Centralized VRF Route-Leaking for VXLAN BGP EVPN Fabrics

VXLAN BGP EVPN uses MP-BGP and its route-policy concept to import and export prefixes. The ability of this very extensive route-policy model allows to leak routes from one VRF to another VRF and vice-versa; any combination of custom VRF or VRF default can be used. VRF route-leaking is a switch-local function at specific to a location in the network, the location where the cross-VRF route-target import/export configuration takes place (leaking point). The forwarding between the different VRFs follows the control-plane, the location of where the configuration for the route-leaking is performed - hence Centralized VRF route-leaking. With the addition of VXLAN BGP EVPN, the leaking point requires to advertise the cross-VRF imported/exported route and advertise them towards the remote VTEPs or External Routers.

The advantage of Centralized VRF route-leaking is that only the VTEP acting as leaking point requires the special capabilities needed, while all other VTEPs in the network are neutral to this function.

You can also enable the VRFs in the EVPN fabric to access shared services in a specific VRF. These shared services are provided by a GOLF or Data Center Interconnect (DCI) device. The hosts located behind the internet or core can access the hosts inside the Application Centric Infrastructure (ACI) fabric. Traffic emanating from the north or core towards the south or ACI fabric will reach the default or shared VRF. The routes are identified for the traffic to reach the host located in the ACI fabric by leaking of host routes from tenant VRFs into the default VRF or internet VRF. This functionality is supported only on Cisco Nexus 7000 M3-Series I/O modules.

The **show ip route** command outputs for a scenario in which the routes from VRF A are leaked to VRF B are as given below. The output logs display the leaked routes along with the source VRF from which the routes were leaked.

```
device# show ip route 100.1.1.0/24 vrf VRF-A
IP Route Table for VRF "VRF-A"
'*' denotes best ucast next-hop
'**' denotes best mcast next-hop
'[x/y]' denotes [preference/metric]
'%<string>' in via output denotes VRF <string>
100.1.1.0/24, ubest/mbest: 1/0 time
*via 10.1.0.34%default, [20/0], 19:34:43, bgp-200, external, tag 6500 (evpn),
segid: 2850822 tunnelid: 0xa010022 encap: VXLAN

device# show ip route 100.1.1.0/24 vrf VRF-B
IP Route Table for VRF "VRF-B"
'*' denotes best ucast next-hop
'**' denotes best mcast next-hop
'[x/y]' denotes [preference/metric]
'%<string>' in via output denotes VRF <string>
100.1.1.0/24, ubest/mbest: 1/0 time
```

```
*via 10.1.0.34%default, [20/0], 19:34:50, bgp-200, external, tag 6500 (evpn),
segid: 2850822 tunnelid: 0xa010022 src vrf: VRF-A encap: VXLAN
```

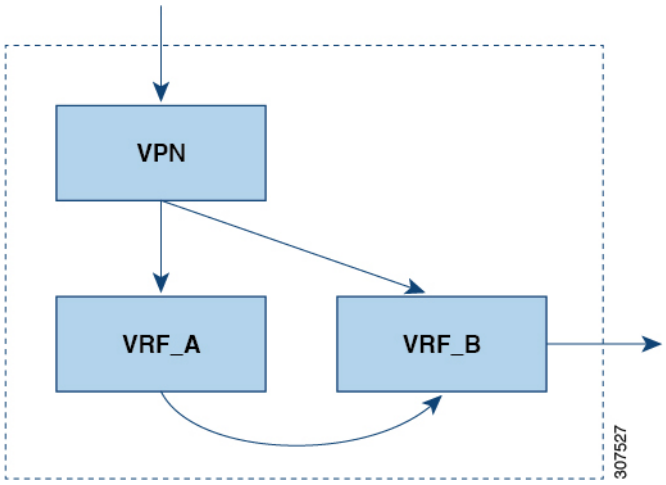

Configuring Route Leaking

Guidelines and Limitations for Centralized VRF Route-Leaking

The following are the guidelines and limitations for Centralized VRF Route-Leaking:

- Each prefix must be imported into each VRF for full cross-VRF reachability.
- The **feature bgp** command is required for the **export vrf default** command.
- If a VTEP has a less specific local prefix in its VRF, the VTEP might not be able to reach a more specific prefix in a different VRF.
- VXLAN routing in hardware and packet reencapsulation at VTEP is required for Centralized VRF Route-Leaking with BGP EVPN.
- The **feature bgp** command has to be used to enable usage of the **export vrf** and **import vrf** commands.
- Duplicate paths pointing to the same remote path may exist in case there are VRFs on which reimporting of routes has been enabled along with the same route-targets. This may have an impact on the performance and memory.

  There may be path duplication due to improper usage of route targets. Consider the following topology followed by the configuration used.




```
vrf context VRF_A
 address-family ipv4 unicast
  route-target both RT_A
  export vrf allow-vpn
vrf context VRF_B
 address-family ipv4 unicast
  route-target import RT_A
```

This configuration leads to VRF_B importing the same route via the VPN and VRF_A. This results in loss of multi-path as there are two separate paths in VRF_B having the same next hop but with different source route distinguishers.

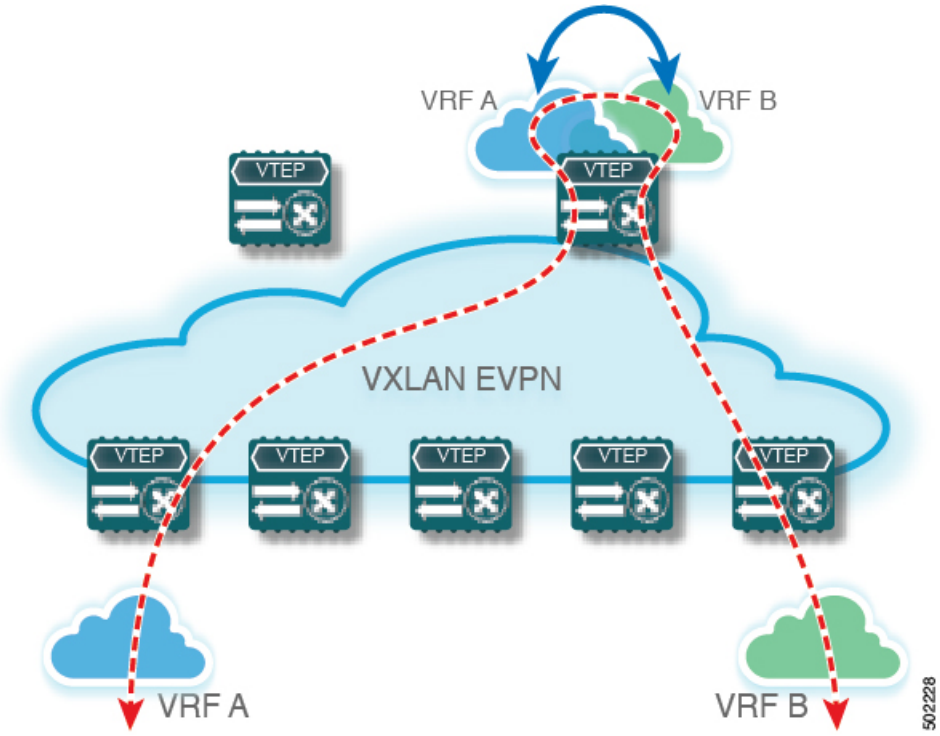
- Be careful when using local route leaking in a leaf-to-leaf case, where border-leaf routers (BLs) are leaking between the same VRFs. This scenario is more prone to routing loops. We recommend using inbound route-maps to exclude the imported routes from other BLs.
- After a remote path gets withdrawn, it can take up to 20 seconds more for BGP to completely clean up the path.

Centralized VRF Route-Leaking Brief - Specific Prefixes Between Custom VRF

Some pointers are given below:

- The Centralized VRF route-leaking for VXLAN BGP EVPN fabrics is depicted within Figure 2.
- BGP EVPN prefixes are cross-VRF leaked by exporting them from VRF Blue with an import into VRF Red and vice-versa. The Centralized VRF route-leaking is performed on the centralized Routing-Block (RBL) and could be any or multiple VTEPs.
- Configured less specific prefixes (aggregates) are advertised from the Routing-Block to the remaining VTEPs in the respective destination VRF.
- BGP EVPN does not export prefixes that were previously imported to prevent the occurrence of routing loops.

Figure 1: Centralized VRF Route-Leaking - Specific Prefixes with Custom VRF



Configuring Centralized VRF Route-Leaking - Specific Prefixes between Custom VRF

Configuring VRF Context on the Routing-Block VTEP

This procedure applies equally to IPv6.

Procedure

	Command or Action	Purpose
Step 1	**configure terminal**	Enter global configuration mode.
Step 2	**vrf context** *vrf-name*	Configure the VRF.
Step 3	**vni** *number*	Specify the VNI. The VNI associated with the VRF is often referred to as Layer-3 VNI, L3VNI or L3VPN. The L3VNI is configured as common identifier across the participating VTEPs.
Step 4	**rd auto**	Specify the VRFs Route Distinguisher (RD). The RD uniquely identifies a VTEP within a L3VNI.

	Command or Action	Purpose
Step 5	**address-family ipv4 unicast**	Configure the IPv4 Unicast address-family. Required for IPv4 over VXLAN with IPv4 underlay.
Step 6	**route-target both {auto** | *as:vni*}	Configure the Route Target (RT) for import/export of IPv4 prefixes within the IPv4 unicast address-family The Route Target (RT) is used for a per-VRF prefix import/export policy. If *as:vni* is entered, the value is in the format of ASN:NN, ASN4:NN, or IPv4:NN.
Step 7	**route-target both {auto** | *as:vni* }**evpn**	Configure the Route Target (RT) for import/export of IPv4 prefixes within the IPv4 unicast address-family The Route Target (RT) is used for a per-VRF prefix import/export policy. If *as:vni* is entered, the value is in the format of ASN:NN, ASN4:NN, or IPv4:NN.
Step 8	**route-target import** *rt-from-different-vrf*	Configure the Route Target (RT) for importing IPv4 prefixes from the leaked-from VRF (ie AS:VNI).
Step 9	**route-target import** *rt-from-different-vrf* **evpn**	Configure the Route Target (RT) for importing IPv4 prefixes from the leaked-from VRF (ie AS:VNI).

Configuring the BGP VRF instance on the Routing-Block

This procedure applies equally to IPv6.

Procedure

	Command or Action	Purpose
Step 1	**configure terminal**	Enters global configuration mode.
Step 2	**router bgp** *autonomous-system number*	Configure BGP.
Step 3	**vrf** *vrf-name*	Specify the VRF.
Step 4	**address-family ipv4 unicast**	Configure address family for IPv4
Step 5	**advertise l2vpn evpn**	Enable the advertisement of EVPN routes within IPv4 address-family.
Step 6	**aggregate-address** *prefix/mask*	Create less specific prefix aggregate into the destination VRF.
Step 7	**maximum-paths ibgp** *number*	Enabling equal cost multipathing (ECMP) for iBGP prefixes.

	Command or Action	Purpose
Step 8	**maximum-paths** *number*	Enabling equal cost multipathing (ECMP) for eBGP prefixes

Example - Configuration Centralized VRF Route-Leaking - Specific Prefixes Between Custom VRF

Configuring VXLAN BGP EVPN Routing-Block

The VXLAN BGP EVPN Routing-Block acts as centralized route-leaking point. The leaking configuration is localized such that control-plane leaking and data-path forwarding follow the same path. Most significantly is the VRF configuration of the Routing-Block and the advertisement of the less specific prefixes (aggregates) into the respective destination VRFs.

```
vrf context Blue
  vni 51010
  rd auto
  address-family ipv4 unicast
    route-target both auto
    route-target both auto evpn
    route-target import 65002:51020
    route-target import 65002:51020 evpn
!
vlan 2110
  vn-segment 51010
!
interface Vlan2110
  no shutdown
  mtu 9216
  vrf member Blue
  no ip redirects
  ip forward
!
vrf context Red
  vni 51020
  rd auto
  address-family ipv4 unicast
    route-target both auto
    route-target both auto evpn
    route-target import 65002:51010
    route-target import 65002:51010 evpn
!
vlan 2120
  vn-segment 51020
!
interface Vlan2120
  no shutdown
  mtu 9216
  vrf member Blue
  no ip redirects
  ip forward
!
interface nve1
  no shutdown
  host-reachability protocol bgp
  source-interface loopback1
  member vni 51010 associate-vrf
  member vni 51020 associate-vrf
!
router bgp 65002
  vrf Blue
```

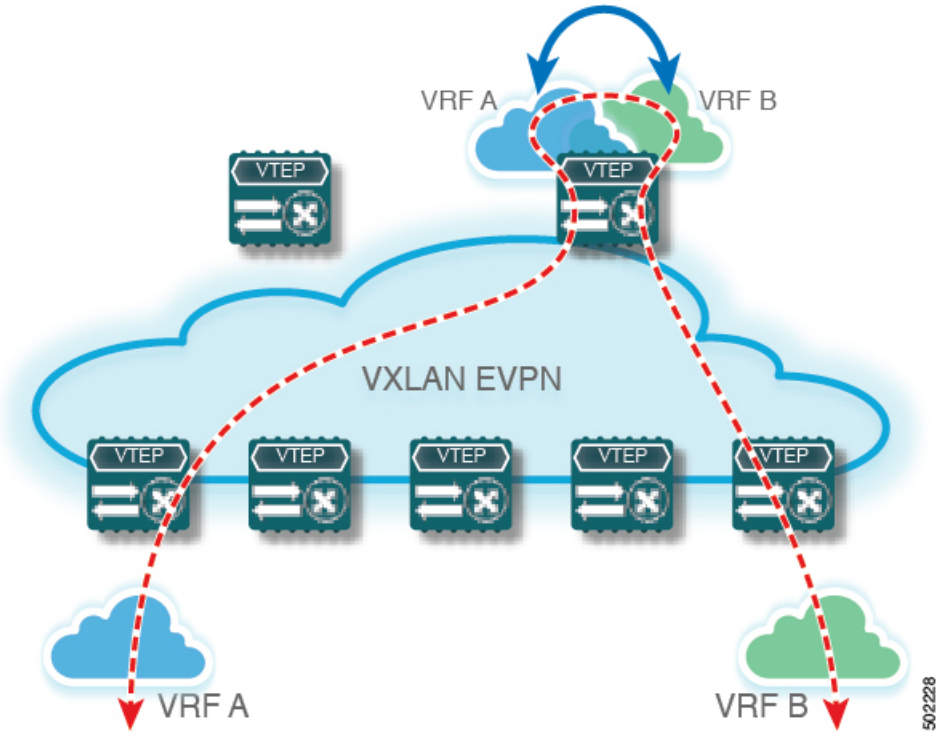
```
  address-family ipv4 unicast
    advertise l2vpn evpn
    aggregate-address 10.20.0.0/16
    maximum-paths ibgp 2
    Maximum-paths 2
vrf Red
  address-family ipv4 unicast
    advertise l2vpn evpn
    aggregate-address 10.10.0.0/16
    maximum-paths ibgp 2
    Maximum-paths 2
```

Centralized VRF Route-Leaking Brief - Shared Internet with Custom VRF

Some pointers follow:

- The Shared Internet with VRF route-leaking for VXLAN BGP EVPN fabrics is depicted in the following figure.
- The default-route is made exported from the Shared Internet VRF and re-advertisement within VRF Blue and VRF Red on the Border Node.
- Ensure the default-route in VRF Blue and VRF Red is not leaked to the Shared Internet VRF.
- The less specific prefixes for VRF Blue and VRF Red are exported for the Shared Internet VRF and re-advertised as necessary.
- Configured less specific prefixes (aggregates) that are advertised from the Border Node to the remaining VTEPs to the destination VRF (Blue or Red).
- BGP EVPN does not export prefixes that were previously imported to prevent the occurrence of routing loops.

Figure 2: Centralized VRF Route-Leaking - Shared Internet with Custom VRF



Configuring Centralized VRF Route-Leaking - Shared Internet with Custom VRF

Configuring Internet VRF on Border Node

This procedure applies equally to IPv6.

Procedure

	Command or Action	Purpose
Step 1	**configure terminal**	Enter global configuration mode.
Step 2	**vrf context** *vrf-name*	Configure the VRF.
Step 3	**vni** *number*	Specify the VNI. The VNI associated with the VRF is often referred to as Layer-3 VNI, L3VNI or L3VPN. The L3VNI is configured as common identifier across the participating VTEPs.
Step 4	**ip route 0.0.0.0/0** *next-hop*	Configure default-route in shared internet VRF to external router (example).
Step 5	**rd auto**	Specify the VRFs Route Distinguisher (RD).

	Command or Action	Purpose
		The RD uniquely identifies a VTEP within a L3VNI.
Step 6	**address-family ipv4 unicast**	Configure the IPv4 Unicast address-family. Required for IPv4 over VXLAN with IPv4 underlay.
Step 7	**route-target both {auto** | *as:vni*}	Configure the Route Target (RT) for import/export of EVPN and IPv4 prefixes within the IPv4 unicast address-family.
Step 8	**route-target both** *shared-vrf-rt* **evpn**	Configure a special Route Target (RT) for the import/export of the shared IPv4 prefixes. Additional import/export map for further qualification is supported

Configuring Shared Internet BGP Instance on the Border Node

This procedure applies equally to IPv6.

Procedure

	Command or Action	Purpose
Step 1	**configure terminal**	Enters global configuration mode.
Step 2	**router bgp** *autonomous-system number*	Configure BGP.
Step 3	**vrf** *vrf-name*	Specify the VRF.
Step 4	**address-family ipv4 unicast**	Configure address family for IPv4
Step 5	**advertise l2vpn evpn**	Enable the advertisement of EVPN routes within IPv4 address-family.
Step 6	**aggregate-address** *prefix/mask*	Create less specific prefix aggregate into the destination VRF.
Step 7	**maximum-paths ibgp** *number*	Enabling equal cost multipathing (ECMP) for iBGP prefixes.
Step 8	**maximum-paths** *number*	Enabling equal cost multipathing (ECMP) for eBGP prefixes.

Configuring Custom VRF on Border Node

This procedure applies equally to IPv6

Procedure

	Command or Action	Purpose
Step 1	**configure terminal**	Enters global configuration mode.
Step 2	**ip prefix-list** *name* **seq 5 permit 0.0.0.0/0**	Configure IPv4 prefix-list for default-route filtering.
Step 3	**route-map** *name* **deny 10**	Create route-map with leading deny statement to prevent the default-route of being leaked.
Step 4	**match ip address prefix-list** *name*	Match against the IPv4 prefix-list that contains the default-route.
Step 5	**route-map** *name* **permit 20**	Create route-map with trailing allow statement to advertise non-matching routes via route-leaking.

Configuring Custom VRF Context on the Border Node - 1

This procedure applies equally to IPv6.

Procedure

	Command or Action	Purpose
Step 1	**configure terminal**	Enters global configuration mode.
Step 2	**vrf context** *vrf-name*	Configure the VRF.
Step 3	**vni** *number*	Specify the VNI. The VNI associated with the VRF is often referred to as Layer-3 VNI, L3VNI or L3VPN. The L3VNI is configured as the common identifier across the participating VTEPs.
Step 4	**rd auto**	Specify the VRFs Route Distinguisher (RD). The Route Distinguisher (RD) uniquely identifies a VTEP within a L3VNI.
Step 5	**ip route 0.0.0.0/0 Null0**	Configure default-route in common VRF to attract traffic towards Border Node with Shared Internet VRF.
Step 6	**address-family ipv4 unicast**	Configure the IPv4 Unicast address-family. Required for IPv4 over VXLAN with IPv4 underlay.
Step 7	**route-target both {auto** | *as:vni***}**	Configure the Route Target (RT) for import/export of IPv4 prefixes within the IPv4 unicast address-family The Route Target (RT) is used for a per-VRF prefix import/export

	Command or Action	Purpose
		policy. If *as:vni* is entered, the value is in the format of ASN:NN, ASN4:NN, or IPv4:NN.
Step 8	**route-target both {auto** | *as:vni***} evpn**	Configure the Route Target (RT) for import/export of IPv4 prefixes within the IPv4 unicast address-family The Route Target (RT) is used for a per-VRF prefix import/export policy. If *as:vni* is entered, the value is in the format of ASN:NN, ASN4:NN, or IPv4:NN.
Step 9	**import map** *name*	Apply a route-map on routes being imported into this routing table.

Configuring Custom VRF Instance in BGP on the Border Node

This procedure applies equally to IPv6.

Procedure

	Command or Action	Purpose
Step 1	**configure terminal**	Enters global configuration mode.
Step 2	**router bgp** *autonomous-system-number*	Configure BGP.
Step 3	**vrf** *vrf-name*	Specify the VRF.
Step 4	**address-family ipv4 unicast**	Configure address family for IPv4.
Step 5	**advertise l2vpn evpn**	Enable the advertisement of EVPN routes within IPv4 address-family.
Step 6	**network 0.0.0.0/0**	Creating IPv4 default-route network statement.
Step 7	**maximum-paths ibgp** *number*	Enabling equal cost multipathing (ECMP) for iBGP prefixes.
Step 8	**maximum-paths** *number*	Enabling equal cost multipathing (ECMP) for eBGP prefixes.

Example - Configuration Centralized VRF Route-Leaking - Shared Internet with Custom VRF

An example of Centralized VRF route-leaking with Shared Internet VRF

Configuring VXLAN BGP EVPN Border Node for Shared Internet VRF

The VXLAN BGP EVPN Border Node provides a centralized Shared Internet VRF. The leaking configuration is localized such that control-plane leaking and data-path forwarding following the same path. Most significantly is the VRF configuration of the Border Node and the advertisement of the default-route and less specific prefixes (aggregates) into the respective destination VRFs.

```
vrf context Shared
  vni 51099
  ip route 0.0.0.0/0 10.9.9.1
  rd auto
  address-family ipv4 unicast
    route-target both auto
    route-target both auto evpn
    route-target both 99:99
    route-target both 99:99 evpn
!
vlan 2199
  vn-segment 51099
!
interface Vlan2199
  no shutdown
  mtu 9216
  vrf member Shared
  no ip redirects
  ip forward
!
ip prefix-list PL_DENY_EXPORT seq 5 permit 0.0.0.0/0
!
route-map RM_DENY_IMPORT deny 10
 match ip address prefix-list PL_DENY_EXPORT
route-map RM_DENY_IMPORT permit 20
!
vrf context Blue
  vni 51010
  ip route 0.0.0.0/0 Null0
  rd auto
  address-family ipv4 unicast
    route-target both auto
    route-target both auto evpn
    route-target both 99:99
    route-target both 99:99 evpn
    import map RM_DENY_IMPORT
!
vlan 2110
  vn-segment 51010
!
interface Vlan2110
  no shutdown
  mtu 9216
  vrf member Blue
  no ip redirects
  ip forward
!
vrf context Red
  vni 51020
  ip route 0.0.0.0/0 Null0
  rd auto
  address-family ipv4 unicast
    route-target both auto
    route-target both auto evpn
    route-target both 99:99
    route-target both 99:99 evpn
    import map RM_DENY_IMPORT
!
vlan 2120
  vn-segment 51020
!
interface Vlan2120
  no shutdown
  mtu 9216
```

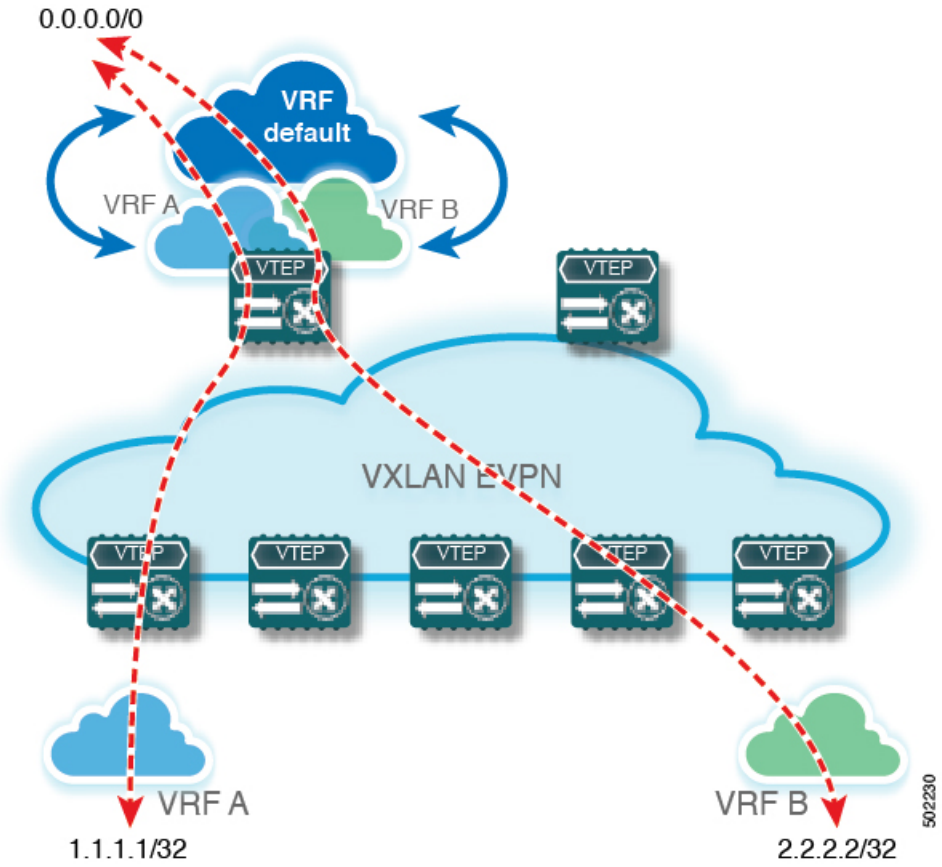
```
  vrf member Blue
  no ip redirects
  ip forward
!
interface nve1
  no shutdown
  host-reachability protocol bgp
  source-interface loopback1
  member vni 51099 associate-vrf
  member vni 51010 associate-vrf
  member vni 51020 associate-vrf
!
router bgp 65002
  vrf Shared
    address-family ipv4 unicast
      advertise l2vpn evpn
      aggregate-address 10.10.0.0/16
      aggregate-address 10.20.0.0/16
      maximum-paths ibgp 2
      maximum-paths 2
  vrf Blue
    address-family ipv4 unicast
      advertise l2vpn evpn
      network 0.0.0.0/0
      maximum-paths ibgp 2
      maximum-paths 2
  vrf Red
    address-family ipv4 unicast
      advertise l2vpn evpn
      network 0.0.0.0/0
      maximum-paths ibgp 2
      maximum-paths 2
```

Centralized VRF Route-Leaking Brief - Shared Internet with VRF Default

Some pointers are given below:

- The Shared Internet with VRF route-leaking for VXLAN BGP EVPN fabrics is depicted within Figure 4.
- The default-route is made exported from VRF default and re-advertisement within VRF Blue and VRF Red on the Border Node.
- Ensure the default-route in VRF Blue and VRF Red is not leaked to the Shared Internet VRF
- The less specific prefixes for VRF Blue and VRF Red are exported to VRF default and re-advertised as necessary.
- Configured less specific prefixes (aggregates) that are advertised from the Border Node to the remaining VTEPs to the destination VRF (Blue or Red).
- BGP EVPN does not export prefixes that were previously imported to prevent the occurrence of routing loops.

Figure 3: Centralized VRF Route-Leaking - Shared Internet with VRF Default



Configuring Centralized VRF Route-Leaking - Shared Internet with VRF Default

Configuring VRF Default on Border Node

This procedure applies equally to IPv6.

Procedure

	Command or Action	Purpose
Step 1	**configure terminal**	Enters global configuration mode.
Step 2	**ip route 0.0.0.0/0** *next-hop*	Configure default-route in VRF default to external router (example)

Configuring BGP Instance for VRF Default on the Border Node

This procedure applies equally to IPv6.

Procedure

	Command or Action	Purpose
Step 1	**configure terminal**	Enters global configuration mode.
Step 2	**router bgp** *autonomous-system number*	Configure BGP.
Step 3	**address-family ipv4 unicast**	Configure address family for IPv4.
Step 4	**aggregate-address** *prefix/mask*	Create less specific prefix aggregate in VRF default.
Step 5	**maximum-paths** *number*	Enabling equal cost multipathing (ECMP) for eBGP prefixes.

Configuring Custom VRF on Border Node

This procedure applies equally to IPv6

Procedure

	Command or Action	Purpose
Step 1	**configure terminal**	Enters global configuration mode.
Step 2	**ip prefix-list** *name* **seq 5 permit 0.0.0.0/0**	Configure IPv4 prefix-list for default-route filtering.
Step 3	**route-map** *name* **deny 10**	Create route-map with leading deny statement to prevent the default-route of being leaked.
Step 4	**match ip address prefix-list** *name*	Match against the IPv4 prefix-list that contains the default-route.
Step 5	**route-map** *name* **permit 20**	Create route-map with trailing allow statement to advertise non-matching routes via route-leaking.

Configuring Filter for Permitted Prefixes from VRF Default on the Border Node

This procedure applies equally to IPv6.

Procedure

	Command or Action	Purpose
Step 1	**configure terminal**	Enters global configuration mode.
Step 2	**route-map** *name* **permit 10**	Create route-map with allow statement to advertise routes via route-leaking to the customer VRF and subsequently remote VTEPs.

Configuring Custom VRF Context on the Border Node - 2

This procedure applies equally to IPv6.

Procedure

	Command or Action	Purpose
Step 1	**configure terminal**	Enters global configuration mode.
Step 2	**vrf context** *vrf-name*	Configure the VRF.
Step 3	**vni** *number*	Specify the VNI. The VNI associated with the VRF is often referred to as Layer-3 VNI, L3VNI or L3VPN. The L3VNI is configured as common identifier across the participating VTEPs.
Step 4	**rd auto**	Specify the VRFs Route Distinguisher (RD). The Route Distinguisher (RD) uniquely identifies a VTEP within a L3VNI.
Step 5	**ip route 0.0.0.0/0 Null0**	Configure default-route in common VRF to attract traffic towards Border Node with Shared Internet VRF.
Step 6	**address-family ipv4 unicast**	Configure the IPv4 Unicast address-family. Required for IPv4 over VXLAN with IPv4 underlay.
Step 7	**route-target both auto** | *AS:VNI*	Configure the Route Target (RT) for import/export of EVPN and IPv4 prefixes within the IPv4 unicast address-family.
Step 8	**route-target both auto** | *AS:VNI* **evpn**	Configure the Route Target (RT) for import/export of EVPN and IPv4 prefixes within the IPv4 unicast address-family.
Step 9	**route-target both** *shared-vrf-rt*	Configure a special Route Target (RT) for the import/export of the Shared IPv4 prefixes. Additional import/export map for further qualification is supported
Step 10	**route-target both** *shared-vrf-rt* **evpn**	Configure a special Route Target (RT) for the import/export of the Shared IPv4 prefixes. Additional import/export map for further qualification is supported
Step 11	**import vrf default map** *name*	Permits all routes, from VRF default, from being imported into the custom VRF according to the specific route-map.

Configuring Custom VRF Instance in BGP on the Border Node

This procedure applies equally to IPv6.

Procedure

	Command or Action	Purpose
Step 1	**configure terminal**	Enters global configuration mode.
Step 2	**router bgp** *autonomous-system-number*	Configure BGP.
Step 3	**vrf** *vrf-name*	Specify the VRF.
Step 4	**address-family ipv4 unicast**	Configure address family for IPv4.
Step 5	**advertise l2vpn evpn**	Enable the advertisement of EVPN routes within IPv4 address-family.
Step 6	**network 0.0.0.0/0**	Creating IPv4 default-route network statement.
Step 7	**maximum-paths ibgp** *number*	Enabling equal cost multipathing (ECMP) for iBGP prefixes.
Step 8	**maximum-paths** *number*	Enabling equal cost multipathing (ECMP) for eBGP prefixes.

Example - Configurating Centralized VRF Route-Leaking - VRF Default with Custom VRF

An example of Centralized VRF route-leaking with VRF default

Configuring VXLAN BGP EVPN Border Node for VRF Default

The VXLAN BGP EVPN Border Node provides centralized access to VRF default. The leaking configuration is localized such that control-plane leaking and data-path forwarding following the same path. Most significantly is the VRF configuration of the Border Node and the advertisement of the default-route and less specific prefixes (aggregates) into the respective destination VRFs.

```
ip route 0.0.0.0/0 10.9.9.1
!
ip prefix-list PL_DENY_EXPORT seq 5 permit 0.0.0.0/0
!
route-map permit 10
match ip address prefix-list PL_DENY_EXPORT
route-map RM_DENY_EXPORT permit 20
route-map RM_PERMIT_IMPORT permit 10
!
vrf context Blue
  vni 51010
  ip route 0.0.0.0/0 Null0
  rd auto
  address-family ipv4 unicast
    route-target both auto
    route-target both auto evpn
    import vrf default map RM_PERMIT_IMPORT
    export vrf default 100 map RM_DENY_EXPORT allow-vpn
!
vlan 2110
```

```
  vn-segment 51010
!
interface Vlan2110
  no shutdown
  mtu 9216
  vrf member Blue
  no ip redirects
  ip forward
!
vrf context Red
  vni 51020
  ip route 0.0.0.0/0 Null0
  rd auto
  address-family ipv4 unicast
    route-target both auto
    route-target both auto evpn
    import vrf default map RM_PERMIT_IMPORT
    export vrf default 100 map RM_DENY_EXPORT allow-vpn
!
vlan 2120
  vn-segment 51020
!
interface Vlan2120
  no shutdown
  mtu 9216
  vrf member Blue
  no ip redirects
  ip forward
!
interface nve1
  no shutdown
  host-reachability protocol bgp
  source-interface loopback1
  member vni 51010 associate-vrf
  member vni 51020 associate-vrf
!
router bgp 65002
  address-family ipv4 unicast
      aggregate-address 10.10.0.0/16
      aggregate-address 10.20.0.0/16
      maximum-paths 2
      maximum-paths ibgp 2
  vrf Blue
    address-family ipv4 unicast
      advertise l2vpn evpn
      network 0.0.0.0/0
      maximum-paths ibgp 2
      maximum-paths 2
  vrf Red
    address-family ipv4 unicast
      advertise l2vpn evpn
      network 0.0.0.0/0
      maximum-paths ibgp 2
      maximum-paths 2
```

Configuring Routes Imported from a VPN to Leak into the Default VRF

You can configure a VRF to allow routes that are imported from a BGP VPN to be exported to the default VRF. Use this procedure for a non-default VRF. Use this procedure for a non-default VRF.

If you have not already enabled BGP, enable it now using the **feature bgp** command.

Procedure

	Command or Action	Purpose
Step 1	**configure terminal**	Enters global configuration mode.
Step 2	**vrf context** *vrf-name*	Creates a new VRF and enters VRF configuration mode. The name can be any case-sensitive, alphanumeric string up to 32 characters.
Step 3	**address-family ipv4 unicast**	Configure the address family for IPv4.
Step 4	**export vrf default** [*prefix-limit*]**map** *route-map* **allow-vpn**	Configures the current VRF to allow routes that are imported from a BGP VPN to be exported to the default VRF.

Configuring Routes Leaked from the Default-VRF to Export to a VPN

You can configure a VRF to allow routes leaked from the default VRF to be exported to a BGP VPN. Use this procedure for a non-default VRF.

If you have not already enabled BGP, enable it now using the **feature bgp** command.

Procedure

	Command or Action	Purpose
Step 1	**configure terminal**	Enters global configuration mode.
Step 2	**vrf context** *vrf-name*	Creates a new VRF and enters VRF configuration mode. The name can be any case-sensitive, alphanumeric string up to 32 characters.
Step 3	**address-family** *address-family sub-family*	Configure the address family for IPv4.
Step 4	**import vrf default** [*prefix-limit*]**map** *route-map* **advertise-vpn**	Configures the current VRF to allow routes imported from the default VRF to be exported to a BGP VPN.

Configuring Routes Imported from a VPN to Export to a VRF

You can configure a VRF to allow VPN imported routes to be exported to another VRF. Use this procedure for non-default VRFs.

If you have not already enabled BGP, enable it now using the **feature bgp** command.

Procedure

	Command or Action	Purpose
Step 1	**configure terminal**	Enters global configuration mode.
Step 2	**vrf context** *vrf-name*	Creates a new VRF and enters VRF configuration mode. The name can be any case-sensitive, alphanumeric string up to 32 characters.
Step 3	**address-family** *address-family sub-family*	Configure the address family for IPv4.
Step 4	**export vrf allow-vpn**	Configures a VRF to allow routes imported from a BGP VPN to be exported to a non-default VRF.

Configuring Routes Imported from a VRF to Export to a VPN

You can configure a VRF to allow routes imported from another VRF to be exported to a BGP VPN. Use this procedure for non-default VRFs.

If you have not already enabled BGP, enable it now using the **feature bgp** command.

Procedure

	Command or Action	Purpose
Step 1	**configure terminal**	Enters global configuration mode.
Step 2	**vrf context** *vrf-name*	Creates a new VRF and enters VRF configuration mode. The name can be any case-sensitive, alphanumeric string up to 32 characters.
Step 3	**address-family** *address-family sub-family*	Configure the address family for IPv4.
Step 4	**import vrf advertise-vpn**	Configures the current VRF to allow routes that are imported from another VRF to be exported to a BGP VPN.

Configuration Examples

Configuring BGP VPN to Default VRF Reachability

In this example, the configuration enables route re-importation through an intermediate VRF, called VRF_A, which is between the VPN and the default VRF.

```
vrf context VRF_A
 address-family ipv4 unicast
  route-target both auto evpn
  import vrf default map MAP_1 advertise-vpn
  export vrf default map MAP_1 allow-vpn
```

Route re-importation is enabled by using the **advertise-vpn** option to control importing routes from the VPN into VRF_A, and **allow-vpn** for the export map to control exporting VPN-imported routes from VRF_A out to the default VRF. Configuration occurs on the intermediate VRF.

Configuring VPN to VRF-Lite Reachability

In this example, the VPN connects to a tenant VRF, called VRF_A. VRF_A connects a VRF-Lite, called VRF-B. The configuration enables VPN imported routes to be leaked from VRF_A to VRF_B.

```
vrf context VRF_A
 address-family ipv4 unicast
  route-target both auto
  route-target both auto evpn
  route-target import 3:3
  route-target export 2:2
  import vrf advertise-vpn
  export vrf allow-vpn
vrf context VRF_B
 address-family ipv4 unicast
  route-target both 1:1
  route-target import 2:2
  route-target export 3:3
```

Route leaking between the two is enabled by using the **allow-vpn** in an export map configured in VRF_A (tenant). The export map in VRF_A allows route imported from the VPN to be leaked into the VRF_B. Routes processed by the export map have the **route-map export** and **export-map** attributes added to the route's set of route targets. The import map uses **advertise-vpn** which enables routes that are imported from the VRF-Lite for be exported out to the VPN.

After a route leaks between the VRFs, it is reoriginated and its route targets are replaced by the route target export and export map attributes specified by the new VRF's configuration.

Leaf-to-Leaf Reachability

In this example, two VPNs exist and two VRFs exist. VPN_1 is connected to VRF_A and VPN_2 is connected to VRF_B. Both VRFs are route distinguishers (RDs).

```
vrf context VRF_A
 address-family ipv4 unicast
  route-target both auto
  route-target both auto evpn
  route-target import 3:3
  route-target export 2:2
  import vrf advertise-vpn
  export vrf allow-vpn
vrf context VRF_B
 address-family ipv4 unicast
  route-target both 1:1
  route-target import 2:2
  route-target export 3:3
  import vrf advertise-vpn
  export vrf allow-vpn
```

Route leaking between the two is enabled by using the **allow-vpn**in an export map configured in VRF_A and VRF_B. VPN imported routes have **route-map export** and **export-map** attributes added to the route's set of route targets. The import map uses **advertise-vpn** option which enables routes that are imported from each VRF to be exported out to the VPN.

After a route leaks between the VRFs, it is reoriginated and its route targets are replaced by the route target export and export map attributes specified by the new VRF's configuration.

Leaf-to-Leaf with Loop Prevention

In the leaf-to-leaf configuration, you can inadvertently cause loops between the BLs that are leaking between the same VRFs unless you are careful with your route maps:

- You can use an inbound route map in each BL to deny updates from every other BL.
- If a BL originates a route, a standard community can be applied, which enables other BLs to accept the routes. This community is then stripped in the receiving BL.

In the following example, VTEPs 3.3.3.3, 4.4.4.4 and 5.5.5.5 are the BLs.

```
ip prefix-list BL_PREFIX_LIST seq 5 permit 3.3.3.3/32
ip prefix-list BL_PREFIX_LIST seq 10 permit 4.4.4.4/32
ip prefix-list BL_PREFIX_LIST seq 20 permit 5.5.5.5/32
ip community-list standard BL_COMMUNITY seq 10 permit 123:123
route-map INBOUND_MAP permit 5
match community BL_COMMUNITY
set community none
route-map INBOUND_MAP deny 10
match ip next-hop prefix-list BL_PREFIX_LIST
route-map INBOUND_MAP permit 20
route-map OUTBOUND_SET_COMM permit 10
match evpn route-type 2 mac-ip
set community 123:123
route-map SET_COMM permit 10
set community 123:123
route-map allow permit 10

vrf context vni100
vni 100
address-family ipv4 unicast
route-target import 2:2
route-target export 1:1
route-target both auto
route-target both auto evpn
import vrf advertise-vpn
export vrf allow-vpn

vrf context vni200
vni 200
address-family ipv4 unicast
route-target import 1:1
route-target export 2:2
route-target both auto
route-target both auto evpn
import vrf advertise-vpn
export vrf allow-vpn

router bgp 100
template peer rr
remote-as 100
update-source loopback0
```

```
address-family l2vpn evpn
send-community
send-community extended
route-map INBOUND_MAP in
route-map OUTBOUND_SET_COMM out
neighbor 101.101.101.101
inherit peer rr
neighbor 102.102.102.102
inherit peer rr
vrf vni100
address-family ipv4 unicast
network 3.3.3.100/32 route-map SET_COMM
vrf vni200
address-family ipv4 unicast
network 3.3.3.200/32 route-map SET_COMM
```

In this example, the tenant VRFs for the border leaf (BL) router can leak traffic by enabling extra import export flows, and the route targets in the route maps determine where the routes are imported from or exported to.

Multipath in a VRF

In this example, a VPN has multiple incoming paths. This configuration enables route leaking through an intermediate VRF, called VRF_A, which is between the VPN and another VRF, named VRF_B. Assume that multipathing is enabled in VRF_A.

```
vrf context VRF_A
 address-family ipv4 unicast
  route-target both auto evpn
  route-target export 3:3
  export vrf allow-vpn
vrf context VRF_B
 address-family ipv4 unicast
  route-target import 3:3
```

Route leaking is enabled by **allow-vpn** in the export map configured in VRF_A. When two paths for a given prefix are learnt from a VPN and imported into VRF_A, two different paths exist in VRF_B with the same source RD (VRF_A's local RD). Each route is distinguished by the original source RD (remote RD).

Path Duplication

In this example, the configuration enables a single VPN path to be imported into both VRF_A and VRF_B. Because VRF_A is configured with **export vrf allow-vpn**, VRF_A also leaks its routes into VRF_B. VRF_B then has two paths with same source RD (VRF_A's local RD), each one distinguished by the original source RD (remote RD).

```
vrf context VRF_A
 address-family ipv4 unicast
  route-target import 1:1 evpn
  route-target export 1:1 evpn
  route-target export 2:2
export vrf allow-vpn
 vrf context VRF_B
  address-family ipv4 unicast
  route-target import 1:1 evpn
  route-target import 2:2
```

This configuration creates a situation in which multipathing does not exist.

Displaying Centralized Route Leaking Information

The following table shows the commands that display information about the Centralized Route Leaking feature.

Command	Action
show bgp vrf *vrf-name* process	For a default or non-default VRF, shows the enabled state (Yes or No) of the **import advertise-vpn** and **export allow-vpn** options.
show bgp vrf *vrf-name* ipv4 unicast *prefix*	Shows information about imported paths, including a list of destinations a route has been imported from.

Route re-importation is enabled by using the **advertise-vpn** option to control importing routes from the VPN into VRF_A, and **allow-vpn** for the export map to control exporting VPN-imported routes from VRF_A out to the default VRF. Configuration occurs on the intermediate VRF.

Configuring VPN to VRF-Lite Reachability

In this example, the VPN connects to a tenant VRF, called VRF_A. VRF_A connects a VRF-Lite, called VRF-B. The configuration enables VPN imported routes to be leaked from VRF_A to VRF_B.

```
vrf context VRF_A
 address-family ipv4 unicast
  route-target both auto
  route-target both auto evpn
  route-target import 3:3
  route-target export 2:2
  import vrf advertise-vpn
  export vrf allow-vpn
vrf context VRF_B
 address-family ipv4 unicast
  route-target both 1:1
  route-target import 2:2
  route-target export 3:3
```

Route leaking between the two is enabled by using the **allow-vpn** in an export map configured in VRF_A (tenant). The export map in VRF_A allows route imported from the VPN to be leaked into the VRF_B. Routes processed by the export map have the **route-map export** and **export-map** attributes added to the route's set of route targets. The import map uses **advertise-vpn** which enables routes that are imported from the VRF-Lite for be exported out to the VPN.

After a route leaks between the VRFs, it is reoriginated and its route targets are replaced by the route target export and export map attributes specified by the new VRF's configuration.

Leaf-to-Leaf Reachability

In this example, two VPNs exist and two VRFs exist. VPN_1 is connected to VRF_A and VPN_2 is connected to VRF_B. Both VRFs are route distinguishers (RDs).

```
vrf context VRF_A
 address-family ipv4 unicast
  route-target both auto
  route-target both auto evpn
```

```
  route-target import 3:3
  route-target export 2:2
  import vrf advertise-vpn
  export vrf allow-vpn
vrf context VRF_B
 address-family ipv4 unicast
  route-target both 1:1
  route-target import 2:2
  route-target export 3:3
  import vrf advertise-vpn
  export vrf allow-vpn
```

Route leaking between the two is enabled by using the **allow-vpn**in an export map configured in VRF_A and VRF_B. VPN imported routes have **route-map export** and **export-map** attributes added to the route's set of route targets. The import map uses **advertise-vpn** option which enables routes that are imported from each VRF to be exported out to the VPN.

After a route leaks between the VRFs, it is reoriginated and its route targets are replaced by the route target export and export map attributes specified by the new VRF's configuration.

Leaf-to-Leaf with Loop Prevention

In the leaf-to-leaf configuration, you can inadvertently cause loops between the BLs that are leaking between the same VRFs unless you are careful with your route maps:

- You can use an inbound route map in each BL to deny updates from every other BL.
- If a BL originates a route, a standard community can be applied, which enables other BLs to accept the routes. This community is then stripped in the receiving BL.

In the following example, VTEPs 3.3.3.3, 4.4.4.4 and 5.5.5.5 are the BLs.

```
ip prefix-list BL_PREFIX_LIST seq 5 permit 3.3.3.3/32
ip prefix-list BL_PREFIX_LIST seq 10 permit 4.4.4.4/32
ip prefix-list BL_PREFIX_LIST seq 20 permit 5.5.5.5/32
ip community-list standard BL_COMMUNITY seq 10 permit 123:123
route-map INBOUND_MAP permit 5
match community BL_COMMUNITY
set community none
route-map INBOUND_MAP deny 10
match ip next-hop prefix-list BL_PREFIX_LIST
route-map INBOUND_MAP permit 20
route-map OUTBOUND_SET_COMM permit 10
match evpn route-type 2 mac-ip
set community 123:123
route-map SET_COMM permit 10
set community 123:123
route-map allow permit 10

vrf context vni100
vni 100
address-family ipv4 unicast
route-target import 2:2
route-target export 1:1
route-target both auto
route-target both auto evpn
import vrf advertise-vpn
export vrf allow-vpn

vrf context vni200
vni 200
```

```
address-family ipv4 unicast
route-target import 1:1
route-target export 2:2
route-target both auto
route-target both auto evpn
import vrf advertise-vpn
export vrf allow-vpn

router bgp 100
template peer rr
remote-as 100
update-source loopback0
address-family l2vpn evpn
send-community
send-community extended
route-map INBOUND_MAP in
route-map OUTBOUND_SET_COMM out
neighbor 101.101.101.101
inherit peer rr
neighbor 102.102.102.102
inherit peer rr
vrf vni100
address-family ipv4 unicast
network 3.3.3.100/32 route-map SET_COMM
vrf vni200
address-family ipv4 unicast
network 3.3.3.200/32 route-map SET_COMM
```

In this example, the tenant VRFs for the border leaf (BL) router can leak traffic by enabling extra import export flows, and the route targets in the route maps determine where the routes are imported from or exported to.

Multipath in a VRF

In this example, a VPN has multiple incoming paths. This configuration enables route leaking through an intermediate VRF, called VRF_A, which is between the VPN and another VRF, named VRF_B. Assume that multipathing is enabled in VRF_A.

```
vrf context VRF_A
 address-family ipv4 unicast
  route-target both auto evpn
  route-target export 3:3
  export vrf allow-vpn
vrf context VRF_B
 address-family ipv4 unicast
  route-target import 3:3
```

Route leaking is enabled by **allow-vpn** in the export map configured in VRF_A. When two paths for a given prefix are learnt from a VPN and imported into VRF_A, two different paths exist in VRF_B with the same source RD (VRF_A's local RD). Each route is distinguished by the original source RD (remote RD).

Path Duplication

In this example, the configuration enables a single VPN path to be imported into both VRF_A and VRF_B. Because VRF_A is configured with **export vrf allow-vpn**, VRF_A also leaks its routes into VRF_B. VRF_B then has two paths with same source RD (VRF_A's local RD), each one distinguished by the original source RD (remote RD).

```
vrf context VRF_A
 address-family ipv4 unicast
```

```
  route-target import 1:1 evpn
  route-target export 1:1 evpn
  route-target export 2:2
export vrf allow-vpn
 vrf context VRF_B
  address-family ipv4 unicast
  route-target import 1:1 evpn
  route-target import 2:2
```

This configuration creates a situation in which multipathing does not exist.

Displaying Centralized Route Leaking Information

To display information about the RTs that have been configured and to check if the **allow-vpn** and **advertise-vpn** keywords have been configured, use the **show bgp vrf***vrf* **process** command.

```
switch# show bgp vrf vni100 process
Information regarding configured VRFs:
BGP Information for VRF vni100
VRF Id : 3
VRF state : UP
VNID : 100 (valid)
Topo Id : 100
Encap type : VXLAN
VTEP IP : 3.3.3.3
VTEP Virtual IP : 0.0.0.0
VTEP VIP-R : 0.0.0.0
Router-MAC : 5254.000e.7996
VIP Derived MAC : 5254.000e.7996
VIP-R Derived MAC : 0000.0000.0000
Router-ID : 3.3.3.100
Configured Router-ID : 0.0.0.0
Confed-ID : 0
Cluster-ID : 0.0.0.0
No. of configured peers : 0
No. of pending config peers : 0
No. of established peers : 0
VRF RD : 3.3.3.3:3
Information for address family IPv4 Unicast in VRF vni100
Table Id : 0x3
Table state : UP
Peers Active-peers Routes Paths Networks Aggregates
0 0 11 11 1 0
Redistribution
None
Auto RT is configured
EVPN Auto RT is configured
Export RT list:
1:1 100:100
Import RT list:
2:2 100:100
EVPN Export RT list:
100:100
EVPN Import RT list:
100:100
MVPN Export RT list:
100:100
MVPN Import RT list:
100:100
Label mode: per-vrf
Import default limit : 1000
Import default prefix count : 1
```

```
Import default map : allow
Import default advertise-vpn : Yes
Import VRF advertise-vpn : Yes
Export default limit : 1000
Export default prefix count : 6
Export default map : NO_DEFAULT_ROUTE
Export default allow-vpn : Yes
Export VRF allow-vpn : Yes
Nexthop trigger-delay
critical 3000 ms
non-critical 10000 ms
```

To verify if a specific route is being leaked from VRF to another, use the **show bgp vrf** *vrf* **ipv4 unicast** *prefix* command and check the 'Imported to <number-of-destinations> destinations' field in the ouput logs.

```
switch# show bgp vrf vni100 ipv4 unicast 4.4.4.100
BGP routing table information for VRF vni100, address family IPv4 Unicast
BGP routing table entry for 4.4.4.100/32, version 50
Paths: (1 available, best #1)
Flags: (0x8008021e) (high32 00000000) on xmit-list, is in urib, is best urib rou
te, is in HW
vpn: version 97, (0x100002) on xmit-list
Multipath: iBGP
Advertised path-id 1, VPN AF advertised path-id 1
Path type: internal, path is valid, is best path, no labeled nexthop, in rib
Imported from 4.4.4.4:3:[5]:[0]:[0]:[32]:[4.4.4.100]/224
Imported to 2 destination(s)
Imported paths list: vni200 default
AS-Path: NONE, path sourced internal to AS
4.4.4.4 (metric 81) from 101.101.101.101 (101.101.101.101)
Origin IGP, MED not set, localpref 100, weight 0
Received label 100
Extcommunity: RT:1:1 RT:100:100 ENCAP:8 Router MAC:5254.00cd.a816
Originator: 4.4.4.4 Cluster list: 101.101.101.101
VRF advertise information:
Path-id 1 not advertised to any peer
VPN AF advertise information:
Path-id 1 not advertised to any peer
```

Additional References

Table 1: Related Documents

Related Topic	Document Title
Cisco NX-OS Licensing Guide	Cisco NX-OS Licensing Guide

Feature History for Centralized VRF Route Leaking

Table 2: Feature History for IP TCP MSS

Feature Name	Release	Feature Information
Centralized VRF Route Leaking	8.4(1)	VXLAN BGP EVPN uses MP-BGP and its route-policy concept to import and export prefixes. The ability of this very extensive route-policy model allows to leak routes from one VRF to another VRF and vice-versa; any combination of custom VRF or VRF default can be used. VRF route-leaking is a switch-local function at specific to a location in the network, the location where the cross-VRF route-target import/export configuration takes place (leaking point). The forwarding between the different VRFs follows the control-plane, the location of where the configuration for the route-leaking is performed- hence Centralized VRF route-leaking. With the addition of VXLAN BGP EVPN, the leaking point requires to advertise the cross-VRF imported/exported route and advertise them towards the remote VTEPs or External Routers.