/32" annotation="" descr="" name="" nameAlias=""/>

            <l3extInfraNodeP annotation="" descr="" fabricExtCtrlPeering="no"
fabricExtIntersiteCtrlPeering="no" name="" nameAlias="" spineRole=""/>
        </l3extRsNodeL3OutAtt>
        <l3extLIfP annotation="" descr="" name="interface" nameAlias="" ownerKey=""
ownerTag="" tag="yellow-green">
            <ospfIfP annotation="" authKeyId="1" authType="none" descr="" name=""
nameAlias="">
                <ospfRsIfPol annotation="" tnOspfIfPolName=""/>
            </ospfIfP>
            <l3extRsPathL3OutAtt addr="36.1.1.1/24" annotation="" autostate="disabled"
descr="" encap="vlan-3063" encapScope="local" ifInstT="ext-svi" ipv6Dad="enabled" llAddr="::"
 mac="00:22:BD:F8:19:FF" mode="regular" mtu="inherit"
tDn="topology/pod-1/paths-104/pathep-[accBndlGrp_104_pc13]" targetDscp="unspecified"/>
            <l3extRsNdIfPol annotation="" tnNdIfPolName=""/>
            <l3extRsIngressQosDppPol annotation="" tnQosDppPolName=""/>
            <l3extRsEgressQosDppPol annotation="" tnQosDppPolName=""/>
        </l3extLIfP>
    </l3extLNodeP>
    <l3extInstP annotation="" descr="" exceptionTag="" floodOnEncap="disabled"
matchT="AtleastOne" name="epg" nameAlias="" prefGrMemb="exclude" prio="unspecified"
targetDscp="unspecified">
        <l3extRsInstPToProfile annotation="" direction="export"
tnRtctrlProfileName="rpm_with_catch_all"/>
       <l3extSubnet aggregate="" annotation="" descr="" ip="0.0.0.0/0" name="" nameAlias=""
 scope="import-security"/>
        <fvRsCustQosPol annotation="" tnQosCustomPolName=""/>
    </l3extInstP>
```

```
</l3extOut>


<rtctrlAttrP annotation="" descr="" dn="uni/tn-t9/attr-set_metric_type" name="set_metric_type"
 nameAlias="">
   <rtctrlSetRtMetricType annotation="" descr="" metricType="ospf-type1" name="" nameAlias=""
 type="metric-type"/>
</rtctrlAttrP>


<rtctrlSubjP annotation="" descr="" dn="uni/tn-t9/subj-catch_all_ip" name="catch_all_ip"
nameAlias="">
    <rtctrlMatchRtDest aggregate="yes" annotation="" descr="" ip="0.0.0.0/0" name=""
nameAlias=""/>
</rtctrlSubjP>
```

With this route map, what we would expect with 0.0.0.0/0 is that all the routes would go with the property `metricType="ospf-type1"`, but only for the OSPF route.

In addition, we also have a subnet configured under a bridge domain (for example, 209.165.201.0/27), with a bridge domain to L3Out relation, using a route map with a pervasive subnet (fvSubnet) for a static route. However, even though the route map shown above is combinable, we do not want it applied for the subnet configured under the bridge domain, because we want 0.0.0.0/0 in the route map above to apply only for the transit route, not on the static route.

Following is the output for the `show route-map` and `show ip prefix-list` commands, where `exp-ctx-st-2555939` is the name of the outbound route map for the subnet configured under the bridge domain, and the name of the prefix list is provided within the output from the `show route-map` command:

```
leaf4# show route-map exp-ctx-st-2555939
route-map exp-ctx-st-2555939, deny, sequence 1
  Match clauses:
    tag: 4294967295
  Set clauses:
route-map exp-ctx-st-2555939, permit, sequence 15801
  Match clauses:
    ip address prefix-lists: IPv4-st16391-2555939-exc-int-inferred-export-dst
    ipv6 address prefix-lists: IPv6-deny-all
  Set clauses:

leaf4# show ip prefix-list IPv4-st16391-2555939-exc-int-inferred-export-dst
ip prefix-list IPv4-st16391-2555939-exc-int-inferred-export-dst: 1 entries
   seq 1 permit 209.165.201.0/27

leaf4#
```

In this situation, everything behaves as expected, because when the bridge domain subnet goes out, it is not applying the `rpm_with_catch_all` route map policies.

Scenario 2

For the second scenario, we configure a "default-export" route map for export route control, where an explicit prefix-list (Match Prefix rule) is assigned to the "default-export" route map, using a configuration post similar to the following:

```
<l3extOut annotation="" descr="" dn="uni/tn-t9/out-L3-out" enforceRtctrl="export"
name="L3-out" nameAlias="" ownerKey="" ownerTag="" targetDscp="unspecified">
    <rtctrlProfile annotation="" descr="" name="default-export" nameAlias="" ownerKey=""
ownerTag="" type="combinable">
        <rtctrlCtxP action="permit" annotation="" descr="" name="set-rule" nameAlias=""
order="0">
```

```
            <rtctrlScope annotation="" descr="" name="" nameAlias="">
                <rtctrlRsScopeToAttrP annotation="" tnRtctrlAttrPName="set_metric_type"/>
            </rtctrlScope>
        </rtctrlCtxP>
    </rtctrlProfile>
    <ospfExtP annotation="" areaCost="1" areaCtrl="redistribute,summary" areaId="backbone"
 areaType="regular" descr="" multipodInternal="no" nameAlias=""/>
    <l3extRsEctx annotation="" tnFvCtxName="ctx0"/>
   <l3extLNodeP annotation="" configIssues="" descr="" name="leaf" nameAlias="" ownerKey=""
 ownerTag="" tag="yellow-green" targetDscp="unspecified">
      <l3extRsNodeL3OutAtt annotation="" configIssues="" rtrId="20.2.0.2" rtrIdLoopBack="no"
 tDn="topology/pod-1/node-104">
          <l3extLoopBackIfP addr="14.1.1.1/32" annotation="" descr="" name="" nameAlias=""/>

            <l3extInfraNodeP annotation="" descr="" fabricExtCtrlPeering="no"
fabricExtIntersiteCtrlPeering="no" name="" nameAlias="" spineRole=""/>
        </l3extRsNodeL3OutAtt>
        <l3extLIfP annotation="" descr="" name="interface" nameAlias="" ownerKey=""
ownerTag="" tag="yellow-green">
            <ospfIfP annotation="" authKeyId="1" authType="none" descr="" name=""
nameAlias="">
                <ospfRsIfPol annotation="" tnOspfIfPolName=""/>
            </ospfIfP>
            <l3extRsPathL3OutAtt addr="36.1.1.1/24" annotation="" autostate="disabled"
descr="" encap="vlan-3063" encapScope="local" ifInstT="ext-svi" ipv6Dad="enabled" llAddr="::"
 mac="00:22:BD:F8:19:FF" mode="regular" mtu="inherit"
tDn="topology/pod-1/paths-104/pathep-[accBndlGrp_104_pc13]" targetDscp="unspecified"/>
            <l3extRsNdIfPol annotation="" tnNdIfPolName=""/>
            <l3extRsIngressQosDppPol annotation="" tnQosDppPolName=""/>
            <l3extRsEgressQosDppPol annotation="" tnQosDppPolName=""/>
        </l3extLIfP>
    </l3extLNodeP>
    <l3extInstP annotation="" descr="" exceptionTag="" floodOnEncap="disabled"
matchT="AtleastOne" name="epg" nameAlias="" prefGrMemb="exclude" prio="unspecified"
targetDscp="unspecified">
       <l3extSubnet aggregate="" annotation="" descr="" ip="0.0.0.0/0" name="" nameAlias=""
 scope="import-security"/>
        <fvRsCustQosPol annotation="" tnQosCustomPolName=""/>
    </l3extInstP>
</l3extOut>
```

Notice that this `default-export` route map has similar information as the `rpm_with_catch_all` route map, where the IP is set to 0.0.0.0/0 (`ip=0.0.0.0/0`), and the set rule in the `default-export` route map is configured only with the Set Metric Type (`tnRtctrlAttrPName=set_metric_type`).

Similar to the situation in the previous example, we also have the same subnet configured under the bridge domain, with a bridge domain to L3Out relation, as we did in the previous example.

However, following is the output in this scenario for the `show route-map` and `show ip prefix-list` commands:

```
leaf4# show route-map exp-ctx-st-2555939
route-map exp-ctx-st-2555939, deny, sequence 1
  Match clauses:
    tag: 4294967295
  Set clauses:
route-map exp-ctx-st-2555939, permit, sequence 8201
  Match clauses:
    ip address prefix-lists:
IPv4-st16391-2555939-exc-int-out-default-export2set-rule0pfx-only-dst
    ipv6 address prefix-lists: IPv6-deny-all
  Set clauses:
    metric-type type-1
```

```
leaf4# show ip prefix-list IPv4-st16391-2555939-exc-int-inferred-export-dst
% Policy IPv4-st16391-2555939-exc-int-inferred-export-dst not found
ifav82-leaf4# show ip prefix-list
IPv4-st16391-2555939-exc-int-out-default-export2set-rule0pfx-only-dst
ip prefix-list IPv4-st16391-2555939-exc-int-out-default-export2set-rule0pfx-only-dst: 1
entries
   seq 1 permit 209.165.201.0/27

leaf4#
```

Notice that in this situation, when the bridge domain subnet goes out, it is applying the `default-export` route map policies. In this situation, that route map matches all routes, including BD subnets and directly-connected networks. This is inconsistent behavior.

Guidelines and Limitations

- You must choose one of the following two methods to configure your route maps. If you use both methods, it will result in double entries and undefined route maps.
  - Add routes under the bridge domain (BD) and configure a BD to Layer 3 Outside relation
  - Configure the match prefix under rtctrlSubjP match profiles.
- Starting 2.3(x), **deny-static** implicit entry has been removed from Export Route Map. The user needs to configure explicitly the permit and deny entries required to control the export of static routes.
- Route-map per peer in an L3Oout is not supported. Route-map can only be applied on L3Out as a whole.

  Following are possible workarounds to this issue:
  - Block the prefix from being advertised from the other side of the neighbor.
  - Block the prefix on the route-map on the existing L3Out where you don't want to learn the prefix, and move the neighbor to another L3Out where you want to learn the prefix and create a separate route-map.
- Creating route-maps using a mixture of GUI and API commands is not supported. As a possible workaround, you can create a route-map different from the default route-map using the GUI, but the route-map created through the GUI on an L3Out cannot be applied to per-peer.

Configuring a Route Map/Profile with Explicit Prefix List Using the GUI

Before you begin

- Tenant and VRF must be configured.
- The VRF must be enabled on the leaf switch.

Procedure

Step 1 On the menu bar, click **Tenant**, and in the **Navigation** pane, expand **Tenant_name** > **Networking** > **External Routed Networks** > **Match Rules for Route Maps**.

Step 2 Right click **Match Rules for Route Maps**, and click **Create Match Rule for a Route Map**.

Step 3 In the **Create Match Rule for a Route Map** dialog box, enter a name for the rule and choose the desired community terms.

Step 4 In the **Create Match Rule** dialog box, expand **Match Prefix** and perform the following actions:

a) In the **IP** field, enter the explicit prefix list.

The explicit prefix can denote a BD subnet or an external network.

b) Check the **Aggregate** check box only if you desire an aggregate prefix. Click **Update**, and click **Submit**.

The match rule can have one or more of the match destination rules and one or more match community terms. Across the match types, the AND filter is supported, so all conditions in the match rule must match for the route match rule to be accepted. When there are multiple match prefixes in **Match Destination Rules**, the OR filter is supported. Any one match prefix is accepted as a route type if it matches.

Step 5 Under **External Routed Networks**, click and choose the available default layer 3 out.

If you desire another layer 3 out, you can choose that instead.

Step 6 Right-click **Route Maps/Profiles**, and click **Create Route Map/Profile**.

Step 7 In the **Create Route Map** dialog box, use a default route map, or enter a name for the desired route map.

For the purpose of this example, we use **default_export** route map.

Step 8 In the **Type** field, choose **Match Routing Policy Only**.

The Match Routing policy is the global RPC match destination route. The other option in this field is Match Prefix and Routing Policy which is the combinable RPC match destination route.

Step 9 Expand the + icon to display the **Create Route Control Context** dialog box.

Step 10 Enter a name for route control context, and choose the desired options for each field. To deny routes that match criteria defined in match rule (**Step 11**), select the action **deny**. The default action is **permit**.

Step 11 In the **Match Rule** field, choose the rule that was created earlier.

Step 12 In the **Set Rule** field, choose **Create Set Rules for a Route Map**.

Typically in the route map/profile you have a match and so the prefix list is allowed in and out, but in addition some attributes are being set for these routes, so that the routes with the attributes can be matched further.

Step 13 In the **Create Set Rules for a Route Map** dialog box, enter a name for the action rule and check the desired check boxes. Click **Submit**.

Step 14 In the **Create Route Control Context** dialog box, click **OK**. And in the **Create Route Map/Profile** dialog box, click **Submit**.

This completes the creation of the route map/profile. The route map is a combination of match action rules and set action rules. The route map is associated with export profile or import profile or redistribute profile as desired by the user. You can enable a protocol with the route map.

Configuring Route Map/Profile with Explicit Prefix List Using NX-OS Style CLI

Before you begin

- Tenant and VRF must be configured through the NX-OS CLI.
- The VRF must be enabled on the leaf switch through the NX-OS CLI.

Procedure

	Command or Action	Purpose
Step 1	**configure** **Example:** `apic1# configure`	Enters configuration mode.
Step 2	**leaf** *node-id* **Example:** `apic1(config)# leaf 101`	Specifies the leaf to be configured.
Step 3	**template route group** *group-name* **tenant** *tenant-name* **Example:** `apic1(config-leaf)# template route group g1 tenant exampleCorp`	Creates a route group template. **Note** The route group (match rule) can have one or more of the IP prefixes and one or more match community terms. Across the match types, the AND filter is supported, so all conditions in the route group must match for the route match rule to be accepted. When there are multiple IP prefixes in route group, the OR filter is supported. Any one match prefix is accepted as a route type if it matches.
Step 4	**ip prefix permit** *prefix/masklen* [**le**{**32 | 128** }] **Example:** `apic1(config-route-group)# ip prefix permit 15.15.15.0/24`	Add IP prefix to the route group. **Note** The IP prefix can denote a BD subnet or an external network. Use optional argument le 32 for IPv4 and le 128 for IPv6 if you desire an aggregate prefix.
Step 5	**community-list** [**standard | expanded**] *community-list-name expression* **Example:** `apic1(config-route-group)# community-list standard com1 65535:20`	This is an optional command. Add match criteria for community if community also needs to be matched along with IP prefix.

	Command or Action	Purpose

Step 6	**exit**
	Example:
	`apic1(config-route-group)# exit`
	`apic1(config-leaf)#`
		Exit template mode.

Step 7	**vrf context tenant** *tenant-name* **vrf** *vrf-name* [**l3out** {**BGP | EIGRP | OSPF | STATIC** }] **Example:** `apic1(config-leaf)# vrf context tenant exampleCorp vrf v1`	Enters a tenant VRF mode for the node. **Note** If you enter the optional **l3out** string, the L3Out must be an L3Out that you configured through the NX-OS CLI.
Step 8	**template route-profile** *profile-name* [*route-control-context-name order-value*] **Example:** `apic1(config-leaf-vrf)# template route-profile rp1 ctxl 1`	Creates a template containing set actions that should be applied to the matched routes.
Step 9	**set** *attribute value* **Example:** `apic1(config-leaf-vrf-template-route-profile)# set metric 128`	Add desired attributes (set actions) to the template.

Step 10	**exit**
	Example:
	`apic1(config-leaf-vrf-template-route-profile)# exit`
	`apic1(config-leaf-vrf)#`
		Exit template mode.

Step 11	**route-map** *map-name* **Example:** `apic1(config-leaf-vrf)# route-map bgpMap`	Create a route-map and enter the route-map configuration mode.
Step 12	**match route group** *group-name* [**order** *number*] [**deny**] **Example:** `apic1(config-leaf-vrf-route-map)# match route group g1 order 1`	Match a route group that has already been created, and enter the match mode to configure the route- profile. Additionally choose the keyword **Deny** if routes matching the match criteria defined in route group needs to be denied. The default is **Permit**.
Step 13	**inherit route-profile** *profile-name* **Example:** `apic1(config-leaf-vrf-route-map-match)# inherit route-profile rp1`	Inherit a route-profile (set actions). **Note** These actions will be applied to the matched routes. Alternatively, the set actions can be configured inline instead of inheriting a route-profile.

	Command or Action	Purpose

Step 14	**exit**
	Example:
	`apic1(config-leaf-vrf-route-map-match)# exit`
	`apic1(config-leaf-vrf-route-map)#`
		Exit match mode.

Step 15	**exit**
	Example:
	`apic1(config-leaf-vrf-route-map)# exit`
	`apic1(config-leaf-vrf)#`
		Exit route map configuration mode.

Step 16	**exit**
	Example:
	`apic1(config-leaf-vrf)# exit`
	`apic1(config-leaf)#`
		Exit VRF configuration mode.

Step 17	**router bgp** *fabric-asn* **Example:** `apic1(config-leaf)# router bgp 100`	Configure the leaf node.
Step 18	**vrf member tenant** *t1* **vrf** *v1* **Example:** `apic1(config-leaf-bgp)# vrf member tenant t1 vrf v1`	Set the BGP's VRF membership and the tenant for the BGP policy.
Step 19	**neighbor** *IP-address-of-neighbor* **Example:** `apic1(config-leaf-bgp-vrf)# neighbor 15.15.15.2`	Configure a BGP neighbor.
Step 20	**route-map** *map-name* {**in** | **out** } **Example:** `apic1(config-leaf-bgp-vrf-neighbor)# route-map bgpMap out`	Configure the route map for a BGP neighbor.

Configuring Route Map/Profile with Explicit Prefix List Using REST API

Before you begin

- Tenant and VRF must be configured.

Procedure

Configure the route map/profile using explicit prefix list.

Example:

```
<?xml version="1.0" encoding="UTF-8"?>
<fvTenant name="PM" status="">
   <rtctrlAttrP name="set_dest">
      <rtctrlSetComm community="regular:as2-nn2:5:24" />
   </rtctrlAttrP>
   <rtctrlSubjP name="allow_dest">
      <rtctrlMatchRtDest ip="192.169.0.0/24" />
      <rtctrlMatchCommTerm name="term1">
         <rtctrlMatchCommFactor community="regular:as2-nn2:5:24" status="" />
         <rtctrlMatchCommFactor community="regular:as2-nn2:5:25" status="" />
      </rtctrlMatchCommTerm>
      <rtctrlMatchCommRegexTerm commType="regular" regex="200:*" status="" />
   </rtctrlSubjP>
   <rtctrlSubjP name="deny_dest">
      <rtctrlMatchRtDest ip="192.168.0.0/24" />
   </rtctrlSubjP>
   <fvCtx name="ctx" />
   <l3extOut name="L3Out_1" enforceRtctrl="import,export" status="">
      <l3extRsEctx tnFvCtxName="ctx" />
      <l3extLNodeP name="bLeaf">
         <l3extRsNodeL3OutAtt tDn="topology/pod-1/node-101" rtrId="1.2.3.4" />
         <l3extLIfP name="portIf">
            <l3extRsPathL3OutAtt tDn="topology/pod-1/paths-101/pathep-[eth1/25]"
ifInstT="sub-interface" encap="vlan-1503" addr="10.11.12.11/24" />
            <ospfIfP />
         </l3extLIfP>
         <bgpPeerP addr="5.16.57.18/32" ctrl="send-com" />
         <bgpPeerP addr="6.16.57.18/32" ctrl="send-com" />
      </l3extLNodeP>
      <bgpExtP />
      <ospfExtP areaId="0.0.0.59" areaType="nssa" status="" />
      <l3extInstP name="l3extInstP_1" status="">
         <l3extSubnet ip="17.11.1.11/24" scope="import-security" />
      </l3extInstP>
      <rtctrlProfile name="default-export" type="global" status="">
         <rtctrlCtxP name="ctx_deny" action="deny" order="1">
            <rtctrlRsCtxPToSubjP tnRtctrlSubjPName="deny_dest" status="" />
         </rtctrlCtxP>
         <rtctrlCtxP name="ctx_allow" order="2">
            <rtctrlRsCtxPToSubjP tnRtctrlSubjPName="allow_dest" status="" />
         </rtctrlCtxP>
         <rtctrlScope name="scope" status="">
            <rtctrlRsScopeToAttrP tnRtctrlAttrPName="set_dest" status="" />
         </rtctrlScope>
      </rtctrlProfile>
   </l3extOut>
   <fvBD name="testBD">
      <fvRsBDToOut tnL3extOutName="L3Out_1" />
      <fvRsCtx tnFvCtxName="ctx" />
      <fvSubnet ip="40.1.1.12/24" scope="public" />
      <fvSubnet ip="40.1.1.2/24" scope="private" />
      <fvSubnet ip="2003::4/64" scope="public" />
   </fvBD>
</fvTenant>
```

Routing Control Protocols

About Configuring a Routing Control Protocol Using Import and Export Controls

This topic provides a typical example that shows how to configure a routing control protocol using import and export controls. It assumes that you have configured Layer 3 outside network connections with BGP. You can also perform these tasks for a Layer 3 outside network configured with OSPF.

Note Cisco ACI does not support IP fragmentation. Therefore, when you configure Layer 3 Outside (L3Out) connections to external routers, or Multi-Pod connections through an Inter-Pod Network (IPN), it is recommended that the interface MTU is set appropriately on both ends of a link. On some platforms, such as Cisco ACI, Cisco NX-OS, and Cisco IOS, the configurable MTU value does not take into account the Ethernet headers (matching IP MTU, and excluding the 14-18 Ethernet header size), while other platforms, such as IOS-XR, include the Ethernet header in the configured MTU value. A configured value of 9000 results in a max IP packet size of 9000 bytes in Cisco ACI, Cisco NX-OS, and Cisco IOS, but results in a max IP packet size of 8986 bytes for an IOS-XR untagged interface.

For the appropriate MTU values for each platform, see the relevant configuration guides.

We highly recommend that you test the MTU using CLI-based commands. For example, on the Cisco NX-OS CLI, use a command such as `ping 1.1.1.1 df-bit packet-size 9000 source-interface ethernet 1/1`.

Configuring a Route Control Protocol to Use Import and Export Controls, With the GUI

This example assumes that you have configured the Layer 3 outside network connections using BGP. It is also possible to perform these tasks for a network configured using OSPF.

This task lists steps to create import and export policies. By default, import controls are not enforced, so the import control must be manually assigned.

Before you begin

- The tenant, private network, and bridge domain are created.
- The Layer 3 outside for tenant networks is created.

Procedure

Step 1 On the menu bar, click **TENANTS** > ***Tenant_name*** > **Networking** > **External Routed Networks** > ***Layer3_Outside_name*** .

Step 2 Right click *Layer3_Outside_name* and click **Create Route Map**.

Step 3 In the **Create Route Map** dialog box, perform the following actions:

a) From the **Name** field drop-down list, choose the appropriate route profile.

Depending on your selection, whatever is advertised on the specific outside is automatically used.

b) In the **Type** field, choose **Match Prefix AND Routing Policy**.

c) Expand **Order**.

Step 4 In the **Create Route Control Context** dialog box, perform the following actions:

a) In the **Order** field, choose the desired order number.

b) In the **Name** field, enter a name for the route control private network.

c) From the **Match Rule** field drop-down list, click **Create Match Rule For a Route Map**.

d) In the **Create Match Rule** dialog box, in the **Name** field, enter a route match rule name. Click **Submit**.

Specify the match community regular expression term and match community terms as desired. Match community factors will require you to specify the name, community and scope.

e) From the **Set Attribute** drop-down list, choose **Create Set Rules For a Route Map**.

f) In the **Create Set Rules For a Route Map** dialog box, in the **Name** field, enter a name for the rule.

g) Check the check boxes for the desired rules you want to set, and choose the appropriate values that are displayed for the choices. Click **Submit**.
The policy is created and associated with the action rule.

h) Click **OK**.

i) In the **Create Route Map** dialog box, click **Submit**.

Step 5 In the **Navigation** pane, choose **Route Profile** > ***route_profile_name*** > ***route_control_private_network_name***.
In the **Work** pane, under **Properties** the route profile policy and the associated action rule name are displayed.

Step 6 In the **Navigation** pane, click the ***Layer3_Outside_name***.
In the **Work** pane, the **Properties** are displayed.

Step 7 (Optional) Click the **Route Control Enforcement** field and check the **Import** check box to enable the import policy.

The import control policy is not enabled by default but can be enabled by the user. The import control policy is supported for BGP and OSPF, but not for EIGRP. If the user enables the import control policy for an unsupported protocol, it will be automatically ignored. The export control policy is supported for BGP, EIGRP, and OSPF.

Note If BGP is established over OSPF, then the import control policy is applied only for BGP and ignored for OSPF.

Step 8 To create a customized export policy, right-click **Route Map/Profiles**, click **Create Route Map**, and perform the following actions:

a) In the **Create Route Map** dialog box, from the drop-down list in the **Name** field, choose or enter a name for the export policy.

b) Expand the + sign in the dialog box.

c) In the **Create Route Control Context** dialog box, in the **Order** field, choose a value.

d) In the **Name** field, enter a name for the route control private network.

e) (Optional) From the **Match Rule** field drop-down list, choose **Create Match Rule For a Route Map**, and create and attach a match rule policy if desired.

f) From the **Set Attribute** field drop-down list, choose **Create Set Rules For a Route Map** and click **OK**.
Alternatively, if desired, you can choose an existing set action, and click **OK**

g) In the **Create Set Rules For A Route Map** dialog box, in the **Name** field, enter a name.

h) Check the check boxes for the desired rules you want to set, and choose the appropriate values that are displayed for the choices. Click **Submit**.

In the **Create Route Control Context** dialog box, the policy is created and associated with the action rule.

i) Click **OK**.

j) In the **Create Route Map** dialog box, click **Submit**.

In the **Work** pane, the export policy is displayed.

Note To enable the export policy, it must first be applied. For the purpose of this example, it is applied to all the subnets under the network.

Step 9 In the **Navigation** pane, expand **External Routed Networks** > ***External_Routed_Network_name*** > **Networks** > ***Network_name***, and perform the following actions:

a) Expand **Route Control Profile**.

b) In the **Name** field drop-down list, choose the policy created earlier.

c) In the **Direction** field drop-down list, choose **Route Control Profile**. Click **Update**.

Configuring a Route Control Protocol to Use Import and Export Controls, With the NX-OS Style CLI

This example assumes that you have configured the Layer 3 outside network connections using BGP. It is also possible to perform these tasks for a network configured using OSPF.

This section describes how to create a route map using the NX-OS CLI:

Before you begin

- The tenant, private network, and bridge domain are created.
- The Layer 3 outside tenant network is configured.

Procedure

Step 1 Import Route control using match community, match prefix-list

Example:

```
apic1# configure
apic1(config)# leaf 101
    # Create community-list
apic1(config-leaf)# template community-list standard CL_1 65536:20 tenant exampleCorp
apic1(config-leaf)# vrf context tenant exampleCorp vrf v1

     #Create Route-map and use it for BGP import control.
apic1(config-leaf-vrf)# route-map bgpMap
    # Match prefix-list and set route-profile actions for the match.
apic1(config-leaf-vrf-route-map)# ip prefix-list list1 permit 13.13.13.0/24
apic1(config-leaf-vrf-route-map)# ip prefix-list list1 permit 14.14.14.0/24
apic1(config-leaf-vrf-route-map)# match prefix-list list1
apic1(config-leaf-vrf-route-map-match)# set tag 200
apic1(config-leaf-vrf-route-map-match)# set local-preference 64
apic1(config-leaf)# router bgp 100
apic1(config-bgp)# vrf member tenant exampleCorp vrf v1
```

```
apic1(config-leaf-bgp-vrf)# neighbor 3.3.3.3
apic1(config-leaf-bgp-vrf-neighbor)# route-map bgpMap in
```

Step 2 Export Route Control using match BD, default-export route-profile

Example:

```
# Create custom and "default-export" route-profiles
apic1(config)# leaf 101
apic1(config-leaf)# vrf context tenant exampleCorp vrf v1
apic1(config-leaf-vrf)# template route-profile default-export
apic1(config-leaf-vrf-template-route-profile)# set metric 256
apic1(config-leaf-vrf)# template route-profile bd-rtctrl
apic1(config-leaf-vrf-template-route-profile)# set metric 128

#Create a Route-map and match on BD, prefix-list
apic1(config-leaf-vrf)# route-map bgpMap
apic1(config-leaf-vrf-route-map)# match bridge-domain bd1
apic1(config-leaf-vrf-route-map-match)#exit
apic1(config-leaf-vrf-route-map)# match prefix-list p1
apic1(config-leaf-vrf-route-map-match)#exit
apic1(config-leaf-vrf-route-map)# match bridge-domain bd2
apic1(config-leaf-vrf-route-map-match)# inherit route-profile bd-rtctrl
```

Note In this case, public-subnets from bd1 and prefixes matching prefix-list p1 are exported out using route-profile “default-export”, while public-subnets from bd2 are exported out using route-profile “bd-rtctrl”.

Configuring a Route Control Protocol to Use Import and Export Controls, With the REST API

This example assumes that you have configured the Layer 3 outside network connections using BGP. It is also possible to perform these tasks for a network using OSPF.

Before you begin

- The tenant, private network, and bridge domain are created.
- The Layer 3 outside tenant network is configured.

Procedure

Configure the route control protocol using import and export controls.

Example:

```
<l3extOut descr="" dn="uni/tn-Ten_ND/out-L3Out1" enforceRtctrl="export" name="L3Out1"
ownerKey="" ownerTag="" targetDscp="unspecified">
        <l3extLNodeP descr="" name="LNodeP1" ownerKey="" ownerTag="" tag="yellow-green"
targetDscp="unspecified">
            <l3extRsNodeL3OutAtt rtrId="1.2.3.4" rtrIdLoopBack="yes"
```

```
tDn="topology/pod-1/node-101">
                <l3extLoopBackIfP addr="2000::3" descr="" name=""/>
            </l3extRsNodeL3OutAtt>
            <l3extLIfP descr="" name="IFP1" ownerKey="" ownerTag="" tag="yellow-green">
                <ospfIfP authKeyId="1" authType="none" descr="" name="">
                    <ospfRsIfPol tnOspfIfPolName=""/>
                </ospfIfP>
                <l3extRsNdIfPol tnNdIfPolName=""/>
                <l3extRsPathL3OutAtt addr="10.11.12.10/24" descr="" encap="unknown"
ifInstT="l3-port"
llAddr="::" mac="00:22:BD:F8:19:FF" mtu="1500" tDn="topology/pod-1/paths-101/pathep-[eth1/17]"
 targetDscp="unspecified"/>
            </l3extLIfP>
        </l3extLNodeP>
        <l3extRsEctx tnFvCtxName="PVN1"/>
        <l3extInstP descr="" matchT="AtleastOne" name="InstP1" prio="unspecified"
targetDscp="unspecified">
            <fvRsCustQosPol tnQosCustomPolName=""/>
            <l3extSubnet aggregate="" descr="" ip="192.168.1.0/24" name="" scope=""/>
        </l3extInstP>
        <ospfExtP areaCost="1" areaCtrl="redistribute,summary" areaId="0.0.0.1"
areaType="nssa" descr=""/>
        <rtctrlProfile descr="" name="default-export" ownerKey="" ownerTag="">
            <rtctrlCtxP descr="" name="routecontrolpvtnw" order="3">
                <rtctrlScope descr="" name="">
                    <rtctrlRsScopeToAttrP tnRtctrlAttrPName="actionruleprofile2"/>
                </rtctrlScope>
            </rtctrlCtxP>
        </rtctrlProfile>
    </l3extOut>
```