Engineering Simplicity

Network Configuration Example

Remote Port Mirroring for EVPN-VXLAN Fabrics

Published
2022-09-29

Juniper Networks, Inc.
1133 Innovation Way
Sunnyvale, California 94089
USA
408-745-2000
www.juniper.net

YEAR 2000 NOTICE

END USER LICENSE AGREEMENT

Table of Contents

About This Guide

Use this network configuration example (NCE) to learn how to configure remote port mirroring for EVPN-VXLAN fabrics.

1

CHAPTER

Example: Configuring Remote Port Mirroring for EVPN-VXLAN Fabrics

How to Configure Remote Port Mirroring for EVPN-VXLAN Fabrics

IN THIS SECTION

About This Network Configuration Example

This network configuration example (NCE) shows how to configure remote port mirroring for EVPN-VXLAN fabrics. Port mirroring copies a traffic flow and sends it to a remote monitoring (RMON) station using a GRE tunnel. Like ERSPAN, remote port mirroring of the tenant traffic with VXLAN encapsulation is often used in the data center environment for troubleshooting or monitoring.

We demonstrate how to add remote port mirroring at lean spine switches and at the edge-routed leaf devices in an EVPN-VXLAN fabric. Our sample fabric is based on an edge-routed bridging (ERB) architecture. VXLAN port mirroring is also supported in centrally-routed bridging (CRB) fabrics.

SEE ALSO

Understanding Port Mirroring Analyzers

Example: Configuring an EVPN-VXLAN Edge-Routed Bridging Fabric With a Virtual Gateway

Use Case Overview

IN THIS SECTION

Remote Port Mirroring and EVPN-VXLAN Fabrics

Remote port mirroring copies a traffic flow and delivers the mirrored traffic to one or more mirroring destination hosts. The traffic flow is identified by the ingress or egress interface, in some cases using a firewall filter for added granularity. The mirroring destination hosts are connected to a leaf switch that is part of the same fabric as the source switch.

For EVPN-VXLAN IP fabrics, the mirrored traffic flow is encapsulated in generic routing encapsulation (GRE) and delivered through the underlay IP fabric to the monitoring host's IP address. You use this host as a monitoring station to capture and decode the traffic flow.

Remote port mirroring with a GRE tunnel is a perfect match for IP fabrics like EVPN-VXLAN because you can connect the monitoring station to any leaf switch in the fabric by advertising the host subnet into the EBGP underlay. Additionally, the destination switch connected to the monitoring station doesn’t have to perform GRE de-encapsulation before it delivers the GRE stream to the monitoring station. The GRE tunnel can cross multiple intermediate IP nodes or be sent outside of the fabric.

Port Mirroring Methods: Analyzer Instance and Port Mirroring Instance

Two methods are available for port mirroring: an analyzer instance and a port mirroring instance. Each approach offers different advantages and is compatible with different architectures. In both cases, the GRE tunnel for mirroring purposes is created automatically, so there is no need to configure a GRE tunnel.

Analyzer instance is port mirroring without any filtering criteria. An analyzer is the simplest form of traffic mirroring to implement. You only need to specify the interface that is the source of the mirrored traffic and whether the traffic to be mirrored on that interface is egress, ingress, or both. You also specify the IP address for the destination of the mirrored traffic. There is no need to configure or apply a firewall filter.

Because analyzer instance is not tenant-specific, it is the best approach when you do not have information about the tenant stream.

Port mirroring instance uses tenant traffic-specific criteria to mirror traffic. The administrator of the fabric decides which specific tenant source IP address, protocol, port, or so on, is of interest on the interface being mirrored. Use a port mirroring instance when specific traffic needs to be mirrored.

Juniper Networks supports three main data center architectures. All support remote port mirroring. For each architecture, we recommend the following method of port mirroring:

- Bridged Overlay: Analyzer instance
- Centrally-Routed Bridging (CRB): Port mirroring instance
- Edge-Routed Bridging (ERB): Port mirroring instance

Technical Overview

IN THIS SECTION

Remote Port Mirroring in an EVPN-VXLAN ERB Fabric

This NCE covers remote port mirroring for an ERB architecture with a lean spine.

In an EVPN-VXLAN ERB fabric with a lean spine, tenant-specific inner flows can't be selectively sent into the tunnel. Only outer header-based filtering (underlay) is supported at a lean spine device. For example, a spine switch can filter the packets coming from a specific source IPv4 loopback address and send the traffic to the monitoring station connected to another leaf device using the GRE tunnel.

Remote port mirroring with GRE for tenant-specific flow is supported on leaf devices in an ERB architecture. In this case, you can implement the remote port mirroring filter at the integrated routing and bridging (IRB) interface to mirror inter-VLAN (routed) traffic. To mirror intra-VLAN (bridged) traffic, apply an interface filter to the ingress interface attached to the server or host device.

In either method, spine or leaf, the mirrored traffic flows to the remote analyzer over a GRE tunnel.

Remote Port Mirroring with QFX Series Switches

In the following examples, we demonstrate different ways to use remote port mirroring for an ERB architecture based on QFX Series switches.

QFX5110 and QFX5120 switches perform well as leaf devices in an ERB reference data center architecture because these models can perform inter-virtual network identifier (VNI) routing.

Table 1 on page 5 shows port mirroring instance type support for various use cases when using QFX10002 and QFX10008 switches. Table 2 on page 5 shows the same when using QFX5110 and QFX5120 switches.

Table 1: Remote Port Mirroring on QFX10002 and QFX10008

Use Case	Sub Use Case	Analyzer Instance	Port Mirroring Instance
Spine providing IP transit	Ingress from leaf to spine	Supported	Supported
	Egress from spine to leaf	Supported	Supported
Spine in a CRB scenario	Ingress of IRB that terminates traffic	Only ge, xe, et and ae interfaces are supported	Not supported
	Egress of IRB that terminates traffic	Only ge, xe, et and ae interfaces are supported	Not supported
Border encapsulation	Access ingress of ESI-LAG	Supported	Supported
Border de-encapsulation	Egress of the fabric	Not supported	Not supported

Table 2: Remote Port Mirroring on QFX5110 and QFX5120

Use Case	Sub Use Case	Analyzer Instance	Port Mirroring Instance
Lean spine providing IP transit	Ingress from leaf to spine	Supported	Supported
	Egress from spine to leaf	Supported	Not supported
Leaf in an ERB scenario	Ingress of IRB that terminates traffic	Only ge, xe, et and ae interfaces are supported	Supported

Table 2: Remote Port Mirroring on QFX5110 and QFX5120 *(Continued)*

Use Case	Sub Use Case	Analyzer Instance	Port Mirroring Instance
	Egress of IRB that terminates traffic	Only ge, xe, et and ae interfaces are supported	Not supported
Border encapsulation	Access ingress of ESI-LAG	Supported	Supported
	Access egress of ESI-LAG	Supported	Not supported
Border de-encapsulation	Ingress of the fabric	Not supported	Not supported
	Egress of the fabric	Not supported	Not supported

All QFX switches in this example run Junos OS Release 18.4R2 or higher. QFX5110 and QFX5120 switches running Junos OS Release 18.4R2 support these scaling numbers:

- Default analyzer sessions: 4
- Port mirroring sessions: 3 port mirroring sessions + 1 global port mirroring session

This NCE covers the following use cases:

1. "Example: Ingress/Egress Solution for an EVPN-VXLAN ERB Fabric Spine Device" on page 9 shows you how to mirror ingress and egress traffic through a lean spine device in an ERB fabric.
2. "Example: Ingress Solution for an EVPN-VXLAN ERB Fabric Leaf Device" on page 21 demonstrates how to mirror ingress traffic through a leaf device in an ERB scenario.
3. "Example: Enable a Remote Mirroring Instance at an ESI-LAG Interface" on page 30 shows you how to use a port mirroring instance or an analyzer instance in the border encapsulation use case for access to ingress and egress traffic through an ESI-LAG interface.
4. "Example: Enable a Remote Analyzer Instance at an ESI-LAG Interface" on page 41 shows you how to use an analyzer instance to mirror ingress and egress traffic through an ESI-LAG interface.

Use the procedures described in these examples to enable remote port mirroring for your particular use case.

Topology

IN THIS SECTION

Requirements

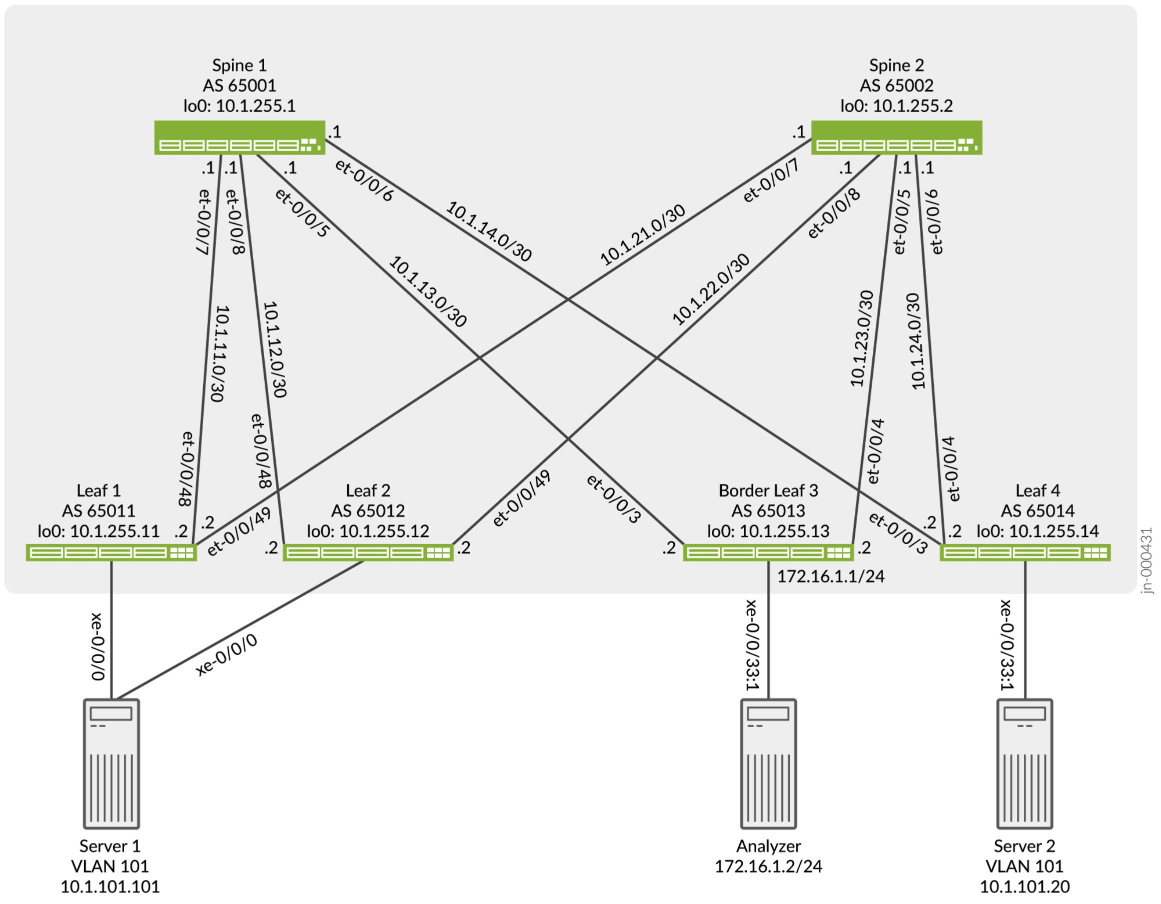
This configuration example uses the following devices running Junos OS Release 18.4R2 or higher. This example was re-validated on QFX switches running Junos 22.2R1.

- Two QFX5110 switches as lean spine devices.
- One QFX10002 switch as a border leaf device.
- Two QFX5110 or 5120 switches as server Leaf 1 and server Leaf 2 devices.
- One QFX10002 as server Leaf 4. If desired, you can use a QFX5110 or QFX5120 instead of a QFX10002.
- Two servers that send and receive traffic over the EVPN-VXLAN fabric.
- A monitoring station equipped with an analyzer application. In this example, we use Wireshark.

Baseline Topology

This NCE focuses on how to add different types of port mirroring to an existing EVPN-VXLAN fabric. Figure 1 on page 8 details the baseline topology for all of the following examples.

Figure 1: Port Mirroring Baseline Topology



Spines 1 and 2 are lean spines devices that don't perform any VXLAN encapsulation or de-encapsulation. Servers 1 and 2 share VLAN 101 and the 10.1.101.0/24 subnet. Server 1 begins single-homed to Leaf 1. In a later sections, we attach it to Leaf 2 to form an ESI-LAG interface. Server 2 attaches to Leaf 4. Leaf 3 functions as a border leaf. Its configuration is the same as those used on the server leaves. Note that in this example, Leaf 4 is a QFX10000 switch attached to the analyzer device on the xe-0/0/33:1 interface.

The topology supports multihoming with ESI-LAG between Server 1 and Leaves 1 and 2. Not all scenarios in this NCE use this multihomed capability. For the full starting device configurations, see "Appendix: Full Device Configurations" on page 51.

Example: Ingress/Egress Solution for an EVPN-VXLAN ERB Fabric Spine Device

IN THIS SECTION

Overview

In this example, we enable remote port mirroring on a lean spine switch. The mirrored traffic is sent through a GRE tunnel to the remote monitoring station. This use case is based on the port mirroring instance method.

This example uses an EVPN-VXLAN ERB architecture where unicast inter-virtual network identifier (VNI) routing takes place at the leaf devices. The lean spine devices do not terminate any VXLAN tunnels. They provide IP forwarding and, in the case of an IBGP-based overlay, route reflection capabilities. Our overlay is EBGP-based, so we do not use route reflection.

This configuration mirrors all traffic sent between the leaf devices. The monitoring station captures both intra-VLAN (bridged) and inter-VLAN (routed) traffic for all VLANs on the leaves. The tenant’s specific information is not provisioned at the lean spine devices, but you can enable the mirroring sessions at the lean spine.

Be aware of the capacity of the analyzer port. A large amount of traffic passes through the analyzer port because you are mirroring all overlay traffic sent between the leaf pairing. The leaf devices are attached to the spine with two 40GE interfaces, so they have the capacity to carry high volumes of traffic. Switches have limited buffering capabilities. Frame drops occur if you mirror more traffic than the monitoring station supports.

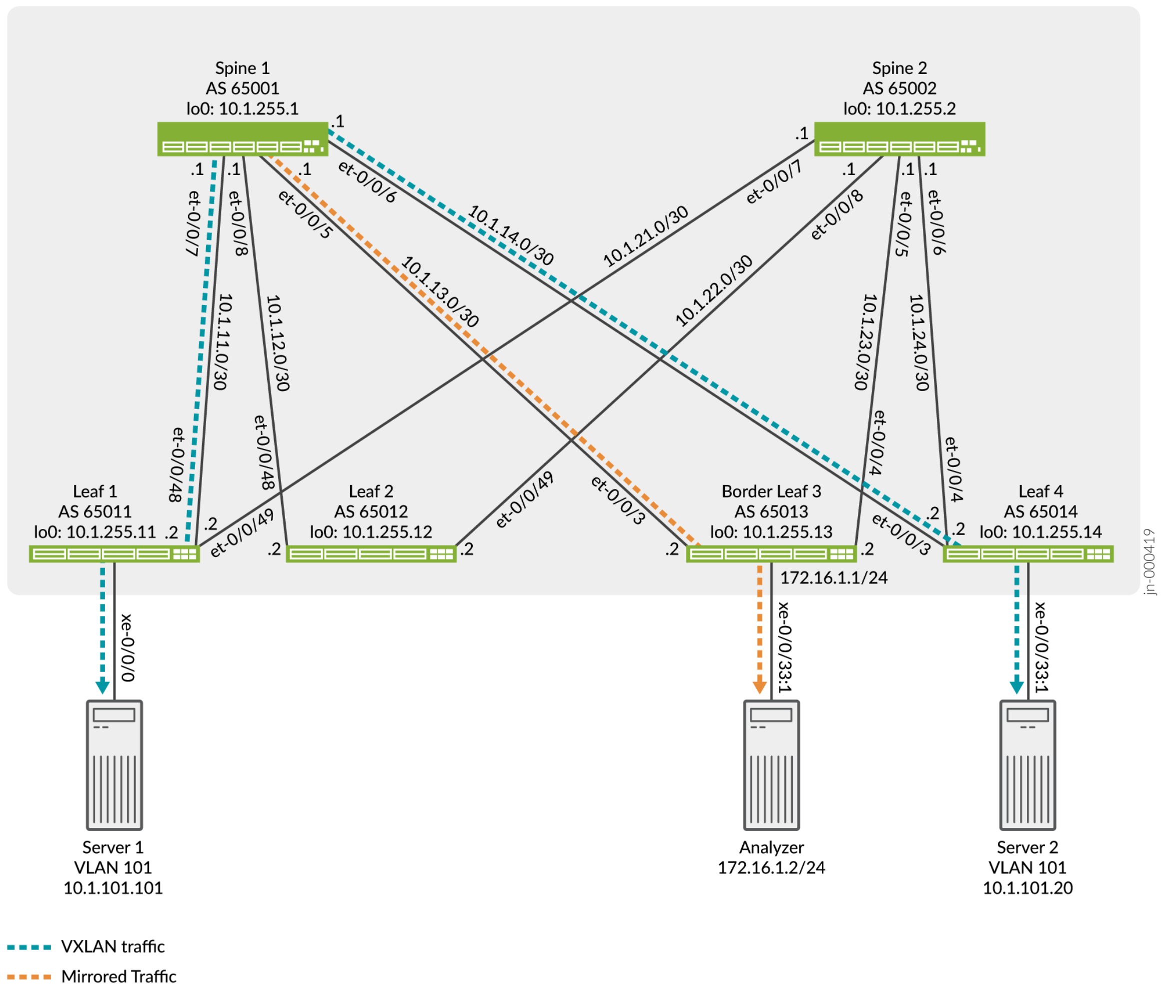
To add capacity, use an aggregated Ethernet interface with multiple link members. Reduce the traffic load by performing mirroring at the leaf devices and selectively mirroring traffic for only certain tenants.

Topology

This configuration example uses the hardware and software components shown in "Requirements" on page 7.The devices in the topology begin with their baseline configurations as provided in "Appendix: Full Device Configurations" on page 51.

As shown in Figure 2 on page 10, Spine 1 mirrors ingress and egress overlay traffic flowing between Leaf 1 and Leaf 4 through a firewall filter applied to its et-0/0/7 interface. This interface attaches to server Leaf 1. Overlay traffic is generated by sending ping traffic between Server 1 and Server 2. The mirrored traffic is sent by Spine 1 to the remote monitoring station connected to the xe-0/0/33:1 interface on border Leaf 3.

Figure 2: Topology of Remote Port Mirroring Through Spine Device



The spine uses a GRE tunnel to send the mirrored traffic to the monitoring station. GRE is necessary as the mirrored traffic may be Layer 2, or overlay Layer 3, and therefore not able to be directed routed over the fabric underlay. The GRE tunnel automatically forms once IP reachability towards the monitoring subnet 172.16.1.0/24 is established via the underlay.

Server 1 is single-homed to Leaf 1 in this scenario.

Configuration

Use this example to mirror traffic passing through Spine 1. To mirror all traffic, repeat the configuration on Spine 2. When port mirroring is configured on Spines 1 and 2, traffic is mirrored regardless of which ECMP fabric next hop is selected by the leaf.

1. (Optional) Deactivate BGP on Spine 2. For our testing purposes, we deactivate the BGP stanza on Spine 2. This way we are able to focus only on Spine 1 and ensure that all traffic between Leaf 1 and Leaf 4 is mirrored.

   ```
   user@spine2# deactivate protocols bgp
   user@spine2# commit
   ```

   After deactivating BGP on Spine 2, Leaf 1 has a single next hop to reach the loopback of Leaf 4. This is through Spine 1 via its et-0/0/48 interface.

   ```
   user@leaf1> show route 10.1.255.14

   inet.0: 11 destinations, 12 routes (11 active, 0 holddown, 0 hidden)
   + = Active Route, - = Last Active, * = Both

   10.1.255.14/32     *[BGP/170] 04:10:11, localpref 100
                         AS path: 65001 65014 I, validation-state: unverified
                       >  to 10.1.11.1 via et-0/0/48.0
   ```

2. Create a mirroring instance at Spine 1 to send the mirrored traffic to the remote monitoring station.

   ```
   user@spine1# set forwarding-options port-mirroring instance mirror-leaf1-and-leaf4 family
   inet output ip-address 172.16.1.2
   ```

3. Lean spines are not overlay-aware in an ERB fabric. All VXLAN-encapsulated traffic sent between leaf devices uses the local VTEP address. The VTEP address normally matches the device's loopback address. This means we can use a filter on the spine devices to match on the loopback addresses of the related leaf devices to direct traffic into the port mirroring instance.

At Spine 1, create firewall filters that match the source and destination IP addresses of Leaf 1 and Leaf 4. By matching on both the source and destination IP, all overlay traffic sent between these leafs is mirrored into the GRE tunnel.

NOTE: Because the lean spines are not VXLAN-aware, this method mirrors all overlay traffic sent by the leaf device. For example, if the leaf has 100 VLAN or VXLAN VNIs defined, then traffic for all 100 VNIs is mirrored. If you want to mirror at a finer level of granularity, see "Example: Ingress Solution for an EVPN-VXLAN ERB Fabric Leaf Device" on page 21. In that example, the mirror function occurs at the VXLAN-aware leaf device.

Set a firewall filter for traffic sent from Leaf 1. In our example, all VXLAN traffic sent by Leaf 1 (through the spines) has a source address of 10.1.255.11. All VXLAN traffic sent by Leaf 4 has a source address of 10.1.255.14.

```
[edit firewall family inet filter port-mirror-from-leaf1]
user@spine1# set term term1 from source-address 10.1.255.11/32
user@spine1# set term term1 from destination-address 10.1.255.14/32
user@spine1# set term term1 then count from-leaf1-vtep
user@spine1# set term term1 then port-mirror-instance mirror-leaf1-and-leaf4
user@spine1# set term term1 then accept
user@spine1# set term term2 then accept
```

Set a firewall filter for traffic sent from Leaf 4.

```
[edit firewall family inet filter port-mirror-from-leaf4]
user@spine1# set term term1 from source-address 10.1.255.14/32
user@spine1# set term term1 from destination-address 10.1.255.11/32
user@spine1# set term term1 then count from-leaf4-vtep
user@spine1# set term term1 then port-mirror-instance mirror-leaf1-and-leaf4
user@spine1# set term term1 then accept
user@spine1# set term term2 then accept
```

4. Apply the firewall filters to the Leaf 1 and Leaf 4-facing fabric interfaces at Spine 1.

```
user@spine1# set interfaces et-0/0/7 unit 0 family inet filter input port-mirror-from-leaf1
user@spine1# set interfaces et-0/0/6 unit 0 family inet filter input port-mirror-from-leaf4
```

Note the directionality of the filter application. Our filters accommodate a QFX5110 platform, which supports only ingress filters for port mirroring, as per Table 2 on page 5. By using input filters that match on the local leaf's VTEP as a source address and the remote leaf's VTEP as a destination address, we ensure that only overlay traffic between these two leaves is mirrored. If you wish to mirror all VXLAN traffic sent by a leaf, regardless of the destination leaf, simply match on the local leaf's source address in your filter.

5. (Optional) Add any additional filters.

   In this example so far, we have assumed that Server 1 is single-homed only to Leaf 1. If the server is multihomed to Leaf 1 and Leaf 2, you need to adapt the filters shown to also catch traffic sent from Leaf 2 to Leaf 4, and vice versa.

   As an example, these statements create an input filter for Spine 1's Leaf 2-facing interface:

   ```
   [edit firewall family inet filter port-mirror-from-leaf4]
   user@spine1# set term term1 from source-address 10.1.255.12/32
   user@spine1# set term term1 from destination-address 10.1.255.14/32
   user@spine1# set term term1 then count from-leaf2-vtep
   user@spine1# set term term1 then port-mirror-instance mirror-leaf1-and-leaf4
   user@spine1# set term term1 then accept
   user@spine1# set term term2 then accept
   ```

   ```
   [edit]
   user@spine1# set interfaces et-0/0/8 unit 0 family inet filter input port-mirror-from-leaf2
   ```

   A quick modification to the existing `port-mirror-from-leaf4` filter is needed to also match on the destination address of Leaf's 2 VTEP.

   ```
   [edit firewall family inet filter port-mirror-from-leaf4]
   user@spine1# set term term1 from destination-address 10.1.255.12/32
   ```

6. Commit the changes at Spine 1.

   The changes from the starting EVPN-VXLAN baseline are shown:

   ```
   user@spine1# show | compare rollback 1
   [edit interfaces et-0/0/6 unit 0 family inet]
   +       filter {
   +           input port-mirror-from-leaf4;
   +       }
   [edit interfaces et-0/0/7 unit 0 family inet]
   ```

```
+       filter {
+           input port-mirror-from-leaf1;
+       }
[edit]
+  forwarding-options {
+      port-mirroring {
+          instance {
+              mirror-leaf1-and-leaf4 {
+                  family inet {
+                      output {
+                          ip-address 172.16.1.2;
+                      }
+                  }
+              }
+          }
+      }
+  }
+  firewall {
+      family inet {
+          filter port-mirror-from-leaf1 {
+              term term1 {
+                  from {
+                      source-address {
+                          10.1.255.11/32;
+                      }
+                      destination-address {
+                          10.1.255.14/32;
+                      }
+                  }
+                  then {
+                      count from-leaf1-vtep;
+                      port-mirror-instance mirror-leaf1-and-leaf4;
+                      accept;
+                  }
+              }
+              term term2 {
+                  then accept;
+              }
+          }
+          filter port-mirror-from-leaf4 {
+              term term1 {
+                  from {
+                      source-address {
```

```
+                           10.1.255.14/32;
+                       }
+                       destination-address {
+                           10.1.255.11/32;
+                       }
+                   }
+                   then {
+                       count from-leaf4-vtep;
+                       port-mirror-instance mirror-leaf1-and-leaf4;
+                       accept;
+                   }
+               }
+               term term2 {
+                   then accept;
+               }
+           }
+       }
+   }
```

7. Modify the configuration at Leaf 3 to configure the xe-0/0/33:1 interface connected to the monitor station.

   ```
   user@bl-leaf3# set interfaces xe-0/0/33:1 unit 0 family inet address 172.16.1.1/24
   ```

   The leaf device connected to the monitoring station does not need explicit GRE configuration. This is because the GRE tunnel terminates at the monitoring station. You must ensure that the monitoring station's IP address is reachable in the fabric underlay. Stated differently, the spine cannot send the GRE encapsulated traffic to the monitor station if it does not have underlay reachability to the monitor station.

8. Modify the underlay export policy at border Leaf 3 to advertise the monitoring station's prefix. Our topology uses an EBGP underlay. We simply add a new term to the existing underlay export policy.

   ```
   [edit policy-options policy-statement send-direct]
   user@bl-leaf3# set term 2 from protocol direct
   user@bl-leaf3# set term 2 from route-filter 172.16.1.0/24 exact
   user@bl-leaf3# set term 2 then accept
   ```

9. Commit the changes on border Leaf 3.

The changes to the starting baseline are shown:

```
user@bl-leaf3# show | compare rollback 1
[edit interfaces]
+   xe-0/0/33:1 {
+       unit 0 {
+           family inet {
+               address 172.16.1.1/24;
+           }
+       }
+   }
[edit policy-options policy-statement send-direct]
     term 1 { ... }
+    term 2 {
+        from {
+            protocol direct;
+            route-filter 172.16.1.0/24 exact;
+        }
+        then accept;
+    }
```

Verification

1. Verify the underlay connectivity from Spine 1 to the monitoring station.

   On Spine 1, display the route to 172.16.1.0/24.

```
user@spine1> show route 172.16.1.0/24

inet.0: 16 destinations, 17 routes (16 active, 0 holddown, 0 hidden)
+ = Active Route, - = Last Active, * = Both

172.16.1.0/24      *[BGP/170] 00:32:00, localpref 100
                      AS path: 65013 I, validation-state: unverified
                    >  to 10.1.13.2 via et-0/0/5.0
```

NOTE: Strictly speaking, only simple connectivity between the spine and monitoring station is required. We have configured a static route on the monitoring station to allow it to reach the fabric underlay. This permits ping testing for fault isolation.

Ping the monitoring station to confirm connectivity.

```
user@spine1> ping 172.16.1.2 count 2
PING 172.16.1.2 (172.16.1.2): 56 data bytes
64 bytes from 172.16.1.2: icmp_seq=0 ttl=63 time=1.116 ms
64 bytes from 172.16.1.2: icmp_seq=1 ttl=63 time=1.046 ms

--- 172.16.1.2 ping statistics ---
2 packets transmitted, 2 packets received, 0% packet loss
round-trip min/avg/max/stddev = 1.046/1.081/1.116/0.035 ms
```

The output confirms expected underlay reachability from the spine to the monitoring station.

2. Confirm the port mirroring instances on Spine 1. On Spine 1, verify that the remote port mirroring state is `up`.

```
user@spine1> show forwarding-options port-mirroring detail
Instance Name: mirror-leaf1-and-leaf4
  Instance Id: 2
  Input parameters:
    Rate                  : 1
    Run-length            : 0
    Maximum-packet-length : 0
  Output parameters:
    Family              State     Destination          Next-hop
    inet                up        172.16.1.2           .local..0
```

The output confirms correct definition of the port mirroring instance. The state of `up` indicates that the spine has an underlay route to the monitoring station. Recall that GRE is a stateless protocol. This is why its a good idea to verify the connectivity between the spine and monitoring station, as we did in the previous step.

3. Verify filters applied to Spine 1 correctly reflect the test traffic.

Clear the counters just before generating any pings.

```
user@server1> clear firewall all
```

Start pings between Server 1 and Server 2 over the fabric.

```
user@server1> ping 10.1.101.20 count 10 rapid
PING 10.1.101.20 (10.1.101.20): 56 data bytes
!!!!!!!!!!
--- 10.1.101.20 ping statistics ---
10 packets transmitted, 10 packets received, 0% packet loss
round-trip min/avg/max/stddev = 0.731/0.800/1.300/0.167 ms
```

Now display the firewall counters on Spine 1. Verify the filters applied to Spine 1 correctly reflect the test traffic. Our example topology has only one VLAN and one pair of servers. This makes it easy to correlate test traffic to filter matches. Any non-zero count is a good sign the filter is working.

```
user@spine1> show firewall

Filter: port-mirror-from-leaf1
Counters:
Name                                                Bytes              Packets
from-leaf1-vtep                                      1520                   10

Filter: port-mirror-from-leaf4
Counters:
Name                                                Bytes              Packets
from-leaf4-vtep                                      1520                   10
```

The firewall counters correctly reflects the test traffic between the two servers.

4. Confirm that the traffic sent between Server 1 and Server 2 is mirrored at Spine 1 and sent to the monitoring Station.

   Once again generate traffic between Server 1 and 2 (not shown for brevity). Use the `rapid` option in your ping command to generate a larger volume of test traffic that makes correlation easier.

   Monitor interface traffic counts on the interface at border Leaf 3 that connects to the monitoring station. Confirm the egress packet counts reflect the generated test traffic.

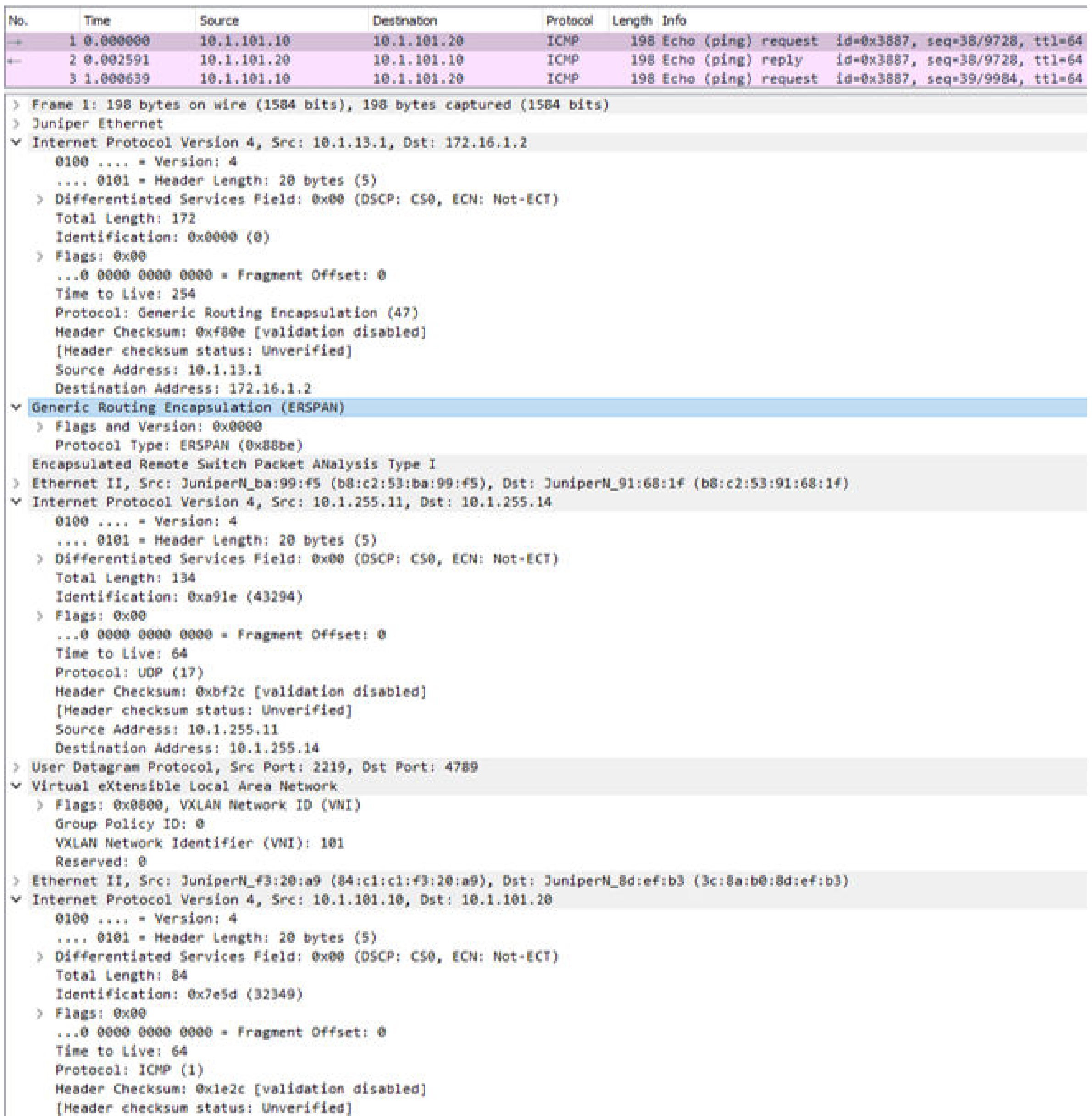
```
user@bl-leaf3> monitor interface xe-0/0/33:1
bl-leaf3                          Seconds: 17                  Time: 19:58:30
                                                            Delay: 1/0/1
Interface: xe-0/0/33:1, Enabled, Link is Up
Encapsulation: Ethernet, Speed: 10000mbps
Traffic statistics:                                              Current delta
```

```
    Input bytes:                    1019608 (0 bps)                            [0]
    Output bytes:                   5699872 (4752824 bps)                [3382190]
    Input packets:                    10115 (0 pps)                            [0]
    Output packets:                   34752 (3126 pps)                     [17801]
```

The output confirms traffic is mirrored to the monitor station. The lack of input packets is expected as the monitor station does not generate any responses to the received GRE traffic.

5. Use Wireshark or an equivalent analyzer application to capture and decode the mirrored traffic.

Figure 3: Mirrored Traffic Analysis



```
No.     Time        Source          Destination     Protocol  Length Info
→     1 0.000000    10.1.101.10     10.1.101.20     ICMP         198 Echo (ping) request  id=0x3887, seq=38/9728, ttl=64
←     2 0.002591    10.1.101.20     10.1.101.10     ICMP         198 Echo (ping) reply    id=0x3887, seq=38/9728, ttl=64
      3 1.000639    10.1.101.10     10.1.101.20     ICMP         198 Echo (ping) request  id=0x3887, seq=39/9984, ttl=64
```

```
> Frame 1: 198 bytes on wire (1584 bits), 198 bytes captured (1584 bits)
> Juniper Ethernet
v Internet Protocol Version 4, Src: 10.1.13.1, Dst: 172.16.1.2
    0100 .... = Version: 4
    .... 0101 = Header Length: 20 bytes (5)
  > Differentiated Services Field: 0x00 (DSCP: CS0, ECN: Not-ECT)
    Total Length: 172
    Identification: 0x0000 (0)
  > Flags: 0x00
    ...0 0000 0000 0000 = Fragment Offset: 0
    Time to Live: 254
    Protocol: Generic Routing Encapsulation (47)
    Header Checksum: 0xf80e [validation disabled]
    [Header checksum status: Unverified]
    Source Address: 10.1.13.1
    Destination Address: 172.16.1.2
v Generic Routing Encapsulation (ERSPAN)
  > Flags and Version: 0x0000
    Protocol Type: ERSPAN (0x88be)
  Encapsulated Remote Switch Packet ANalysis Type I
> Ethernet II, Src: JuniperN_ba:99:f5 (b8:c2:53:ba:99:f5), Dst: JuniperN_91:68:1f (b8:c2:53:91:68:1f)
v Internet Protocol Version 4, Src: 10.1.255.11, Dst: 10.1.255.14
    0100 .... = Version: 4
    .... 0101 = Header Length: 20 bytes (5)
  > Differentiated Services Field: 0x00 (DSCP: CS0, ECN: Not-ECT)
    Total Length: 134
    Identification: 0xa91e (43294)
  > Flags: 0x00
    ...0 0000 0000 0000 = Fragment Offset: 0
    Time to Live: 64
    Protocol: UDP (17)
    Header Checksum: 0xbf2c [validation disabled]
    [Header checksum status: Unverified]
    Source Address: 10.1.255.11
    Destination Address: 10.1.255.14
> User Datagram Protocol, Src Port: 2219, Dst Port: 4789
v Virtual eXtensible Local Area Network
  > Flags: 0x0800, VXLAN Network ID (VNI)
    Group Policy ID: 0
    VXLAN Network Identifier (VNI): 101
    Reserved: 0
> Ethernet II, Src: JuniperN_f3:20:a9 (84:c1:c1:f3:20:a9), Dst: JuniperN_8d:ef:b3 (3c:8a:b0:8d:ef:b3)
v Internet Protocol Version 4, Src: 10.1.101.10, Dst: 10.1.101.20
    0100 .... = Version: 4
    .... 0101 = Header Length: 20 bytes (5)
  > Differentiated Services Field: 0x00 (DSCP: CS0, ECN: Not-ECT)
    Total Length: 84
    Identification: 0x7e5d (32349)
  > Flags: 0x00
    ...0 0000 0000 0000 = Fragment Offset: 0
    Time to Live: 64
    Protocol: ICMP (1)
    Header Checksum: 0x1e2c [validation disabled]
    [Header checksum status: Unverified]
```

The capture shows that Server 1 sends ping traffic to Server 2, and that Server 2 replies. This confirms that overlay traffic sent between the Leaf 1 and Leaf 4 pairing is correctly mirrored at Spine 1. Note that the VXLAN encapsulated traffic (UDP port 4789) is in turn encapsulated into GRE (IP protocol 47). The protocol type of the GRE header is “ERSPAN” (comparable to remote port mirroring) and all the other flags are set to zero.

The decode allows you to see the leaf VTEP or loopback addresses used in the underlay (10.1.255.x). Also present is the original Layer 2 frame sent by the servers, now shown as encapsulated in VXLAN using VNI 101. The IP packet as sent by the servers is also decoded. The IP addresses used by the servers in the overlay are visible (10.1.100.x/24), as are the details of the IP/ICMP packets they generated.

You have successfully configured port mirroring on your topology.

Example: Ingress Solution for an EVPN-VXLAN ERB Fabric Leaf Device

IN THIS SECTION

Overview

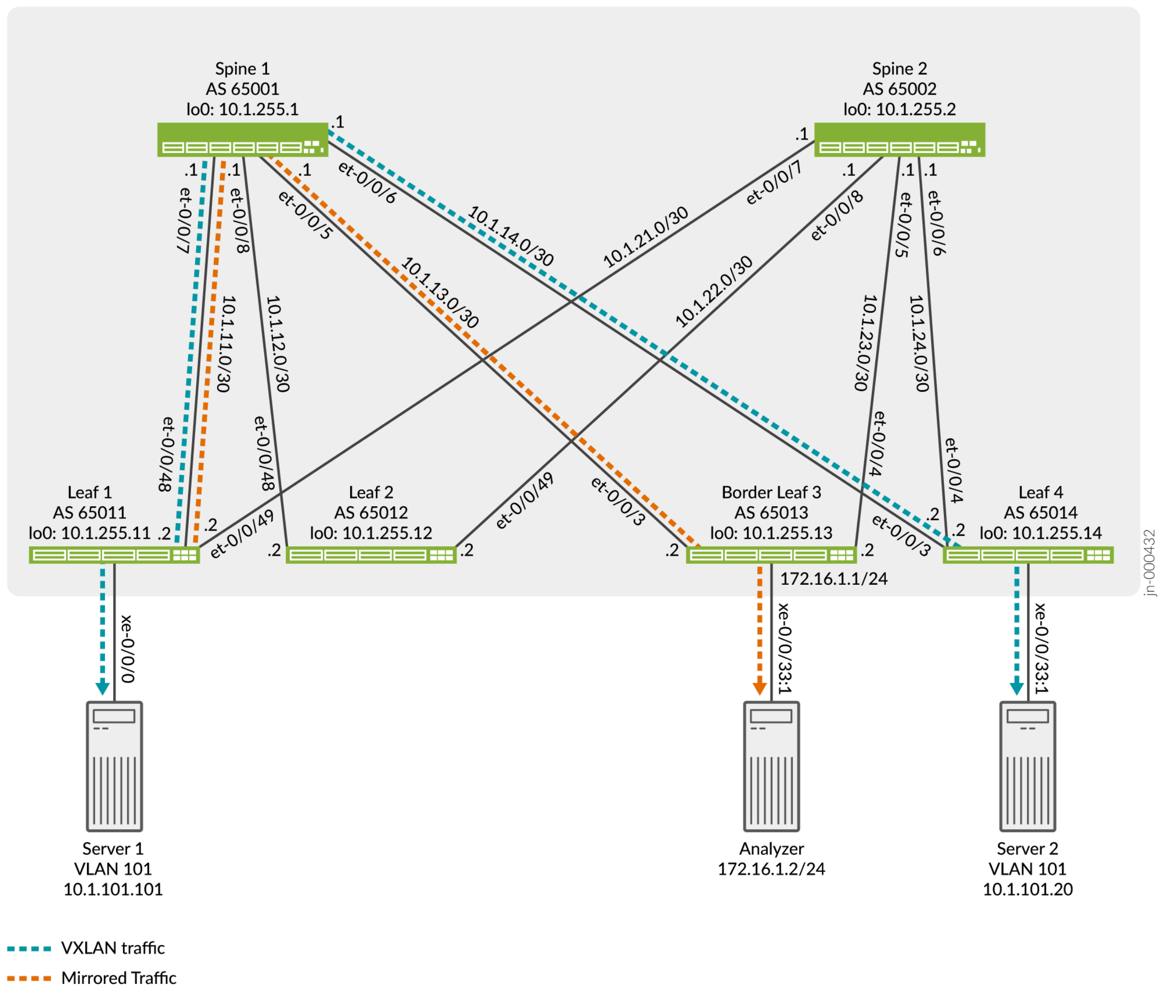
Use the following example to mirror traffic flowing through a leaf device on an EVPN-VXLAN ERB fabric. Unlike the lean spine-based port mirroring scenario covered in the first example, the leaf is tenant- and VLAN-aware in an ERB design. This means we can mirror specific traffic sent between our leaf devices rather than all overlay traffic. The tenant flows that match the filtering criteria on the ERB leaf device are mirrored into a GRE tunnel that terminates at the monitoring station attached to border Leaf 3.

Topology

This configuration example uses the hardware and software components shown in "Requirements" on page 7.The devices in the topology begin with their baseline configurations as provided in "Appendix: Full Device Configurations" on page 51.

Figure 4 on page 22 shows the topology for remote port mirroring with tenant flow filtering capabilities.

Figure 4: Topology of Remote Port Mirroring Through Leaf Device



Note that Server 1 is still single-homed only to Leaf 1. The difference is the filters and port mirroring instance are now defined on the leaf device. The goal is to match on all traffic sent by Server 1 to the 10.1.101.0/24 subnet associated with VLAN 101. Traffic sent from Server 1 to VLAN 101 destinations is matched on and sent to the port mirroring station at border Leaf 3.

Configuration

IN THIS SECTION

NOTE: In the first scenario, we deactivated BGP on Spine 2 to focus on filter application at Spine 1. In the remaining examples, both Spine 1 and Spine 2 are fully operational.

1. Configure a port mirroring instance at Leaf 1. Use IP address 172.16.1.2 for the monitoring station.

```
user@leaf1# set forwarding-options port-mirroring instance mirror-leaf1-v101 family inet
output ip-address 172.16.1.2
```

2. Next, create a firewall filter that matches on the subnet assigned to VLAN 101. The goal is to mirror all intra-VLAN 101 traffic sent by Server 1 through Leaf 1. To catch inter-VLAN traffic, specify the target VLAN's subnet for the destination address.

```
[edit firewall family ethernet-switching filter mirror-leaf-1]
user@leaf1# set term 1 from ip-source-address 10.1.101.0/24
user@leaf1# set term 1 from ip-destination-address 10.1.101.0/24
user@leaf1# set term 1 then accept
user@leaf1# set term 1 then port-mirror-instance mirror-leaf1-v101
user@leaf1# set term 1 then count from-s1-v101
user@leaf1# set term 2 then accept
```

NOTE: If you wish to mirror only routed traffic (inter-VLAN), use a `family inet` filter applied to the VLAN's IRB interface. Only one IRB interface is defined for each VLAN. A filter applied to the VLAN's IRB interface catches all inter-VLAN traffic for the associated VLAN, regardless of which front panel port the traffic arrives on.

The method shown here works for inter-VLAN and intra-VLAN traffic, but requires filters be applied to server facing interfaces based on their VLAN membership. An `inet` filter applied to an IRB interface does not see bridged (intra-VLAN) traffic.

3. Apply the remote port mirroring firewall filter to the access port attached to Server 1 in the input direction.

```
user@leaf1# set interfaces xe-0/0/0 unit 0 family ethernet-switching filter input mirror-
leaf-1
```

NOTE: On QFX5110 and QFX5120 switches, the firewall filter cannot be applied in the output direction or used on the fabric interfaces connected to the spine devices.

4. Border Leaf 3 does not need to be configured to perform GRE de-encapsulation since the GRE tunnel terminates at the monitoring station. However, Leaf 3 does need to advertise reachability for the monitoring station's subnet into the fabric's underlay. Configure the following on Leaf 3 to establish reachability to the monitoring station.

   Begin by configuring the xe-0/0/33:1 interface attached to the monitoring station.

```
user@bl-leaf3# set interfaces xe-0/0/33:1 unit 0 family inet address 172.16.1.1/24
```

5. Modify the underlay export policy at border Leaf 3 to advertise the monitoring station's prefix. Our topology uses an EBGP underlay. We simply add a new term to the existing underlay export policy.

```
[edit policy-options policy-statement send-direct]
user@bl-leaf3# set term 2 from protocol direct
user@bl-leaf3# set term 2 from route-filter 172.16.1.0/24 exact
user@bl-leaf3# set term 2 then accept
```

6. Commit the configuration.

   The changes from the starting baseline are shown at Leaf 1.

```
user@bl-leaf1# show | compare rollback 1
[edit interfaces xe-0/0/0 unit 0 family ethernet-switching]
+       filter {
+           input mirror-leaf-1;
+       }
[edit]
+  forwarding-options {
+      port-mirroring {
```

```
+         instance {
+             mirror-leaf1 {
+                 family inet {
+                     output {
+                         ip-address 172.16.1.2;
+                     }
+                 }
+             }
+         }
+     }
+  }
+  firewall {
+     family ethernet-switching {
+         filter mirror-leaf-1 {
+             term 1 {
+                 from {
+                     ip-source-address {
+                         10.1.101.0/24;
+                     }
+                     ip-destination-address {
+                         10.1.101.0/24;
+                     }
+                 }
+                 then {
+                     accept;
+                     port-mirror-instance mirror-leaf1;
+                     count from-s1;
+                 }
+             }
+             term 2 {
+                 then accept;
+             }
+         }
+     }
+  }
```

You have successfully configured remote port mirroring through a leaf device on an EVPN-VXLAN ERB fabric.

Verification

1. Verify underlay connectivity from Leaf 1 to the monitor station.

On Leaf 1, display the route to 172.16.1.0/24.

```
user@leaf1> show route 172.16.1.0/24

inet.0: 12 destinations, 13 routes (12 active, 0 holddown, 0 hidden)
+ = Active Route, - = Last Active, * = Both

172.16.1.0/24      *[BGP/170] 4d 00:56:23, localpref 100
                      AS path: 65001 65013 I, validation-state: unverified
                    >  to 10.1.11.1 via et-0/0/48.0
```

NOTE: Strictly speaking, only simple connectivity between the leaf and monitoring station is required. We have configured a static route on the monitoring station to allow it to reach the fabric underlay. This permits ping testing to aid in fault isolation.

Ping the monitoring station to confirm connectivity.

```
user@leaf1> ping 172.16.1.2 count 2 source 10.1.255.11
PING 172.16.1.2 (172.16.1.2): 56 data bytes
64 bytes from 172.16.1.2: icmp_seq=0 ttl=62 time=1.380 ms
64 bytes from 172.16.1.2: icmp_seq=1 ttl=62 time=7.614 ms

--- 172.16.1.2 ping statistics ---
2 packets transmitted, 2 packets received, 0% packet loss
round-trip min/avg/max/stddev = 1.380/4.497/7.614/3.117 ms
```

The output confirms expected underlay reachability from Leaf 1 to the monitoring station. We must source the ping from the leaf's loopback address because our underlay export policy only advertises loopback address reachability. Sourcing from the loopback address was not needed in the lean spine example. This is because the spine and border leaf are directly attached over a fabric link. Thus, the border leaf is able to route the ping reply back to the spine when it is sourced from the spine's leaf facing fabric interface.

2. Confirm the port mirroring instance on Leaf 1.

   On Leaf 1, verify that the remote port mirroring state is `up`.

```
user@leaf1> show forwarding-options port-mirroring detail
Instance Name: mirror-leaf1
```

```
  Instance Id: 2
  Input parameters:
    Rate                  : 1
    Run-length            : 0
    Maximum-packet-length : 0
  Output parameters:
    Family              State     Destination          Next-hop
    inet                up        172.16.1.2           et-0/0/48.0
```

The output confirms correct definition of the port mirroring instance. The state of `up` indicates that the leaf has an underlay route to the monitoring station. Recall that GRE is a stateless protocol. This is why its a good idea to verify the connectivity between the leaf and monitoring station, as we did in the previous step.

3. Verify the filter applied to Leaf 1 correctly reflects test traffic.

   Start pings between Server 1 and Server 2 over the fabric. Verify the filter applied to Leaf 1 correctly reflects the traffic. Our example topology has only one VLAN and one pair of servers. This makes it easy to correlate test traffic to filter matches. Any non-zero count is a good sign the filter is working.

```
user@server1> ping 10.1.101.20 count 10 rapid
PING 10.1.101.20 (10.1.101.20): 56 data bytes
!!!!!!!!!!
--- 10.1.101.20 ping statistics ---
10 packets transmitted, 10 packets received, 0% packet loss
round-trip min/avg/max/stddev = 0.731/0.800/1.300/0.167 ms
```

   Clear the counters just before generating any pings.

```
user@server1> clear firewall all
```

   Now display the firewall counters on Spine 1.

```
user@leaf1> show firewall

Filter: mirror-leaf-1
Counters:
Name                                                Bytes              Packets
from-s1                                              1020                   10
```

   The firewall counters correctly reflect the test traffic between the two servers.

4. Confirm that the traffic sent between Server 1 and Server 2 is mirrored at Leaf 1 to the monitoring station.

   Once again generate traffic between Server 1 and 2 (not shown for brevity). We used the rapid option for the ping command to generate a larger volume of packets to make detection easier.

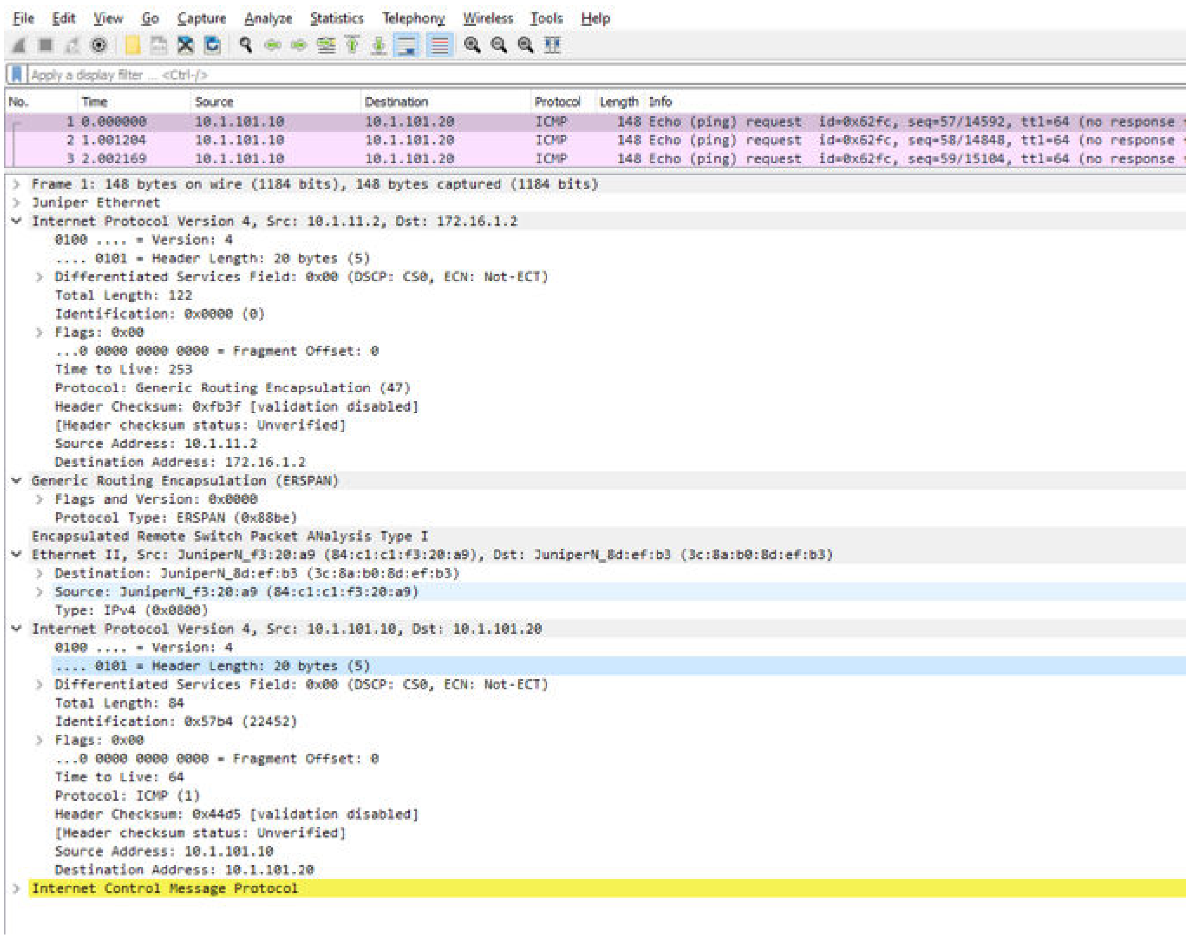
   Monitor interface traffic counters at border Leaf 3 to verify the test traffic is being sent to the monitoring station.

   ```
   user@bl-leaf3> monitor interface xe-0/0/33:1
   bl-leaf3                         Seconds: 17                 Time: 19:58:30
                                                            Delay: 1/0/1
   Interface: xe-0/0/33:1, Enabled, Link is Up
   Encapsulation: Ethernet, Speed: 10000mbps
   Traffic statistics:                                              Current delta
     Input bytes:                  1019608 (0 bps)                            [0]
     Output bytes:                 5699872 (4752824 bps)                [3382190]
     Input packets:                  10115 (0 pps)                            [0]
     Output packets:                 34752 (3126 pps)                     [17801]
   ```

   The output confirms traffic is mirrored to the monitor station. The lack of input packets is expected because the monitor station does not generate any responses to the GRE traffic it receives.

5. Use Wireshark or an equivalent analyzer application to capture and decode the mirrored traffic.

Figure 5: Mirrored Traffic Analysis


File Edit View Go Capture Analyze Statistics Telephony Wireless Tools Help
Apply a display filter ... <Ctrl-/>

```
No.     Time        Source          Destination     Protocol  Length Info
   1 0.000000    10.1.101.10     10.1.101.20     ICMP      148 Echo (ping) request  id=0x62fc, seq=57/14592, ttl=64 (no response
   2 1.001204    10.1.101.10     10.1.101.20     ICMP      148 Echo (ping) request  id=0x62fc, seq=58/14848, ttl=64 (no response
   3 2.002169    10.1.101.10     10.1.101.20     ICMP      148 Echo (ping) request  id=0x62fc, seq=59/15104, ttl=64 (no response

> Frame 1: 148 bytes on wire (1184 bits), 148 bytes captured (1184 bits)
> Juniper Ethernet
v Internet Protocol Version 4, Src: 10.1.11.2, Dst: 172.16.1.2
    0100 .... = Version: 4
    .... 0101 = Header Length: 20 bytes (5)
  > Differentiated Services Field: 0x00 (DSCP: CS0, ECN: Not-ECT)
    Total Length: 122
    Identification: 0x0000 (0)
  > Flags: 0x00
    ...0 0000 0000 0000 = Fragment Offset: 0
    Time to Live: 253
    Protocol: Generic Routing Encapsulation (47)
    Header Checksum: 0xfb3f [validation disabled]
    [Header checksum status: Unverified]
    Source Address: 10.1.11.2
    Destination Address: 172.16.1.2
v Generic Routing Encapsulation (ERSPAN)
  > Flags and Version: 0x0000
    Protocol Type: ERSPAN (0x88be)
  Encapsulated Remote Switch Packet ANalysis Type I
v Ethernet II, Src: JuniperN_f3:20:a9 (84:c1:c1:f3:20:a9), Dst: JuniperN_8d:ef:b3 (3c:8a:b0:8d:ef:b3)
  > Destination: JuniperN_8d:ef:b3 (3c:8a:b0:8d:ef:b3)
  > Source: JuniperN_f3:20:a9 (84:c1:c1:f3:20:a9)
    Type: IPv4 (0x0800)
v Internet Protocol Version 4, Src: 10.1.101.10, Dst: 10.1.101.20
    0100 .... = Version: 4
    .... 0101 = Header Length: 20 bytes (5)
  > Differentiated Services Field: 0x00 (DSCP: CS0, ECN: Not-ECT)
    Total Length: 84
    Identification: 0x57b4 (22452)
  > Flags: 0x00
    ...0 0000 0000 0000 = Fragment Offset: 0
    Time to Live: 64
    Protocol: ICMP (1)
    Header Checksum: 0x44d5 [validation disabled]
    [Header checksum status: Unverified]
    Source Address: 10.1.101.10
    Destination Address: 10.1.101.20
> Internet Control Message Protocol
```

The capture shows that Server 1 sends ping traffic to Server 2. The reply traffic is not mirrored because we have applied our filter in the input direction at Leaf 1 only. Apply a similar port mirror instance and filter configuration at Leaf 4 to also mirror return traffic. Recall that in the lean spine case the mirrored traffic showed VXLAN encapsulation. In our ERB leaf case we mirror Layer 2 traffic before it has been encapsulated into VXLAN. The filter used to mirror traffic is applied as an input filter on the server-facing interface. At ingress there is no VXLAN encapsulation.

This is similar to the case of applying the filter to a VLAN's IRB interface. In the IRB case the mirrored traffic does not contain VXLAN encapsulation. IRB based filters process inter-VLAN traffic at the IP layer. Thus, for the IRB filter case only Layer 3 IP traffic is mirrored.

The decode shows the GRE tunnel is between Leaf 1 and the monitoring station through the underlay. The Ethernet frame and related IP packet, as sent by Server 1, is decoded. The IP addresses assigned to the servers are visible (10.1.100.x/24), as are the details of the IP/ICMP packet being generated by Server 1.

You have successfully configured port mirroring on your topology.

Example: Enable a Remote Mirroring Instance at an ESI-LAG Interface

IN THIS SECTION

Overview

An Ethernet segment identifier (ESI) is the unique 10-byte number that identifies an Ethernet segment in an EVPN-VXLAN fabric connected to the server. The ESI is configured on the interface that connects to the server. When multiple links form a link aggregation group (LAG) and are connected to the server, this forms an ESI-LAG, also known as EVPN LAG, interface. In some cases, you will need to enable port mirroring for all or part of the traffic entering or leaving the ESI-LAG interface.

Use this section to enable remote port mirroring at the ESI-LAG level of an EVPN-VXLAN ERB fabric.

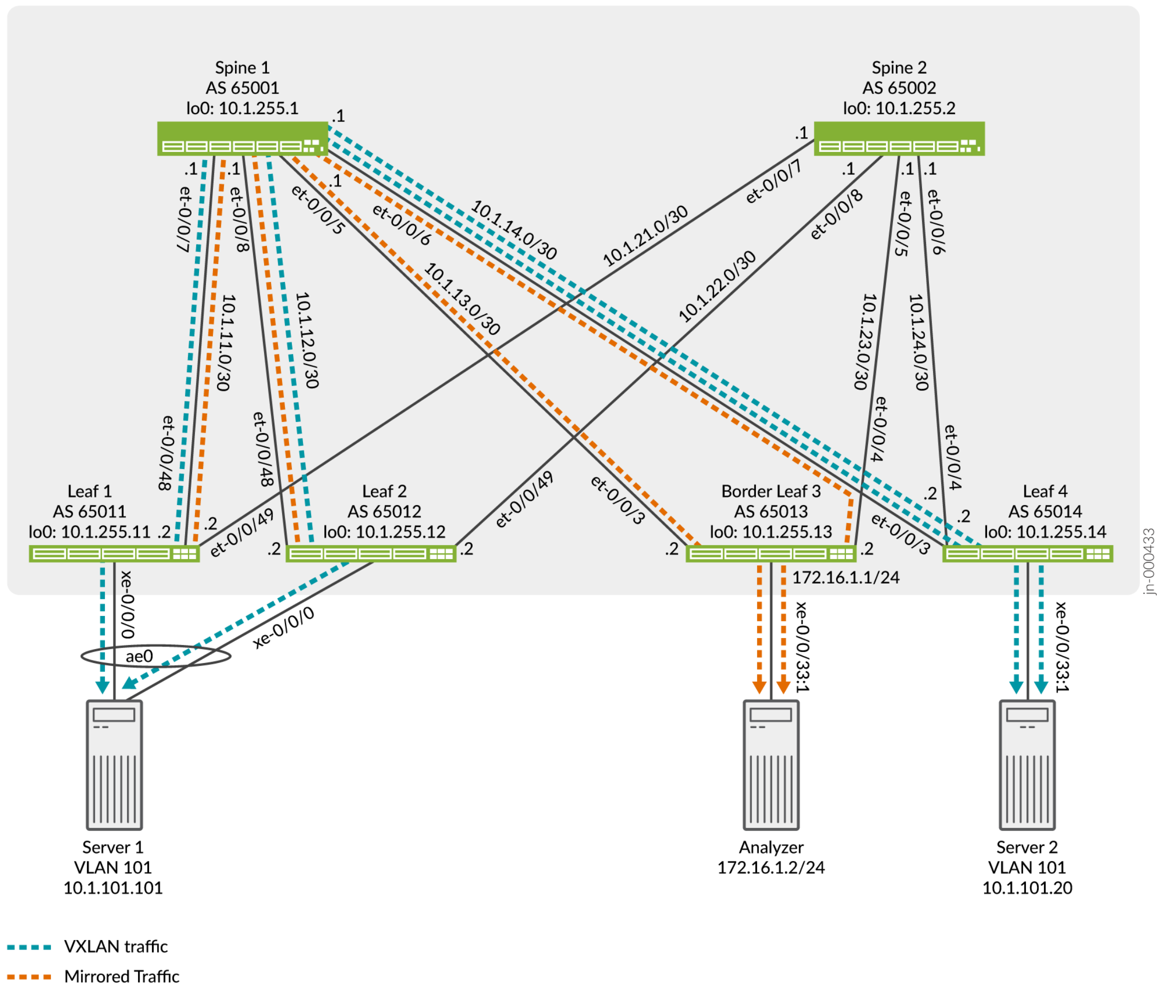
Topology

This configuration example uses the hardware and software components shown in "Requirements" on page 7. The devices in the topology begin with their baseline configurations as provided in "Appendix: Full Device Configurations" on page 51.

In this case we modify the EVPN fabric to support multihoming of Server 1 to leaves 1 and 2. We do this using ESI-LAG, also known as EVPN LAG, interfaces. See EVPN LAG Configuration Best Practices for best practices.

Figure 6 on page 31 shows the topology for this example. Both spines are operational in this topology. ECMP means that Leaf 1 and Leaf 2 can send traffic through either spine device. To simplify the illustration, we show the traffic and mirror paths with Spine 1 as the focus.

Figure 6: Topology of Remote Port Mirroring at ESI-LAG Interface



Interface ae0.0 is the ESI-LAG interface attached to Server 1. Server 1 is assigned IP address 10.1.101.10/24, and is associated with VLAN 101. The goal is to filter on, then mirror, traffic sent from Server 1 as it passes through ae0.0 on either leaf. The mirrored traffic is placed into a GRE tunnel and sent to the monitoring station. As before, you must ensure the monitoring station subnet is reachable in the EBGP underlay.

Before You Begin

If you are starting from the baseline configuration shown in "Appendix: Full Device Configurations" on page 51, follow these steps to modify your topology before you begin.

1. Modify the configuration of Leaf 1 and Leaf 2 to support multihomed attachment of Server 1. Begin by renaming the existing xe-0//0/0 interface to become the new ae0.0 interface. This caries over any configuration from the xe-0/0/0 interface to save some time.

```
user@leaf1# rename interfaces xe-0/0/0 to ae0
user@leaf1# set interfaces ae0 esi 00:02:02:02:02:02:02:02:00:04
user@leaf1# set interfaces ae0 esi all-active
user@leaf1# set interfaces ae0 aggregated-ether-options lacp active
user@leaf1# set interfaces ae0 aggregated-ether-options lacp system-id 00:00:02:02:00:04
```

2. The configuration specifics for the server's aggregated Ethernet interface, often called a bonded interface in Linux, is beyond the scope of this NCE. Note that for the server, there is only a standard aggregated Ethernet configuration. The statements related to ESI-LAG are applied only to the leaf devices.

NOTE: The starting baseline configuration enables an aggregated Ethernet interface through the `set chassis aggregated-devices ethernet device-count 1` configuration statement. You must enable chassis support for aggregated devices or the aggregated interface is not created.

3. After applying and committing these changes to both leaf devices and the server, confirm the ae0 interface is up on both leaves. The example output below has been shortened for clarity.

```
user@leaf2> show interfaces ae0
Physical interface: ae0, Enabled, Physical link is Up
  Physical interface: ae0, Enabled, Physical link is Up
[...]
  Current address: b8:c2:53:ba:8b:80, Hardware address: b8:c2:53:ba:8b:80
  Ethernet segment value: 00:02:02:02:02:02:02:02:00:04, Mode: all-active
  Last flapped   : 2022-09-07 14:57:53 UTC (01:06:43 ago)
[...]
  Egress queues: 12 supported, 5 in use
  Queue counters:       Queued packets  Transmitted packets      Dropped packets
    0                            81129                81129                    0
    3                                0                    0                    0
    4                                0                    0                    0
    7                             4135                 4135                    0
    8                               64                   64                    0
[...]
    Protocol eth-switch, MTU: 1514, Generation: 266, Route table: 7, Mesh Group: __all_ces__,
EVPN multi-homed status: Forwarding,
    EVPN multi-homed ESI Split Horizon Label: 0, Next-hop: 1665, vpls-status: up
    Local Bias Logical Interface Name: vtep.32769, Index: 554, VXLAN Endpoint Address:
10.1.255.11
      Flags: Is-Primary
```

Also verify that Server 1 is able to ping Server 2 (not shown for brevity).

Configuration

This example shows how to enable remote port mirroring at the ESI-LAG level using a remote port mirroring instance. This method supports tenant-specific (within an overlay VLAN) match criteria to selectively mirror traffic.

1. Configure a port mirroring instance at both Leaf 1 and Leaf 2. Server 1 can send traffic for VLAN 101 over either of its LAG member interfaces. ECMP means that traffic can arrive at either leaf. IP address 172.16.1.2 is the monitoring station.

```
user@leaf1# set forwarding-options port-mirroring instance mirror-leaf1-v101 family inet
output ip-address 172.16.1.2
```

2. Next, create a firewall filter that matches on the subnet assigned to VLAN 101 on both leaves. The goal is to mirror all intra-VLAN 101 traffic sent by Server 1 through Leaf 1 or Leaf 2. To catch inter-VLAN traffic, specify the target VLAN's subnet for the destination address.

   If you need to filter on multiple source or destination IP addresses, consider using a `source-prefix-list` and/or a `destination-prefix-list` in your filter. This way you can make changes to the prefix list without having to modify the filter definition.

```
[edit firewall family ethernet-switching filter mirror-leaf-1]
user@leaf1# set term 1 from ip-source-address 10.1.101.0/24
user@leaf1# set term 1 from ip-destination-address 10.1.101.0/24
user@leaf1# set term 1 then accept
user@leaf1# set term 1 then port-mirror-instance mirror-leaf1-v101
user@leaf1# set term 1 then count from-s1-v101
user@leaf1# set term 2 then accept
```

NOTE: If you want to mirror only routed traffic (inter-VLAN), use a `family inet` filter applied to the VLAN's IRB interface. Only one IRB interface is defined for each VLAN. A filter applied to the VLAN's IRB interface therefore catches all inter-VLAN traffic for the associated VLAN, regardless of which front panel port the traffic ingresses on.

The method shown in this example works for inter-VLAN and intra-VLAN traffic, but requires filters be applied to server-facing interfaces based on their VLAN membership. An `inet` filter applied to an IRB interface does not see bridged (intra-VLAN) traffic.

3. Apply the remote port mirroring firewall filter to the ae0.0 interface as an input filter on both leaves.

```
user@leaf1# set interfaces ae0 unit 0 family ethernet-switching filter input mirror-leaf-1
```

NOTE: On QFX5110 and QFX5120 switches, the firewall filter cannot be enabled in the outgoing direction or used at the interfaces connected to the spine devices.

4. Border Leaf 3 does not need to be configured to perform GRE de-encapsulation since the GRE tunnel terminates at the monitoring station. However, Leaf 3 does need to advertise reachability for the monitoring station's subnet into the fabric’s underlay. Configure the xe-0/0/33:1 interface that attaches to the monitoring station.

```
user@bl-leaf3# set interfaces xe-0/0/33:1 unit 0 family inet address 172.16.1.1/24
```

   Modify the underlay export policy at border Leaf 3 to advertise the monitoring station's prefix. Our topology uses an EBGP underlay. We simply add a new term to the existing underlay export policy.

```
[edit policy-options policy-statement send-direct]
user@bl-leaf3# set term 2 from protocol direct
user@bl-leaf3# set term 2 from route-filter 172.16.1.0/24 exact
user@bl-leaf3# set term 2 then accept
```

5. The changes to the starting baseline are shown at Leaf 1.

```
#user@leaf1# show | compare base
[edit interfaces xe-0/0/0]
+    gigether-options {
+        802.3ad ae0;
+    }
[edit interfaces xe-0/0/0]
-    unit 0 {
-        family ethernet-switching {
-            vlan {
-                members v101;
-            }
-        }
-    }
[edit interfaces]
+   ae0 {
+       esi {
+           00:02:02:02:02:02:02:02:00:04;
+           all-active;
+       }
+       aggregated-ether-options {
+           lacp {
+               active;
+               system-id 00:00:02:02:00:04;
+           }
+       }
+       unit 0 {
+           family ethernet-switching {
+               interface-mode access;
+               vlan {
+                   members v101;
+               }
+               filter {
+                   input mirror-leaf-1;
+               }
+           }
+       }
+   }
[edit]
+  forwarding-options {
+      port-mirroring {
+          instance {
+              mirror-leaf1 {
+                  family inet {
+                      output {
+                          ip-address 172.16.1.2;
+                      }
+                  }
+               }
+           }
+       }
+   }
+   firewall {
+       family ethernet-switching {
+           filter mirror-leaf-1 {
+               term 1 {
+                   from {
+                       ip-source-address {
+                           10.1.101.0/24;
+                       }
+                       ip-destination-address {
+                           10.1.101.0/24;
+                       }
+                   }
+                   then {
+                       accept;
+                       port-mirror-instance mirror-leaf1;
+                       count from-s1;
+                   }
+               }
+               term 2 {
+                   then accept;
+               }
+           }
+       }
+   }
```

You have successfully configured remote port mirroring through a leaf device on an EVPN-VXLAN ERB fabric.

Verification

1. Verify underlay connectivity from Leaf 1 to the monitoring station.

   On Leaf 1, display the route to 172.16.1.0/24.

```
user@leaf1> show route 172.16.1.0/24

inet.0: 12 destinations, 13 routes (12 active, 0 holddown, 0 hidden)
+ = Active Route, - = Last Active, * = Both
```

```
172.16.1.0/24      *[BGP/170] 4d 00:56:23, localpref 100
                      AS path: 65001 65013 I, validation-state: unverified
                    >  to 10.1.11.1 via et-0/0/48.0
```

NOTE: Strictly speaking, only simple connectivity is required between the leaf and monitoring station. We have configured a static route on the monitoring station to allow it to reach the fabric underlay. This permits ping testing as a fault isolation tool.

Ping the monitoring station to confirm connectivity.

```
user@leaf1> ping 172.16.1.2 count 2 source 10.1.255.11
PING 172.16.1.2 (172.16.1.2): 56 data bytes
64 bytes from 172.16.1.2: icmp_seq=0 ttl=62 time=1.380 ms
64 bytes from 172.16.1.2: icmp_seq=1 ttl=62 time=7.614 ms

--- 172.16.1.2 ping statistics ---
2 packets transmitted, 2 packets received, 0% packet loss
round-trip min/avg/max/stddev = 1.380/4.497/7.614/3.117 ms
```

The output confirms expected underlay reachability from Leaf 1 to the monitoring station. We must source the ping from the leaf's loopback address because our underlay export policy only advertises loopback address reachability. Sourcing from the loopback address is not needed in the lean spine case. This is because the spine and border leaf share a directly connected fabric link.

2. Confirm the port mirroring instance on Leaf 1.

   On Leaf 1, verify that the remote port mirroring state is `up`.

```
user@leaf1> show forwarding-options port-mirroring detail
Instance Name: mirror-leaf1
  Instance Id: 2
  Input parameters:
    Rate                  : 1
    Run-length            : 0
    Maximum-packet-length : 0
  Output parameters:
    Family              State     Destination          Next-hop
    inet                up        172.16.1.2           et-0/0/48.0
```

The output confirms correct definition of the port mirroring instance. The state of `up` indicates that the leaf has an underlay route to the monitoring station. Recall that GRE is a stateless protocol. This is why its a good idea to verify the connectivity between the leaf and monitoring station, as we did in the previous step.

3. Verify the filter applied to Leaf 1 and Leaf 2 correctly reflects test traffic. Start pings between Server 1 and Server 2 over the fabric.

```
user@server1> ping 10.1.101.20 count 10 rapid
PING 10.1.101.20 (10.1.101.20): 56 data bytes
!!!!!!!!!!
--- 10.1.101.20 ping statistics ---
10 packets transmitted, 10 packets received, 0% packet loss
round-trip min/avg/max/stddev = 0.731/0.800/1.300/0.167 ms
```

Verify the filter applied to Leaf 1 and Leaf 2 correctly reflects the traffic. Our example topology has only one VLAN and one pair of servers. This makes it easy to correlate test traffic to filter matches. Any non-zero count is a good sign the filter is working.

Because of ECMP behavior, the single ping flow hashes to only one of the aggregated Ethernet interface's member links. This means the test traffic is expected to be sent to one leaf, or the other, but not both. If Server 1 generates numerous flows, expect to see packets forwarded to both leaves using both member interfaces.

Clear the counters just before generating any pings.

```
user@server1> clear firewall all
```

Now display the firewall counters on Leaf 1 and Leaf 2.

```
user@leaf1> show firewall
user@leaf1# run show firewall

Filter: mirror-leaf-1
Counters:
Name                                                Bytes              Packets
from-s1                                                 0                    0
```

The output shows no packets are seen at Leaf 1. This indicates that for this flow, the server is hashing to Leaf 2.

Check the firewall counters at Leaf 2:

```
user@leaf2> show firewall

Filter: mirror-leaf-1
Counters:
Name                                                    Bytes              Packets
from-s1                                                  1020                   10
```

The firewall counter at Leaf 2 correctly reflects the generated test traffic.

4. Confirm that the traffic sent between Server 1 and Server 2 is mirrored at Leaf 1 and/or Leaf 2 for transmission to the monitoring station.

   Once again generate traffic between Server 1 and 2 (not shown for brevity). Use the rapid option for the ping command to generate a larger volume of packets to make detection easier.

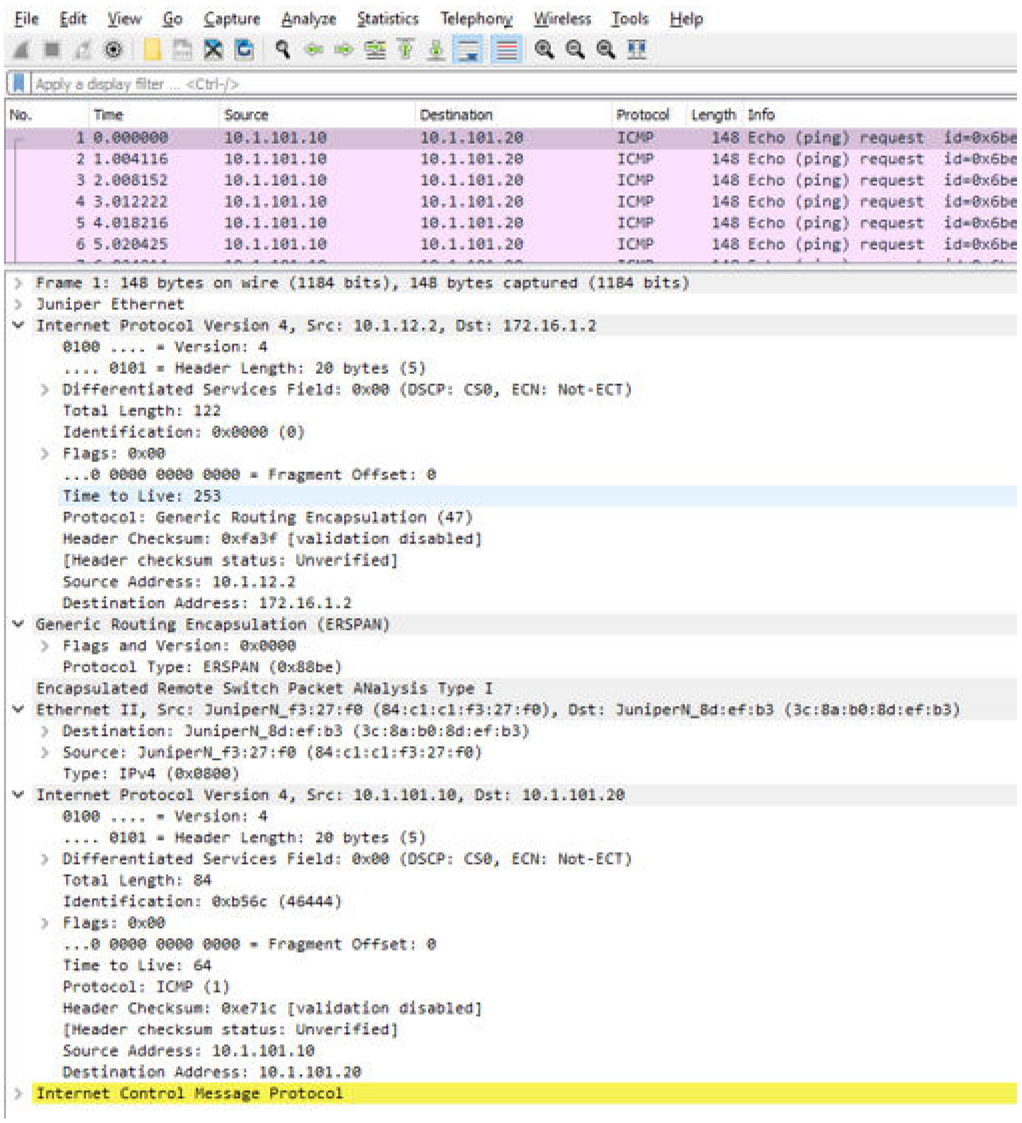
   Monitor interface traffic counters at border Leaf 3 to verify test traffic arrives at the monitoring station.

```
user@bl-leaf3>  monitor interface xe-0/0/33:1
bl-leaf3                          Seconds: 17                 Time: 19:58:30
                                                          Delay: 1/0/1
Interface: xe-0/0/33:1, Enabled, Link is Up
Encapsulation: Ethernet, Speed: 10000mbps
Traffic statistics:                                                Current delta
  Input bytes:                 1019608 (0 bps)                              [0]
  Output bytes:                5699872 (4752824 bps)                  [3382190]
  Input packets:                 10115 (0 pps)                              [0]
  Output packets:                34752 (3126 pps)                       [17801]
```

The output confirms traffic is sent to the monitor station. The lack of input packets is expected because the monitor station does not generate any responses to the GRE traffic it receives.

5. Use Wireshark or an equivalent analyzer application to capture and decode the mirrored traffic.

Figure 7: Mirrored Traffic Analysis


File Edit View Go Capture Analyze Statistics Telephony Wireless Tools Help
Apply a display filter ... <Ctrl-/>

```
No.   Time        Source         Destination    Protocol  Length Info
    1 0.000000    10.1.101.10    10.1.101.20    ICMP         148 Echo (ping) request  id=0x6be
    2 1.004116    10.1.101.10    10.1.101.20    ICMP         148 Echo (ping) request  id=0x6be
    3 2.008152    10.1.101.10    10.1.101.20    ICMP         148 Echo (ping) request  id=0x6be
    4 3.012222    10.1.101.10    10.1.101.20    ICMP         148 Echo (ping) request  id=0x6be
    5 4.018216    10.1.101.10    10.1.101.20    ICMP         148 Echo (ping) request  id=0x6be
    6 5.020425    10.1.101.10    10.1.101.20    ICMP         148 Echo (ping) request  id=0x6be
```

```
> Frame 1: 148 bytes on wire (1184 bits), 148 bytes captured (1184 bits)
> Juniper Ethernet
v Internet Protocol Version 4, Src: 10.1.12.2, Dst: 172.16.1.2
    0100 .... = Version: 4
    .... 0101 = Header Length: 20 bytes (5)
  > Differentiated Services Field: 0x00 (DSCP: CS0, ECN: Not-ECT)
    Total Length: 122
    Identification: 0x0000 (0)
  > Flags: 0x00
    ...0 0000 0000 0000 = Fragment Offset: 0
    Time to Live: 253
    Protocol: Generic Routing Encapsulation (47)
    Header Checksum: 0xfa3f [validation disabled]
    [Header checksum status: Unverified]
    Source Address: 10.1.12.2
    Destination Address: 172.16.1.2
v Generic Routing Encapsulation (ERSPAN)
  > Flags and Version: 0x0000
    Protocol Type: ERSPAN (0x88be)
  Encapsulated Remote Switch Packet ANalysis Type I
v Ethernet II, Src: JuniperN_f3:27:f0 (84:c1:c1:f3:27:f0), Dst: JuniperN_8d:ef:b3 (3c:8a:b0:8d:ef:b3)
  > Destination: JuniperN_8d:ef:b3 (3c:8a:b0:8d:ef:b3)
  > Source: JuniperN_f3:27:f0 (84:c1:c1:f3:27:f0)
    Type: IPv4 (0x0800)
v Internet Protocol Version 4, Src: 10.1.101.10, Dst: 10.1.101.20
    0100 .... = Version: 4
    .... 0101 = Header Length: 20 bytes (5)
  > Differentiated Services Field: 0x00 (DSCP: CS0, ECN: Not-ECT)
    Total Length: 84
    Identification: 0xb56c (46444)
  > Flags: 0x00
    ...0 0000 0000 0000 = Fragment Offset: 0
    Time to Live: 64
    Protocol: ICMP (1)
    Header Checksum: 0xe71c [validation disabled]
    [Header checksum status: Unverified]
    Source Address: 10.1.101.10
    Destination Address: 10.1.101.20
> Internet Control Message Protocol
```

The capture shows that Server 1 sends ping traffic to Server 2. The reply traffic is not mirrored because we have applied our filter at Leaf 1 and Leaf 2 in the input direction only. Apply a similar port mirror instance and filter configuration at Leaf 4 to mirror return traffic. Recall that in the lean spine case, the mirrored traffic showed VXLAN encapsulation. In this ERB leaf case, we mirror Layer 2 traffic before it has been encapsulated into VXLAN. The filter used to mirror traffic is applied as an input filter on the server-facing ae0.0 interface. There is no VXLAN encapsulation at ingress.

This is similar to the case of applying a the filter to the VLAN's IRB interface. In the IRB filtering case, the mirrored traffic also does not contain VXLAN encapsulation. IRB filters only process inter-VLAN traffic at the IP layer. Thus, for the IRB filter case, only Layer 3 IP traffic is mirrored.

The decode shows the GRE tunnel is between Leaf 2 and the monitoring station through the underlay. The Ethernet frame and IP packet sent by Server 1 is also decoded. The IP addresses assigned to the servers are visible (10.1.100.x/24), as are the details of the IP/ICMP packet generated by Server 1.

You have successfully configured port mirroring on your topology.

Example: Enable a Remote Analyzer Instance at an ESI-LAG Interface

IN THIS SECTION

Overview

An Ethernet segment identifier (ESI) is the unique 10-byte number that identifies an Ethernet segment. The ESI is enabled at the interface connected to the server. When multiple links form a link aggregation group (LAG) the result is an ESI-LAG, also known as EVPN LAG, interface. In some cases, you will need to enable port mirroring directly for all or part of the traffic entering or leaving the ESI-LAG interface.

Both egress and ingress analyzer instances are supported for ESI-LAG interfaces on leaf devices in an EVPN-VXLAN fabric. This is useful in a data center where the administrator needs to send all the traffic entering or leaving a given ESI-LAG port to, or from, a remote host. Use this section to enable an analyzer instance at the ESI-LAG level of an EVPN-VXLAN ERB fabric.

An analyzer instance differs from a port mirroring instance in that the analyzer works in the input, in the output, or in both directions. Also, it mirrors *all* traffic on the interface. For example, if Server 1 has 10 VLANs using a trunk interface, traffic from all 10 VLANs is sent to the monitoring station. In the previous port mirroring example, a filter is used to allow selective mirroring of traffic from one VLAN or from particular IP addresses in a VLAN.

Analyzer instances don't use firewall filters for granular matching. All traffic in the specified direction on the specified interface is mirrored.

Topology

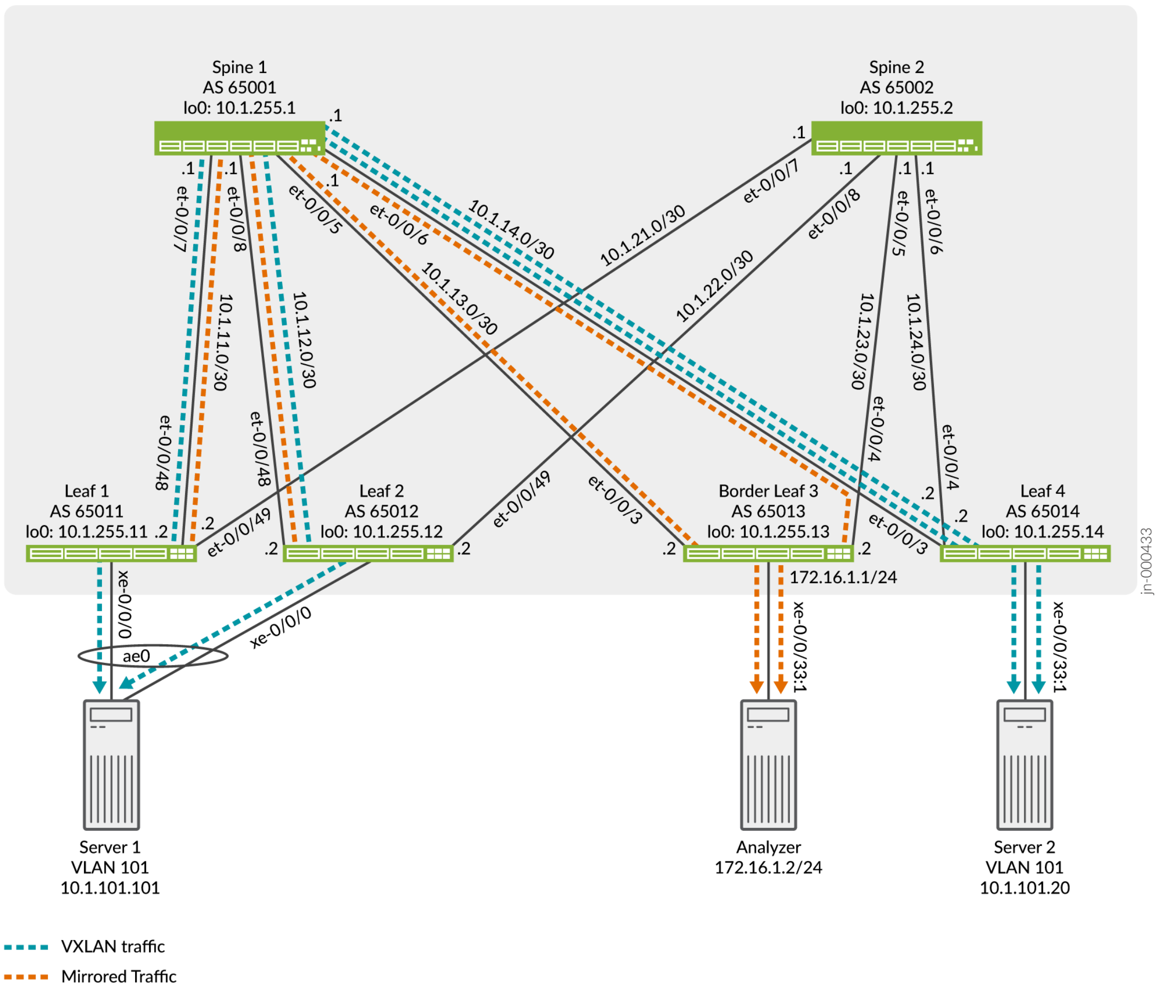
This configuration example uses the hardware and software components shown in "Requirements" on page 7.The devices in the topology begin with their baseline configurations as provided in "Appendix: Full Device Configurations" on page 51.

In this example an ESI-LAG, also known as an EVPN LAG, interface is used between Server 1 and Leaf 1 and Leaf 2. See EVPN LAG Configuration Best Practices for best practices.

NOTE: You can configure the analyzer instance on an ACX7100 switch running Junos OS Evolved 22.3R1 or later.

Figure 8 on page 43 shows the topology for this example.

Figure 8: Topology of Remote Analyzer on an ESI-LAG Interface



Interface ae0.0 is the ESI-LAG interface attached to Server 1. Server 1 is assigned IP address 10.1.101.10/24, and is associated with VLAN 101. The goal is to configure an analyzer instance to forward all traffic sent and received on the ae0.0 interface at both Leaf 1 and Leaf 2. The mirrored traffic is placed into a GRE tunnel and sent to the monitoring station. As before, you must ensure the monitoring station subnet is reachable in the EBGP underlay.

NOTE: Both spines are operational in this topology. ECMP means that leaf 1 and leaf 2 can send traffic through either spine device. To reduce clutter in the drawing we show the traffic and mirror paths with the Spine 1 device as the focus.

Before You Begin

If you are starting from the baseline configuration shown in "Appendix: Full Device Configurations" on page 51, follow these steps to modify your topology before you begin.

1. Modify the configuration of Leaf 1 and Leaf 2 to support multihomed attachment of Server 1. We begin by renaming the existing xe-0//0/0 interface to become the new ae0.0 interface. This caries over any configuration from the xe-0/0/0 interface to save some time.

```
user@leaf1# rename interfaces xe-0/0/0 to ae0
user@leaf1# set interfaces ae0 esi 00:02:02:02:02:02:02:02:00:04
user@leaf1# set interfaces ae0 esi all-active
user@leaf1# set interfaces ae0 aggregated-ether-options lacp active
user@leaf1# set interfaces ae0 aggregated-ether-options lacp system-id 00:00:02:02:00:04
```

   The configuration specifics for the server's aggregated Ethernet interface, often called a bonded interface in Linux, is beyond the scope of this NCE. Of note is that, for the server, there is only a standard aggregated Ethernet configuration. The statements related to ESI-LAG are applied only to the leaf devices.

   NOTE: The starting baseline configuration enables an aggregated Ethernet device using the `set chassis aggregated-devices ethernet device-count 1` configuration statement. You must enable chassis support for aggregated devices or the aggregated interface is not created.

2. After applying and committing these changes to both leaf devices and the server, confirm the ae0 interface is up on both leaves. The example output below has been shortened for clarity.

```
user@leaf2> show interfaces ae0
Physical interface: ae0, Enabled, Physical link is Up
  Physical interface: ae0, Enabled, Physical link is Up
[...]
  Current address: b8:c2:53:ba:8b:80, Hardware address: b8:c2:53:ba:8b:80
  Ethernet segment value: 00:02:02:02:02:02:02:02:00:04, Mode: all-active
  Last flapped   : 2022-09-07 14:57:53 UTC (01:06:43 ago)
```

```
[...]
  Queue counters:       Queued packets  Transmitted packets      Dropped packets
    0                            81129                81129                    0
    3                                0                    0                    0
    4                                0                    0                    0
    7                             4135                 4135                    0
    8                               64                   64                    0
[...]
    Protocol eth-switch, MTU: 1514, Generation: 266, Route table: 7, Mesh Group: __all_ces__, 
EVPN multi-homed status: Forwarding, 
    EVPN multi-homed ESI Split Horizon Label: 0, Next-hop: 1665, vpls-status: up
    Local Bias Logical Interface Name: vtep.32769, Index: 554, VXLAN Endpoint Address: 
10.1.255.11
      Flags: Is-Primary
```

Verify that Server 1 is able to ping Server 2 (not shown for brevity).

Configuration

This example shows how to enable remote port mirroring at the ESI-LAG level using a remote analyzer instance.

1. The ae0.0 interface is the ESI-LAG interface where the tenant Server 1 connects. This ESI-LAG terminates at both Leaf 1 and Leaf 2. Configure the analyzer instance on both leaves for the ae0.0 interface. Note that you configure a remote analyzer for both the input and output directions.

```
[edit forwarding-options analyzer my-analyzer1]
user@leaf1# set input ingress interface ae0.0
user@leaf1# set input egress interface ae0.0
user@leaf1# set output ip-address 172.16.1.2
```

2. Border Leaf 3 does not need to be configured to perform GRE decapsulation since the GRE tunnel terminates at the monitoring station. However, Leaf 3 does need to advertise reachability for the monitoring station's subnet into the fabric’s underlay.

   Configure the xe-0/0/33:1 interface that attaches to the monitoring station.

```
user@bl-leaf3# set interfaces xe-0/0/33:1 unit 0 family inet address 172.16.1.1/24
```

Modify the underlay export policy at border Leaf 3 to advertise the monitoring station's prefix. Our topology uses an EBGP underlay. We simply add a new term to the existing underlay export policy.

```
[edit policy-options policy-statement send-direct]
user@bl-leaf3# set term 2 from protocol direct
user@bl-leaf3# set pterm 2 from route-filter 172.16.1.0/24 exact
user@bl-leaf3# set term 2 then accept
```

3. The changes from the starting baseline are shown at Leaf 1.

```
user@leaf1# show | compare base
[edit interfaces xe-0/0/0]
+    gigether-options {
+        802.3ad ae0;
+    }
[edit interfaces xe-0/0/0]
-    unit 0 {
-        family ethernet-switching {
-            vlan {
-                members v101;
-            }
-        }
-    }
[edit interfaces]
+   ae0 {
+       esi {
+           00:02:02:02:02:02:02:02:00:04;
+           all-active;
+       }
+       aggregated-ether-options {
+           lacp {
+               active;
+               system-id 00:00:02:02:00:04;
+           }
+       }
+       unit 0 {
+           family ethernet-switching {
+               interface-mode access;
+               vlan {
+                   members v101;
+               }
+           }
+       }
+   }
[edit]
+  forwarding-options {
+      analyzer {
+          my-analyzer1 {
+              input {
+                  ingress {
+                      interface ae0.0;
+                  }
+                  egress {
+                      interface ae0.0;
+                  }
+              }
+              output {
+                  ip-address 172.16.1.2;
+              }
+          }
+      }
+  }
```

You have successfully configured remote port mirroring through a leaf device on an EVPN-VXLAN ERB fabric.

Verification

1. Verify underlay connectivity from Leaf 1 and Leaf 2 to the monitor station.

   On Leaf 1, display the route to 172.16.1.0/24.

```
user@leaf1> show route 172.16.1.0/24

inet.0: 12 destinations, 13 routes (12 active, 0 holddown, 0 hidden)
+ = Active Route, - = Last Active, * = Both

172.16.1.0/24      *[BGP/170] 4d 00:56:23, localpref 100
                      AS path: 65001 65013 I, validation-state: unverified
                    >  to 10.1.11.1 via et-0/0/48.0
```

NOTE: Strictly speaking, only simple connectivity is required between the leaf and monitoring station. We have configured a static route on the monitoring station to allow it to reach the fabric underlay. This permits ping testing as a fault isolation tool.

Ping the monitoring station to confirm connectivity.

```
user@leaf1> ping 172.16.1.2 count 2 source 10.1.255.11
PING 172.16.1.2 (172.16.1.2): 56 data bytes
64 bytes from 172.16.1.2: icmp_seq=0 ttl=62 time=1.380 ms
64 bytes from 172.16.1.2: icmp_seq=1 ttl=62 time=7.614 ms

--- 172.16.1.2 ping statistics ---
2 packets transmitted, 2 packets received, 0% packet loss
round-trip min/avg/max/stddev = 1.380/4.497/7.614/3.117 ms
```

The output confirms expected underlay reachability from Leaf 1 to the monitoring station. We must source the ping from the leaf's loopback address because our underlay export policy only advertises loopback address reachability. Sourcing from the loopback address was not needed in the lean spine example because the spine and border leaf are directly attached through a shared fabric link subnet.

2. Confirm the analyzer instance on Leaf 1 and Leaf 2.

   On Leaf 1, verify that the remote analyzer state is `up`.

```
user@leaf1> show forwarding-options analyzer
  Analyzer name                    : my-analyzer1
  Mirror rate                      : 1
  Maximum packet length            : 0
  State                            : up
  Ingress monitored interfaces     : ae0.0
  Egress monitored interfaces      : ae0.0
 Destination IP                    : 172.16.1.2
```

   The output confirms correct definition of the analyzer instance. The state of `up` indicates that the leaf has an underlay route to the monitoring station. Recall that GRE is a stateless protocol. This is why its a good idea to verify the connectivity between the leaf and monitoring station, as we did in the previous step.

3. Confirm the traffic sent between Server 1 and Server 2 is mirrored by Leaf 1 and/or Leaf 2 to the monitoring station.

Once again generate traffic between Server 1 and 2 (not shown for brevity). We used the rapid option for the ping command to generate a larger volume of packets to make detection easier.

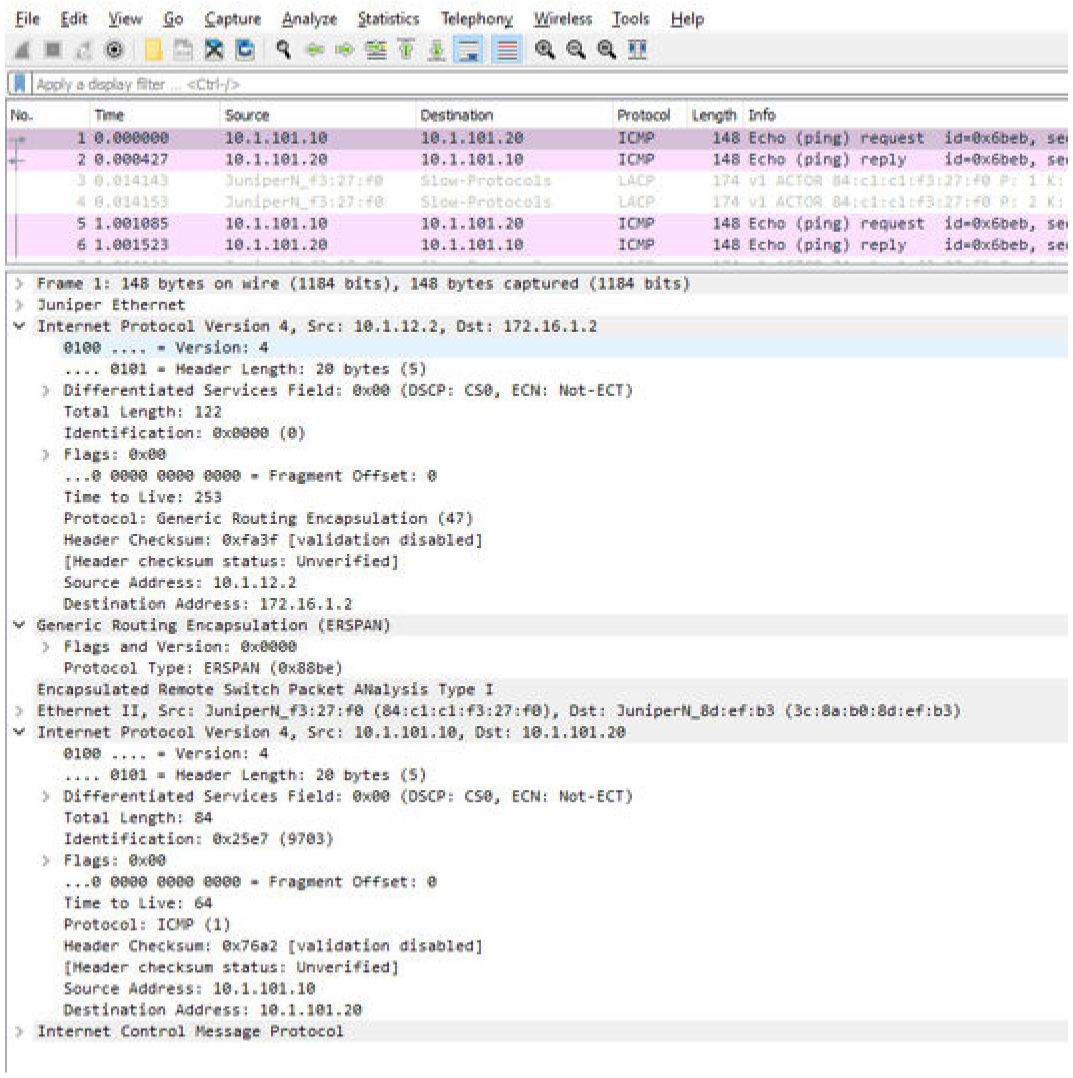
Monitor interface traffic counters at border Leaf 3 to verify test traffic arrives at the monitoring station.

```
user@bl-leaf3>  monitor interface xe-0/0/33:1
bl-leaf3                           Seconds: 17                  Time: 19:58:30
                                                            Delay: 1/0/1
Interface: xe-0/0/33:1, Enabled, Link is Up
Encapsulation: Ethernet, Speed: 10000mbps
Traffic statistics:                                                   Current delta
  Input bytes:                    1019608 (0 bps)                               [0]
  Output bytes:                   5699872 (4752824 bps)                   [3382190]
  Input packets:                    10115 (0 pps)                               [0]
  Output packets:                   34752 (3126 pps)                        [17801]
```

The output confirms traffic arrives at the monitor station. The lack of input packets is expected as the monitor station does not generate any responses to the GRE traffic it receives.

4. Use Wireshark or an equivalent analyzer application to capture and decode the mirrored traffic.

Figure 9: Mirrored Traffic Analysis



```
No.      Time        Source               Destination          Protocol  Length Info
       1 0.000000    10.1.101.10          10.1.101.20          ICMP      148 Echo (ping) request  id=0x6beb, se
       2 0.000427    10.1.101.20          10.1.101.10          ICMP      148 Echo (ping) reply    id=0x6beb, se
       3 0.014143    JuniperN_f3:27:f0    Slow-Protocols       LACP      174 v1 ACTOR 84:c1:c1:f3:27:f0 P: 1 K:
       4 0.014153    JuniperN_f3:27:f0    Slow-Protocols       LACP      174 v1 ACTOR 84:c1:c1:f3:27:f0 P: 2 K:
       5 1.001085    10.1.101.10          10.1.101.20          ICMP      148 Echo (ping) request  id=0x6beb, se
       6 1.001523    10.1.101.20          10.1.101.10          ICMP      148 Echo (ping) reply    id=0x6beb, se
```

```
> Frame 1: 148 bytes on wire (1184 bits), 148 bytes captured (1184 bits)
> Juniper Ethernet
v Internet Protocol Version 4, Src: 10.1.12.2, Dst: 172.16.1.2
    0100 .... = Version: 4
    .... 0101 = Header Length: 20 bytes (5)
  > Differentiated Services Field: 0x00 (DSCP: CS0, ECN: Not-ECT)
    Total Length: 122
    Identification: 0x0000 (0)
  > Flags: 0x00
    ...0 0000 0000 0000 = Fragment Offset: 0
    Time to Live: 253
    Protocol: Generic Routing Encapsulation (47)
    Header Checksum: 0xfa3f [validation disabled]
    [Header checksum status: Unverified]
    Source Address: 10.1.12.2
    Destination Address: 172.16.1.2
v Generic Routing Encapsulation (ERSPAN)
  > Flags and Version: 0x0000
    Protocol Type: ERSPAN (0x88be)
  Encapsulated Remote Switch Packet ANalysis Type I
> Ethernet II, Src: JuniperN_f3:27:f0 (84:c1:c1:f3:27:f0), Dst: JuniperN_8d:ef:b3 (3c:8a:b0:8d:ef:b3)
v Internet Protocol Version 4, Src: 10.1.101.10, Dst: 10.1.101.20
    0100 .... = Version: 4
    .... 0101 = Header Length: 20 bytes (5)
  > Differentiated Services Field: 0x00 (DSCP: CS0, ECN: Not-ECT)
    Total Length: 84
    Identification: 0x25e7 (9703)
  > Flags: 0x00
    ...0 0000 0000 0000 = Fragment Offset: 0
    Time to Live: 64
    Protocol: ICMP (1)
    Header Checksum: 0x76a2 [validation disabled]
    [Header checksum status: Unverified]
    Source Address: 10.1.101.10
    Destination Address: 10.1.101.20
> Internet Control Message Protocol
```

The capture shows that Server 1 sends ping traffic to Server 2. Due to the analyzer instances being applied to both directions of the ae0.0 interface, the reply traffic is also mirrored. Because the analyzer works at the interface level, LACP, which is a link level protocol used on the aggregated Ethernet interface, is also mirrored. Recall that in the lean spine case the mirrored traffic showed VXLAN encapsulation. In this ERB leaf case we capture Layer 2 traffic before its been encapsulated into VXLAN. At ingress there is no VXLAN encapsulation.

This is similar to the case of applying a the filter to a VLAN's IRB interface. In that case the mirrored traffic also does not contain VXLAN encapsulation. IRB based filters only catch inter-VLAN traffic at the IP layer. Thus, for the IRB filter case, only non-VXLAN encapsulated Layer 3 IP traffic is mirrored.

The decode shows the GRE tunnel is between Leaf 2 (10.1.12.2) and the monitoring station through the underlay. The Ethernet frame and related IP packet, as sent by Server 1, is also decoded. The IP addresses assigned to the servers are visible (10.1.100.x/24), as are the details of the IP/ICMP packet generated by Server 1. The reply from Server 2 is also mirrored because the analyzer instance is applied bidirectionally in our example.

You have successfully configured port mirroring on your topology.

RELATED DOCUMENTATION

Data Center EVPN-VXLAN Fabric Architecture Guide

Network Management and Monitoring Guide

Example: Configuring a QFX5110 Switch as Layer 2 and 3 VXLAN Gateways in an EVPN-VXLAN Edge-Routed Bridging Overlay

Appendix: Full Device Configurations

IN THIS SECTION

This appendix provides the starting configuration for all devices in the example topology shown in "Baseline Topology" on page 8. To quickly configure this topology, modify the configurations to match your network and copy them onto your devices.

Lean Spine Configurations

IN THIS SECTION

Spine 1

```
set system host-name spine1
set interfaces et-0/0/7 unit 0 family inet address 10.1.11.1/30
set interfaces et-0/0/8 unit 0 family inet address 10.1.12.1/30
set interfaces et-0/0/5 unit 0 family inet address 10.1.13.1/30
set interfaces et-0/0/6 unit 0 family inet address 10.1.14.1/30
set interfaces lo0 unit 0 family inet address 10.1.255.1/32
set policy-options policy-statement load-balancing-policy then load-balance per-packet
set policy-options policy-statement send-direct term 1 from protocol direct
set policy-options policy-statement send-direct term 1 from route-filter 10.1.255.1/32 exact
set policy-options policy-statement send-direct term 1 then accept
set routing-options router-id 10.1.255.1
set routing-options autonomous-system 65001
set routing-options forwarding-table export load-balancing-policy
set protocols bgp group underlay type external
set protocols bgp group underlay export send-direct
set protocols bgp group underlay multipath multiple-as
set protocols bgp group underlay neighbor 10.1.11.2 peer-as 65011
set protocols bgp group underlay neighbor 10.1.12.2 peer-as 65012
set protocols bgp group underlay neighbor 10.1.13.2 peer-as 65013
set protocols bgp group underlay neighbor 10.1.14.2 peer-as 65014
set protocols bgp group overlay type external
set protocols bgp group overlay multihop no-nexthop-change
set protocols bgp group overlay local-address 10.1.255.1
set protocols bgp group overlay family evpn signaling
set protocols bgp group overlay multipath multiple-as
set protocols bgp group overlay neighbor 10.1.255.11 peer-as 65011
set protocols bgp group overlay neighbor 10.1.255.12 peer-as 65012
set protocols bgp group overlay neighbor 10.1.255.13 peer-as 65013
set protocols bgp group overlay neighbor 10.1.255.14 peer-as 65014
set protocols bgp group overlay neighbor 10.1.255.2 peer-as 65002
```

Spine 2

```
set system host-name spine2
set system ports console log-out-on-disconnect
set interfaces et-0/0/7 unit 0 family inet address 10.1.21.1/30
set interfaces et-0/0/8 unit 0 family inet address 10.1.22.1/30
set interfaces et-0/0/5 unit 0 family inet address 10.1.23.1/30
set interfaces et-0/0/6 unit 0 family inet address 10.1.24.1/30
set interfaces lo0 apply-groups-except global
set interfaces lo0 unit 0 family inet address 10.1.255.2/32
set policy-options policy-statement load-balancing-policy then load-balance per-packet
set policy-options policy-statement send-direct term 1 from protocol direct
set policy-options policy-statement send-direct term 1 from route-filter 10.1.255.2/32 exact
set policy-options policy-statement send-direct term 1 then accept
set routing-options router-id 10.1.255.2
set routing-options autonomous-system 65002
set routing-options forwarding-table export load-balancing-policy
set protocols bgp group underlay type external
set protocols bgp group underlay export send-direct
set protocols bgp group underlay multipath multiple-as
set protocols bgp group underlay neighbor 10.1.21.2 peer-as 65011
set protocols bgp group underlay neighbor 10.1.22.2 peer-as 65012
set protocols bgp group underlay neighbor 10.1.23.2 peer-as 65013
set protocols bgp group underlay neighbor 10.1.24.2 peer-as 65014
set protocols bgp group overlay type external
set protocols bgp group overlay multihop no-nexthop-change
set protocols bgp group overlay local-address 10.1.255.2
set protocols bgp group overlay family evpn signaling
set protocols bgp group overlay multipath multiple-as
set protocols bgp group overlay neighbor 10.1.255.11 peer-as 65011
set protocols bgp group overlay neighbor 10.1.255.12 peer-as 65012
set protocols bgp group overlay neighbor 10.1.255.13 peer-as 65013
set protocols bgp group overlay neighbor 10.1.255.14 peer-as 65014
set protocols bgp group overlay neighbor 10.1.255.1 peer-as 65001
```

Leaf Device Configurations

IN THIS SECTION

Leaf 1

```
set system host-name leaf1
set chassis aggregated-devices ethernet device-count 1
set interfaces et-0/0/48 unit 0 family inet address 10.1.11.2/30
set interfaces et-0/0/49 unit 0 family inet address 10.1.21.2/30
set interfaces xe-0/0/3 unit 0 family ethernet-switching vlan members v101
set interfaces irb unit 101 virtual-gateway-accept-data
set interfaces irb unit 101 family inet address 10.1.101.1/24 virtual-gateway-address
10.1.101.254
set interfaces irb unit 102 virtual-gateway-accept-data
set interfaces irb unit 102 family inet address 10.1.102.1/24 virtual-gateway-address
10.1.102.254
set interfaces lo0 unit 0 family inet address 10.1.255.11/32
set policy-options policy-statement load-balancing-policy then load-balance per-packet
set policy-options policy-statement send-direct term 1 from protocol direct
set policy-options policy-statement send-direct term 1 from route-filter 10.1.255.11/32 exact
set policy-options policy-statement send-direct term 1 then accept
set routing-instances vrf101 instance-type vrf
set routing-instances vrf101 routing-options auto-export
set routing-instances vrf101 interface irb.101
set routing-instances vrf101 route-distinguisher 10.1.255.11:101
set routing-instances vrf101 vrf-import vrf101_vrf_imp
set routing-instances vrf101 vrf-target target:1001:1
set routing-options router-id 10.1.255.11
set routing-options autonomous-system 65011
set routing-options forwarding-table export load-balancing-policy
set protocols bgp group underlay type external
set protocols bgp group underlay export send-direct
set protocols bgp group underlay multipath multiple-as
set protocols bgp group underlay neighbor 10.1.11.1 peer-as 65001
set protocols bgp group underlay neighbor 10.1.21.1 peer-as 65002
set protocols bgp group overlay type external
set protocols bgp group overlay multihop
set protocols bgp group overlay local-address 10.1.255.11
set protocols bgp group overlay family evpn signaling
set protocols bgp group overlay multipath multiple-as
set protocols bgp group overlay neighbor 10.1.255.1 peer-as 65001
set protocols bgp group overlay neighbor 10.1.255.2 peer-as 65002
set protocols evpn encapsulation vxlan
set protocols evpn default-gateway no-gateway-community
set protocols evpn multicast-mode ingress-replication
deactivate protocols evpn multicast-mode
set protocols evpn vni-options vni 101 vrf-target target:101:1
set protocols evpn extended-vni-list all
set switch-options vtep-source-interface lo0.0
set switch-options route-distinguisher 10.1.255.11:1
set switch-options vrf-target target:1:1
set switch-options vrf-target auto
set vlans v101 vlan-id 101
set vlans v101 l3-interface irb.101
set vlans v101 vxlan vni 101
```

Leaf 2

```
set system host-name leaf2
set chassis aggregated-devices ethernet device-count 1
set interfaces et-0/0/48 unit 0 family inet address 10.1.12.2/30
set interfaces et-0/0/49 unit 0 family inet address 10.1.22.2/30
set interfaces xe-0/0/3 unit 0 family ethernet-switching vlan members v101
set interfaces irb unit 101 virtual-gateway-accept-data
set interfaces irb unit 101 family inet address 10.1.101.2/24 virtual-gateway-address
10.1.101.254
set interfaces lo0 unit 0 family inet address 10.1.255.12/32
set policy-options policy-statement load-balancing-policy then load-balance per-packet
set policy-options policy-statement send-direct term 1 from protocol direct
set policy-options policy-statement send-direct term 1 from route-filter 10.1.255.12/32 exact
set policy-options policy-statement send-direct term 1 then accept
set routing-instances vrf101 instance-type vrf
set routing-instances vrf101 interface irb.101
set routing-instances vrf101 route-distinguisher 10.1.255.12:101
set routing-instances vrf101 vrf-import vrf101_vrf_imp
set routing-instances vrf101 vrf-target target:1001:1
set routing-options router-id 10.1.255.12
set routing-options autonomous-system 65012
set routing-options forwarding-table export load-balancing-policy
set protocols bgp group underlay type external
set protocols bgp group underlay export send-direct
set protocols bgp group underlay multipath multiple-as
set protocols bgp group underlay neighbor 10.1.12.1 peer-as 65001
set protocols bgp group underlay neighbor 10.1.22.1 peer-as 65002
set protocols bgp group overlay type external
set protocols bgp group overlay multihop
set protocols bgp group overlay local-address 10.1.255.12
set protocols bgp group overlay family evpn signaling
set protocols bgp group overlay multipath multiple-as
set protocols bgp group overlay neighbor 10.1.255.1 peer-as 65001
set protocols bgp group overlay neighbor 10.1.255.2 peer-as 65002
set protocols evpn encapsulation vxlan
set protocols evpn default-gateway no-gateway-community
set protocols evpn vni-options vni 101 vrf-target target:101:1
set protocols evpn extended-vni-list all
set switch-options vtep-source-interface lo0.0
set switch-options route-distinguisher 10.1.255.12:1
set switch-options vrf-target target:1:1
set switch-options vrf-target auto
set vlans v101 vlan-id 101
set vlans v101 l3-interface irb.101
set vlans v101 vxlan vni 101
```

Leaf 3 (QFX10000 Border Leaf)

```
set system host-name bl-leaf3
set chassis fpc 0 pic 0 port 33 channel-speed 10g
set interfaces et-0/0/3 unit 0 family inet address 10.1.13.2/30
set interfaces et-0/0/4 unit 0 family inet address 10.1.23.2/30
set interfaces irb unit 101 virtual-gateway-accept-data
set interfaces irb unit 101 family inet address 10.1.101.3/24 virtual-gateway-address
10.1.101.254
set interfaces lo0 unit 0 family inet address 10.1.255.13/32
set policy-options policy-statement load-balancing-policy then load-balance per-packet
set policy-options policy-statement send-direct term 1 from protocol direct
set policy-options policy-statement send-direct term 1 from route-filter 10.1.255.13/32 exact
set policy-options policy-statement send-direct term 1 then accept
set routing-instances vrf101 instance-type vrf
set routing-instances vrf101 routing-options auto-export
set routing-instances vrf101 interface irb.101
set routing-instances vrf101 route-distinguisher 10.1.255.13:101
set routing-instances vrf101 vrf-import vrf101_vrf_imp
set routing-instances vrf101 vrf-target target:1001:1
set routing-options router-id 10.1.255.13
set routing-options autonomous-system 65013
set routing-options forwarding-table export load-balancing-policy
set protocols bgp group underlay type external
set protocols bgp group underlay export send-direct
set protocols bgp group underlay multipath multiple-as
set protocols bgp group underlay neighbor 10.1.13.1 peer-as 65001
set protocols bgp group underlay neighbor 10.1.23.1 peer-as 65002
set protocols bgp group overlay type external
set protocols bgp group overlay multihop
set protocols bgp group overlay local-address 10.1.255.13
set protocols bgp group overlay family evpn signaling
set protocols bgp group overlay multipath multiple-as
set protocols bgp group overlay neighbor 10.1.255.1 peer-as 65001
set protocols bgp group overlay neighbor 10.1.255.2 peer-as 65002
set protocols evpn encapsulation vxlan
set protocols evpn default-gateway no-gateway-community
set protocols evpn multicast-mode ingress-replication
deactivate protocols evpn multicast-mode
set protocols evpn vni-options vni 101 vrf-target target:101:1
set protocols evpn vni-options vni 102 vrf-target target:102:1
set protocols evpn extended-vni-list all
set switch-options vtep-source-interface lo0.0
set switch-options route-distinguisher 10.1.255.13:1
set switch-options vrf-target target:1:1
set switch-options vrf-target auto
set vlans v101 vlan-id 101
set vlans v101 l3-interface irb.101
set vlans v101 vxlan vni 101
```

Leaf 4 (QFX10000 Server Leaf)

```
set system host-name leaf4
set chassis fpc 0 pic 0 port 33 channel-speed 10g
set interfaces et-0/0/3 unit 0 family inet address 10.1.14.2/30
set interfaces et-0/0/4 unit 0 family inet address 10.1.24.2/30
set interfaces xe-0/0/33:1 unit 0 family ethernet-switching interface-mode access
set interfaces xe-0/0/33:1 unit 0 family ethernet-switching vlan members v101
set interfaces em0 unit 0 family inet address 10.1.1.238/24
set interfaces irb unit 101 virtual-gateway-accept-data
set interfaces irb unit 101 family inet address 10.1.101.4/24 virtual-gateway-address
10.1.101.254
set interfaces lo0 unit 0 family inet address 10.1.255.14/32
set policy-options policy-statement load-balancing-policy then load-balance per-packet
set policy-options policy-statement send-direct term 1 from protocol direct
set policy-options policy-statement send-direct term 1 from route-filter 10.1.255.12/32 exact
set policy-options policy-statement send-direct term 1 from route-filter 10.1.255.14/32 exact
set policy-options policy-statement send-direct term 1 then accept
set policy-options policy-statement vrf101_vrf_imp term 1 from community vrf102
set policy-options policy-statement vrf101_vrf_imp term 1 then accept
set policy-options policy-statement vrf102_vrf_imp term 1 from community vrf101
set policy-options policy-statement vrf102_vrf_imp term 1 then accept
set policy-options community vrf101 members target:1001:1
set policy-options community vrf102 members target:1002:1
set routing-instances vrf101 instance-type vrf
set routing-instances vrf101 interface irb.101
set routing-instances vrf101 route-distinguisher 10.1.255.12:101
set routing-instances vrf101 vrf-import vrf101_vrf_imp
set routing-instances vrf101 vrf-target target:1001:1
set routing-instances vrf101 routing-options auto-export
set routing-options static route 10.1.1.0/24 next-hop 10.1.1.254
set routing-options router-id 10.1.255.14
set routing-options autonomous-system 65014
set routing-options forwarding-table export load-balancing-policy
set protocols bgp group underlay type external
set protocols bgp group underlay export send-direct
set protocols bgp group underlay multipath multiple-as
set protocols bgp group underlay neighbor 10.1.12.1 peer-as 65001
set protocols bgp group underlay neighbor 10.1.22.1 peer-as 65002
set protocols bgp group underlay neighbor 10.1.14.1 peer-as 65001
set protocols bgp group underlay neighbor 10.1.24.1 peer-as 65002
set protocols bgp group overlay type external
set protocols bgp group overlay multihop
set protocols bgp group overlay local-address 10.1.255.14
set protocols bgp group overlay family evpn signaling
set protocols bgp group overlay multipath multiple-as
set protocols bgp group overlay neighbor 10.1.255.1 peer-as 65001
set protocols bgp group overlay neighbor 10.1.255.2 peer-as 65002
set protocols evpn vni-options vni 101 vrf-target target:101:1
set protocols evpn vni-options vni 102 vrf-target target:102:1
set protocols evpn vni-options vni 103 vrf-target target:103:1
set protocols evpn encapsulation vxlan
set protocols evpn default-gateway no-gateway-community
set protocols evpn extended-vni-list all
set switch-options vtep-source-interface lo0.0
set switch-options route-distinguisher 10.1.255.14:1
set switch-options vrf-target target:1:1
set vlans v101 vlan-id 101
set vlans v101 l3-interface irb.101
set vlans v101 vxlan vni 101
```