Inter-VRF Traffic Inspection in Smart Switches

Inter-VRF traffic inspection

Inter-VRF routing involves routing traffic between different VRFs by leaking routes from the source VRF to the target VRF through routing protocols. Traffic entering the target VRF is sent to the Data Processing Unit (DPU) for firewall inspection applicable to both the target and source VRFs, and then it egresses out of the source VRF. This process ensures security inspection while enabling cross-VRF communication.

Asymmetric configuration of VRFs are not supported. For example, if source VRF is a service VRF and destination VRF is a non-service VRF or if source VRF is a non-service VRF and destination VRF is a service VRF.

A route can only have a single VRF redirect destination. ECMP (Equal-Cost Multi-Path) of VRF redirect destinations is not supported.

A route cannot have a next hop in the local VRF and a VRF redirect destination simultaneously. The URIB or Client filters out and sends only the best path.

When the route lookup result is the egress VRF, for example, VRF B, this involves a second route lookup in the VRF, for example, VRF B, in the NPU, which leads to reduced bandwidth.

Only Global Consistency Checker (CC) and troubleshoot commands are supported. Single route CC is not supported for leaked routes. For more information, refer to Cisco Smart Switches Troubleshooting Guide for DPU Security Mode.

Configure Inter-VRF traffic inspection

Summary

Perform these procedures to configure inter-VRF traffic inspection:

Workflow

1. Assign source and target VRFs to service system hypershield. For more information, refer to Configure service VRFs.
2. Configure IP prefix lists and route maps for IPv4 and IPv6 filtering

3. Configure BGP Route Redistribution in VRF
4. Configure VRF Route Import and Export with Route-Targets and Route-Map

   For more information about configuring BGP route leaking, refer to Cisco Nexus 9000 Series NX-OS Unicast Routing Configuration Guide, Release 10.6(x).

Configure IP prefix lists and route maps for IPv4 and IPv6 filtering

Use this procedure to configure prefix lists to filter IPv4 and IPv6 prefixes and associate them with route maps to control route advertisement or filtering policies on the device. Configuring prefix list and route-maps helps in filtering the IP addresses that need to be leaked.

Procedure

Step 1 Use the **configure terminal** command to enter the global configuration mode.

Example:

```
switch# configure terminal
```

Step 2 Perform step a or b to create an IPv4 or IPv6 prefix list to permit all prefixes with prefix length up to 32 for IPv4 and up to 128 for IPv6 respectively. The prefix-list name can be any alphanumeric string up to 63 characters.

a) Use the **ip prefix-list** *prefix-list-name* **seq** *sequence-number* **permit** *ipv4-prefix* **le** *max-prefix-length* command for IPv4.

Example:

```
switch(config)# ip prefix-list test seq 5 permit 0.0.0.0/0 le 32
```

b) Use the **ipv6 prefix-list** *prefix-list-name* **seq** *sequence-number* **permit** *ipv6-prefix* **le** *max-prefix-length* command for IPv6.

Example:

```
switch(config)# ipv6 prefix-list testv6 seq 5 permit 0::/0 le 128
```

Step 3 Use the **route-map** *route-map name* **permit** *sequence number* command to define an IPv4 or IPv6 route-map with a permit statement sequence.

Example:

```
switch(config)# route-map test permit 10
```

Example:

```
switch(config)# route-map testv6 permit 10
```

Step 4 Use the **match ip address prefix-list** *prefix-list-name* command to match IP route against the prefix list within the route map for IPv4 or IPv6.

Example:

```
switch(config)# match ip address prefix-list test
```

Example:

```
switch(config)# match ipv6 address prefix-list testv6
```

Configure BGP route redistribution in VRF

This configuration enables BGP, enters the BGP routing process for autonomous system number (AS) 100, and configures redistribution of multiple routing protocols and sources into BGP within the VRF named red. The route-map statements allow filtering or policy control of redistributed routes. The examples cover both IPv4 and IPv6 address families under the VRF context.

Procedure

Step 1 Use the **configure terminal** command to enter the global configuration mode.

Example:

```
switch# configure terminal
```

Step 2 Enable **feature BGP**.

Example:

```
switch(config)# feature BGP
```

Step 3 Enter BGP router configuration mode with AS number 100 using the **router bgp 100** command.

Example:

```
switch(config)# router bgp 100
```

Step 4 Configure IPv4 or IPv6 unicast address family for VRF **red**. Perform step a or b for IPv4 or IPv6 respectively.

a) Use the **address-family ipv4 unicast vrf** *vrf-name* command for IPv4.

Example:

```
switch(config)# address-family ipv4 unicast vrf red
```

b) Use the **address-family ipv6 unicast vrf** *vrf-name* command for IPv6.

Example:

```
switch(config)# address-family ipv6 unicast vrf red
```

Step 5 Redistribute routes in VRF red from various clients within BGP. Perform step a or b for IPv4 or IPv6 respectively.

a) For IPv4, redistribute Adjacency Manager (am), OSPF process 100, directly connected, or static routes with route-map.

Example:

```
switch(config)# redistribute am route-map test
switch(config)# redistribute ospf 100 route-map test
switch(config)# redistribute direct route-map test
switch(config)# redistribute static route-map test
```

b) For IPv6, redistribute OSPFv3 process 100, directly connected, and static routes with route map.

Example:

```
switch(config)# redistribute ospfv3 100 route-map testv6
switch(config)# redistribute direct route-map testv6
switch(config)# redistribute static route-map testv6
```

Step 6 Exit the current address family configuration mode.

Example:

```
switch(config)# exit-address-family
```

Configure VRF route import and export with route-targets and route-maps

Use this procedure to configure route import and export between two VRFs, for example, green and red, using route-targets and route maps to control route leaking.

A brief explanation on how route leaking works in this example:

- Routes exported from VRF green are tagged with 3:3. VRF red imports routes with route-target 3:3, so it receives routes exported by green.
- Conversely, routes exported from VRF red are tagged with 2:2. VRF green imports routes with route-target 2:2, so it receives routes exported by red.
- The use of export map and import map (test and testv6) allows filtering or modifying routes during export/import.
- The additional import vrf default map test in VRF red allows it to import routes from the default VRF, applying the test route map.
- This configuration enables bidirectional route leaking between VRF green and VRF red, controlled by route-targets and route maps.

Procedure

Step 1 Configure VRF green.

a) Use the **configure terminal** command to enter the global configuration mode.

Example:

```
switch# configure terminal
```

b) Enable **feature BGP**.

Example:

```
switch(config)# feature BGP
```

c) Configure VRF green using the **vrf context** *vrf-name* command.

Example:

```
switch(config)# vrf context green
```

d) Enter the IPv4 or IPv6 unicast address family configuration mode within the VRF green.

i. Use the **address-family ipv4 unicast** command for IPv4.

Example:

```
switch(config)# address-family ipv4 unicast
```

ii. Use the **address-family ipv6 unicast** command for IPv6.

Example:

```
switch(config)# address-family ipv6 unicast
```

e) Configure the VRF to import routes tagged with the route-target extended community 2:2 for IPv4 or IPv6 using the **route-target import 2:2** command.

Example:

```
switch(config)# route-target import 2:2
```

f) Configure the VRF to export routes tagged with the route-target extended community 3:3 for IPv4 or IPv6 using the **route-target export 3:3** command.

Example:

```
switch(config)# route-target export 3:3
```

g) Apply the route-map for IPv4 or IPv6 to filter or control routes being exported from the VRF green. Use the **export map** *route-map* command for IPv4 or IPv6 respectively.

Example:

```
switch(config)# export map test
```

Example:

```
switch(config)# export map test6
```

h) Apply the route-map for IPv4 or IPv6 to filter or control routes being imported into the VRF green. Use the **import map** *route-map* command for IPv4 or IPv6 respectively.

Example:

```
switch(config)# import map test
```

Example:

```
switch(config)# import map test6
```

i) Exit the current address family configuration mode.

Example:

```
switch# exit-address-family
```

Step 2 Configure VRF red.

a) Configure **feature BGP** as mentioned in Step a and b of Step 1, and then configure VRF red.

Example:

```
switch# configure terminal
switch(config)# feature BGP
switch(config)# vrf context red
```

b) Enter the IPv4 or IPv6 unicast address family configuration mode within the VRF red.

i. Use the **address-family ipv4 unicast** command for IPv4.

Example:

```
switch(config)# address-family ipv4 unicast
```

ii. Use the **address-family ipv6 unicast** command for IPv6.

Example:

```
switch(config)# address-family ipv6 unicast
```

c) Configure the VRF to import routes tagged with the route-target extended community 3:3 for IPv4 or IPv6 using the **route-target import 3:3** command.

Example:

```
switch(config)# route-target import 3:3
```

d) Configure the VRF to export routes tagged with the route-target extended community 2:2 for IPv4 or IPv6 using the **route-target export 2:2** command.

Example:

```
switch(config)# route-target export 2:2
```

e) Apply the route-map for IPv4 or IPv6 to filter or control routes being exported from the VRF red. Use the **export map** *route-map* command for IPv4 or IPv6 respectively.

Example:

```
switch(config)# export map test
```

Example:

```
switch(config)# export map test6
```

f) Use the **import vrf default map** *route-map* command to import routes from the default VRF.

Example:

```
switch(config)# import vrf default map test
```

g) Apply the route-map for IPv4 or IPv6 to filter or control routes being imported into the VRF red. Use the **import map** *route-map* command for IPv4 or IPv6 respectively.

Example:

```
switch(config)# import map test
```

Example:

```
switch(config)# import map test6
```

h) Exit the current address family configuration mode.

Example:

```
switch# exit-address-family
```

Verify inter-VRF traffic inspection

This section includes examples for inter-VRF traffic inspection.

In this example, route is learned on the source VRF red.

```
show ip route 12.10.1.0/24 vrf red
IP Route Table for VRF "red"
'*' denotes best ucast next-hop
'**' denotes best mcast next-hop
'[x/y]' denotes [preference/metric]
'%<string>' in via output denotes VRF <string>

12.10.1.0/24, ubest/mbest: 1/0
*via 1.10.1.2, [1/0], 00:02:58, static
```

In this example, route is leaked to the target VRF green.

```
show ip route 12.10.1.0/24 vrf green
IP Route Table for VRF "green"
'*' denotes best ucast next-hop
'**' denotes best mcast next-hop
```

```
'[x/y]' denotes [preference/metric]
'%<string>' in via output denotes VRF <string>

12.10.1.0/24, ubest/mbest: 1/0
*via 1.10.1.2%red, [20/0], 00:13:58, bgp-100, external, tag 100
```

In the output of the **show vrf red detail** command, the table id or hardware VRF id for VRF red is 6 and the table id for VRF green is 4.

```
show vrf red detail
VRF-Name: red, VRF-ID: 7, State: Up
VPNID: unknown
RD: 0:0
Max Routes: 0 Mid-Threshold: 0
Table-ID: 0x80000006, AF: IPv6, Fwd-ID: 0x80000006, State: Up
Table-ID: 0x00000006, AF: IPv4, Fwd-ID: 0x00000006, State: Up
VRF Type: Service

show vrf green detail
VRF-Name: green, VRF-ID: 5, State: Up
VPNID: unknown
RD: 0:0
Max Routes: 0 Mid-Threshold: 0
Table-ID: 0x80000004, AF: IPv6, Fwd-ID: 0x80000004, State: Up
Table-ID: 0x00000004, AF: IPv4, Fwd-ID: 0x00000004, State: Up
VRF Type: Service
```