Configure and Verify VXLAN VRF Leaking on Nexus 9000

Contents

Introduction

Background information

Prerequisites

Requirements

Components Used

Diagram

Default VRF to Tenant-VRF

- Verify routing table
- Filter route
  - Configure
- Import route to BGP
  - Configure
  - Verify BGP table
- Import route to Tenant VRF
  - Configure
- Summary Steps
- Verify
  - Verify route is imported to L2VPN.
  - Verify route is imported to Tenant VRF

Tenant-VRF to Default VRF

- Verify routing table
- Filter route
  - Configure
- Export route to default VRF from tenant-a VRF
  - Configure
- Summary Steps
- Verify
  - Verify route is imported to BGP IPV4 address family on Default VRF
  - Verify route is imported to default VRF routing table

Tenant-VRF to Tenant-VRF

- Verify routing table
- Filter route
  - Identify Route Target
  - Configure
- Import route to tenant-a VRF from tenant-a VRF
  - Configure
- Summary Steps
- Verify
  - Verify route is imported to BGP on tenant-b VRF
  - Verify route is imported to routing table on tenant-b VRF

Introduction

This document describes how to configure and verify VRF leaking on a VXLAN environment.

Background information

In a VXLAN (Virtual Extensible LAN) environment, connecting VXLAN hosts to external hosts from the fabric often requires the use of VRF leaking and Border Leaf devices.

VRF leaking is crucial for enabling communication between VXLAN hosts and external hosts while maintaining network segmentation and security.

The Border Leaf device serves as a gateway between the VXLAN fabric and external networks, playing a pivotal role in facilitating this communication.

The importance of VRF leaking in this scenario can be summarized with the next statements:

1. **Interconnection with External Networks**: VRF leaking allows VXLAN hosts within the fabric to communicate with external hosts outside the fabric. This enables access to resources, services, and applications hosted on external networks, such as the internet or other data centers.
2. **Network Segmentation and Isolation**: VRF leaking maintains network segmentation and isolation within the VXLAN fabric while enabling selective communication with external networks. This ensures that VXLAN hosts remain isolated from each other based on their VRF assignments while still being able to access external resources as needed.
3. **Policy Enforcement**: VRF leaking enables administrators to enforce network policies and access controls for traffic flowing between VXLAN hosts and external hosts. This ensures that communication uses predefined security policies and prevents unauthorized access to sensitive resources.
4. **Scalability and Flexibility**: VRF leaking enhances the scalability and flexibility of VXLAN deployments by allowing VXLAN hosts to seamlessly communicate with external hosts. It enables dynamic allocation and sharing of resources between VXLAN and external networks, adapting to changing network requirements without disrupting existing configurations.

Filtering routes in VRF (Virtual Routing and Forwarding) leaking is crucial for maintaining network security, optimizing routing efficiency, and preventing unintended data leakage. VRF leaking allows communication between virtual networks while keeping them logically separate.

The importance of filtering routes in VRF leaking is important can be summarized with the next statements:

1. **Security**: Filtering routes ensures that only specific routes are leaked between VRF instances, reducing the risk of unauthorized access or data breaches. By controlling which routes are allowed to cross VRF boundaries, administrators can enforce security policies and prevent sensitive information

from being exposed to unauthorized entities.

2. **Isolation**: VRFs are designed to provide network segmentation and isolation, allowing different tenants or departments to operate independently within the same physical infrastructure. Filtering routes in VRF leaking helps maintain this isolation by limiting the scope of route propagation between VRF instances, preventing unintended communication and potential security vulnerabilities.

3. **Optimized Routing**: Filtering routes allows administrators to selectively leak only the necessary routes between VRFs, optimizing routing efficiency and reducing unnecessary traffic across the network. By filtering out irrelevant routes, administrators can ensure that traffic uses the most efficient paths while minimizing congestion and latency.

4. **Resource Utilization**: By filtering routes, administrators can control the flow of traffic between VRF instances, optimizing resource utilization and bandwidth allocation. This helps prevent network congestion and ensures that critical resources are available for priority applications or services.

5. **Compliance**: Filtering routes in VRF leaking helps organizations maintain compliance with regulatory requirements and industry standards. By restricting the leakage of routes to only authorized entities, organizations can demonstrate compliance with data protection regulations and ensure the integrity of sensitive information.

6. **Granular Control**: Filtering routes provides administrators with granular control over the communication between VRF instances, allowing them to define specific policies based on their unique requirements. This flexibility enables organizations to tailor their network configurations to meet the needs of different applications, users, or departments.

Prerequisites

Existing VXLAN environment with a Border Router

Requirements

Cisco recommends that you have knowledge of these topics:

- NXOS Platform
- VXLAN
- VRF
- BGP

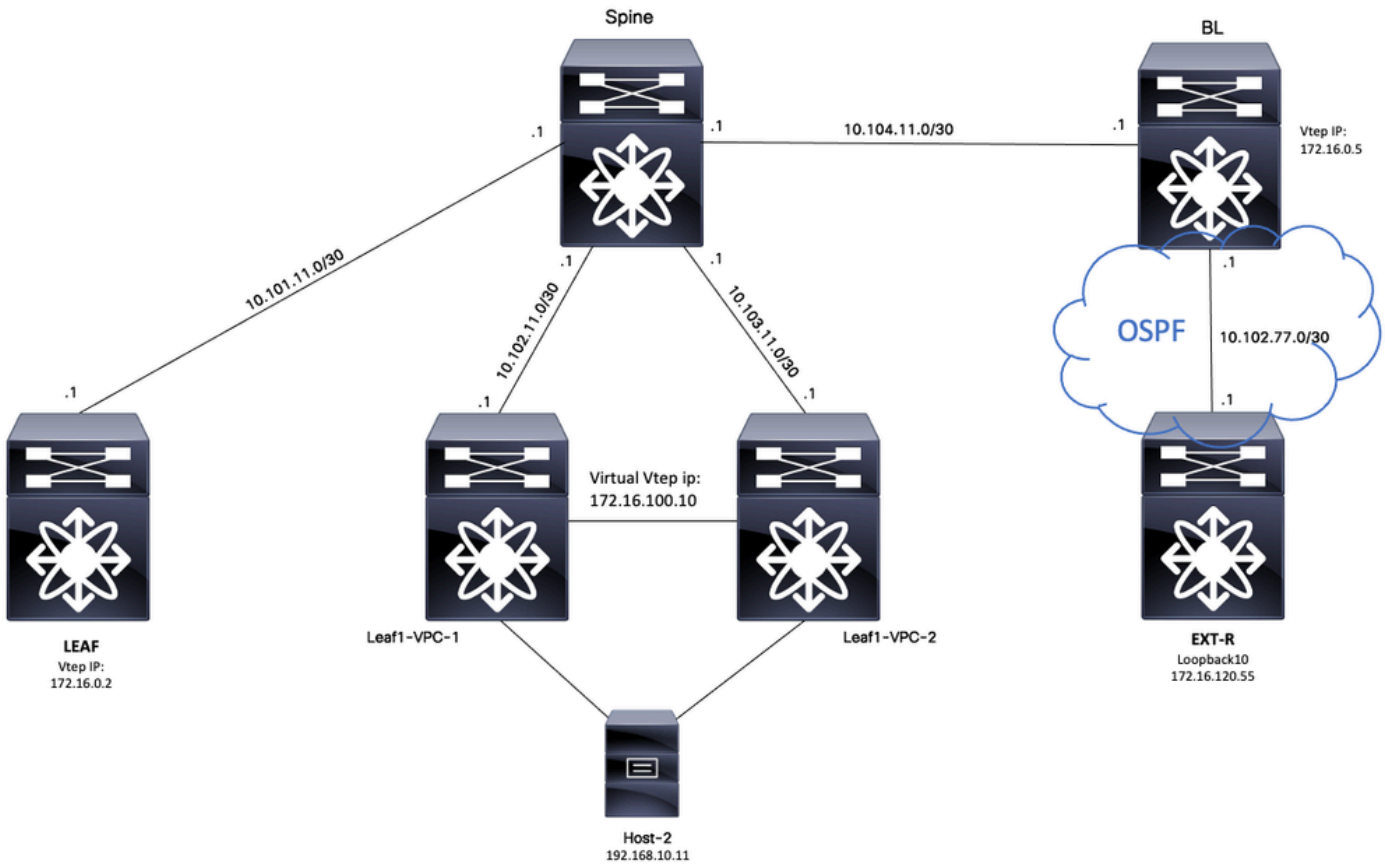
Components Used

The information in this document was created from the devices in a specific lab environment. All of the devices used in this document started with a cleared (default) configuration. If your network is live, ensure that you understand the potential impact of any command.

Name	Platform	Version

HOST-2	N9K-C92160YC-X	9.3(6)
Leaf-VPC-1	N9K-C93180YC-EX	9.3(9)
Leaf-VPC-2	N9K-C93108TC-EX	9.3(9)
LEAF	N9K-C9332D-GX2B	10.2(6)
BL	N9K-C9348D-GX2A	10.2(5)
EXT-R	N9K-C9348D-GX2A	10.2(3)
SPINE	N9K-C93108TC-FX3P	10.1(1)

Diagram



Considering BGP as an application, BGP is the application that is used to perform leak between VRFs

Default VRF to Tenant-VRF

For this example Border VTEP (**BL**) is receiving 172.16.120.55 from external device via OSFP in default VRF that is going to be leaked to Tenant VRF.

Verify routing table

```
BL# sh ip route 172.16.120.55
IP Route Table for VRF "default"
'*' denotes best ucast next-hop
'**' denotes best mcast next-hop
'[x/y]' denotes [preference/metric]
'%<string>' in via output denotes VRF <string>

172.16.120.55/32, ubest/mbest: 1/0
*via 10.105.100.2, Eth1/41.2, [110/2], 00:00:10, ospf-1, intra
```

Filter route

In NXOS a route-map is required as a parameter to filter and redistribute routes, for this example prefix 172.16.120.55/32 is going to be filtered.

Configure

	Command or Action	**Purpose**
Step 1	BL# configure terminal Enter configuration commands, one per line. End with CNTL/Z.	Enters configuration mode.
Step 2	BL(config)# ip prefix-list VXLAN-VRF-default-to-Tenant permit 172.16.120.55/32	Create prefix-list matching host.
Step 3	BL(config)# route-map VXLAN-VRF-default-to-Tenant	Create route-map.
Step 4	BL(config-route-map)# match ip address prefix-list VXLAN-VRF-default-to-Tenant	Match prefix-list created on step 2.

Import route to BGP

Once it is verified that route exist on default VRF, route must be imported to BGP process.

Configure

	Command or Action	**Purpose**

Step 1	BL# configure terminal Enter configuration commands, one per line. End with CNTL/Z.	Enters configuration mode.
Step 2	BL(config)# router bgp 65000	Enters BGP configuration.
Step 3	BL(config-router)# address-family ipv4 unicast	Enter BGP address-famili IPV4.
Step 4	BL(config-router-af)# redistribute ospf 1 route-map VXLAN-VRF-default-to-Tenant	Redistribute route from OSPF to BGP using route-map created on step 3.

Verify BGP table

```
BL(config-router-af)# show ip bgp 172.16.120.55
BGP routing table information for VRF default, address family IPv4 Unicast
BGP routing table entry for 172.16.120.55/32, version 16
Paths: (1 available, best #1)
Flags: (0x000002) (high32 00000000) on xmit-list, is not in urib

Advertised path-id 1
Path type: redist, path is valid, is best path, no labeled nexthop
AS-Path: NONE, path locally originated
0.0.0.0 (metric 0) from 0.0.0.0 (172.16.0.5)
Origin incomplete, MED 2, localpref 100, weight 32768
Extcommunity: OSPF RT:0.0.0.0:0:0
```

Import route to Tenant VRF

Once route it is imported to BGP, route now can be imported to target VRF (tenant-a).

Configure

	Command or Action	**Purpose**
Step 1	BL(config)# vrf context tenant-a	Enters VRF configuration.
Step 2	BL(config-vrf)# address-family ipv4 unicast	Enters IPV4 address family.

Step 3	BL(config-vrf-af-ipv4)# import vrf default map VXLAN-VRF-default-to-Tenant **advertise-vpn**	Import route from VRF default to Tenant VRF advertising VPN

⚠ **Caution**: By default, the maximum number of IP prefixes that can be imported from the default VRF into a non-default VRF is 1000 routes. This value can be changed with command under VRF address-family IPV4: import vrf <number **of prefixes**> default map <route-map name> advertise-vpn.

Summary Steps

1. configure terminal
2. ip prefix-list VXLAN-VRF-default-to-Tenant permit 172.16.120.55/32
3. route-map VXLAN-VRF-default-to-Tenant
4. match ip address prefix-list VXLAN-VRF-default-to-Tenant
5. router bgp 65000
6. address-family ipv4 unicast
7. redistribute ospf 1 route-map VXLAN-VRF-default-to-Tenant
8. vrf context tenant-a
9. address-family ipv4 unicast
10. import vrf default map VXLAN-VRF-default-to-Tenant **advertise-vpn**

Verify

Verify route is imported to L2VPN.

```
BL# sh bgp l2vpn evpn 172.16.120.55
BGP routing table information for VRF default, address family L2VPN EVPN
Route Distinguisher: 172.16.0.5:3 (L3VNI 303030)
BGP routing table entry for [5]:[0]:[0]:[32]:[172.16.120.55]/224, version 38
Paths: (1 available, best #1)
Flags: (0x000002) (high32 00000000) on xmit-list, is not in l2rib/evpn
Multipath: Mixed

Advertised path-id 1
Path type: local, path is valid, is best path, no labeled nexthop
Gateway IP: 0.0.0.0
AS-Path: NONE, path locally originated
172.16.0.5 (metric 0) from 0.0.0.0 (172.16.0.5)
Origin incomplete, MED 2, localpref 100, weight 32768
Received label 303030
Extcommunity: RT:65000:303030 ENCAP:8 Router MAC:20cf.ae54.fa3b
OSPF RT:0.0.0.0:0:0

Path-id 1 advertised to peers:
10.104.11.1
```

Verify route is imported to Tenant VRF

```
BL# sh ip route 172.16.120.55 vrf tenant-a
IP Route Table for VRF "tenant-a"
'*' denotes best ucast next-hop
'**' denotes best mcast next-hop
'[x/y]' denotes [preference/metric]
'%<string>' in via output denotes VRF <string>

172.16.120.55/32, ubest/mbest: 1/0
*via 172.16.0.5%default, [200/2], 00:02:47, bgp-65000, internal, tag 65000, segid: 303030 tunnelid: 0xa
```

Tenant-VRF to Default VRF

For this example Border VTEP (**BL**) is receiving route 192.168.10.11 via VXLAN on tenant-a VRF that going to be be leaked to default VRF.

Verify routing table

```
BL# sh ip route 192.168.10.11 vrf tenant-a
IP Route Table for VRF "tenant-a"
'*' denotes best ucast next-hop
'**' denotes best mcast next-hop
'[x/y]' denotes [preference/metric]
'%<string>' in via output denotes VRF <string>

192.168.10.11/32, ubest/mbest: 1/0
*via 172.16.100.10%default, [200/0], 01:15:04, bgp-65000, internal, tag 65000, segid: 303030 tunnelid:
```

Filter route

In NXOS a route-map is required as a parameter to filter and redistribute routes, for this example prefix 172.16.120.55/32 is going to be be filtered.

Configure

	Command or Action	**Purpose**
Step 1	BL# configure terminal	Enters configuration mode.

	Enter configuration commands, one per line. End with CNTL/Z.	
Step 2	BL(config)# ip prefix-list VXLAN-VRF-Tenant-to-default permit 192.168.10.11/32	Create prefix-list matching host.
Step 3	BL(config)# route-map VXLAN-VRF-Tenant-to-default	Create route-map.
Step 4	BL(config-route-map)# match ip address prefix-list VXLAN-VRF-Tenant-to-default	Match prefix-list created on step 2.

Export route to default VRF from tenant-a VRF

Since route is already on BGP L2VPN process it only needs to be exported to VRF default.

Configure

	Command or Action	**Purpose**
Step 1	BL# configure terminal Enter configuration commands, one per line. End with CNTL/Z.	Enters configuration mode.
Step 2	BL(config)# vrf context tenant-a	Enters VRF configuration.
Step 3	BL(config-vrf)# address-family ipv4 unicast	Enter VRF address-family IPV4.
Step 4	BL(config-vrf-af-ipv4)# export vrf default map VXLAN-VRF-Tenant-to-default **allow-vpn**	Export route from Tenant VRF to default VRF allowing VPN



Caution: By default, the maximum number of IP prefixes that can be exported from the non-

default VRF into a default VRF is 1000 routes. This value can be changed with command under VRF address-family IPV4: export vrf default <**number of prefixes**> map <route-map name> allow-vpn.

Summary Steps

1. configure terminal
2. ip prefix-list VXLAN-VRF-Tenant-to-default permit 192.168.10.11/32
3. route-map VXLAN-VRF-Tenant-to-default
4. match ip address prefix-list VXLAN-VRF-Tenant-to-default
5. vrf context tenant-a
6. address-family ipv4 unicast
7. export vrf default map VXLAN-VRF-Tenant-to-default **allow-vpn**

Verify

Verify route is imported to BGP IPV4 address family on Default VRF

```
BL(config-router-vrf-neighbor)# sh ip bgp 192.168.10.11
BGP routing table information for VRF default, address family IPv4 Unicast
BGP routing table entry for 192.168.10.11/32, version 55
Paths: (1 available, best #1)
Flags: (0x8000001a) (high32 00000000) on xmit-list, is in urib, is best urib route, is in HW

Advertised path-id 1
Path type: internal, path is valid, is best path, no labeled nexthop, in rib
Imported from 172.16.0.5:3:192.168.10.11/32 (VRF tenant-a)
Original source: 172.16.100.1:32777:[2]:[0]:[0]:[48]:[0027.e380.6059]:[32]:[192.168.10.11]/272
AS-Path: NONE, path sourced internal to AS
172.16.100.10 (metric 45) from 10.104.11.1 (192.168.0.11)
Origin IGP, MED not set, localpref 100, weight 0
Received label 101010 303030
Extcommunity: RT:65000:101010 RT:65000:303030 SOO:172.16.100.10:0 ENCAP:8
Router MAC:70db.9855.f52f
Originator: 172.16.100.1 Cluster list: 192.168.0.11

Path-id 1 not advertised to any peer
```

Verify route is imported to default VRF routing table

```
BL(config-router-vrf-neighbor)# show ip route 192.168.10.11
IP Route Table for VRF "default"
'*' denotes best ucast next-hop
'**' denotes best mcast next-hop
'[x/y]' denotes [preference/metric]
'%<string>' in via output denotes VRF <string>
```

```
192.168.10.11/32, ubest/mbest: 1/0
*via 172.16.100.10, [200/0], 00:03:51, bgp-65000, internal, tag 65000, segid: 303030 tunnelid: 0xac1064
```

Tenant-VRF to Default VRF

Tenant-VRF to Tenant-VRF

For this example nexus **LEAF** is receiving route 172.16.120.55/32 tenant-a that is going to be be leaked to VRF tenant-b

Verify routing table

```
show ip route 172.16.120.55/32 vrf tenant-a
IP Route Table for VRF "tenant-a"
'*' denotes best ucast next-hop
'**' denotes best mcast next-hop
'[x/y]' denotes [preference/metric]
'%<string>' in via output denotes VRF <string>

172.16.120.55/32, ubest/mbest: 1/0
*via 172.16.0.5%default, [200/2], 4d02h, bgp-65000, internal, tag 65000, segid: 303030 tunnelid: 0xac10
```

Filter route

In order to filter routes two steps are needed, the filtering between VRFs are done vie Route Targets (RT),RT is conformed by **<BGP Process ID>:L3VNI ID>** and filtering specific subnets. If second step is not used all routes from source VRF is going to be be leaked to destination VRF.



Note: The Route Target (RT) format is expressed as ASN:L3VNI, where ASN represents the Autonomous System Number of the tenant originating the route, and L3VNI is the Layer 3 VXLAN Network Identifier. However, for compatibility reasons when using 4-byte ASNs in BGP, the ASN portion is set to the special-purpose value 23456, known as AS_TRANS. This substitution ensures interoperability with devices that do not support 4-byte ASNs.

Identify Route Target

<#root>

```
LEAF# show nve vni
<Snipped>
```

```
Interface VNI Multicast-group State Mode Type [BD/VRF] Flags
--------- -------- ----------------- ----- ---- ------------------ -----
nve1 50500 n/a Up CP L3 [tenant-b]
nve1 101010 224.10.10.10 Up CP L2 [10]
nve1 202020 224.10.10.10 Up CP L2 [20]
nve1

303030

 n/a Up CP L3 [

tenant-a

]

LEAF# show run bgp | include ignore-case router
router bgp

65000


router-id 172.16.0.2
```

For this example Route Target equals to: **65000:303030** and route 172.16.120.55/32 is going to be filtered.

Configure

	Command or Action	Purpose
Step 1	LEAF# configure terminal Enter configuration commands, one per line. End with CNTL/Z.	Enters configuration mode.
Step 2	LEAF(config)# ip prefix-list filter-tenant-a-to-tenant-b permit 172.16.120.55/32	Create prefix-list matching host.
Step 3	LEAF(config)# route-map tenantA-to-tenantB	Create route-map.
Step 4	LEAF(config-route-map)# match ip address prefix-listfilter-tenant-a-to-tenant-b	Match prefix-list created on step 2.

Import route to tenant-a VRF from tenant-a VRF

Once RT is identified and filtering is configured, route can be imported to destination VRF **(tenant-b)**

Configure

	Command or Action	**Purpose**
Step 1	LEAF# configure terminal Enter configuration commands, one per line. End with CNTL/Z.	Enters configuration mode.
Step 2	LEAF(config)# vrf context tenant-b	Enters VRF configuration.
Step 3	LEAF(config-vrf)# address-family ipv4 unicast	Enter VRF address-family IPV4.
Step 4	LEAF(config-vrf-af-ipv4)# import map tenantA-to-tenantB	Import route filtered with route-map
Step 5	LEAF(config-vrf-af-ipv4)# route-target import 65000:303030	Import Route Target
Step 6	LEAF(config-vrf-af-ipv4)# route-target import 65000:303030 **evpn**	Import Route Target evpn



Caution: Not using an import map can allow all routes from origin VRF leaking them to target VRF. The use of import map can allow to control the routes to be leaked.

Summary Steps

1. configure terminal
2. ip prefix-list filter-tenant-a-to-tenant-b permit 172.16.120.55/32
3. route-map tenantA-to-tenantB
4. match ip address prefix-listfilter-tenant-a-to-tenant-b
5. vrf context tenant-b
6. address-family ipv4 unicast
7. import map tenantA-to-tenantB
8. route-target import 65000:303030
9. route-target import 65000:303030 **evpn**

Verify

Verify route is imported to BGP on tenant-b VRF

```
LEAF(config-vrf-af-ipv4)# show ip bgp 172.16.120.55/32 vrf tenant-b
BGP routing table information for VRF tenant-b, address family IPv4 Unicast
BGP routing table entry for 172.16.120.55/32, version 311
Paths: (1 available, best #1)
Flags: (0x8008021a) (high32 00000000) on xmit-list, is in urib, is best urib route, is in HW
vpn: version 456, (0x00000000100002) on xmit-list

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 172.16.0.5:3:[5]:[0]:[0]:[32]:[172.16.120.55]/224
AS-Path: NONE, path sourced internal to AS
172.16.0.5 (metric 45) from 10.101.11.1 (192.168.0.11)
Origin incomplete, MED 2, localpref 100, weight 0
Received label 303030
Extcommunity: RT:65000:303030 ENCAP:8 Router MAC:20cf.ae54.fa3b
OSPF RT:0.0.0.0:0:0
Originator: 172.16.0.5 Cluster list: 192.168.0.11

VRF advertise information:
Path-id 1 not advertised to any peer

VPN AF advertise information:
Path-id 1 not advertised to any peer
```

Verify route is imported to routing table on tenant-b VRF

```
LEAF# show ip route 172.16.120.55/32 vrf tenant-b
IP Route Table for VRF "tenant-b"
'*' denotes best ucast next-hop
'**' denotes best mcast next-hop
'[x/y]' denotes [preference/metric]
'%<string>' in via output denotes VRF <string>

172.16.120.55/32, ubest/mbest: 1/0
*via 172.16.0.5%default, [200/2], 00:00:08, bgp-65000, internal, tag 65000, segid: 303030 (Asymmetric)
```