Mellanox Networking for Nutanix Backup and Disaster Recovery Solution

Rev 2.0
NOTE:
THIS HARDWARE, SOFTWARE OR TEST SUITE PRODUCT ("PRODUCT(S)") AND ITS RELATED DOCUMENTATION ARE PROVIDED BY MELLANOX TECHNOLOGIES “AS-IS” WITH ALL FAULTS OF ANY KIND AND SOLELY FOR THE PURPOSE OF AIDING THE CUSTOMER IN TESTING APPLICATIONS THAT USE THE PRODUCTS IN DESIGNATED SOLUTIONS. THE CUSTOMER’S MANUFACTURING TEST ENVIRONMENT HAS NOT MET THE STANDARDS SET BY MELLANOX TECHNOLOGIES TO FULLY QUALIFY THE PRODUCT(S) AND/OR THE SYSTEM USING IT. THEREFORE, MELLANOX TECHNOLOGIES CANNOT AND DOES NOT GUARANTEE OR WARRANT THAT THE PRODUCTS WILL OPERATE WITH THE HIGHEST QUALITY. ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT ARE DISCLAIMED. IN NO EVENT SHALL MELLANOX BE LIABLE TO CUSTOMER OR ANY THIRD PARTIES FOR ANY DIRECT, INDIRECT, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES OF ANY KIND (INCLUDING, BUT NOT LIMITED TO, PAYMENT FOR PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY FROM THE USE OF THE PRODUCT(S) AND RELATED DOCUMENTATION EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.
Table of Contents

1 Overview ........................................................................................................................................... 4

2 Effective Disaster Recovery Plan Requirements .................................................................................. 4

3 Implementation .................................................................................................................................. 5

  3.1 Mellanox DCI Solution .................................................................................................................... 6

  3.2 Nutanix and Mellanox Spectrum SN2010 DCI Setup Diagram .......................................................... 7

    3.2.1 Setup overview: ............................................................................................................................ 7

    3.2.2 MLAG configuration on Cumulus Linux ....................................................................................... 9

  3.3 Underlay Network Configuration ....................................................................................................... 16

    3.3.1 Underlay Network Configuration using BGP .............................................................................. 16

    3.3.2 Overlay Network Configuration Using VXLAN with BGP EVPN ................................................. 18

    3.3.3 VLAN to VNI mapping on Cumulus Linux. ................................................................................. 20

  3.4 Nutanix Prism Central Plug-in for Mellanox NEO ............................................................................. 21

    3.4.1 Verifying site-to-site connectivity using EVPN/VXLAN .............................................................. 35

4 Appendix A: Manual DCI with VXLAN +EVPN and MLAG configuration using Cumulus Linux CLI 36

  4.1 SNMP and LLDP configuration ......................................................................................................... 36

  4.2 Manual MLAG configuration using Cumulus Linux CLI: ............................................................... 36

    4.2.1 Configure Peerlink Interface ........................................................................................................ 36

    4.2.2 Configure MLAG Ports (Downlinks to hosts) ............................................................................ 37

    4.2.3 VNI and VLAN interfaces creation ............................................................................................. 38

5 Configuration Verification .................................................................................................................... 42
1 Overview

More and more businesses relay on storing and sharing digital data for internal and external purposes. Today, this data is exposed to an unprecedented number of threats, from cyber threats to environmental events and natural disasters, and as companies are moving their business applications and data to a cloud-based solution, especially into a hybrid cloud solution, the risks increase and become more challenging to solve.

The main concern for any CIO is to manage these risks and devise an elaborate business continuity plan which includes a clear strategy on which backup method to use and what disaster recovery plan to follow.

There are many backup methods currently being deployed in the market which vary in data flow volume within data centers and the required processing power to deliver the desired performance.

The following are a few of the backup methods popular among businesses:

- Private Cloud – Back up VMs running in production
- Hybrid Cloud Solution – Back up VMs on both a private and public cloud interchangeably
- Cloud Storage – Both the data and the system are replicated off-site.

This method enables uninterrupted access to systems and data, even after a disaster.

2 Effective Disaster Recovery Plan Requirements

The disaster recovery strategy is well defined in the RPO (recovery point objective) and RTO (recovery time objective) sections of a business continuity plan. The core of these objectives is to define which data to restore first and how fast.

The frequency and severity of weather-related events is increasing and the reliance on a complex network of technology and supply chains is expanding. Both trends leave businesses susceptible to a variety of existing and emerging risks. Managing these risks by developing a business continuity strategy is key to the survival of any organization.

An efficient backup system can recover data at fast speeds. In case of data center failure, the lower the interruption period of data access, the better the data backup and recovery system is.
3 Implementation

3.1 Introduction

This document describes the implementation of DCI as shown in the setup diagram below where Mellanox Spectrum switches were utilized as the host's gateways. There are other implementations of VXLAN/EVPN involving centralized or distributed gateways, routers, firewalls and more which are not covered in this document.

Mellanox DCI/VXLAN solution has virtualization integration with VMware NSX-V/MH, Midokura MidoNet and OpenStack. With this integration, Mellanox SN2000 switch series running Cumulus Linux can function as hardware VTEP gateways while the controller provides consistent provisioning across virtual and physical server infrastructures. For more information read “Virtualization Integrations of Hardware VTEPs”

3.2 Use Cases

Backup from the main site to a disaster recovery site was previously done by changing the DNS entry from an old DNS entry to a new one pointing the client to a new location. However, this method is not efficient as changing each DNS entry takes a lot of time and the network should be designed in a way that it can take care of any workload down situation.

With the cloud evolution, the network has also evolved and can now provide multiple designs on how the cloud network can manage the workload in the cloud instead of locally or on-premise data center. With these innovations comes EVPN, which provides a neat solution for DCI (Data Center Interconnect).

By using an EVPN based DCI, businesses can stretch a layer2 network between data centers, and VMs can move easily with the same IP and gateway in the event of a disaster. The EVPN based control plane will automatically update the location of the VM (the host on which it is sitting) while the client accessing the data from this VM will not be affected and will not notice the change. The MAC move community takes care of moving the mac address to the right host.

EVPN on the network enables high speeds (up to 100Gb/s), low latency (required for most business applications), better buffer and tuning for big data, Artificial Inelegance (AI) and Machine Learning (ML) applications.

While EVPN is pure software implementation the rest of the mentioned attributes come from the network switching ASIC.
3.3 Mellanox DCI Solution

Mellanox SN2000 series of switches are the industry leader in ESF (Ethernet Storage Fabric), as these switches were designed for storage and have the right combination of performance, form factor, power consumption, buffers, automation hooks, integrated management and price point.

At the core of the ESF solution for Nutanix is the Mellanox Spectrum SN2010 switch, which is the ideal Top-of-Rack (ToR) switch for this use case.

Mellanox Spectrum SN2010 switch is the industry’s best cost-to-performance solution which allows Nutanix customers to easily deploy a highly scalable, fully transparent networking cloud by utilizing the systems 18 ports of 25GbE and 4 ports of 100GbE. With EVPN enabled these switches support the DCI solution and have a clean data backup and recovery solution integrated within Nutanix.

Mellanox also provides a centralized network management application called NEO™ which is a powerful platform for data-center network orchestration, designed to simplify network provisioning, monitoring and operation for the modern data-center. NEO™ offers robust automation capabilities that extend the existing tools features set, from network staging and bring-up, to day-to-day operations. It allows to easily automate network configuration using various configuration templates for Mellanox Onyx and Cumulus Linux as well as custom made templates.

The integration of NEO™ with Nutanix Prism Central provided a seamless DCI/VXLAN L2 stretch from host perspective. When a user creates a network and a VM from Prism Central, The API triggers Mellanox NEO to configure the network switches for VLAN to VXLAN mapping to stretch L2 across the data-center interconnect infrastructure, so data backup and recovery can be realized.

NOTE:

Mellanox NEO can be installed on any host (bare-metal/virtual machine) in the network and the NEO virtual appliance is integrated with Nutanix AHV and validated as Nutanix Ready for Networking. To install NEO on Nutanix AHV, follow installation guide in Mellanox NEO User Manual.

The configuration is based on EVPN DCI where VXLAN is stretched from primary data location to the backup data location.

3.4 Supported Software

Below is a list of DCI solution supported software:

- NEO, ver. 2.3 and above
- CL, ver. 3.6.2 and above
- Nutanix AOS, ver. 5.5 and above
- Nutanix Prism Central, ver. 5.7.1 and above
3.5 Nutanix and Mellanox Spectrum SN2010 DCI Setup Diagram

The below diagram shows the logical connections of the Mellanox and Nutanix DCI/VXLAN solution.

The following chapter will guide you through the steps to configure the above solution using Nutanix Prism Central, NEO and SN2010 switches running Cumulus Linux.

3.5.1 Setup overview:

- Main Site – 4x Nutanix nodes and 2x Mellanox SN2010 switches (connected as MLAG peers)
- Disaster Recovery Site – 4x Nutanix nodes and 1x Mellanox SN2010 switch
### Main Site

<table>
<thead>
<tr>
<th></th>
<th>IP Address</th>
</tr>
</thead>
<tbody>
<tr>
<td>VM A</td>
<td>192.168.1.10/24</td>
</tr>
<tr>
<td>VM B</td>
<td>192.168.2.11/24</td>
</tr>
<tr>
<td>VM C</td>
<td>192.168.3.12/24</td>
</tr>
<tr>
<td>VM D</td>
<td>192.168.4.13/24</td>
</tr>
<tr>
<td>SW-1 VLAN10</td>
<td>192.168.1.251/24</td>
</tr>
<tr>
<td>SW-1 VLAN20</td>
<td>192.168.2.251/24</td>
</tr>
<tr>
<td>SW-1 VLAN30</td>
<td>192.168.3.251/24</td>
</tr>
<tr>
<td>SW-1 VLAN40</td>
<td>192.168.4.251/24</td>
</tr>
<tr>
<td>SW-1 VTEP</td>
<td>10.10.10.1/32</td>
</tr>
<tr>
<td>SW-2 VTEP</td>
<td>10.10.10.2/32</td>
</tr>
<tr>
<td>SW-2 VLAN10</td>
<td>192.168.1.252/24</td>
</tr>
<tr>
<td>SW-2 VLAN20</td>
<td>192.168.2.252/24</td>
</tr>
<tr>
<td>SW-2 VLAN30</td>
<td>192.168.3.252/24</td>
</tr>
<tr>
<td>SW-2 VLAN40</td>
<td>192.168.4.252/24</td>
</tr>
<tr>
<td>CLAG VRR VLAN10 (GW)</td>
<td>192.168.1.254/24</td>
</tr>
<tr>
<td>CLAG VRR VLAN20 (GW)</td>
<td>192.168.2.254/24</td>
</tr>
<tr>
<td>CLAG VRR VLAN30 (GW)</td>
<td>192.168.3.254/24</td>
</tr>
<tr>
<td>CLAG VRR VLAN40 (GW)</td>
<td>192.168.4.254/24</td>
</tr>
<tr>
<td>CLAG Anycast IP</td>
<td>10.10.10.10/32</td>
</tr>
</tbody>
</table>

### Disaster Recovery Site

<table>
<thead>
<tr>
<th></th>
<th>IP Address</th>
</tr>
</thead>
<tbody>
<tr>
<td>VM 1</td>
<td>192.168.1.106/24</td>
</tr>
<tr>
<td>VM 2</td>
<td>192.168.2.107/24</td>
</tr>
<tr>
<td>VM 3</td>
<td>192.168.3.108/24</td>
</tr>
<tr>
<td>VM 4</td>
<td>192.168.4.109/24</td>
</tr>
<tr>
<td>SW-3 VTEP (Loopback)</td>
<td>100.100.100.100/32</td>
</tr>
<tr>
<td>SW-3 VLAN10 (GW)</td>
<td>192.168.1.254/24</td>
</tr>
<tr>
<td>SW-3 VLAN20 (GW)</td>
<td>192.168.2.254/24</td>
</tr>
<tr>
<td>SW-3 VLAN30 (GW)</td>
<td>192.168.3.254/24</td>
</tr>
<tr>
<td>SW-3 VLAN40 (GW)</td>
<td>192.168.4.254/24</td>
</tr>
</tbody>
</table>

### NOTE:

In this solution, we assume that the LAN gateways are Mellanox Spectrum switches (ToR). ToR switches may not act as LAN gateway in other deployments.
NOTE:
In deployments where there are more VMs in the same L2 domain (VLAN) as Nutanix Host (on site), L3 VLAN interfaces with trunks should be configured on the ToR’s uplinks instead of L3 Router Port interfaces.

Mellanox DCI solution for Nutanix multi-cloud was accomplished by using a small number of network technologies: MLAG, underlay connectivity using any protocol, overlay BGP-EVPN control plane and VXLAN encapsulation for the data.

MLAG should be enabled and configured in the main site before configuring the underlay network as hosts in the main site have network high-availability (HA) provided by the MLAG protocol in the ToR switches to allow hosts to stay connected even when one of the ToR switches fail.

3.5.2 MLAG configuration on Cumulus Linux

Mellanox NEO offers a built in simplified MLAG configuration method on Cumulus Linux but an initial installation of the software is required. to install Mellanox NEO, follow the installation guide in Mellanox NEO User Manual.

There are two methods to configure MLAG on Cumulus Linux:

1. Automated MLAG configuration

To configure MLAG using the automated method, run the following template in NEO:

```
net add bond <bond_name> bond slaves <<interface_list>>
net add interface <bond_name>.<vlan_id> ip address <<ipl_ip>>/<<netmask_length>>
net add interface <bond_name>.<vlan_id> clag peer-ip <<peer_ip>>
net add interface <bond_name>.<vlan_id> clag backup-ip <<backup_ip>>
net add interface <bond_name>.<vlan_id> clag sys-mac <virtual_mac>
net add bond <LAG_name> clag id <CLAG_ID>
net commit
```

2. Manual MLAG configuration

To manually configure MLAG using NEO, both switches must be added to the management software.

To add them to the NEO management software, follow the below steps:

1. Go to “Managed Elements” and add device:
2. Add both switches in the main site:

![Add New Devices](image)

3. After the switches are successfully added to NEO, the following window will appear:

![Adding systems](image)

4. LLDP and SNMP protocols must be enabled on the switches for NEO to discover all the switch information, such as ports’ connection information needed for the MLAG Wizard.

There are 3 ways to enable the LLDP and SNMP protocols:
i. Predefined task
   To configure using the Task, go “Tasks” and “Run”:

ii. Predefined template
   To configure using a predefined template, go to “Managed Elements” and select the switches you want to configure. Right click and choose “Provisioning”:

   In the “Provisioning” tab that opened, press “Templates”:

   In this window, search for initial Cumulus configuration and load it:
Enter the Management IPs for the switches and start the template configuration:

Add Nutanix servers to NEO (same as adding a switch):

After LLDP and SNMP are enabled on all switches and the servers added:

5. Go to “Services” and click on “+Add” button to open MLAG Wizard and add the MLAG service:

6. Add MLAG cluster parameters through the following:
a. Set name, description, select OS type and choose one of the added switches (the rest will be auto filled – ports and 2nd switch).

b. Under “Advanced” you can manually enter peerlink IP addresses for the cluster.

7. Configure MLAG networks:

8. Set subnet and VLAN ID for each of the VLANs in main site (10, 20, 30, and 40):
9. Create MLAG Port-Channels (bonds) for downlinks:

![MLAG Wizard]

10. Repeat for the rest of the VLANs and VMs.
    Note: For this configuration, previously added Nutanix servers are mandatory.

11. Press “Finish” to save MLAG service.
12. Apply the configuration on the switches:

![MLAG Configuration Diagram]

i. MLAG is now configured with all the parameters on the switches. CLI (see Appendix A).

Appendix A shows manual MLAG configuration using Cumulus Linux CLI.
3.6 Underlay Network Configuration

Underlay network reachability needs to be configured on the setup so that each ToR switch will have L3 connectivity to other switches (both physical and loopback interfaces-VTEPs).

The configuration method depends on the cross-site connection or the routing configuration already in use:

- OSPF Routing protocol
- BGP Routing protocol
- Static Routing

In our solution with Nutanix, BGP routing protocol underlay is used, but any of the above protocols are supported.

To create a ToR to remote ToR underlay network connectivity, BGP IPv4 neighborships must be established from each ToR to its next hop switch (there is an option to establish only BGP EVPN neighborship straight to the remote ToR in case of a different underlay protocol).

In the diagram above, only ToR switches are shown. The uplink interfaces will be used to connect to the exit point of the data center (generally with ECMP).

**NOTE:**

In case there is more than one rack on site, all racks should be connected using Spine switches. To connect the racks, an IP (including the undelay routing protocols) must be configured to enable intra and inter-site underlay L3 (IP) reachability between all leafs (Racks). Any dynamic or static routing protocol can be used (here we cover BGP IPv4 address-family) for establishing IP connectivity. VTEP (Loopback) and MLAG Anycast-IP addresses (if exist) must be advertised via underlay routing protocol or set with static routing for further building the overlay VXLAN tunnel.

3.6.1 Underlay Network Configuration using BGP

We will demonstrate this process using 2 spine switches on each site. Each ToR is connected to both spines for ECMP.

Setup Overview:

- Main Site – spine1 and spine2.
- Disaster Recovery Site – spine3 and spine4.
To configure the underlay network follow the below steps:

- **Run the following commands on SW-1:**

  ```
  cumulus@SW-1:$ net add bgp autonomous-system 65002
  cumulus@SW-1:$ net add bgp router-id 10.10.10.1
  cumulus@SW-1:$ net add bgp bestpath as-path multipath-relax
  cumulus@SW-1:$ net add bgp maximum-paths 2
  cumulus@SW-1:$ net add bgp neighbor [Spine1 interface IP] remote-as [Spine1 AS]**
  cumulus@SW-1:$ net add bgp neighbor [Spine2 interface IP] remote-as [Spine2 AS]**
  **On Spines switches, use leaf’s IP address/AS numbers for BGP neighborhood (SW-1 uplinks IP addresses).
  ```

  ```
  cumulus@SW-1:$ net add bgp ipv4 unicast network 10.10.10.1/32
  cumulus@SW-1:$ net add bgp ipv4 unicast network 10.10.10.10/32
  cumulus@SW-1:$ net pending
  cumulus@SW-1:$ net commit
  ```

- **Run the following commands on SW-2:**

  ```
  cumulus@SW-1:$ net add bgp autonomous-system 65002
  cumulus@SW-1:$ net add bgp router-id 10.10.10.2
  cumulus@SW-1:$ net add bgp bestpath as-path multipath-relax
  cumulus@SW-1:$ net add bgp maximum-paths 2
  cumulus@SW-1:$ net add bgp neighbor [Spine1 interface IP] remote-as [Spine1 AS]**
  cumulus@SW-1:$ net add bgp neighbor [Spine2 interface IP] remote-as [Spine2 AS]**
  **On Spines switches, use leaf’s IP address/AS numbers for BGP neighborhood (SW-2 uplinks IP addresses).
  ```

  ```
  cumulus@SW-1:$ net add bgp ipv4 unicast network 10.10.10.1/32
  cumulus@SW-1:$ net add bgp ipv4 unicast network 10.10.10.10/32
  cumulus@SW-1:$ net pending
  cumulus@SW-1:$ net commit
  ```

- **Run the following commands on SW-3:**

  ```
  cumulus@SW-3:$ net add bgp autonomous-system 65003
  cumulus@SW-3:$ net add bgp router-id 100.100.100.100
  cumulus@SW-3:$ net add bgp bestpath as-path multipath-relax
  cumulus@SW-3:$ net add bgp maximum-paths 2
  cumulus@SW-3:$ net add bgp neighbor [Spine3 interface IP] remote-as [Spine3 AS]**
  cumulus@SW-3:$ net add bgp neighbor [Spine4 interface IP] remote-as [Spine4 AS]**
  **On Spines switches, use leaf’s IP address/AS numbers for BGP neighborhood (SW-3 uplinks IP addresses).
  ```

  ```
  cumulus@SW-3:$ net add bgp ipv4 unicast network 100.100.100.100/32
  cumulus@SW-3:$ net pending
  cumulus@SW-3:$ net commit
  ```

If the BGP IPv4 connectivity is configured correctly, IP reachability should be successful between SW-1 and 2 loopbacks addresses in the main site and the SW-3 loopback on Disaster Recovery Site.

BGP EVPN control plane is used to “stretch” L2 network over the L3 underlay physical environment. Each leaf switch is used as VXLAN Tunnel End-Point (VTEP) that translates VLAN to VXLAN encapsulation and vice versa. As we have established undelay IP connectivity, BGP EVPN neighborship must be established between the sites and VTEP IP addresses must be advertised using BGP IPv4.
3.6.2 Overlay Network Configuration Using VXLAN with BGP EVPN

VTEP's Configuration:

To configure Loopback interface that will be used as VTEP tunnel IP address for VXLAN encapsulation follow the below steps:

- Run the following commands on SW-1:
  
  ```
  cumulus@SW-1:~$ net add loopback lo ip address 10.10.10.1/32
  cumulus@SW-1:~$ net pending
  cumulus@SW-1:~$ net commit
  ```

- Run the following commands on SW-2:
  
  ```
  cumulus@SW-2:~$ net add loopback lo ip address 10.10.10.2/32
  cumulus@SW-2:~$ net pending
  cumulus@SW-2:~$ net commit
  ```

- Run the following commands on SW-3:
  
  ```
  cumulus@SW-3:~$ net add loopback lo ip address 100.100.100.100/32
  cumulus@SW-3:~$ net pending
  cumulus@SW-3:~$ net commit
  ```

In our case MLAG deployment exist on the main site, so both peers should be considered as “single” device from the VXLAN perspective and “Anycast-IP” should be configured.

To configure Anycast-IP follow the below steps:

- Run the following commands on SW-1:
  
  ```
  cumulus@SW-1:~$ net add loopback lo clag VXLAN -anycast-ip 10.10.10.10
  cumulus@SW-1:~$ net pending
  cumulus@SW-1:~$ net commit
  ```

- Run the following commands on SW-2:
  
  ```
  cumulus@SW-2:~$ net add loopback lo clag VXLAN -anycast-ip 10.10.10.10
  cumulus@SW-2:~$ net pending
  cumulus@SW-2:~$ net commit
  ```

BGP IPv4 neighbors are now established but both sites need to establish BGP EVPN neighborships.

For that, BGP EVPN address-family should be configured and neighbors activated within it. Each leaf will then be able to advertise L2 (MAC+IP) advertisements of VM’s LAN networks to other site. To do that, follow the below steps:

- Run the following commands on SW-1:
  
  ```
  cumulus@SW-1:~$ net add bgp neighbor 100.100.100.100 remote-as 65005
  cumulus@SW-1:~$ net add bgp neighbor 100.100.100.100 update-source lo
  cumulus@SW-1:~$ net add bgp neighbor 100.100.100.100 activate
  ```

- Run the following commands on SW-2:
  
  ```
  cumulus@SW-2:~$ net add bgp neighbor 100.100.100.100 remote-as 65005
  cumulus@SW-2:~$ net add bgp neighbor 100.100.100.100 update-source lo
  cumulus@SW-2:~$ net add bgp neighbor 100.100.100.100 activate
  ```

- Run the following commands on SW-3:
In case the cross-site has routing enabled, eBGP multihop must be used. For Dark fibre connection between sites, eBGP multihop is not required.

The following configuration is made per neighbor.

- Run the following commands on SW-1:

  ```
cumulus@SW-1:~$ net add bgp neighbor 10.10.10.1 remote-as 65002
cumulus@SW-1:~$ net add bgp neighbor 10.10.10.1 update-source lo

cumulus@SW-1:~$ net add bgp neighbor 10.10.10.1 activate

cumulus@SW-1:~$ net add bgp neighbor 10.10.10.2 remote-as 65002

cumulus@SW-1:~$ net add bgp neighbor 10.10.10.2 update-source lo

cumulus@SW-1:~$ net add bgp neighbor 10.10.10.2 activate

cumulus@SW-1:~$ net pending

cumulus@SW-1:~$ net commit
  ```

- Run the following commands on SW-2:

  ```
cumulus@SW-2:~$ net add bgp neighbor 10.10.10.2 remote-as 65002

cumulus@SW-2:~$ net add bgp neighbor 10.10.10.2 update-source lo

cumulus@SW-2:~$ net add bgp neighbor 10.10.10.2 activate

cumulus@SW-2:~$ net pending

cumulus@SW-2:~$ net commit
  ```

- Run the following commands on SW-3:

  ```
cumulus@SW-3:~$ net add bgp neighbor 10.10.10.1 remote-as 65002

cumulus@SW-3:~$ net add bgp neighbor 10.10.10.1 update-source lo

cumulus@SW-3:~$ net add bgp neighbor 10.10.10.1 activate

cumulus@SW-3:~$ net add bgp neighbor 10.10.10.2 remote-as 65002

cumulus@SW-3:~$ net add bgp neighbor 10.10.10.2 update-source lo

cumulus@SW-3:~$ net add bgp neighbor 10.10.10.2 activate

cumulus@SW-3:~$ net pending

cumulus@SW-3:~$ net commit
  ```

In addition, EVPN control plane should be able to advertise all VLAN→VNI mappings where NEO configures VLAN → VNI mapping automatically as per cluster networking in Prism Central.

To enable this functionality, run the following commands to configure under EVP

- Run the following commands on SW-1:

  ```
cumulus@SW-1:~$ net add bgp l2vpn evpn advertise all-vni

cumulus@SW-1:~$ net pending

cumulus@SW-1:~$ net commit
  ```

- Run the following commands on SW-2:

  ```
cumulus@SW-2:~$ net add bgp l2vpn evpn advertise all-vni

cumulus@SW-2:~$ net pending

cumulus@SW-2:~$ net commit
  ```

- Run the following commands on SW-3:

  ```
cumulus@SW-3:~$ net add bgp l2vpn evpn advertise all-vni

cumulus@SW-3:~$ net pending

cumulus@SW-3:~$ net commit
  ```

To cover various failure scenarios, BGP neighborship should be established between both MLAG peers on the peerlink interfaces IP addresses. This way in case of a failure, there will still be a way to transfer VXLAN traffic between peers.
To do that, BGP IPv4 and EVPN address-families neighborships must be established between the MLAG pair switches using the following commands:

- Run the following commands on SW-1:

```
cumulus@SW-1:$ net add bgp neighbor 169.254.1.2 remote-as 65002
cumulus@SW-1:$ net add bgp l2vpn evpn neighbor 169.254.1.2 activate
cumulus@SW-1:$ net pending
cumulus@SW-1:$ net commit
```

- Run the following commands on SW-2:

```
cumulus@SW-2:$ net add bgp neighbor 169.254.1.1 remote-as 65002
cumulus@SW-2:$ net add bgp l2vpn evpn neighbor 169.254.1.1 activate
cumulus@SW-2:$ net pending
cumulus@SW-2:$ net commit
```

After establishing IP reachability on the underlay and ensuring EVPN control plane is configured, now we need to create a VLAN per VM and assign it to a VXLAN interface (VNI).

### 3.6.3 VLAN to VNI mapping on Cumulus Linux

When a new VM is configured using Nutanix Prism, NEO gets the APIs and automates the interface to VLAN (VLAN to VNI mapping). When the mapping done, each VXLAN interface created per VNI will have all the parameters needed for it and the LAN gateway address is configured with Virtual IP for the VMs.

Below is the NEO™ VLAN to VXLAN configuration template:

```
net add vlan <vlan_id>
net add vlan <vlan_id> ip address <<vlan_addr>>
net add vlan <vlan_id> ip address-virtual <mac_addr> <gw_addr>
net add bridge bridge ports <vni_name>
net add bridge bridge vids <vlan_id>
net add VXLAN <vni_name> VXLAN_id <vni_id>
net add VXLAN <vni_name> bridge access <vlan_id>
net add VXLAN <vni_name> bridge arp-nd-suppress on
net add VXLAN <vni_name> bridge learning off
0net add VXLAN <vni_name> stp bpduguard
net add VXLAN <vni_name> stp portbdupfilter
net add VXLAN <vni_name> VXLAN local-tunnelip <<tunnel_ip>>
net add VXLAN <vni_name> mtu <mtu>*
net add <interface_type> <interface_name> bridge vids <VLAN_range_list>
net pending
net commit
```

After the above configuration takes place in Nutanix Prism, all LAN addresses should be advertised to other sites via BGP EVPN control plane to ensure L2 network is “stretched” to the Disaster Recovery site.

This is an automated process as we already configured EVPN to advertise all VNIs using “net add bgp l2vpn evpn advertise-all-vni” command under EVPN address-family.

As the environment will use VXLAN encapsulation, Jumbo MTU (9216B) should be set to support VXLAN headers that adds 50B to each packet.
NOTE:
Cross-site connection (ISP, dark fibre etc.) must support this MTU to enable VXLAN traffic to pass between sites.

### 3.7 Nutanix Prism Central Plug-in for Mellanox NEO

For Nutanix DCI solution, Nutanix Prism Central plugin needs to be installed on the NEO server to connect to Nutanix Prism Central.

The software plug-in is an add-on that offers enhanced functionality to Mellanox and Nutanix customers. Nutanix Prism Central offers enhanced network capabilities, including a set of APIs to use Prism Central's accumulated VM data. Mellanox uses these APIs to establish the integration solution between Prism Central and Mellanox, which adds network automation for Nutanix AHV. This integration addresses the most common use-cases of the Nutanix hyperconverged cloud: VLAN auto-provisioning on Mellanox switches for Nutanix VM creation, migration and deletion. By using this plug-in, customers can establish a connection to Nutanix AHV's events and have the infrastructure VLAN and VLAN-VNI mapping provisioned transparently. The plug-in can be installed and run on any CentOS/RHEL server that has connectivity to both Nutanix Prism Central and the NEO server.
The following table details the APIs used in this solution:

<table>
<thead>
<tr>
<th>Scope</th>
<th>URL</th>
<th>Version</th>
<th>URL</th>
<th>Method</th>
</tr>
</thead>
<tbody>
<tr>
<td>Prism/Cluster</td>
<td>/PrismGateway/services/rest/</td>
<td>v1.0</td>
<td>hosts</td>
<td>GET</td>
</tr>
<tr>
<td>Prism</td>
<td>/PrismGateway/services/rest/</td>
<td>v1.0</td>
<td>Hosts/&lt;uuid&gt;</td>
<td>GET</td>
</tr>
<tr>
<td>Cluster</td>
<td>/PrismGateway/services/rest/</td>
<td>v1.0</td>
<td>switches</td>
<td>GET</td>
</tr>
<tr>
<td>Prism</td>
<td>/PrismGateway/services/rest/</td>
<td>v1.0</td>
<td>clusters</td>
<td>GET</td>
</tr>
<tr>
<td>Cluster</td>
<td>/PrismGateway/services/rest/</td>
<td>v2.0</td>
<td>vms/&lt;uuid&gt;/virtual_nics</td>
<td>GET</td>
</tr>
<tr>
<td>Cluster</td>
<td>/PrismGateway/services/rest/</td>
<td>v2.0</td>
<td>vms/&lt;uuid&gt;/nics</td>
<td>GET</td>
</tr>
<tr>
<td>Prism</td>
<td>/api/nutanix/</td>
<td>v3.0</td>
<td>vms/&lt;uuid&gt;</td>
<td>GET</td>
</tr>
<tr>
<td>Cluster</td>
<td>/api/nutanix/</td>
<td>v3.0</td>
<td>webhooks/list</td>
<td>POST</td>
</tr>
<tr>
<td>Cluster</td>
<td>/api/nutanix/</td>
<td>v3.0</td>
<td>webhooks</td>
<td>POST</td>
</tr>
<tr>
<td>Cluster</td>
<td>/api/nutanix/</td>
<td>v3.0</td>
<td>subnets/list</td>
<td>POST</td>
</tr>
<tr>
<td>Cluster</td>
<td>/api/nutanix/</td>
<td>v3.0</td>
<td>vms/list</td>
<td>POST</td>
</tr>
</tbody>
</table>

Before installing the plugin, make sure you are using Nutanix Prism Central 5.7.1 and NEO v2.2 or above, SNMP and LLDP configured on the switches, Nutanix nodes physical connectivity to the switch established and Prism Central to NEO VM IP connectivity is established.

Download the plug-in from MyMellanox portal and run the RPM installation using the command “yum install nutanix-neo-1.3.0-3.x86_64.rpm”

To configure the plugin, fill the required details in the plugin configuration file located in /opt/nutanix-neo/config/nutanix-neo-plugin.cfg

Since this solution connects to the Prism Central, the credentials are those of the PC in use.
Start the service using “service nutanix-neo start” command to enable the plugin.

After plugin configuration file filled with the required information and the service started, we can start deploying the configuration using NEO™ automation.

Now we can create both clusters in Nutanix Prism. To do that follow the below:

1. Add the switches to the Nutanix Prism Element web UI. Click the wrench symbol on the right -> Network switch.

2. Create the Nutanix cluster network using Prism. Click the wrench symbol on the right -> Network Configuration. Make sure to edit the new network and identify the IP ranges if needed.
3. Repeat same steps to create the Disaster Recovery Site and add the switch to it.

For further information about Nutanix cluster network configuration, please refer to the Nutanix AHV Networking Best Practice Guide.
Now that the clusters were created, switches added to Prism Element and the plugin installed, let’s see per tenant automation in-action:

1. Create Networks on the “Main Site” cluster:

![Image of create networks on Main Site]

2. Select the cluster and create virtual network in it:

![Image of selecting cluster and creating network]

3. Define new network and set its VLAN and IP addressing for it (IP+Mask and default gateway):

   ![Image of defining network]

   In this example, each VM separated in different VLAN and has Network name of “Tenant_Net_[VLAN ID]”. 
4. Network “Tenant_Net_10” created with VLAN 10:

```
<table>
<thead>
<tr>
<th>NAME</th>
<th>VLAN ID</th>
</tr>
</thead>
<tbody>
<tr>
<td>Tenant_Net_10</td>
<td>10</td>
</tr>
</tbody>
</table>
```

5. Repeat the steps above for each network in Main Site from the diagram:
6. The same configuration will be made on “Disaster Recovery Site”. Same network names, VLANs and IP address assignments will be used here:
Now that the networks were created on both Nutanix clusters, NEO receives their respective network properties and configures the network switches accordingly.

1. To see the configuration, log into the NEO server and go to “Jobs”:

2. A job will be presented for each created network (on both clusters):

### Network Configuration

<table>
<thead>
<tr>
<th>Virtual Networks</th>
<th>Internal Interfaces</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Create Network</td>
</tr>
<tr>
<td>NAME</td>
<td>VLAN ID</td>
</tr>
<tr>
<td>Tenant_Net_10</td>
<td>vlan.10</td>
</tr>
<tr>
<td>Tenant_Net_20</td>
<td>vlan.20</td>
</tr>
<tr>
<td>Tenant_Net_30</td>
<td>vlan.30</td>
</tr>
<tr>
<td>Tenant_Net_40</td>
<td>vlan.40</td>
</tr>
</tbody>
</table>
3. To see the configuration commands template NEO automatically executed you can press “View Summary”. After pressed, a window will appear with the switches we want to look at (in our case there is MLAG in Main Site, so the configuration will be made on two switches.

4. Select the switch you would like to review and configuration commands on the switch will be displayed.

5. Now, we will create VMs in the respective networks.
6. Select the desired cluster to create the VM:

7. Configure the VM itself (name, compute details, NICs and disks):

You can scroll down for more parameters.
8. Now add HD to the VM by pressing “+Add New Disk” and choose its type:

   ![Add New Disk](image)

   ![Add Disk](image)

   In this example we are using HD to clone an image of CentOS7

9. Create Network card for the VM (NIC):

   ![Create VM](image)
10. Attach the VM to VLAN (from created networks) and assign IP address for the VM:

![Create NIC window](image)

11. Check and save VM configuration:

![Create VM window](image)
Now, the VM appears in VM list:

12. Create VMs B, C and D on Main Site and VMs 1, 2, 3 and 4 on “DR_Site” using the same steps.

13. “Power on” all the VMs:

That’s it! You now have DCI between the VMs in Main site and the VMs in Disaster Recovery Site for the same subnets.
### 3.7.1 Verifying site-to-site connectivity using EVPN/VXLAN

To verify the site-to-site connectivity status:

Connectivity test from VM_A on Main site to VM_1 on Disaster Recovery Site (VLAN10\(\rightarrow\)VNI10010):

```
[root@localhost ]# ping 192.168.1.106 -c 5
PING 192.168.1.106 (192.168.1.106) 56(84) bytes of data.
64 bytes from 192.168.1.106: icmp_seq=1 ttl=64 time=0.224 ms
64 bytes from 192.168.1.106: icmp_seq=2 ttl=64 time=0.116 ms
64 bytes from 192.168.1.106: icmp_seq=3 ttl=64 time=0.103 ms
64 bytes from 192.168.1.106: icmp_seq=4 ttl=64 time=0.136 ms
64 bytes from 192.168.1.106: icmp_seq=5 ttl=64 time=0.101 ms
--- 192.168.1.106 ping statistics ---
5 packets transmitted, 5 received, 0% packet loss, time 3999ms
rtt min/avg/max/mdev = 0.101/0.136/0.224/0.045 ms
```

Success!

Connectivity from VM_B on Main site to VM_2 on Disaster Recovery Site (VLAN20\(\rightarrow\)VNI10020):

```
[root@localhost ]# ping 192.168.2.107 -c 5
PING 192.168.2.107 (192.168.2.107) 56(84) bytes of data.
64 bytes from 192.168.2.107: icmp_seq=1 ttl=64 time=0.416 ms
64 bytes from 192.168.2.107: icmp_seq=2 ttl=64 time=0.215 ms
64 bytes from 192.168.2.107: icmp_seq=3 ttl=64 time=0.211 ms
64 bytes from 192.168.2.107: icmp_seq=4 ttl=64 time=0.137 ms
64 bytes from 192.168.2.107: icmp_seq=5 ttl=64 time=0.215 ms
--- 192.168.2.107 ping statistics ---
5 packets transmitted, 5 received, 0% packet loss, time 399ms
rtt min/avg/max/mdev = 0.137/0.250/0.416/0.095 ms
```

Success!

Connectivity from VM_C on Main site to VM_3 on Disaster Recovery Site (VLAN30\(\rightarrow\)VNI10030):

```
[root@localhost ]# ping 192.168.3.108 -c 5
PING 192.168.3.108 (192.168.3.108) 56(84) bytes of data.
64 bytes from 192.168.3.108: icmp_seq=1 ttl=64 time=0.436 ms
64 bytes from 192.168.3.108: icmp_seq=2 ttl=64 time=0.235 ms
64 bytes from 192.168.3.108: icmp_seq=3 ttl=64 time=0.238 ms
64 bytes from 192.168.3.108: icmp_seq=4 ttl=64 time=0.238 ms
64 bytes from 192.168.3.108: icmp_seq=5 ttl=64 time=0.199 ms
--- 192.168.3.108 ping statistics ---
5 packets transmitted, 5 received, 0% packet loss, time 4000ms
rtt min/avg/max/mdev = 0.199/0.284/0.436/0.080 ms
```

Success!

Connectivity from VM_D on Main site to VM_4 on Disaster Recovery Site (VLAN40\(\rightarrow\)VNI10040):

```
[root@localhost ]# ping 192.168.4.109 -c 5
PING 192.168.4.109 (192.168.4.109) 56(84) bytes of data.
64 bytes from 192.168.4.109: icmp_seq=1 ttl=64 time=0.486 ms
64 bytes from 192.168.4.109: icmp_seq=2 ttl=64 time=0.224 ms
64 bytes from 192.168.4.109: icmp_seq=3 ttl=64 time=0.244 ms
64 bytes from 192.168.4.109: icmp_seq=4 ttl=64 time=0.220 ms
64 bytes from 192.168.4.109: icmp_seq=5 ttl=64 time=0.247 ms
--- 192.168.4.109 ping statistics ---
5 packets transmitted, 5 received, 0% packet loss, time 4000ms
rtt min/avg/max/mdev = 0.220/0.284/0.486/0.102 ms
```

Success!
We can see that VMs on the same vlan have L2 connectivity across the sites. This indicates that L2 networks are “stretched” across the two sites connected over L3 networks.

Mellanox NEO allows for automated provisioning of switch MLAG, and VLAN and VNI creation per network. Appendix A shows how these steps can be manually configured through Cumulus Linux CLI, and how you can verify BGP EVPN control plane.

4 Appendix A: Manual DCI with VXLAN +EVPN and MLAG configuration using Cumulus Linux CLI

4.1 SNMP and LLDP configuration

Configure SNMP and LLDP on a Cumulus Linux switch to manage using NEO.

The following script is a prerequisite to the configuration process:

```bash
sudo service lldpd start
sudo systemctl start snmpd.service
sudo systemctl enable snmpd.service
net add snmp-server readonly
net add snmp-server listening-address all
net commit
```

4.2 Manual MLAG configuration using Cumulus Linux CLI:

4.2.1 Configure Peerlink Interface

Peerlink interface, a port-channel (bond) between switches that maintains state information of MLAG and MLAG-ports.

- Run the following commands on SW-1:

  ```bash
  cumulus@SW-1:~$ net add bond peerlink bond slaves swp5-6
  cumulus@SW-1:~$ net add interface peerlink.4094 ip address 169.254.1.10/30
  cumulus@SW-1:~$ net add interface peerlink.4094 clag peer-ip 169.254.1.2
  cumulus@SW-1:~$ net add interface peerlink.4094 clag backup-ip 192.0.2.50
  cumulus@SW-1:~$ net add interface peerlink.4094 clag sys-mac 44:38:39:FF:40:94
  cumulus@SW-1:~$ net pending
  cumulus@SW-1:~$ net commit
  ```

- Run the following commands on SW-2:

  ```bash
  cumulus@SW-2:~$ net add bond peerlink bond slaves swp11-12
  cumulus@SW-2:~$ net add interface peerlink.4094 ip address 169.254.1.12/30
  cumulus@SW-2:~$ net add interface peerlink.4094 clag peer-ip 169.254.1.10
  cumulus@SW-2:~$ net add interface peerlink.4094 clag backup-ip 192.0.2.50
  cumulus@SW-2:~$ net add interface peerlink.4094 clag sys-mac 44:38:39:FF:40:94
  cumulus@SW-2:~$ net pending
  cumulus@SW-2:~$ net commit
  ```
Notes:

- MLAG backup-ip used for MLAG peerlink failure detection and should be an IP address reachable via leaf’s management network
- Do not use 169.254.0.1 as the MLAG peerlink IP address. This address is used for BGP unnumbered in Cumulus Linux
- Same MAC address for different MLAG pairs cannot be used. Make sure you specify a different “clag sys-mac” setting for each MLAG pair in the network. To prevent MAC address conflicts with other interfaces in the same bridged network, use MAC address from MLAG specially reserved MAC range - 44:38:39:ff:00:00 to 44:38:39:ff:ff:ff.
- For high availability, we recommend having more than one physical link within this LAG
- All VLANs are open on the peerlink interface
- No need to add VLAN 4094 to the bridge VLAN list

To enable MLAG, peerlink must be added to bridge (or VLAN-aware bridge).

- Run the following commands on SW-1:

```bash
cumulus@SW-1:$ net add bridge bridge ports peerlink
cumulus@SW-1:$ net pending
cumulus@SW-1:$ net commit
```

- Run the following commands on SW-2:

```bash
cumulus@SW-2:$ net add bridge bridge ports peerlink
cumulus@SW-2:$ net pending
cumulus@SW-2:$ net commit
```

### 4.2.2 Configure MLAG Ports (Downlinks to hosts)

There are 4 MLAG ports—one for each host.

Host A is connected to MLAG-port (bond) 1, host B to MLAG-port 2, host C to MLAG-port 3 and host D to MLAG-port 4.

- Run the following commands on SW-1:

```bash
cumulus@SW-1:$ net add bond HostA bond slaves swp1
cumulus@SW-1:$ net add clag port bond HostA interface swp1 clag-id 1
cumulus@SW-1:$ net add bond HostB bond slaves swp2
cumulus@SW-1:$ net add clag port bond HostB interface swp2 clag-id 2
cumulus@SW-1:$ net add bond HostC bond slaves swp3
cumulus@SW-1:$ net add clag port bond HostC interface swp3 clag-id 3
cumulus@SW-1:$ net add bond HostD bond slaves swp4
cumulus@SW-1:$ net add clag port bond HostD interface swp4 clag-id 4
cumulus@SW-1:$ net pending
cumulus@SW-1:$ net commit
```

- Run the following commands on SW-2:

```bash
cumulus@SW-2:$ net add bond HostA bond slaves swp1
cumulus@SW-2:$ net add clag port bond HostA interface swp1 clag-id 1
cumulus@SW-2:$ net add bond HostB bond slaves swp2
cumulus@SW-2:$ net add clag port bond HostB interface swp2 clag-id 2
cumulus@SW-2:$ net add bond HostC bond slaves swp3
cumulus@SW-2:$ net add clag port bond HostC interface swp3 clag-id 3
cumulus@SW-2:$ net add bond HostD bond slaves swp4
cumulus@SW-2:$ net add clag port bond HostD interface swp4 clag-id 4
cumulus@SW-2:$ net pending
cumulus@SW-2:$ net commit
```
NOTE:
By default MLAG interfaces set to Active LACP mode, use “balance-xor” mode only if you cannot use LACP from some reason.

- Run the following commands on SW-1:

  ```
  cumulus@SW-1:$ net add bond HostA bond mode balance-xor
  cumulus@SW-1:$ net add bond HostB bond mode balance-xor
  cumulus@SW-1:$ net add bond HostC bond mode balance-xor
  cumulus@SW-1:$ net add bond HostD bond mode balance-xor
  ```

  ```
  cumulus@SW-1:$ net pending
  ```

  ```
  cumulus@SW-1:$ net commit
  ```

- Run the following commands on SW-2:

  ```
  cumulus@SW-2:$ net add bond HostA bond mode balance-xor
  cumulus@SW-2:$ net add bond HostB bond mode balance-xor
  cumulus@SW-2:$ net add bond HostC bond mode balance-xor
  cumulus@SW-2:$ net add bond HostD bond mode balance-xor
  ```

  ```
  cumulus@SW-2:$ net pending
  ```

  ```
  cumulus@SW-2:$ net commit
  ```

### 4.2.3 VNI and VLAN interfaces creation

- Run the following commands on SW-1:

  ```
  cumulus@SW-1:$ net add vlan 10 vlan-id 10
  cumulus@SW-1:$ net add vlan 20 vlan-id 20
  cumulus@SW-1:$ net add vlan 30 vlan-id 30
  cumulus@SW-1:$ net add vlan 40 vlan-id 40
  ```

  ```
  cumulus@SW-1:$ net add bridge bridge vids 10,20,30,40
  ```

  ```
  cumulus@SW-1:$ net add bond HostA bridge vids 10,20,30,40
  cumulus@SW-1:$ net add bond HostB bridge vids 10,20,30,40
  cumulus@SW-1:$ net add bond HostC bridge vids 10,20,30,40
  cumulus@SW-1:$ net add bond HostD bridge vids 10,20,30,40
  ```

  ```
  cumulus@SW-1:$ net add vlan 10 ip address 192.168.1.251/24
  ```

  ```
  cumulus@SW-1:$ net add vlan 10 ip address virtual 00:00:00:11:22:33 192.168.1.254/24
  ```

  ```
  cumulus@SW-1:$ net add vlan 20 ip address 192.168.2.251/24
  ```

  ```
  cumulus@SW-1:$ net add vlan 20 ip address virtual 00:00:00:11:22:33 192.168.2.254/24
  ```

  ```
  cumulus@SW-1:$ net add vlan 30 ip address 192.168.3.251/24
  ```

  ```
  cumulus@SW-1:$ net add vlan 30 ip address virtual 00:00:00:11:22:33 192.168.3.254/24
  ```

  ```
  cumulus@SW-1:$ net add vlan 40 ip address 192.168.4.251/24
  ```

  ```
  cumulus@SW-1:$ net add vlan 40 ip address virtual 00:00:00:11:22:33 192.168.4.254/24
  ```

  ```
  cumulus@SW-1:$ net add VXLAN vni10 VXLAN id 10010
  ```

  ```
  cumulus@SW-1:$ net add VXLAN vni10 bridge access 10
  ```

  ```
  cumulus@SW-1:$ net add VXLAN vni10 bridge arp-nd-suppress on
  ```

  ```
  cumulus@SW-1:$ net add VXLAN vni10 bridge learning off
  ```

  ```
  cumulus@SW-1:$ net add VXLAN vni10 stp bpduguard
  ```

  ```
  cumulus@SW-1:$ net add VXLAN vni10 stp portbpdufilter
  ```

  ```
  cumulus@SW-1:$ net add VXLAN vni10 VXLAN local-tunnelip 10.10.10.1
  ```

  ```
  cumulus@SW-1:$ net add VXLAN vni10 mtu 9216
  ```

  ```
  cumulus@SW-1:$ net add VXLAN vni20 VXLAN id 10020
  ```

  ```
  cumulus@SW-1:$ net add VXLAN vni20 bridge access 20
  ```

  ```
  cumulus@SW-1:$ net add VXLAN vni20 bridge arp-nd-suppress on
  ```
Run the following commands on SW-2:

```bash
cumulus@SW-2:$ net add VXLAN vni20 bridge learning off
cumulus@SW-2:$ net add VXLAN vni20 stp bpduguard
cumulus@SW-2:$ net add VXLAN vni20 stp portbpdudfilter
cumulus@SW-2:$ net add VXLAN vni20 VXLAN local-tunnelip 10.10.10.1
cumulus@SW-2:$ net add VXLAN vni20 mtu 9216*
cumulus@SW-2:$ net add VXLAN vni30 VXLAN id 10030
cumulus@SW-2:$ net add VXLAN vni30 bridge access 30
cumulus@SW-2:$ net add VXLAN vni30 bridge arp-md-suppress on
cumulus@SW-2:$ net add VXLAN vni30 bridge learning off
cumulus@SW-2:$ net add VXLAN vni30 stp bpduguard
cumulus@SW-2:$ net add VXLAN vni30 stp portbpdudfilter
cumulus@SW-2:$ net add VXLAN vni30 VXLAN local-tunnelip 10.10.10.1
cumulus@SW-2:$ net add VXLAN vni30 mtu 9216*
cumulus@SW-2:$ net add VXLAN vni40 VXLAN id 10040
cumulus@SW-2:$ net add VXLAN vni40 bridge access 40
cumulus@SW-2:$ net add VXLAN vni40 bridge arp-md-suppress on
cumulus@SW-2:$ net add VXLAN vni40 bridge learning off
cumulus@SW-2:$ net add VXLAN vni40 stp bpduguard
cumulus@SW-2:$ net add VXLAN vni40 stp portbpdudfilter
cumulus@SW-2:$ net add VXLAN vni40 VXLAN local-tunnelip 10.10.10.1
cumulus@SW-2:$ net add VXLAN vni40 mtu 9216*
cumulus@SW-2:$ net add bridge bridge ports vni10, vni20, vni30, vni40
```

- Run the following commands on SW-2:

```bash
cumulus@SW-2:$ net add vlan 10 vlan-id 10
cumulus@SW-2:$ net add vlan 20 vlan-id 20
cumulus@SW-2:$ net add vlan 30 vlan-id 30
cumulus@SW-2:$ net add vlan 40 vlan-id 40
```
Run the following commands on SW-3:

```
cumulus@SW-3:~$ net add vxlan vni10 vxlan id 10010
cumulus@SW-3:~$ net add vxlan vni10 bridge access 10
cumulus@SW-3:~$ net add vxlan vni10 bridge nd-suppress on
.cumulus@SW-3:~$ net add vxlan vni10 bridge learning off
.cumulus@SW-3:~$ net add vxlan vni10 stpduguard
.cumulus@SW-3:~$ net add vxlan vni10 stp portbpdufilter
.cumulus@SW-3:~$ net add vxlan vni10 vxlan local-tunnelip 100.100.100.100
.cumulus@SW-3:~$ net add vxlan vni10 vxlan mtu 9216*
.cumulus@SW-3:~$ net add vxlan vni20 vxlan id 10020
.cumulus@SW-3:~$ net add vxlan vni20 bridge access 20
.cumulus@SW-3:~$ net add vxlan vni20 bridge nd-suppress on
.cumulus@SW-3:~$ net add vxlan vni20 bridge learning off
.cumulus@SW-3:~$ net add vxlan vni20 stpduguard
.cumulus@SW-3:~$ net add vxlan vni20 stp portbpdufilter
.cumulus@SW-3:~$ net add vxlan vni20 vxlan local-tunnelip 100.100.100.100
.cumulus@SW-3:~$ net add vxlan vni20 vxlan mtu 9216*
.cumulus@SW-3:~$ net add vxlan vni30 vxlan id 10030
.cumulus@SW-3:~$ net add vxlan vni30 bridge access 30
.cumulus@SW-3:~$ net add vxlan vni30 bridge nd-suppress on
.cumulus@SW-3:~$ net add vxlan vni30 bridge learning off
.cumulus@SW-3:~$ net add vxlan vni30 stpduguard
.cumulus@SW-3:~$ net add vxlan vni30 stp portbpdufilter
.cumulus@SW-3:~$ net add vxlan vni30 vxlan local-tunnelip 100.100.100.100
.cumulus@SW-3:~$ net add vxlan vni30 vxlan mtu 9216*
.cumulus@SW-3:~$ net add vxlan vni40 vxlan id 10040
.cumulus@SW-3:~$ net add vxlan vni40 bridge access 40
.cumulus@SW-3:~$ net add vxlan vni40 bridge nd-suppress on
.cumulus@SW-3:~$ net add vxlan vni40 bridge learning off
.cumulus@SW-3:~$ net add vxlan vni40 stpduguard
.cumulus@SW-3:~$ net add vxlan vni40 stp portbpdufilter
.cumulus@SW-3:~$ net add vxlan vni40 vxlan local-tunnelip 100.100.100.100
.cumulus@SW-3:~$ net add vxlan vni40 vxlan mtu 9216*
.cumulus@SW-3:~$ net add bridge bridge ports vni10,vni20,vni30,vni40
.cumulus@SW-3:~$ net pending
.cumulus@SW-3:~$ net add bridge bridge ports vni10,vni20,vni30,vni40
.cumulus@SW-3:~$ net add bridge bridge ports vni20,vni30,vni40
.cumulus@SW-3:~$ net add bridge bridge ports vni30,vni40
.cumulus@SW-3:~$ net add bridge bridge ports vni40
.cumulus@SW-3:~$ net pending
.cumulus@SW-3:~$ net add bridge bridge ports vni10,vni20,vni30,vni40
.cumulus@SW-3:~$ net add bridge bridge ports vni20,vni30,vni40
.cumulus@SW-3:~$ net add bridge bridge ports vni30,vni40
.cumulus@SW-3:~$ net add bridge bridge ports vni40
.cumulus@SW-3:~$ net pending
```

```
cumulus@SW-3:$ net add VXLAN vni30 mtu 9216
cumulus@SW-3:$ net add VXLAN vni40 VXLAN id 10040
cumulus@SW-3:$ net add VXLAN vni40 bridge access 40
cumulus@SW-3:$ net add VXLAN vni40 bridge arp-nd-suppress on
cumulus@SW-3:$ net add VXLAN vni40 bridge learning off
cumulus@SW-3:$ net add VXLAN vni40 stp bpduguard
cumulus@SW-3:$ net add VXLAN vni40 stp portbpdufilter
cumulus@SW-3:$ net add VXLAN vni40 VXLAN local-tunnelip 100.100.100.100
cumulus@SW-3:$ net add VXLAN vni40 mtu 9216
cumulus@SW-3:$ net add bridge bridge ports vni10,vni20,vni30,vni40
cumulus@SW-3:$ net pending
cumulus@SW-3:$ net commit
5  Configuration Verification

After all configuration configured correctly on the setup. Both sites will see each other as same L2 domain using BGP EVPN control plane.

Only SW-1 and SW-3 will be shown in the below outputs, SW-2 should have the same outputs as SW-1 (MLAG).

- BGP IPv4 and EVPN neighborships (for underlay and Overlay route advertisement)

```bash
cumulus@SW-1:~$ net show bgp summary

show bgp ipv4 unicast summary

BGP router identifier 10.10.10.1, local AS number 65002 vrf-id 0
BGP table version 19
RIB entries 7, using 1064 bytes of memory
Peers 4, using 77 KiB of memory

Neighbor     V  AS MsgRcvd MsgSent   TblVer  InQ OutQ  Up/Down State/PfxRcd
1.1.1.2      4   65001  176052  176046        0    0    0 00:19:43
1.1.2.2      4   65004  201856  171989        0    0    0 00:19:43
169.254.1.2  4   65002  150152  150181        0    0    0 5d05h04m
Total number of neighbors 3

show bgp ipv6 unicast summary

show bgp l2vpn evpn summary

BGP router identifier 10.10.10.1, local AS number 65002 vrf-id 0
BGP table version 0
RIB entries 23, using 3496 bytes of memory
Peers 4, using 77 KiB of memory

Neighbor     V  AS MsgRcvd MsgSent   TblVer  InQ OutQ  Up/Down State/PfxRcd
100.100.100.100 4   65005   131    140        0    0    0 00:02:43
12
169.254.1.2  4   65002  150152  150181        0    0    0 5d05h04m
12
Total number of neighbors 2
```
show bgp ipv4 unicast summary
============================================
BGP router identifier 100.100.100.100, local AS number 65005 vrf-id 0
RIB entries 7, using 1064 bytes of memory
Peers 4, using 77 KiB of memory

<table>
<thead>
<tr>
<th>Neighbo</th>
<th>V</th>
<th>AS</th>
<th>MsgRcvd</th>
<th>MsgSent</th>
<th>TblVer</th>
<th>InQ</th>
<th>OutQ</th>
<th>Up/Down</th>
</tr>
</thead>
<tbody>
<tr>
<td>1.1.5.2</td>
<td>4</td>
<td>65006</td>
<td>150529</td>
<td>150433</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>00:24:41</td>
</tr>
<tr>
<td>1.1.6.2</td>
<td>4</td>
<td>65006</td>
<td>150534</td>
<td>150438</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>00:24:41</td>
</tr>
</tbody>
</table>

Total number of neighbors 2

show bgp ipv6 unicast summary
============================================
% No BGP neighbors found

show bgp l2vpn evpn summary
============================================
BGP router identifier 100.100.100.100, local AS number 65005 vrf-id 0
RIB entries 15, using 2280 bytes of memory
Peers 4, using 77 KiB of memory

<table>
<thead>
<tr>
<th>Neighbor</th>
<th>V</th>
<th>AS</th>
<th>MsgRcvd</th>
<th>MsgSent</th>
<th>TblVer</th>
<th>InQ</th>
<th>OutQ</th>
<th>Up/Down</th>
</tr>
</thead>
<tbody>
<tr>
<td>10.10.1.1</td>
<td>4</td>
<td>65002</td>
<td>215</td>
<td>209</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>00:06:46</td>
</tr>
<tr>
<td>10.10.2</td>
<td>4</td>
<td>65002</td>
<td>200</td>
<td>220</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>00:06:46</td>
</tr>
</tbody>
</table>

Total number of neighbors 2

All switches have BGP neighbors established with BGP IPv4 address-family to ensure underlay connectivity. EVPN address-family is also activated on the ToR switches to ensure MAC-IP route advertisements between sites.
• Underlay Routing Table to VTEPs (Loopbacks located on ToR switches)

```
cumulus@SW-1:$ net show route
show ip route

Codes: K - kernel route, C - connected, S - static, R - RIP, O - OSPF, I - IS-IS, B - BGP, E - EIGRP, N - NHRP, T - Table, v - VNC, V - VNC-Direct, A - Babel, D - SHARP, F - PBR, > - selected route, * - FIB route

K>* 0.0.0.0/0 [0/0] via 10.209.20.1, eth0, 22:39:47
C>* 1.1.1.0/24 is directly connected, swp2, 22:39:47
C>* 1.1.2.0/24 is directly connected, swp3, 22:39:47
C>* 10.10.10.0/24 is directly connected, peerlink.4094, 19:28:39
C>* 10.10.10.1/32 is directly connected, lo, 22:39:47
B>* 10.10.10.2/32 [200/0] via 169.254.1.2, peerlink.4094, 01:07:44
C>* 10.10.10.3/24 is directly connected, eth0, 22:39:47
C>* 1.1.5.0/24 is directly connected, swp5, 01:44:24
C>* 1.1.6.0/24 is directly connected, swp6, 01:44:24
B>* 10.10.10.1/32 [200/0] via 169.254.1.2, peerlink.4094, 01:07:44
C>* 10.10.10.2/32 [20/0] via 169.254.1.2, peerlink.4094, 01:07:44
C>* 10.10.10.3/24 is directly connected, eth0, 22:39:47
C>* 1.1.5.0/24 is directly connected, swp5, 01:44:24
C>* 1.1.6.0/24 is directly connected, swp6, 01:44:24
B>* 10.209.20.0/22 is directly connected, eth0, 01:44:24
B>* 10.209.20.1/32 [20/0] via 1.1.2.2, swp2, 00:00:11
C>* 169.254.1.0/30 is directly connected, peerlink.4094, 19:28:39
C * 192.168.1.0/24 is directly connected, vlan10-v0, 19:28:30
C * 192.168.2.0/24 is directly connected, vlan20-v0, 19:28:30
C * 192.168.3.0/24 is directly connected, vlan30-v0, 19:28:30
C * 192.168.4.0/24 is directly connected, vlan40-v0, 19:28:30
```

As we can see, each ToR switch has L3 connectivity (routes) to the destination VTEP on remote site allowing VXLAN tunnel encapsulation and deliver the packets to a remote VTEP.
BGP EVPN Routes

cumulus@SW-1:$ net show bgp evpn route
BGP table version is 15, local router ID is 10.10.10.1
Status codes: s suppressed, d damped, h history, * valid, > best, i - internal
Origin codes: i - IGP, e - EGP, ? - incomplete
EVPN type-2 prefix: [2]:[ESI]:[EthTag]:[MAClen]:[MAC]:[IPlen]:[IP]
EVPN type-3 prefix: [3]:[EthTag]:[IPlen]:[OrigIP]
EVPN type-5 prefix: [5]:[ESI]:[EthTag]:[IPlen]:[IP]

<table>
<thead>
<tr>
<th>Network Route Distinguisher</th>
<th>Next Hop</th>
<th>Metric</th>
<th>LocPrf</th>
<th>Weight</th>
<th>Path</th>
</tr>
</thead>
<tbody>
<tr>
<td>*&gt; [2]:[0]:[48]:[00:00:10:00:00:10]</td>
<td>10.10.10.10</td>
<td>32768</td>
<td>i</td>
<td></td>
<td></td>
</tr>
<tr>
<td>*&gt; [2]:[0]:[48]:[00:00:10:00:00:10]:[32]:[192.168.1.10]</td>
<td>10.10.10.10</td>
<td>32768</td>
<td>i</td>
<td></td>
<td></td>
</tr>
<tr>
<td>*&gt; [3]:[32]:[10.10.10.10]</td>
<td>10.10.10.10</td>
<td>32768</td>
<td>i</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Route Distinguisher: 10.10.10.1:2</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>*&gt; [2]:[0]:[48]:[00:00:30:00:00:12]</td>
<td>10.10.10.10</td>
<td>32768</td>
<td>i</td>
<td></td>
<td></td>
</tr>
<tr>
<td>*&gt; [2]:[0]:[48]:[00:00:30:00:00:12]:[32]:[192.168.3.12]</td>
<td>10.10.10.10</td>
<td>32768</td>
<td>i</td>
<td></td>
<td></td>
</tr>
<tr>
<td>*&gt; [3]:[32]:[10.10.10.10]</td>
<td>10.10.10.10</td>
<td>32768</td>
<td>i</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Route Distinguisher: 10.10.10.1:3</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>*&gt; [2]:[0]:[48]:[00:00:20:00:00:11]</td>
<td>10.10.10.10</td>
<td>32768</td>
<td>i</td>
<td></td>
<td></td>
</tr>
<tr>
<td>*&gt; [2]:[0]:[48]:[00:00:20:00:00:11]:[32]:[192.168.2.11]</td>
<td>10.10.10.10</td>
<td>32768</td>
<td>i</td>
<td></td>
<td></td>
</tr>
<tr>
<td>*&gt; [3]:[32]:[10.10.10.10]</td>
<td>10.10.10.10</td>
<td>32768</td>
<td>i</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Route Distinguisher: 10.10.10.1:4</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>*&gt; [2]:[0]:[48]:[00:00:40:00:00:13]</td>
<td>10.10.10.10</td>
<td>32768</td>
<td>i</td>
<td></td>
<td></td>
</tr>
<tr>
<td>*&gt; [2]:[0]:[48]:[00:00:40:00:00:13]:[32]:[192.168.4.13]</td>
<td>10.10.10.10</td>
<td>32768</td>
<td>i</td>
<td></td>
<td></td>
</tr>
<tr>
<td>*&gt; [3]:[32]:[10.10.10.10]</td>
<td>10.10.10.10</td>
<td>32768</td>
<td>i</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Route Distinguisher: 100.100.100.100:7</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>* [2]:[0]:[48]:[00:00:10:00:01:06]</td>
<td>100.100.100.100</td>
<td>100</td>
<td>0</td>
<td>65005</td>
<td>i</td>
</tr>
<tr>
<td>*&gt; [2]:[0]:[48]:[00:00:10:00:01:06]</td>
<td>100.100.100.100</td>
<td>0</td>
<td>65005</td>
<td>i</td>
<td></td>
</tr>
<tr>
<td>* [2]:[0]:[48]:[00:00:10:00:01:06]:[32]:[192.168.1.106]</td>
<td>100.100.100.100</td>
<td>0</td>
<td>65005</td>
<td>i</td>
<td></td>
</tr>
<tr>
<td>*&gt; [2]:[0]:[48]:[00:00:10:00:01:06]:[32]:[192.168.1.106]</td>
<td>100.100.100.100</td>
<td>0</td>
<td>65005</td>
<td>i</td>
<td></td>
</tr>
<tr>
<td>* [3]:[32]:[100.100.100.100]</td>
<td>100.100.100.100</td>
<td>100</td>
<td>0</td>
<td>65005</td>
<td>i</td>
</tr>
<tr>
<td>*&gt; [3]:[0]:[32]:[100.100.100.100]</td>
<td>100.100.100.100</td>
<td>0</td>
<td>65005</td>
<td>i</td>
<td></td>
</tr>
<tr>
<td>Route Distinguisher: 100.100.100.100:8</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>* [2]:[0]:[48]:[00:00:30:00:01:08]</td>
<td>100.100.100.100</td>
<td>100</td>
<td>0</td>
<td>65005</td>
<td>i</td>
</tr>
<tr>
<td>*&gt; [2]:[0]:[48]:[00:00:30:00:01:08]</td>
<td>100.100.100.100</td>
<td>0</td>
<td>65005</td>
<td>i</td>
<td></td>
</tr>
<tr>
<td>* [2]:[0]:[48]:[00:00:30:00:01:08]:[32]:[192.168.3.108]</td>
<td>100.100.100.100</td>
<td>0</td>
<td>65005</td>
<td>i</td>
<td></td>
</tr>
<tr>
<td>*&gt; [2]:[0]:[48]:[00:00:30:00:01:08]:[32]:[192.168.3.108]</td>
<td>100.100.100.100</td>
<td>0</td>
<td>65005</td>
<td>i</td>
<td></td>
</tr>
<tr>
<td>* [3]:[32]:[100.100.100.100]</td>
<td>100.100.100.100</td>
<td>100</td>
<td>0</td>
<td>65005</td>
<td>i</td>
</tr>
<tr>
<td>*&gt; [3]:[0]:[32]:[100.100.100.100]</td>
<td>100.100.100.100</td>
<td>0</td>
<td>65005</td>
<td>i</td>
<td></td>
</tr>
</tbody>
</table>
Route Distinguisher: 100.100.100.100:9
* i [2]:[0]:[0]:[48]:[00:00:20:00:01:07]
  100.100.100.100
*> [2]:[0]:[0]:[48]:[00:00:20:00:01:07]
  100.100.100.100
* 1[2]:[0]:[0]:[48]:[00:00:20:00:01:07]:[32]:[192.168.2.107]
  100.100.100.100
*> [2]:[0]:[0]:[48]:[00:00:20:00:01:07]:[32]:[192.168.2.107]
  100.100.100.100
* i [3]:[0]:[32]:[100.100.100.100]
  100.100.100.100
*> [3]:[0]:[32]:[100.100.100.100]
  100.100.100.100
Route Distinguisher: 100.100.100.100:10
* i [2]:[0]:[0]:[48]:[00:00:40:00:01:09]
  100.100.100.100
*> [2]:[0]:[0]:[48]:[00:00:40:00:01:09]
  100.100.100.100
* 1[2]:[0]:[0]:[48]:[00:00:40:00:01:09]:[32]:[192.168.4.109]
  100.100.100.100
*> [2]:[0]:[0]:[48]:[00:00:40:00:01:09]:[32]:[192.168.4.109]
  100.100.100.100
* i [3]:[0]:[32]:[100.100.100.100]
  100.100.100.100
*> [3]:[0]:[32]:[100.100.100.100]
  100.100.100.100

Displayed 24 prefixes (36 paths)
```plaintext
cumulus@SW-3:~$ net show bgp evpn route
BGP table version is 18, local router ID is 100.100.100.100
Status codes: s suppressed, d damped, h history, * valid, > best, i - internal
Origin codes: i - IGP, e - EGP, ? - incomplete
EVPN type-2 prefix: [2]:[ESI]:[EthTag]:[MAClen]:[MAC]:[IPlen]:[IP]
EVPN type-3 prefix: [3]:[EthTag]:[IPlen]:[OrigIP]
EVPN type-5 prefix: [5]:[ESI]:[EthTag]:[IPlen]:[IP]

<table>
<thead>
<tr>
<th>Network</th>
<th>Next Hop</th>
<th>Metric</th>
<th>LocPrf</th>
<th>Weight</th>
<th>Path</th>
</tr>
</thead>
</table>
| Route Distinguisher: 10.10.10.1:2  
*> [2]:[0]:[0]:[48]:[00:00:10:00:00:10]  
10.10.10.10  | 0 65002 | i      |
| * [2]:[0]:[0]:[48]:[00:00:10:00:00:10]:[32]:[192.168.1.10]  
10.10.10.10  | 0 65002 | i      |
| * [3]:[0]:[32]:[10.10.10.10]  
10.10.10.10  | 0 65002 | i      |
| Route Distinguisher: 10.10.10.1:3  
*> [2]:[0]:[0]:[48]:[00:00:30:00:00:12]  
10.10.10.10  | 0 65002 | i      |
| * [2]:[0]:[0]:[48]:[00:00:30:00:00:12]:[32]:[192.168.3.12]  
10.10.10.10  | 0 65002 | i      |
| * [3]:[0]:[32]:[10.10.10.10]  
10.10.10.10  | 0 65002 | i      |
| Route Distinguisher: 10.10.10.1:4  
*> [2]:[0]:[0]:[48]:[00:00:20:00:00:11]  
10.10.10.10  | 0 65002 | i      |
| * [2]:[0]:[0]:[48]:[00:00:20:00:00:11]:[32]:[192.168.2.11]  
10.10.10.10  | 0 65002 | i      |
| * [3]:[0]:[32]:[10.10.10.10]  
10.10.10.10  | 0 65002 | i      |
| Route Distinguisher: 10.10.10.1:5  
*> [2]:[0]:[0]:[48]:[00:00:40:00:00:13]  
10.10.10.10  | 0 65002 | i      |
| * [2]:[0]:[0]:[48]:[00:00:40:00:00:13]:[32]:[192.168.4.13]  
10.10.10.10  | 0 65002 | i      |
| * [3]:[0]:[32]:[10.10.10.10]  
10.10.10.10  | 0 65002 | i      |
| Route Distinguisher: 100.100.100.100:7  
*> [2]:[0]:[0]:[48]:[00:00:10:00:00:01:06]  
100.100.100.100  | 32768 | i      |
| * [2]:[0]:[0]:[48]:[00:00:10:00:00:01:06]:[32]:[192.168.1.106]  
100.100.100.100  | 32768 | i      |
| * [3]:[0]:[32]:[100.100.100.100]  
100.100.100.100  | 32768 | i      |
| Route Distinguisher: 100.100.100.100:8  
*> [2]:[0]:[0]:[48]:[00:00:30:00:00:01:08]  
100.100.100.100  | 32768 | i      |
| * [2]:[0]:[0]:[48]:[00:00:30:00:00:01:08]:[32]:[192.168.3.108]  
100.100.100.100  | 32768 | i      |
| * [3]:[0]:[32]:[100.100.100.100]  
100.100.100.100  | 32768 | i      |
| Route Distinguisher: 100.100.100.100:9  
*> [2]:[0]:[0]:[48]:[00:00:20:00:00:01:07]  
100.100.100.100  | 32768 | i      |
| * [2]:[0]:[0]:[48]:[00:00:20:00:00:01:07]:[32]:[192.168.2.107]  
100.100.100.100  | 32768 | i      |
| * [3]:[0]:[32]:[100.100.100.100]  
100.100.100.100  | 32768 | i      |
| Route Distinguisher: 100.100.100.100:10  
*> [2]:[0]:[0]:[48]:[00:00:40:00:00:01:09]  
100.100.100.100  | 32768 | i      |
| * [2]:[0]:[0]:[48]:[00:00:40:00:00:01:09]:[32]:[192.168.4.109]  
100.100.100.100  | 32768 | i      |
| * [3]:[0]:[32]:[100.100.100.100]  
100.100.100.100  | 32768 | i      |
```
As we can see from output above, EVPN control plane is successfully converged and EVPN routes appear in BGP EVPN tables of each ToR. By this, each site will be able to forward encapsulated VXLAN packets to remote site hosts.

- VNI Information

<table>
<thead>
<tr>
<th>Tenant VRF</th>
<th>Type</th>
<th>RD</th>
<th>Import RT</th>
<th>Export RT</th>
</tr>
</thead>
<tbody>
<tr>
<td>* 10030</td>
<td>L2</td>
<td>10.10.10.1:3</td>
<td>65002:10030</td>
<td>65002:10030</td>
</tr>
<tr>
<td></td>
<td>Route</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>* 10010</td>
<td>L2</td>
<td>10.10.10.1:2</td>
<td>65002:10010</td>
<td>65002:10010</td>
</tr>
<tr>
<td></td>
<td>Route</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>* 10040</td>
<td>L2</td>
<td>10.10.10.1:5</td>
<td>65002:10040</td>
<td>65002:10040</td>
</tr>
<tr>
<td></td>
<td>Route</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>* 10020</td>
<td>L2</td>
<td>10.10.10.1:4</td>
<td>65002:10020</td>
<td>65002:10020</td>
</tr>
<tr>
<td></td>
<td>Route</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Shows the configuration for VNI interfaces and route-targets used in BGP to exchange information.
MAC addresses information

cumulus@SW-1:$ net show evpn mac vni all

<table>
<thead>
<tr>
<th>VNI 10030</th>
<th>#MACs (local and remote) 2</th>
</tr>
</thead>
<tbody>
<tr>
<td>MAC</td>
<td>Type</td>
</tr>
<tr>
<td>00:00:30:00:00:12</td>
<td>local</td>
</tr>
<tr>
<td>ec:0d:9a:5f:9f:18</td>
<td>local</td>
</tr>
<tr>
<td>00:00:30:00:00:12</td>
<td>remote</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>VNI 10010</th>
<th>#MACs (local and remote) 2</th>
</tr>
</thead>
<tbody>
<tr>
<td>MAC</td>
<td>Type</td>
</tr>
<tr>
<td>00:00:5e:00:01:01</td>
<td>local</td>
</tr>
<tr>
<td>00:00:10:00:00:06</td>
<td>remote</td>
</tr>
<tr>
<td>ec:0d:9a:5f:9f:18</td>
<td>local</td>
</tr>
<tr>
<td>00:00:00:11:22:33</td>
<td>local</td>
</tr>
<tr>
<td>00:00:10:00:00:10</td>
<td>local</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>VNI 10040</th>
<th>#MACs (local and remote) 2</th>
</tr>
</thead>
<tbody>
<tr>
<td>MAC</td>
<td>Type</td>
</tr>
<tr>
<td>00:00:40:00:00:01:09</td>
<td>remote</td>
</tr>
<tr>
<td>ec:0d:9a:5f:9f:18</td>
<td>local</td>
</tr>
<tr>
<td>00:00:00:11:22:33</td>
<td>local</td>
</tr>
<tr>
<td>00:00:40:00:00:13</td>
<td>local</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>VNI 10020</th>
<th>#MACs (local and remote) 2</th>
</tr>
</thead>
<tbody>
<tr>
<td>MAC</td>
<td>Type</td>
</tr>
<tr>
<td>00:00:20:00:00:01:07</td>
<td>remote</td>
</tr>
<tr>
<td>00:00:20:00:00:11</td>
<td>local</td>
</tr>
<tr>
<td>ec:0d:9a:5f:9f:18</td>
<td>local</td>
</tr>
<tr>
<td>00:00:00:11:22:33</td>
<td>local</td>
</tr>
</tbody>
</table>
```
VLAN Master Interface  MAC                            TunnelDest State Flags
---- ------ -------- ----------- ----------------- ----------- ------- -------
10    bridge HostA  00:00:10:00:00:10
10    bridge bridge 00:00:00:11:22:33      permanent
10    bridge bridge  ec:0d:9a:5f:9f:0c      permanent
10    bridge vni10  00:00:10:00:00:10:06  offload  00:33:56
20    bridge HostA  00:00:20:00:00:11
20    bridge bridge 00:00:00:11:22:33      permanent
20    bridge bridge  ec:0d:9a:5f:9f:0c      permanent
20    bridge vni20  00:00:20:00:00:10:07  offload  00:33:56
30    bridge HostA  00:00:30:00:00:12
30    bridge bridge 00:00:00:11:22:33      permanent
30    bridge bridge  ec:0d:9a:5f:9f:0c      permanent
30    bridge vni30  00:00:30:00:00:10:08  offload  00:33:56
40    bridge HostA  00:00:40:00:00:13
40    bridge bridge 00:00:00:11:22:33      permanent
40    bridge bridge  ec:0d:9a:5f:9f:0c      permanent
40    bridge vni40  00:00:40:00:00:10:09  offload  00:33:56
untagged bridge never 00:00:00:11:22:33      permanent
untagged bridge never 00:00:5e:00:00:01:01  permanent
untagged bridge never 00:00:5e:00:00:01:02  permanent
untagged bridge never 00:00:5e:00:00:01:03  permanent
untagged bridge never 00:00:5e:00:00:01:04  permanent
untagged bridge never  ec:0d:9a:5f:9f:18  permanent
untagged bridge never  vlan10 00:00:00:11:22:33  permanent
untagged bridge never  vlan10 00:00:5e:00:00:01:01  permanent
untagged bridge never  vlan20 00:00:00:11:22:33  permanent
untagged bridge never  vlan20 00:00:5e:00:00:01:02  permanent
untagged bridge never  vlan30 00:00:00:11:22:33  permanent
untagged bridge never  vlan30 00:00:5e:00:00:01:03  permanent
untagged bridge never  vlan40 00:00:00:11:22:33  permanent
untagged bridge never  vlan40 00:00:5e:00:00:01:04  permanent
```
### VNI 10030 #MACs (local and remote) 4

<table>
<thead>
<tr>
<th>MAC</th>
<th>Type</th>
<th>Intf/Remote VTEP</th>
<th>VLAN</th>
</tr>
</thead>
<tbody>
<tr>
<td>ec:0d:9a:fd:61:10</td>
<td>local</td>
<td>vlan30</td>
<td>30</td>
</tr>
<tr>
<td>00:00:30:00:00:12</td>
<td>remote</td>
<td>10.10.10.10</td>
<td></td>
</tr>
<tr>
<td>00:00:00:11:22:33</td>
<td>local</td>
<td>vlan30-v0</td>
<td>30</td>
</tr>
<tr>
<td>00:00:30:00:01:08</td>
<td>local</td>
<td>swp3</td>
<td>30</td>
</tr>
</tbody>
</table>

### VNI 10010 #MACs (local and remote) 4

<table>
<thead>
<tr>
<th>MAC</th>
<th>Type</th>
<th>Intf/Remote VTEP</th>
<th>VLAN</th>
</tr>
</thead>
<tbody>
<tr>
<td>00:00:10:00:01:06</td>
<td>local</td>
<td>swp3</td>
<td>10</td>
</tr>
<tr>
<td>ec:0d:9a:fd:61:18</td>
<td>local</td>
<td>vlan10</td>
<td>10</td>
</tr>
<tr>
<td>00:00:00:11:22:33</td>
<td>local</td>
<td>vlan10-v0</td>
<td>10</td>
</tr>
<tr>
<td>00:00:10:00:00:10</td>
<td>remote</td>
<td>10.10.10.10</td>
<td></td>
</tr>
</tbody>
</table>

### VNI 10040 #MACs (local and remote) 4

<table>
<thead>
<tr>
<th>MAC</th>
<th>Type</th>
<th>Intf/Remote VTEP</th>
<th>VLAN</th>
</tr>
</thead>
<tbody>
<tr>
<td>00:00:40:00:00:01:09</td>
<td>local</td>
<td>swp3</td>
<td>40</td>
</tr>
<tr>
<td>ec:0d:9a:fd:61:10</td>
<td>local</td>
<td>vlan40</td>
<td>40</td>
</tr>
<tr>
<td>00:00:00:11:22:33</td>
<td>local</td>
<td>vlan40-v0</td>
<td>40</td>
</tr>
<tr>
<td>00:00:40:00:00:13</td>
<td>remote</td>
<td>10.10.10.10</td>
<td></td>
</tr>
</tbody>
</table>

### VNI 10020 #MACs (local and remote) 4

<table>
<thead>
<tr>
<th>MAC</th>
<th>Type</th>
<th>Intf/Remote VTEP</th>
<th>VLAN</th>
</tr>
</thead>
<tbody>
<tr>
<td>00:00:20:00:01:07</td>
<td>local</td>
<td>swp3</td>
<td>20</td>
</tr>
<tr>
<td>00:00:20:00:00:11</td>
<td>remote</td>
<td>10.10.10.10</td>
<td></td>
</tr>
<tr>
<td>ec:0d:9a:fd:61:10</td>
<td>local</td>
<td>vlan20</td>
<td>20</td>
</tr>
<tr>
<td>00:00:00:11:22:33</td>
<td>local</td>
<td>vlan20-v0</td>
<td>20</td>
</tr>
</tbody>
</table>
cumulus@SW-3:~$ net show bridge macs

<table>
<thead>
<tr>
<th>VLAN</th>
<th>Master</th>
<th>Interface</th>
<th>MAC</th>
<th>TunnelDest</th>
<th>State</th>
<th>Flags</th>
</tr>
</thead>
<tbody>
<tr>
<td>10</td>
<td>bridge</td>
<td>bridge</td>
<td>00:00:00:11:22:33</td>
<td>permanent</td>
<td></td>
<td></td>
</tr>
<tr>
<td>10</td>
<td>bridge</td>
<td>bridge</td>
<td>ec:0d:9a:fd:61:10</td>
<td>permanent</td>
<td></td>
<td></td>
</tr>
<tr>
<td>10</td>
<td>bridge</td>
<td>swp3</td>
<td>00:00:10:00:00:01</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>bridge</td>
<td>vni10</td>
<td>00:00:10:00:00:10</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>bridge</td>
<td>vni20</td>
<td>00:00:20:00:00:11</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>bridge</td>
<td>vni30</td>
<td>00:00:30:00:00:12</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>bridge</td>
<td>vni40</td>
<td>00:00:40:00:00:13</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>bridge</td>
<td>vlan10</td>
<td>00:00:00:11:22:33</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>bridge</td>
<td>vlan20</td>
<td>00:00:00:11:22:33</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>bridge</td>
<td>vlan30</td>
<td>00:00:00:11:22:33</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>bridge</td>
<td>vlan40</td>
<td>00:00:00:11:22:33</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>bridge</td>
<td>vni10</td>
<td>10.10.10.10</td>
<td>permanent</td>
<td>self</td>
<td></td>
</tr>
<tr>
<td></td>
<td>bridge</td>
<td>vni20</td>
<td>10.10.10.10</td>
<td>permanent</td>
<td>self</td>
<td></td>
</tr>
<tr>
<td></td>
<td>bridge</td>
<td>vni30</td>
<td>10.10.10.10</td>
<td>permanent</td>
<td>self</td>
<td></td>
</tr>
<tr>
<td></td>
<td>bridge</td>
<td>vni40</td>
<td>10.10.10.10</td>
<td>permanent</td>
<td>self</td>
<td></td>
</tr>
</tbody>
</table>

<1 sec
As we can see, both sites have remote MAC addresses stored locally in their MAC address-table (offload to FDB). That means L2 networks are “stretched” successfully and Virtual Machines can communicate over L3 networks between the sites.