Arista OpenStack Deployment Guide

openstack-dev@arista.com

Table of contents

Table of contents

Table of contents

Introduction

Arista Networks was founded to pioneer and deliver software-driven cloud networking solutions for large data centers, storage and computing environments. Arista's award-winning platforms, ranging in Ethernet speeds from 10 to 100 gigabits per second, redefine scalability, agility and resilience. Arista has shipped more than ten million cloud networking ports worldwide with CloudVision and EOS, an advanced network operating system. Committed to open standards, Arista is a founding member of the 25/50GbE consortium. Arista Networks products are available worldwide directly and through partners.

Arista was recognized by Gartner as a "leader" in the "2016 Magic Quadrant for Data Center Networking" based on a number of factors, including high growth, technology solutions and flexible software. The Arista team is comprised of experienced management and engineering talent from leading networking companies. Arista designs revolutionary products and delivers them worldwide through distribution partners, systems integrators and resellers with a strong dedication to partner and customer success.

At the core of Arista's platform is the Extensible Operating System (EOS™), a ground-breaking network operating system with single-image consistency across hardware platforms, and modern core architecture enabling in-service upgrades and application extensibility.

Arista EOS™ has extensive integration with the OpenStack project, giving customers a powerful network platform on which to run OpenStack deployments. By leveraging the Arista ML2 driver and Layer 3 service plugin, operators can automatically provision tenant networks across the physical infrastructure. This combination provides a high-performance OpenStack networking environment over VLAN and VXLAN based fabrics, along with enhanced visibility into how the virtual tenant networks map onto the physical infrastructure.

This guide provides steps to deploy OpenStack Neutron along with Arista type driver, mechanism drivers and/or L3 plugin in order to automate provisioning of Arista Networks switches. It walks through the different components in the deployment, design considerations and finally detailed steps to configure the different components in the solution. Please note that there are other methods of deploying OpenStack with Arista equipment which are out of scope for this document.

It is assumed that the reader is familiar with OpenStack and has a functional OpenStack deployment before they attempt to perform this integration. Please note that the different OpenStack distributions differ slightly when it comes to specifying the configuration files used by OpenStack Neutron. Please refer to the relevant distribution configuration guide for details.

Getting Started

The section below is divided into three (3) subsections, and contains information on supported OpenStack distributions, OpenStack releases and how to obtain support from the Arista Technical Assistance Center (TAC).

Distributions

The Arista OpenStack Neutron Modular Layer 2 (ML2) driver, Layer 3 (L3) plugin and all other software provided in the networking-arista package have been tested against different distributions of OpenStack. No code is specific to a distribution. Installation instructions do vary between distributions, see RHOSP Installation for instructions on deployment through RHOSP director.

[1] https://github.com/openstack/networking-arista

Supported Versions

Table 1: OpenStack release support matrix					
OpenStack Release	**OpenStack EOL Date**	**networking-arista version**	**Support Status**	**Minimum EOS Version**	**Maximum EOS Version**
Yoga	Extended Maintenance - 2023-09-30(est)	>=2022.1,<2022.2	Planned	TBD	TBD
Xena	Extended Maintenance - 2023-04-06	>=2021.2,<2022.1	Supported	4.21.0F	N/A
Wallaby	Extended Maintenance - 2022-10-14	>=2021.1,<2021.2	Supported	4.21.0F	N/A
Victoria	Extended Maintenance - 2022-4-18	>=2020.2,<2021.1	Supported	4.21.0F	N/A
Ussuri	Extended Maintenance - 2021-10-12	>=2020.1,<2020.2	Supported	4.21.0F	N/A
Train	Extended Maintenance - 2021-4-16	>=2019.2,<2020.1	Supported	4.21.0F	N/A
Stein	Extended Maintenance - 2020-10-10	>=2019.1,<2019.2	Supported	4.21.0F	N/A
Rocky	Extended Maintenance - 2020-02-24	>=2018.2,<2019.1	Supported	4.21.0F	N/A
Queens	Extended Maintenance - 2019-08-25	>=2018.1,<2018.2	Best Effort	4.21.0F	N/A
Pike	Extended Maintenance - 2019-03-03	>=2017.2,<2018.1	Best Effort	4.15.5F	N/A
Ocata	2020-05-14 (EOL)	>=2017.1,<2017.2	Best Effort	4.15.5F	N/A
Newton	2017-10-25	>=2016.2,<2017.1	Best Effort	4.15.5F	4.22xM
Mitaka	2017-04-10 (EOL)	>=2016.1,<2016.2	Best Effort	4.15.5F	4.22xM
Liberty	2016-11-17 (EOL)	>=2015.2,<2016.1	Best Effort	4.15.5F	4.22xM
Kilo	2016-05-02 (EOL)	>=2015.1,<2015.2	Best Effort	4.15.5F	4.22xM
Juno	2015-12-07 (EOL)	N/A	Best Effort	4.15.5F	4.22xM
Icehouse	2015-07-02 (EOL)	N/A	Best Effort	4.15.5F	4.22xM
Havana	2014-09-30 (EOL)	N/A	Best Effort	4.15.5F	4.22xM

Table 2: Features and corresponding minimum EOS & OpenStack release versions		
Feature	**Arista EOS Release**	**OpenStack Release**
Arista ML2 mechanism driver support for automatic L2 provisioning	EOS-4.15.5F (or newer)	Havana (or later)
Automatic VLAN to VNI mapping	EOS-4.15.5F (or newer)	Havana (or later)
Arista L3 plugin support	EOS-4.15.5F (or newer)	Icehouse (or later)
CVX HA support	EOS-4.15.5F (or newer)	Liberty (or later)
Arista VLAN type driver support	EOS-4.18.1F	Mitaka (or later)
Neutron Distributed Virtual Router	EOS-4.18.1F	Mitaka (or later)
Neutron ML2 Hierarchical Port Binding	EOS-4.18.1F	Newton (or later)
Bare Metal Provisioning (Ironic)	EOS-4.18.1F	Newton (or later)
Security groups for bare metal	EOS-4.18.1F	Newton (or later)
VLAN Aware VMs (Trunk Ports)	EOS-4.18.1F	Pike (or later)
Baremetal Trunk Ports	EOS-4.21.0F	Queens (or later)
MLAG + VRFs with L3 Plugin	EOS-4.21.0F	Queens (or later)
Neutron HA & Legacy Router Support	EOS-4.21.0F	Queens (or later)
EVPN VxLAN Control Plane Support	EOS-4.23.2F	Havana (or later)
VRF Default Route Support	EOS-4.21.0F	Queens (or later)

Note: Arista switches and Arista CVX run Arista EOS. Except during upgrades, it is required to keep both Arista switches and CVX at the same version level.

Support for OpenStack Releases

Please contact Arista TAC for any issues with Arista equipment or drivers. [2]

Support for End-of-Life (EOL) designated OpenStack releases

Arista provides support for the integration with all OpenStack releases currently supported by the upstream OpenStack project. As code patches do not merge upstream for fixes in a release past it's EOL date, all support for releases marked as EOL are supported best effort and might require manual patch application. Additionally, there will be no new feature additions for releases that are marked EOL.

EOS version required for new OpenStack features

For using features released with a specific OpenStack release, the Arista EOS release must be a version greater than or equal to the minimum corresponding Arista EOS release specified in Table 1 above.

[2]https://www.arista.com/en/support/customer-support

EOS version required for new OpenStack features

OpenStack features supported in a particular Arista EOS release should continue to work with newer OpenStack releases without requiring upgrading the EOS version for that feature set. Once OpenStack is deployed with a version of EOS, upgrades to OpenStack without enabling new features should not require an EOS upgrade. However, in some extreme scenarios, this condition might not be true. Please review the networking-arista release notes and this document for compatibility information.

Deploying the OpenStack master branch

As OpenStack is constantly evolving, deploying and supporting the OpenStack master branch presents a unique set of challenges. With constant code churn, there often are small windows where OpenStack Neutron code is refactored or modified in a way that could render the Arista networking driver incompatible. Caution should be exercised when deploying the OpenStack master branch.

Architecture

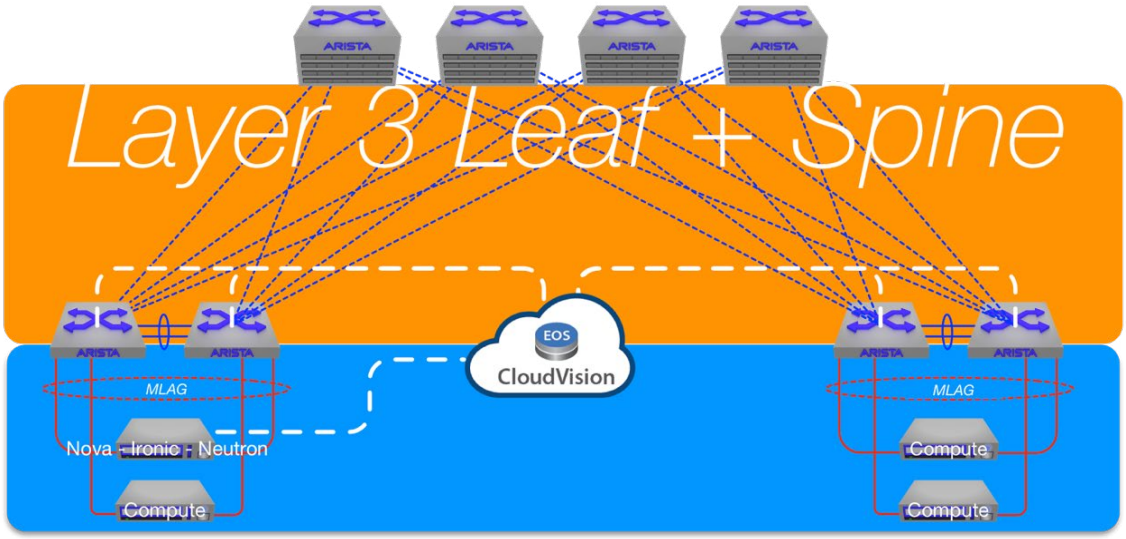

Figure 1: Common OpenStack deployment

Arista CloudVision eXchange (CVX) is an instance or highly available cluster of virtual Arista EOS that communicates with every Arista switch in the deployment. Using Link Layer Discovery Protocol (LLDP) information available from each switch that CVX manages, CVX builds a view of the entire data center topology.

OpenStack Neutron, running the Modular Layer 2 (ML2) plugin, communicates with CVX through the Arista Mechanism Driver in order to share information on activities performed in OpenStack.

In the case of virtual machines, port binding information in OpenStack contains information about the hypervisor or compute node the VM is launched on. For bare metal instances, OpenStack Ironic provides switch identification and switch interface information to OpenStack Neutron which in turn provides this to CVX.

In response, the OpenStack agent running on CVX appropriately provisions top of rack network switches so that bare metal or virtual machines hosted on compute hosts have connectivity over the tenant network by identifying the switch and switch port the compute host is connected to.

Similar to the L2 provisioning above, the OpenStack Neutron api-server handles requests to create or delete logical routers and handles requests to enable or disable routing between different subnets. When configured to run with the Arista L3 plugin, the API calls result in Switch Virtual Interfaces (or SVIs) being created on the Arista switches.

The Arista ML2 driver and the L3 Service Plugin may be deployed independently or together. The decision to deploy one or both comes down to the functionality required in the OpenStack deployment.

Features

Arista integration with OpenStack offers the following features.

Automatic L2 provisioning

Using the Arista ML2 mechanism driver, layer 2 (L2) provisioning, such as creating and dynamically mapping VLANs to physical switch interfaces can be automated.

Automatic VLAN to VNI mapping

Using the automatic VLAN to VNI mapping feature, a VXLAN VNI (Virtual Network Identifier) will be automatically mapped to a VLAN on Arista hardware once it is created by the Arista ML2 mechanism driver.

Arista L3 plugin

The Arista L3 plugin allows for the automatic creation of SVIs (Switch Virtual Interfaces) which allow for routing to occur at the leaf or spine switch, instead of in software on an OpenStack host.

Arista VLAN type driver

As OpenStack logical networks are created, OpenStack Neutron allocates identifiers from an available range of configured VLAN IDs. An alternate to that approach is to use the Arista VLAN type driver which allows a network operator to assign a range of VLANs to OpenStack from CVX.

Apart from bringing visibility and control of how the VLAN space is segmented to a single entity, this has the added benefit of allowing the range of identifiers allocated to OpenStack to be modified without requiring an OpenStack Neutron service restart.

OpenStack Neutron Distributed Virtual Router

OpenStack Neutron Distributed Virtual Routers (DVR) route traffic into the destination network at the virtual switch on the hypervisor. The Arista solution tracks the creation of ports for the distributed router and provisions TOR switches to ensure that the destination network is provisioned (in addition to the network the instance is connected to).

OpenStack Neutron ML2 Hierarchical Port Binding

ML2 Hierarchical Port Binding (HPB) shares all the benefits of the automatic VLAN-to-VNI mapping solution presented above, but is different in a few key ways:

- Instead of Arista CVX allocating VNIs for a network transparently, OpenStack Neutron is aware of the fact that there is a L3 fabric in use and allocates the VNI for each logical network.
- While the L2 agent configuring the virtual switch on the hypervisor still configures it to send out 802.1Q tagged traffic, CVX configures the Arista TOR to map the VLAN into the VNI provided by Neutron.
- It is possible to scale beyond 4096 tenant networks since each TOR switch can map 4096 VLANs to different VNIs.
- OpenStack Neutron logically groups the TOR and all hypervisors connected to it into a physical network, representing the set of devices where a VLAN identifier represents the same logical network. Traffic headed north of this TOR is encapsulated and identified by the VNI associated with the logical network.

Bare Metal Provisioning (OpenStack Ironic)

Through OpenStack Ironic integration with Neutron, it is possible to provision bare metal servers that are attached to Arista switches and connect them to tenant networks. All of the features that Arista supports for provisioning networks for VMs is extended to bare metal servers. This includes automatic VLAN-to-VNI mapping and Hierarchical Port Binding.

Security groups for OpenStack Ironic

Security groups can be applied as ACLs on switch interfaces connected to bare metal servers.

Bare Metal Trunk Ports

Additional VLANs (beyond the initial native VLAN) can be provisioned on bare metal connected interfaces through the use of Neutron trunk ports and subports.

Neutron HA & Legacy Router Support

VLANs required for Neutron HA and legacy routers are automatically provisioned on TORs connected to networking nodes. This primarily addresses the case where HPB is deployed alongside Neutron HA routers as the necessary VLANs cannot be predicted and therefore cannot be manually configured.

EVPN VxLAN Control Plane Support

VLANs and VLAN to VNI mappings required for OpenStack are automatically provisioned through CVX, but host reachability information is distributed via BGP EVPN control plane rather than the Vxlan Control Service (VCS).

Design Considerations

OpenStack offers a number of design choices to the user. Based on the design choices made, one or more subsequent sections in the document might not be applicable for the deployment.

Please note:

- Each solution below has pros and cons and its suitability depends on the required deployment.
- This document focuses on the OpenStack features that require the Arista ML2 driver or L3 plugin.

Tenant Network Types

The section below outlines the differences between deploying L2 VLAN or VXLAN and L3 topologies.

VLAN or VXLAN?

There are different L2 technologies available to isolate tenant traffic from one another in a multi-tenant cloud deployment. The two relevant OpenStack deployment technologies supported by Arista are VLAN and VXLAN.

VLAN isolation is generally well understood and deployments, however they are limited to less than 4096 tenant networks after allocating VLANs to management, storage traffic, etc.

Two compute instances hosted on different compute nodes that are part of the same tenant network will require the VLAN be stretched across the DC fabric, and with a sufficiently large set of hypervisors this will result in a large L2 domain. Beyond scalability challenges, a large L2 domain has inherent limitations, such as requiring spanning tree and very large broadcast domains. For smaller deployments, a VLAN deployment will be the easiest to implement.

For larger environments, a VXLAN deployment may be more appropriate. Arista and VMware co-authored the Virtual eXtensible LAN (VXLAN) protocol to provide virtual overlay networks and extend the cost efficiency gains of virtualized servers in the data center. VXLAN encapsulates network traffic of virtualized workloads into standard IP packets. As a result, multiple VXLAN virtual networks can run over the same physical infrastructure to reduce capital expenditure (CAPEX). VXLAN runs over any standard IP network and benefits from the scaling, performance and reliability characteristics available in current Layer 3 IP data center networks. Standards based IP underlays are open and allow for best of breed, cost efficient, multi-vendor DC environments. IP underlay networks are also more reliable, reducing costly network outages in a VXLAN data center infrastructure. VXLAN doesn't rely on dated spanning tree protocols, TRILL fabrics or 802.1Q VLAN tagging mechanisms that offer limited reliability, fault isolation and scaling. Using Hierarchical Port Binding (HPB) with OpenStack, VXLAN also allows administrators to scale up to 16.7 million unique layer 2 networks in the data center.

VXLAN deployment options

There are three (3) VXLAN deployment options available; software only, hardware only, and software and hardware.

Software VTEPs - In this model, the hypervisor acts as a tunnel endpoint (VTEP) and a 3rd party controller (open-source or proprietary) or the L2 population driver from Neutron is used to exchange information between VTEPs that have endpoints connected to the same virtual network identified by a unique Virtual Network Identifier (VNI).

In this scenario, the Arista equipment acts as an underlay or pure IP forwarder and needs no additional integration with Neutron or OpenStack.

Hardware VTEPs - a second option is to use only Arista switches as a VTEPs. In this model, traffic exiting the hypervisor is VLAN tagged and is mapped into a VNI at the top of rack (TOR) switch. This gives operators the benefit of keeping the hypervisor and virtual switch deployment simple and increasing performance while receiving all the benefits of a L3 fabric.

In order to go beyond the 4096 VLAN tag limit per network due to the 802.1Q protocol reserving 12 bits for the VLAN identifier, it is possible to use hierarchical port binding in OpenStack Neutron which allows VLANs to be mapped into different VNIs on each of the TORs. This allows for creating more than 4096 tenant networks across the deployment while limiting the number of unique networks seen on any given TOR switch to 4096.

VXLAN Control Plane Options

Vxlan Controller Service - Host reachability information is published to CVX and CVX distributes reachability information and flood lists to all connected VTEPs

BGP EVPN - Host reachability information is learned and distributed via BGP. Using a standard-based BGP EVPN control plane allows for third party VTEPs to participate in the EVPN control plane.

See https://eos.arista.com/summary-of-arista-vxlan-control-plane-options/ for a more detailed comparison of control plane options.

L3 Considerations

Logical routers can be used to route between tenant networks. These can be implemented as software routers or on hardware switches with routing enabled and SVIs (Switch Virtual Interfaces) configured.

Software routing options

For software routing, an OpenStack Neutron Highly Available (HA) router is deployed, either on the Neutron node or elsewhere where traffic between tenant networks is required. In this model, the Arista switches require manual and static configuration of the switch port connected to the Neutron network node(s) to ensure that all permissible tenant VLANs are carried on that port.

Alternatively, the user can choose to deploy Neutron's Distributed Virtual Router (DVR) where all routing occurs within the hypervisor, closest to the source of the traffic. This has the advantage of ensuring there is no centralized software router node that traffic must traverse for east-west traffic and is more efficient. However, the solution requires that the virtual switch in the hypervisor be programmed with additional information, which makes it more complex to configure and operate.

In deployments that use an external controller for network configuration, routing is typically accomplished in software, with some controllers requiring specific software perform the routing functionality. The controller is responsible for provisioning the software router in response to OpenStack Neutron API requests from the user.

Hardware routing options

The Arista L3 plugin can be used to configure SVIs on a selected set of hardware switches. This provides the benefit of keeping routing functionality within hardware, which allows for greatly improved performance. For a multi-tenant deployment with overlapping IP addresses, this solution permits creating a virtual routing and forwarding (VRF) instance for each logical router created, but is limited by the number of VRFs supported on the hardware platform. In the Neutron, the software L3 agents provide metadata services, so the "force_metadata" option must to be set to True in the Neutron DHCP agent configuration file to ensure instances have metadata service.

Network Address Translation (NAT)

OpenStack Neutron can perform NAT within the software router. If hardware NAT is a requirement, it is possible to have a separate hardware device handling address translation as traffic enters or exits the OpenStack deployment.

Arista CVX Clustering

While Arista CVX is not involved in the path of data traffic, it is an integral component for auto-provisioning network resources. In the event that the CVX instance restarts or is unavailable, the ability to provision new resources such as networks and instances is lost while CVX remains unavailable.

Running multiple instances of CVX in a highly available cluster reduces this time window significantly. While running multiple CVX instances in a cluster, a single instance is the active node and the Arista drivers communicate with it. If the active instance fails or is restarted, a warm standby instance becomes the new active and the Arista drivers detect the switch over and synchronize state with the new CVX, restoring functionality.

It is highly recommended that a CVX cluster be created with a minimum of three (3) instances.

More information on configuring CVX HA can be found in Appendix A: References.

Scale

While there are multiple metrics that are of interest when it comes to scale, this document focuses on scaling up the number of tenant networks. Scale on other metrics is typically restricted by other factors such as OpenStack performance or the scaling restrictions of a third-party controller and are out of scope for this document.

Previous sections of this document have already touched upon scaling up the number of tenant networks in a deployment by choosing VXLAN over VLANs as the underlying L2 technology and presents the options of using software or hardware VTEPs or a mix of software and hardware.

Another scalability option available in some environments is creating multiple OpenStack deployments (or regions) on the same physical infrastructure. Arista CVX supports multiple OpenStack regions with each region mapped to a separate set of TOR switches. This allows scaling out by creating multiple smaller OpenStack regions. In such a model, an end user needs to know what region they need to create workloads in and all communication between workloads in different regions has to be performed external to the OpenStack environment.

Note - it is possible to have a deployment with OpenStack regions that share TORs as well. However, in that scenario, please ensure that the regions have non-overlapping VLAN/ VNI ranges and that the regions don't have security groups enforcement offloaded to the TOR (via the plugin as security groups from both regions can interfere with one another).

Recommendations

While Arista supports all of the different deployment types described above, most deployments tend to favor using an L3 fabric with VXLAN in order to get the performance, reliability and scaling benefits mentioned above.

A vanilla OpenStack Neutron deployment, with hardware VTEPs removes the need for any additional controller and reduces operational (day-2) complexity as well. This solution scales up with the use of hierarchical port binding or scales out by repeating the pattern across multiple OpenStack deployments.

Finally, the solution is rounded out by using software routing, though in the future software routing can be replaced with routing east-west traffic at the top of rack and moving north-south traffic through additional software or hardware components.

Installing the networking-arista package

This step is not required on RHOSP, but is for all other deployments where the Arista ML2 drivers or L3 plugin are used.

Before beginning

- The steps below assume a functional OpenStack deployment.
- Please make accommodations for file paths, commands etc. as necessary based on:
  - Linux distribution used
  - OpenStack distribution
- The steps listed below are common to all supported releases.

Installing Arista Networking Drivers

Beginning with the Kilo release of OpenStack, in order for OpenStack Neutron to load any Arista drivers, it is necessary that the networking-arista package be installed. It contains both the Arista ML2 Type and Mechanism drivers and the Arista L3 plugin. This package needs to be installed on all nodes where the OpenStack Neutron server is installed.

The networking-arista package can be installed by executing the following command:

```
sudo pip install networking-arista
```

The above command installs the latest version of networking-arista from the master branch. To install drivers for a specific release, run the command corresponding to the release:

```
Xena     sudo pip install -c https://releases.openstack.org/constraints/upper/xena "networking-
arista>=2021.2,<2022.1"
Wallaby  sudo pip install -c https://releases.openstack.org/constraints/upper/wallaby "networking-
arista>=2021.1,<2021.2"
Victoria  sudo pip install -c https://releases.openstack.org/constraints/upper/victoria "networking-
arista>=2020.2,<2021.1"
Ussuri  sudo pip install -c https://releases.openstack.org/constraints/upper/ussuri "networking-
arista>=2020.1,<2020.2"
Train  sudo pip install -c https://releases.openstack.org/constraints/upper/train "networking-
arista>=2019.2,<2020.1"
Stein  sudo pip install -c https://releases.openstack.org/constraints/upper/stein "networking-
arista>=2019.1,<2019.2"
Rocky  sudo pip install -c https://releases.openstack.org/constraints/upper/rocky "networking-
arista>=2018.2,<2019.1"
Queens  sudo pip install -c https://releases.openstack.org/constraints/upper/queens "networking-
arista>=2018.1,<2018.2"
Pike  sudo pip install -c https://releases.openstack.org/constraints/upper/pike "networking-
arista>=2017.2,<2018.1"
Ocata  sudo pip install -c https://releases.openstack.org/constraints/upper/ocata "networking-
arista>=2017.1,<2017.2"
Newton  sudo pip install -c https://releases.openstack.org/constraints/upper/newton "networking-
arista>=2016.2,<2017.1"
Mitaka  sudo pip install -c https://releases.openstack.org/constraints/upper/mitaka "networking-
arista>=2016.1,<2016.2"
Liberty  sudo pip install -c https://releases.openstack.org/constraints/upper/liberty "networking-
arista>=2015.2,<2016.1"
Kilo  sudo pip install -c https://releases.openstack.org/constraints/upper/kilo "networking-
arista>=2015.1.5,<2015.2"
```

Note: The networking-arista module must be accessible to the user that will be running the neutron-server process. On systems running neutron-server under systemd, the neutron user can be determined with systemctl show <neutron service> | grep ^User. Access should be confirmed by running pip list as that user and ensuring networking-arista appears in the output.

Note: For Train and later releases, Neutron may be running with python3. If that is the case, pip3 should be used to install networking-arista

Note: Using pip to install networking-arista ensures that the version deployed is the latest supported version for the release and is not a version bundled along with the OpenStack distribution at a particular point in time.

Offline Installation

In some cases, internet access may not be available on the Neutron controller node. Follow these instructions in order to install networking-arista on a machine without internet access.

On a machine with internet access:

1. Download the upper-constraints.txt file for your release from here: https://opendev.org/openstack/requirements/ (Make sure you select the appropriate release branch)
2. Create a directory to store networking-arista and all of its dependencies "`mkdir networking-arista-deps`"
3. Download networking-arista and all of its dependencies with "`sudo pip download networking-arista==<see version table above> -d ./networking-arista-deps -c <upper-constraints.txt file from step 1>`"
4. Create a tarball with networking-arista and its dependencies "`tar -cvzf networking-arista-deps.tar.gz networking-arista-deps`"

Then, copy networking-arista-deps.tar.gz to the Neutron controller.

On the Neutron controller:

1. Unzip the tarball "`tar -xzvf networking-arista-deps.tar.gz && cd networking-arista-deps`"
2. Install networking-arista along with all of its dependencies from the current directory "`sudo pip install -U networking_arista-<version>.whl -f ./ --no-index`"

Configuring L2 automatic provisioning

The section below outlines installation steps for deployments where automatic layer 2 provisioning of top of rack Arista switch ports is required. This includes both VLAN and hardware VXLAN backed tenant networks.

Summary of steps

1. Prerequisites: This step includes connecting compute hosts to Arista Top of Rack (ToR) switches and configuring the underlay for either VLAN or VXLAN backed networks.
2. Arista TOR Switch configuration: This step needs to be performed on any Arista switch that will participate in automated OpenStack provisioning.
3. Arista Cloud Vision eXchange (CVX) Setup: This step ensures that CVX can provision Arista switches.
4. Configuring OpenStack Neutron to run the Arista driver: This step ensures that OpenStack Neutron loads the Arista mechanism driver on startup.
5. Verification of end-to-end operation: This step ensures that all services are running properly and tenants can create or delete networks and compute instances (VMs). This step also ensures that correct network (VLAN) is provisioned on the appropriate interfaces of Arista Switches.

Step 1 - Prerequisites

The physical network must be properly configured prior to configuring OpenStack. The steps below include both VLAN and VXLAN configuration. While both VLAN and VXLAN options share some configuration on the TOR switches, there are significant differences. For design considerations between the two options, please see VLAN or VXLAN?

Configuring TOR and Spine links for a VLAN backed L2 fabric

For a VLAN backed deployment, ensure that the inter-switch links are configured as trunk interfaces and that all required VLANs are allowed. Either allow all VLANs or prune them to the range that is provided to OpenStack.

```
TOR1(config)#interface Ethernet 32
TOR1(config-if-Et32)#switchport mode trunk
TOR1(config-if-Et32)#switchport trunk allowed vlan all
```

Configuring the network for VXLAN backed fabric

Options for configuring automatic VLAN to VNI mapping, Hierarchical Port Binding and the Arista VLAN type driver are covered in the section Optional L2 configuration.

Configuring switch ports connecting TOR to compute nodes

Interfaces that are expected to carry VM traffic must be placed in VLAN trunk mode. In the event that the operator needs these interfaces to carry VLANs for management or storage traffic, those can be configured statically with the allowed VLAN list matching the required VLANs. All other VLANs including those configured in OpenStack Neutron should not be in the allowed VLAN list and will be automatically provisioned.

In the following example, the first 10 interfaces are to be automatically provisioned by OpenStack and VLAN 11 is configured on these interfaces for management traffic. OpenStack will dynamically add the required VLANs.

```
TOR1(config)#interface Ethernet 1-10
TOR1(config-if-Et1-10)#switchport mode trunk
TOR1(config-if-Et1-10)#switchport trunk allowed vlan 11
TOR1(config-if-Et1-10)#spanning-tree portfast edge
```

In the following example, interfaces 11-20 are configured for attachment to compute nodes and aren't configured to carry VLAN 11. OpenStack will dynamically add the required VLANs. Note that these selected interfaces are directly connected to the hypervisor nodes or bare metal servers.

```
TOR1(config)#interface Ethernet 11-20
TOR1(config-if-Et11-20)#switchport mode trunk
TOR1(config-if-Et11-20)#switchport trunk allowed vlan none
TOR1(config-if-Et11-20)#spanning-tree portfast edge
```

Note: All interfaces attached to a compute node must be configured in trunk mode.

Note: On the hypervisor, interfaces that connect to the TOR need to be added as physical ports to the Open vSwitch (OVS) bridges to ensure that the VM traffic gets forwarded through these interfaces. See an example of this configuration in section VM's are unable to communicate.

Host configuration

The Arista solution requires an LLDP daemon to be run on each OpenStack compute hypervisor. The following example shows the steps required for an Ubuntu 14.04 distribution. If it is a different distribution, please use the appropriate command to accomplish this step.

Ubuntu/Debian:

```
sudo apt-get install lldpd
sudo service lldpd start
```

RHEL/CentOS:

```
sudo yum install lldpad
sudo service lldpad start
sudo lldptool set-lldp -i $INTERFACE adminStatus=rxtx
```

Open vSwitch:

On systems with OVS, it can be used to transmit LLDP information when deployment constraints do not allow for a traditional lldp daemon.

```
sudo ovs-vsctl set interface $INTERFACE lldp:enable=true
# Create dummy AutoAttach record to set hostname for external bridge
sudo ovs-vsctl add-aa-mapping $BRIDGE 30 0
sudo ovs-vsctl set AutoAttach $BRIDGE system_name=$HOSTNAME
```

If the deployment is a different distribution, please use the appropriate command to accomplish this step.

Step 2 - Arista TOR Switch Configuration

The following configuration is necessary on each TOR switch connected to OpenStack compute nodes.

Configure the switch to communicate with CVX

The switch must be configured to communicate with all CVX instances.

```
TOR1(config)#management cvx
TOR1(config-mgmt-cvx)#server host 192.168.6.248
TOR1(config-mgmt-cvx)#no shutdown
```

The IP address above is the management IP address of the CVX controller.

Note: For CVX clusters, please repeat the "server host <IP>" configuration for each instance.

Verify LLDP is running on the TOR switch

The Arista solution requires LLDP to be run on each TOR switch. Though Arista switches run LLDP by default, the following example displays the output seen if LLDP has been manually disabled.

```
TOR1#show lldp
% LLDP is not enabled
```

Enable LLDP on the TOR switch

If LLDP is not running on the TOR switch, it can be enabled with the following command.

```
TOR1(config)#lldp run
```

For VXLAN backed networks

Options for configuring automatic VLAN to VNI mapping, Hierarchical Port Binding and the Arista VLAN type driver are covered in the section Optional L2 configuration.

Verification

Please run the verification steps in Switch Topology Verification to ensure that all the neighbors of a given switch are visible.

Step 3 - Arista CVX Setup

The section below assumes Arista CVX is already running in the environment, for details on installing CVX please see the CloudVision configuration guide.

Note: All CVX instances should have identical configuration (barring hostname, IP address). Repeat the following configuration on all CVX instances

Enable the CVX service on CVX instances

```
CVX1(config)#cvx
CVX1(config-cvx)#no shutdown
```

CVX1(config)#cvx
CVX1(config-cvx)#no shutdown

```
CVX1(config)#cvx
CVX1(config-cvx)#service openstack
CVX1(config-cvx-openstack)#no shutdown
```

Configure CVX to query OpenStack
This step is optional and enables CVX to query OpenStack for VM and Tenant names from OpenStack, granting the network operations staff visibility into OpenStack from Arista switches.

Note: The region name specified above must match that configured in the Arista mechanism driver configuration in Step 4 below.

```
CVX1(config-cvx)#cvx
CVX1(config-cvx)#service openstack
CVX1(config-cvx-openstack)#region <region-name>

CVX1-openstack-<region-name>)#username <admin or valid keystone user>
tenant <tenant-name> password <password for user>

CVX1-openstack-<region-name>)#keystone auth-url <URL of keystone
authentication server>  port  <authentication port number> version
<authentication API version>
```

Note 2: The Keystone service catalog must contain internal endpoints for Neutron and Nova. This limitation will be addressed in a future release.

Note for OpenStack Pike and newer releases: Beginning in the OpenStack Pike release, the community updated it's recommended deployment guide to make Keystone available from an HTTP service running on the traditional HTTPS (443) port or insecurely on HTTP (80) port. Versions of CVX prior to 4.21.3F only support Keystone at the server root '/' and not on a unified OpenStack REST endpoint using a subpath such as '/identity'. The workaround for older CVX instances is to configure the server endpoint to additionally make Keystone service available at the server root on an alternate port such as 5000. Releases of CVX after 4.21.3F will not require this workaround.
Example Devstack Apache2 configuration for port 5000 proxy to the Keystone uwsgi service:

```
Listen 5000
<VirtualHost *:5000>
    ProxyPass "/" "unix:/var/run/uwsgi/keystone-wsgi-public.socket|uwsgi://
uwsgi-uds-keystone-wsgi-public/" retry=0
</VirtualHost>
```

Enable eAPI on CVX

The Arista ML2 and L3 drivers utilize the Arista API (eAPI) to communicate with CVX.

```
CVX1(config)#management api http-commands
CVX1(config-mgmt-api-http-cmds)#no shutdown CVX1(config-mgmt-api-http-cmds)#no
protocol http CVX1(config-mgmt-api-http-cmds)#protocol https
```

Note: OpenStack integration through eAPi is not supported with an enable password set on CVX.

DNS

Arista CVX requires functioning DNS servers and DNS entries for each TOR switch in the Arista network. If DNS is not in place, static DNS entries may be added on all CVX instances.

```
CVX1(config)#ip host TOR1 192.168.6.249
```

For VXLAN backed networks

Options for configuring automatic VLAN to VNI mapping, Hierarchical Port Binding and the Arista VLAN type driver are covered in the section Optional L2 configuration.

Verification

It is advisable to perform verification steps to ensure that:

1. The physical topology is visible in the CVX. Execute the steps described in Topology Verification to ensure that all the hosts and network connectivity is displayed as expected.
2. CVX is reachable from HTTPS port (TCP/443). The Arista driver uses this port to connect with CVX. If CVX is not reachable from this port, the Arista driver will fail causing OpenStack Neutron to exit. Execute steps described in CVX Reachability Verification for more details.

Step 4 - Configure OpenStack Neutron

For RHOSP deployments, please skip ahead to Step 4 (RHOSP) - Configure OpenStack Neutron.

This section describes how to enable the Arista Mechanism Driver within OpenStack Neutron.

Prior to this step, the following steps should be completed:

- OpenStack has been installed
- networking-arista package installed on all the nodes running the OpenStack Neutron server

OpenStack Neutron Configuration

This step configures OpenStack Neutron to run the Arista ML2 plugin.

Verify that OpenStack Neutron is setup to use the ML2 plugin in the OpenStack Neutron configuration by ensuring that the `core_plugin` is set to `neutron.plugins.ml2.plugin.Ml2Plugin`

```
admin@openstack:~$ grep "core_plugin" /etc/neutron/neutron.conf
core_plugin = neutron.plugins.ml2.plugin.Ml2Plugin
```

Optionally, ensure that OpenStack Neutron has OpenStack Keystone authentication parameters configured to allow for the Arista ML2 driver to share information with CVX as CVX polls OpenStack Keystone to query tenant and VM names. If OpenStack Neutron is not configured to use OpenStack Keystone authentication information, provide the OpenStack Keystone authentication parameters via the CVX command line or Tenant and VM names will not be resolved.

Note: It is highly recommended that Keystone v3 be used rather than Keystone v2.0

Search for `[keystone_authtoken]` in the OpenStack Neutron configuration file and add the following line after it:

```
auth_uri = <keystone service endpoint eg. http://ip-addr:35357/identity >
```

```
auth_protocol = <http or https>
auth_host = <keystone service endpoint IP address or DNS>
auth_port = <port that the service is reachable on; usually 5000 or 35357>
```

Or if running a release prior to OpenStack Mitaka, add the following three (3) lines after it:

Note: Please provide the OpenStack Keystone endpoint IP address. Execute the "keystone endpoint-list" command on the OpenStack controller to get this information and ensure that the address is reachable from CVX.

Neutron ML2 Plugin Configuration

By default, the OpenStack Neutron ML2 plugin does not start any vendor specific mechanism drivers. Edit the `/etc/neutron/plugins/ml2/ml2_conf.ini` file and append '`arista`' to the list of mechanism drivers in order to register the Arista driver.

```
admin@openstack:~$ grep "arista" /etc/neutron/plugins/ml2/ml2_conf.ini
mechanism_drivers = openvswitch,arista
```

Note: Do not remove any mechanism drivers already listed, only add `arista` as an additional driver.

Additionally, in the same file ensure that the network type is set to `vlan` and the range of VLANs reflects the VLANs available for the OpenStack deployment.

```
tenant_network_types = vlan

[ml2_type_vlan]
network_vlan_ranges=default:1:100
```

Arista ML2 Driver Configuration

The Arista mechanism driver provides several configuration knobs to help optimize communication between the mechanism driver and CVX based on the deployment.

Configuration options are divided into two parts; mandatory configuration and optional configuration.

Mandatory Configuration

Edit `/etc/neutron/plugins/ml2/ml2_conf_arista.ini` as follows:

```
[ml2_arista]
eapi_host=set a comma separated list of IP addresses for each CVX instance
eapi_username=<user name>
eapi_password=<password for above user>
```

Note: If CVX has been deployed in a highly available (HA) cluster, specify each instance IP separated by a comma.

If the `ml2_conf_arista.ini` file is not present in the `/etc/neutron/plugins/ml2` directory, copy it from `etc/ml2_config_arista.ini` under the networking-arista installation directory or from https://github.com/openstack/networking-arista/blob/master/etc/ml2_conf_arista.ini.

Optional Configuration

The following optional configuration elements can be configured in `/etc/neutron/plugins/ml2/ml2_conf_arista.ini`.

Configuration Element	Default	Description
`use_fqdn`	True	Use this option to ensure that the hypervisor names used within OpenStack matches those learned on CVX through LLDP information from switches.
`sync_interval`	30 Seconds	The mechanism driver periodically checks to see if Neutron and CVX are in synchronization. Use this option to set how frequently sync will occur. Note: Older arista_networking releases had a default `sync_interval` of 180 seconds.
`region_name`	RegionOne	Unique region name of the OpenStack deployment. Note: A single Arista CVX (cluster) may manage one or more OpenStack deployments. The region name must be globally unique to represent each OpenStack controller cluster. This name must also match the region name used when configuring the OpenStack Keystone service used by the OpenStack controller. CVX authenticates this information with the OpenStack Keystone service.

Restart OpenStack Neutron

Please restart the OpenStack Neutron server on all nodes it is running on after completing the configuration steps above and passing the `ml2_conf_arista.ini` file as a `--config-file` parameter to OpenStack Neutron.

Consult the relevant distribution's documentation to determine how to pass additional configuration files to OpenStack Neutron.

Step 4 (RHOSP) - Configure OpenStack Neutron

First, acquire and configure the appropriate Arista OpenStack Neutron container image for your RHOSP release:

```
docker login registry.connect.redhat.com
docker pull registry.connect.redhat.com/arista/openstack<RHOSP release>
docker tag ...
docker push ...
```

A full list of Arista OpenStack Neutron containers is available here: https://catalog.redhat.com/software/containers/search?application_categories_list=Networking&p=1&vendor_name=ARISTA%20NETWORKS

Next, configure the above container as the NeutronApi container in /home/stack/templates/overcloud-images.yaml on the RHOSP director:

```
DockerNeutronApiImage: <director provisioning IP>:8787/arista/openstack13:latest
```

Next create an Arista ML2 configuration template, for example /home/stack/templates/arista_ml2_config.yaml:

```
heat_template_version: <version>

description:
  Arista Ml2 plugin service enablement

parameters:
  servers:
    type: json
  ComputeHostnameFormat:
    type: string
    default: "”
  CVXUsername:
    description: CVX host Username.
    type: string
    default: ‘’
  KeystoneRegion:
    description: KeystoneRegion.
    type: string
    default: ‘’
  CVXPassword:
    type: string
    description: CVX host Password.
    hidden: true
    default: ‘’
  CVXHost:
```

```
    description: List of IP address of all CVX switches.
    type: string
  MechanismDriversAristaService:
    description: List of enabled ML2 pluginsEnable arista service
    type: string
  TypeDrivers:
    description: List of enabled ML2 type drivers
    type: string
  PhysNet:
    description: OpenStack Managed Physical nic's
    type: string
    default: ''
  CVXSyncInterval:
    description: Interval for CVX out-of-sync checks
    type: number
    default: 30
  CVXConnectionTimeout:
    description: Maximum CVX API request length
    type: number
    default: 60
  UseFQDN:
    description: Send FQDN hostnames to CVX
    type: boolean
    default: true
  UseFQDNPhysnet:
    description: Use FQDN switch names in topology lookups for HPB
    type: boolean
    default: true
  IronicSecurityGroupsEnabled:
    description: Enable translation of security groups to ACLs
    type: boolean
    default: false
  IronicSwitchInfo:
    description: eAPI login details for baremetal TORs
    type: comma_delimited_list
    default: ''
  ManageFabric:
    description: Bind VXLAN fabric segments in HPB deployments
    type: boolean
    default: false
  ManagedPhysnets:
    description: Physical networks managed by the Arista ML2 plugin
    type: comma_delimited_list
    default: ''

resources:
  ExtraDeployments:
    type: OS::Heat::StructuredDeployments
    properties:
```

```
      servers:  {get_param: servers}
      config: {get_resource: ExtraConfig}
      # Do this on CREATE/UPDATE (which is actually the default)
      actions: ['CREATE', 'UPDATE']
      input_values:
        deploy_username: {get_param: CVXUsername}
        deploy_password: {get_param: CVXPassword}
        deploy_cvxip: {get_param: CVXHost}
        deploy_kregion: {get_param: KeystoneRegion}
        deploy_ovsphysnet: {get_param: PhysNet}
        deploy_mech_drivers: {get_param: MechanismDriversAristaService}
        deploy_type_drivers: {get_param: TypeDrivers}
        deploy_sync_interval: {get_param: CVXSyncInterval}
        deploy_conn_timeout: {get_param: CVXConnectionTimeout}
        deploy_use_fqdn: {get_param: UseFQDN}
        deploy_use_fqdn_physnet: {get_param: UseFQDNPhysnet}
        deploy_sec_group_support: {get_param: IronicSecurityGroupsEnabled}
        deploy_switch_info: {get_param: IronicSwitchInfo}
        deploy_manage_fabric: {get_param: ManageFabric}
        deploy_managed_physnets: {get_param: ManagedPhysnets}


  ExtraConfig:
    type: OS::Heat::SoftwareConfig
    properties:
      group: script
      inputs:
        - name: deploy_username
        - name: deploy_password
        - name: deploy_cvxip
        - name: deploy_kregion
        - name: deploy_ovsphysnic
        - name: deploy_mech_driversarista
        - name: deploy_type_drivers
        - name: deploy_sync_interval
        - name: deploy_conn_timeout
        - name: deploy_use_fqdn
        - name: deploy_use_fqdn_physnet
        - name: deploy_sec_group_support
        - name: deploy_switch_info
        - name: deploy_manage_fabric
        - name: deploy_managed_physnets
      config: |
            #!/bin/bash
            set -x
            function arista_service () {
              sudo echo "Arista INI file has been configured and ML2 Neutron API
service restarted" >> /root/ml2_automation_logs.txt
              sudo echo [ml2_arista] >> /var/lib/config-data/puppet-generated/neutron/
etc/neutron/plugins/ml2/ml2_conf_arista.ini
              sudo chmod 777 /var/lib/config-data/puppet-generated/neutron/etc/
neutron/plugins/ml2/ml2_conf_arista.ini
```

```
            sudo crudini --set /var/lib/config-data/puppet-generated/neutron/etc/
neutron/plugins/ml2/ml2_conf_arista.ini ml2_arista eapi_username $deploy_username
            sudo crudini --set /var/lib/config-data/puppet-generated/neutron/etc/
neutron/plugins/ml2/ml2_conf_arista.ini ml2_arista eapi_password $deploy_password
            sudo crudini --set /var/lib/config-data/puppet-generated/neutron/etc/
neutron/plugins/ml2/ml2_conf_arista.ini ml2_arista eapi_host $deploy_cvxip
            crudini --set /var/lib/config-data/puppet-generated/neutron/etc/neutron/
plugins/ml2/ml2_conf_arista.ini ml2_arista use_fqdn $deploy_use_fqdn
            crudini --set /var/lib/config-data/puppet-generated/neutron/etc/neutron/
plugins/ml2/ml2_conf_arista.ini ml2_arista use_fqdn_physnet $deploy_use_fqdn_physnet
            crudini --set /var/lib/config-data/puppet-generated/neutron/etc/neutron/
plugins/ml2/ml2_conf_arista.ini ml2_arista sync_interval $deploy_sync_interval
            crudini --set /var/lib/config-data/puppet-generated/neutron/etc/neutron/
plugins/ml2/ml2_conf_arista.ini ml2_arista #onn_timeout  $deploy_conn_timeout
            crudini --set /var/lib/config-data/puppet-generated/neutron/etc/neutron/
plugins/ml2/ml2_conf_arista.ini ml2_arista region_name  $deploy_kregion
            crudini --set /var/lib/config-data/puppet-generated/neutron/etc/neutron/
plugins/ml2/ml2_conf_arista.ini ml2_arista sec_group_support  $deploy_sec_group_
support
            crudini --set /var/lib/config-data/puppet-generated/neutron/etc/neutron/
plugins/ml2/ml2_conf_arista.ini ml2_arista switch_info $deploy_switch_info

            crudini --set /var/lib/config-data/puppet-generated/neutron/etc/neutron/
plugins/ml2/ml2_conf_arista.ini ml2_arista manage_fabric $deploy_manage_fabric
            crudini --set /var/lib/config-data/puppet-generated/neutron/etc/neutron/
plugins/ml2/ml2_conf_arista.ini ml2_arista managed_physnets $deploy_managed_physnets
            crudini --set /var/lib/config-data/puppet-generated/neutron/etc/neutron/
plugins/ml2/ml2_conf.ini ml2 mechanism_drivers  $deploy_mech_drivers
            crudini --set /var/lib/config-data/puppet-generated/neutron/etc/neutron/
plugins/ml2/ml2_conf.ini ml2 type_drivers  $deploy_type_drivers
            sudo mkdir -p /var/lib/config-data/puppet-generated/neutron/etc/neutron/
conf.d/neutron-server
            sudo ln -sf ../../plugins/ml2/ml2_conf_arista.ini /var/lib/config-data/
puppet-generated/neutron/etc/neutron/conf.d/neutron-server/arista.conf
            sudo docker restart neutron_api
            sudo docker restart neutron_ovs_agent
            sudo echo "Arista ML2 service has been executed successfully and
updated to log folder" >> /root/ml2_automation_logs.txt
          }
          function arista_compute () {
            sudo docker restart neutron_ovs_agent
            sudo echo "Compute node neutron ovs agent has been restarted
successfully and updated to log folder" >> /root/ml2_automation_logs.txt
          }

          # The following if statement is to check for compute nodes.
          if hiera -c /etc/puppet/hiera.yaml service_names | grep -q -e neutron_api
-e horizon; then
              arista_service
          elif hiera -c /etc/puppet/hiera.yaml service_names | grep -q nova_
compute; then
              arista_compute
```

```
            else
              sudo echo "Arista ML2 service is not executed as expected" >> /root/
ml2_automation_logs.txt
            fi
```

Finally, deploy the overcloud, ensuring that you pass the arista_ml2_enable.yaml file as an argument (eg.'`openstack overcloud deploy <...> -e /home/stack/templates/arista_ml2_enable.yaml`')

Step 5 - Verification

This section contains several steps that verify that the solution works as configured.

Switch Topology Verification

Verify that each Arista Switch can see its neighbors - this includes all the compute hosts as well as other switches connected to this switch.

On an Arista TOR switch execute '`show lldp neighbors`' command and ensure that all the connected neighbors are listed. If there are missing neighbors or duplicate hosts, please follow steps in Appendix B: Troubleshooting to correct the condition.

Example:

```
TOR1#show lldp neighbors
Last table change time   : 0:00:05 ago
Number of table inserts  : 6
Number of table deletes  : 0
Number of table drops    : 0
Number of table age-outs : 0

Port       Neighbor Device ID            Neighbor Port ID          TTL
Et1        os-ctrl1                      eth5                      180
Et8        os-comp1                      eth3                      180
Et9        os-comp2                      eth4                      180
Et17       TOR2                          Ethernet9                 120
```

Data Center Topology Verification

Verify that the network topology for entire data center shows up correctly in CVX.

Execute the '`show network physical-topology hosts`' command on the CVX cluster to ensure that all hosts and switches that are in the OpenStack deployment are listed.

Example:

```
CVX1#show network physical-topology hosts
Unique Id            Hostname
---------------- ------------------------------
001c.7301.bb01    TOR2
90e2.ba2e.d300    os-comp1
90e2.ba2e.d3ec    os-clrl
90e2.ba21.eb4c    os-comp2
001c.7308.09ec    TOR3
```

Execute 'show network physical-topology neighbors' command and verify that the connectivity information is as expected.

Example:

```
CVX1#show network physical-topology neighbors
Host                  Interface       Neighbor Host          Neighbor Intf
--------------------- --------------- ---------------------- -------------
os-clrl               eth2            TOR2                   Ethernet4
os-clrl               eth3            TOR1                   Ethernet7
os-clrl               eth5            TOR2                   Ethernet1
os-clrl               eth6            TOR2                   Ethernet2
os-comp1              eth3            TOR2                   Ethernet8
os-comp1              eth4            TOR1                   Ethernet3
os-comp2              eth4            TOR2                   Ethernet9
os-comp2              eth5            TOR1                   Ethernet4
TOR1                  Ethernet3       os-comp1               eth4
TOR1                  Ethernet4       os-comp2               eth5
TOR1                  Ethernet7       os-clrl                eth3
TOR1                  Ethernet9       TOR2                   Ethernet17
TOR2                  Ethernet1       os-clrl                eth5
TOR2                  Ethernet2       os-clrl                eth6
TOR2                  Ethernet4       os-clrl                eth2
TOR2                  Ethernet8       os-comp1               eth3
TOR2                  Ethernet9       os-comp2               eth4
TOR2                  Ethernet17      TOR1                   Ethernet9
```

If CVX has no topology information, but LLDP is correctly shown on the switch, this could mean that CVX and TOR switches are not connected or communicating correctly. Please follow the troubleshooting steps described in CVX/TOR Connectivity.

CVX Reachability Verification

Verify Arista eAPI is enabled by launching a browser and entering 'https://<cvx-ip-address>/' in the browser address bar.
A simple command, such as 'show version' can then be executed to ensure that commands are able to be executed on CVX.

Network Creation Verification

Once all the steps above are complete, create a test network using OpenStack Horizon or using 'neutron net-create' in the OpenStack Neutron CLI ().
Verify that the network was successfully created as follows:

1. Log in to OpenStack Horizon and verify that the network was successfully created. Alternatively, use the OpenStack Neutron CLI command 'neutron net-list' to verify the creation of the network. Note the segmentation ID allocation for a given network
2. Log in to Arista CVX and verify the created network. To verify, use the 'show openstack networks' command. Verify that the segmentation ID of the created network matches with the segmentation ID allocated by OpenStack Neutron.

```
CVX1#show openstack networks
Region: RegionOne
  Tenant Name: Tenant-Enterprise
  Tenant Id: c605204020f34320adf70f101ace3c57
      Network Name        Network Id                                Seg Type    Seg Id
   ------------------- ------------------------------------- -------------- ------
      R&D-Network         ffe11aeb-6dc1-4d65-a30b-98ca63bfc24d       vlan        1002
      Sales-Network       f4fe3cbe-66d3-4a9f-a43e-220c0306ff81       vlan        1003

  Tenant Name: Tenant-Retail
  Tenant Id: 7ab54896083645d68c63e43357fb3076

      Network Name            Network Id                                Seg Type    Seg Id
   ---------------------- ------------------------------------- -------------- ------
      HR Network             46162753-279b-4872-8ead-6244c8040527       vlan        1005
      Marketing Network      0195b6c1-63fa-4c71-8b05-5b7a8ff2037e        vlan        1004
```

VM Creation Verification

Using OpenStack Horizon or the nova boot command in the OpenStack Nova CLI, create a VM and attach it to one of the networks which were created in the previous step.

Verify that the VM was successfully created as follows:

1. Log in to OpenStack Horizon and verify that the instance (VM) was successfully created. Alternatively, use OpenStack Nova CLI to verify the creation of the VM using 'nova list')
2. Login to Arista CVX and verify the created VMs. To verify, use 'show openstack vms' command.

```
CVX1#show openstack vms
Region: RegionOne
  Tenant Name: Tenant-Enterprise
  Tenant Id: c605204020f34320adf70f101ace3c57

      VM Name          VM Id                                  Host                                 Network Name
   --------------- ------------------------------------ ------------------------------------ -------------
      Bob's VM         8136bcea-a965-4eb2-8397-99631be52b35      os-h-comp2.sjc.aristanetworks.com    Sales-Network
      DevTest VM       70c53069-5911-4902-b87a-60246388b66f      os-h-ctrl1.sjc.aristanetworks.com    R&D-Network

  Tenant Name: Tenant-Retail
  Tenant Id: 7ab54896083645d68c63e43357fb3076

      VM Name          VM Id                                  Host                                 Network Name
   --------------- ------------------------------------ ------------------------------------ -----------------
      Admin VM 2       80f29a24-ec5b-41f3-bb09-55cec11458b0      os-h-ctrl1.sjc.aristanetworks.com    HR Network
                                                                                                   Marketing Network
      Payroll VM       61c37e04-371d-4d8b-a0bc-9df8120af2e7      os-h-comp2.sjc.aristanetworks.com    HR Network
      Tom's VM         753541b8-009d-4042-a16a-d3e8544140ec      os-h-comp1.sjc.aristanetworks.com    Marketing Network
```

Additionally, verify the details of the network to ensure that the DHCP port was created and matches with the DHCP port selected by Neutron. Use 'show openstack networks detail' command to get this information.

Example:

```
CVX1#sh openstack networks detail
Region: RegionOne
  Tenant Name: Tenant-Enterprise
  Tenant Id: c605204020f34320adf70f101ace3c57

    Network Name: R&D-Network
    Network Id: ffe11aeb-6dc1-4d65-a30b-98ca63bfc24d
    DHCP VM Id: dhcpf30f7f6c-f5de-53c9-b6ac-8ffc7cd9c321-ffe11aeb-6dc1-4d65-a30b-98ca63bfc24d
        Host: os-h-ctrl1.sjc.aristanetworks.com
        Port Id: f618d372-433a-4e0d-ac8b-078dd2ff5f14
        Port Name: Unknown

Host                        Switch Name   Switch Id       Switch Interface   Seg Type  Seg Id
--------------------------- ------------- --------------- ------------------ --------- -------
os-h-ctrl1                  TOR1           001c.730b.0d5e Ethernet17         vlan        1002
```

VLAN Auto Provisioning

After verifying the creation of networks and VMs on CVX, verify that correct VLANs are configured on the individual Arista switches.

On each TOR switch, verify if the VLANs are correctly provisioned. If VLANs are not configured correctly, ensure that the VLAN pools are correctly configured. See Arista VLAN type driver below.

```
TOR1#show vlan
VLAN  Name                            Status    Ports
---- -------------------------------  --------- --------------------------------
1     default                         active    Et17
1000* VLAN1000                        active    Et1, Et2, Et4, Et8, Et9, Et17
1001* VLAN1001                        active    Et1, Et2, Et4, Et8, Et9, Et17
1002* VLAN1002                        active    Et1, Et2, Et4, Et8, Et9, Et17
1003* VLAN1003                        active    Et1, Et2, Et4, Et8, Et17
* indicates a Dynamic VLAN
```

Optional L2 configuration

The section below contains information on optional Layer 2 (L2) configuration, including configuring OpenStack to automatically perform VLAN to Virtual Network Identifier (VNI) mapping which will enable the use of a L3 fabric with hardware VTEPs.

Automating VLAN to VNI mapping

This feature automates mapping a dynamically provisioned VLAN to a VNI on the TOR, enabling using a Layer 3 (L3) fabric to interconnect TOR switches without the use of any software VTEPs.

From OpenStack Neutron's perspective, the logical networks are backed by VLANs. Consequently, the L2 agent on the hypervisor will configure the virtual switch to send out 802.1Q tagged traffic for each tenant network.

Once the VXLAN interfaces are configured on the Arista switches, the OpenStack agent automatically adds the required VLAN to VNI mappings to these VTEPs. Traffic ingressing the TOR switch is mapped into a VXLAN tunnel and on egress, it is mapped back to the VLAN associated with the VNI, making it transparent to all devices other than the participating HW VTEPs.

VLAN to VNI mappings are also specific to an OpenStack region. Multiple regions can be configured to use the same VLAN ranges where each region has its own set of TORs as those VLANs will be mapped into different VNIs which can share the same spine or other IP infrastructure devices that are part of the L3 fabric, such as firewalls.

Note: that if multiple regions share the same set of TOR switches, each region should be configured with a unique VLAN/ VNI range.

VLAN to VNI mapping can be configured on CVX as follows:

```
CVX1(config)#cvx
CVX1(config-cvx)#service openstack
CVX1(config-cvx-openstack)#region RegionOne
CVX1(config-cvx-openstack-R1)#networks map vlan 10-20 vni 100010-100020
```

The above command creates a mapping of VLAN range 10-20 to VNI range 100010-100020 for RegionOne. Once VMs are created using a VLAN in the 10-20 range on the OpenStack compute nodes the VLAN to VNI mapping is created on that TOR switch automatically.

This feature requires the use of the Arista CVX VXLAN Control Service (VCS) to distribute MAC addresses and VTEP flood lists for each VNI to the hardware VTEPs. To enable VCS on CVX:

```
CVX1(config)#cvx
CVX1(config-cvx)#service vxlan
CVX1(config-cvx-vxlan)#no shutdown
```

On the TOR switches, a VXLAN interface needs to be created and the VXLAN controller-client needs to be enabled. This can be configured with the following commands:

Create a loopback interface

Create a loopback interface and assign it an IP address. The assigned IP address is the IP address of the VTEP.

```
TOR1(config)#interface loopback0
TOR1(config-if-loopback0)#ip address A.B.C.D/32
```

Configure a VXLAN interface

Create a VXLAN interface and associate with the loopback that was created above. Additionally, enable the VXLAN control service.

```
TOR1(config)#interface vxlan 1
TOR1(config-if-Vx1)#vxlan source-interface loopback0
TOR1(config-if-Vx1)#vxlan controller-client
```

Optionally starting in 4.23.2F: Disable VCS import (If using EVPN VxLAN control plane)

```
TOR1(config-if-Vx1)#vxlan controller-client import vlan none
```

See https://eos.arista.com/eos-4-18-1f/evpn-vxlan/ for EVPN configuration instructions.

ML2 Hierarchical Port Binding

ML2 Hierarchical Port Binding (HPB) shares all the benefits of the automatic VLAN-to-VNI mapping solution presented above, but is different in a few key ways:

- Instead of Arista CVX allocating VNIs for a network transparently, OpenStack Neutron is aware of the fact that there is a L3 fabric in use and allocates the VNI for each logical network.
- While the L2 agent configuring the virtual switch on the hypervisor still configures it to send out 802.1Q tagged traffic, CVX configures the Arista TOR to map the VLAN into the VNI provided by Neutron.
- It is possible to scale beyond 4096 tenant networks since each TOR switch can map 4096 VLANs to different VNIs.
- OpenStack Neutron logically groups the TOR and all hypervisors connected to it into a physical network (or physnet), representing the set of devices where a VLAN identifier represents the same logical network. Traffic headed north of this TOR is encapsulated and identified by the VNI associated with the logical network.

An OpenStack Neutron network is associated with a VNI that is globally significant across the physical infrastructure. On each of the TOR switches, this VNI is mapped into a locally significant VLAN identifier that is allocated from a range of available VLAN identifiers specific to that TOR switch or port - also known as physnet.

To enable hierarchical port binding, configure OpenStack Neutron to load both the VXLAN and VLAN type drivers. Additionally, configure the range of available VLANs on a per-physnet basis in the OpenStack Neutron configuration file.

Configuration Requirements to support HPB

In order to support HPB the following sections in `ml2_conf.ini`, `linuxbridge_agent.ini` and `openvswitch_agent.ini` on controller and compute nodes should be configured.

MLAG Note

For any physnets that are comprised of an MLAG pair, replace <TOR> with <PEER1>_<PEER2> wherever TOR appears in this section.

On Controller Nodes -

Execute the following changes on OpenStack controller nodes.

Section `'ml2'`:

```
[ml2]
tenant_network_types = vxlan
mechanism_drivers=..., arista
```

The `mechanism_drivers` option should include `'arista'` driver.

`Section 'ml2_type_vlan':`

In this section VLAN ranges per physical network need to be defined for all
physical networks in the topology. The physical network is identified by switch name in the topology.

Example:

```
[ml2_type_vlan]
network_vlan_ranges = TOR1:100:200,TOR2:300:400
```

The configuration above enables VLANs 100-200 for switch TOR1 and 300-400 for switch TOR2.

Note: The format of switch name used above depends on the 'use_fqdn' setting in ml2_conf_arista.ini; if it is set to 'True' the switch FQDN needs to be used.

Section `'ml2_type_vxlan'`

This section defines the VXLAN ranges used by tenant networks in the topology.

Example:

```
[ml2_type_vxlan]
vni_ranges = 5000:6000
```

Section 'vxlan':

(May be in linuxbridge_agent.ini):

In this section managing of vxlan networks needs to be disabled for the linuxbridge/OVS agents.

```
[vxlan]
enable_vxlan = False
```

Example:

Section `'ovs'`

(May be in openvswitch_agent.ini)

In this section the `'bridge_mappings'` option needs to be set. This option defines the mapping between physical network and Open vSwitch (OVS) bridge. This mapping shows the connectivity between physical switches and VM instances on the controller node.

'tunnel_types' must also be empty. This ensures that the OVS agent does not bind vxlan fabric segments and instead allows the Arista ML2 plugin to do this binding.

Example:

```
[ovs]
bridge_mappings = TOR2:br-eth3
[agent]
tunnel_types =
```

Note: The switch name should use the same naming format as section `'ml2_type_vlan'.`

Additonally in `ml2_conf_arista.ini:`

```
[ml2_arista]
manage_fabric = True
```

On Compute Nodes -

Execute the following changes on OpenStack controller nodes.

Section 'ovs'

The setting in this section is similar to the setting configured above for controller nodes. The `bridge_mappings` configuration option should be set to switch name that compute node is connected to.

Example:

```
[ovs]
bridge_mappings = TOR1:br-eth3
```

Bare Metal Provisioning

Through OpenStack Ironic integration with Neutron, it is possible to provision bare metal servers that are attached to Arista switches and connect them to tenant networks. All of the features that Arista supports for provisioning networks for VMs is extended to bare metal servers. This includes automatic VLAN-to-VNI mapping and Hierarchical Port Binding.

Additionally, there are several features that are only supported for bare metal provisioning. The additional features that Arista supports for bare metal provisioning are:

- Automated provisioning network to tenant network switchover during bare metal boot. This feature allows for the use of a quarantine or provisioning network during the initial boot and configuration of a bare metal host on a dedicated network that is separate from tenant networks.
- Security groups can be applied as ACLs on switch interfaces connected to bare metal servers
- Additional VLANs can be added to bare metal connected interfaces through the use of Neutron trunk ports

Documentation for configuring networking for bare metal servers can be found in Appendix A: References.

When creating an OpenStack Ironic port, the chassis ID and switch interface that are connected to the bare metal server must be specified in --local-link-connection. The chassis ID can be found by running "show lldp local-info" on the switch that the host is connected to. The switch interface can be found by locating the switch port the bare metal host is connected to and takes the form EthernetX[/X], for example Ethernet48/1.

Security Groups

In order to enable security group provisioning for bare metal servers, the IP address of the switch's management interface must also be specified in the optional switch_info parameter in ml2_conf_arista.ini.

The following configuration must be added to ml2_conf_arista.ini:

```
[ml2_arista]
sec_group_support = True
switch_info = <switch-IP>:<switch-username>:<switch-password>,<switch2>
```

All switches connected to bare metal servers must be specified in the comma separated `switch_info` parameter.

There are some caveats to note for security group support:

- Only TCP, UDP and ICMP rules are supported
- Only IPv4 ACLs are supported
- Only one security group may be applied to an interface connected to a bare metal server

Port-Channel Support

In order to dynamically provision VLANs and VLAN-toVNI mappings for baremetal servers connected to Port-Channels, an Ironic port group should be created with one port group member for each physical link that forms the Port-Channel.

For example, if there MLAG pair of switch 00:00:00:00:00:00 and switch 11:11:11:11:11:11 where Ethernet5 on each switch forms a Port-Channel connected to a baremetal node, an Ironic port group with two member ports would need to be created. The first member port would be configured with local_link_connection switch_id=00:00:00:00:00:00 and port_id=Ethernet5 and the second member port would be configured with switch_id=11:11:11:11:11:11 and port_id=Ethernet5.

Arista VLAN type driver

As logical networks are created, OpenStack Neutron allocates identifiers from an available range of VLAN IDs that the operator configures in a configuration file. An alternate to that approach is to use the Arista VLAN type driver which allows a network operator to assign a range of VLANs to OpenStack from CVX. Only one VLAN type driver should be enabled within a single OpenStack region and therefore the Arista VLAN type driver may not be used in conjunction with the default VLAN type driver within a single region.

Apart from bringing visibility and control of how the VLAN space is segmented to CVX, this has the added benefit of allowing the range of identifiers allocated to OpenStack to be modified without requiring an OpenStack Neutron service restart.

When this type driver is enabled, the VLANs for tenant networks are allocated from a VLAN pool configured on CVX. The Arista VLAN type driver must be enabled in OpenStack as well as on CVX.

As a caveat, the Arista VLAN type driver currently only supports a single physnet, and it must be named 'default'.

Enable type driver in the ML2 configuration

Edit the ML2 configuration file used when OpenStack Neutron is loaded and alter the fields below.

```
[ml2]
tenant_network_types = vlan
type_drivers = arista_vlan
```

Enable the type driver on CVX

Enable the VLAN type driver and specify a VLAN resource pool in CVX.

```
CVX1(config)# cvx
CVX1(config-cvx)# service openstack
CVX1(config-cvx-openstack)# network type-driver vlan arista
CVX1(config-cvx-openstack)# region RegionOne
CVX1(config-cvx-openstack-regionone)# resource-pool vlan <range>
```

Mandatory Regions

On supported EOS releases, it is possible to configure CVX to require that a specific region be in sync with CVX before CVX configures any switches. This removes the possibility that CVX disrupts the datapath in deployments where a single CVX cluster manages multiple regions and also allows regions to be configured as optional if CVX is managing both production and test networks. Mandatory and optional regions can be configured as follows:

```
CVX1(config)# cvx
CVX1(config-cvx)# service openstack
CVX1(config-cvx-openstack)# region RegionOne
CVX1(config-cvx-openstack-regionone)# provision sync [mandatory|optional]
```

In single region deployments, the sole region is always considered mandatory (regardless of this config.) In multi-region deployments, all regions are considered optional by default.

Configuring the Arista L3 Service Plugin

In addition to the Arista ML2 driver which provisions Layer 2 networks, the Arista L3 Service Plugin provisions SVIs on Arista switches in order to provide routing between different logical networks. L3 functionality can be enabled on either the TOR or Spine switches.

Summary of steps

1. Arista switch configuration
2. Configure Neutron to run the Arista L3 Plugin

Step 1 - Arista switch configuration

Enable Arista eAPI on the switches the Arista L3 plugin will create SVIs on - typically a TOR or spine pair.

```
TOR1(config)#management api http-commands
TOR1(config-mgmt-api-http-cmds)#no shutdown
TOR1(config-mgmt-api-http-cmds)#no protocol http
TOR1(config-mgmt-api-http-cmds)#protocol https
```

Verify that the Arista eAPI is enabled by launching a browser and entering 'https://<switch-ip-address>/' in the browser address bar.

A simple command, such as `'show version'` can then be executed to ensure that commands are able to be executed on the switch.

Step 2 - Configure OpenStack Neutron to run the Arista L3 Plugin

The steps below walk through installing the Arista L3 plugin in OpenStack Neutron.

Prior to this step, the following prerequisites should be completed:

- OpenStack has been installed
- networking-arista package has been installed on all the nodes running the OpenStack Neutron service

OpenStack Neutron Configuration

When OpenStack Neutron L3 services are enabled, the L3_Router service plugin is installed by default. In order to use Arista switches for routing, the Arista L3 service plugin needs to be installed and enabled.

Execute the following steps to configure OpenStack Neutron to run the Arista L3 service plugin:

Edit the OpenStack Neutron configuration file

Edit the /etc/neutron/neutron.conf file and add the Arista L3 driver to service_plugins.

For OpenStack Liberty or newer releases:

```
service_plugins =
arista_l3,neutron.services.loadbalancer.plugin.LoadBalancerPlugin
```

For OpenStack Kilo and older releases:

```
service_plugins =
neutron.services.l3_router.l3_arista.AristaL3ServicePlugin,neutron.services.lo
adbalancer.plugin.LoadBalancerPlugin
```

Note: Please replace `L3RouterPlugin` with `AristaL3ServicePlugin` and ensure that this is the first service plugin in the list.

Arista L3 plugin Configuration

The Arista L3 Plugin provides several configuration knobs to help optimize communication between the mechanism driver and CVX based on the deployment.

Configuration options are divided into two parts; mandatory configuration and optional configuration.

Mandatory Configuration

Edit `/etc/neutron/plugins/ml2/ml2_conf_arista.ini` as follows:

```
[l3_arista]
primary_l3_host = IP address of Arista Switch
primary_l3_host_username = <user name>
primary_l3_host_password = <password>
```

Optional Configuration

The following optional configuration elements can be configured in `/etc/neutron/plugins/ml2/ml2_conf_arista.`

Configuration Element	Default	Description
`secondary_l3_host`	None	This is IP address of the second Arista switch. This address is needed if the Arista switches are configured as MLAG pair. Note: The credentials for the secondary Arista switch must match the primary.
`mlag_config`	False	By default, the plugin assumes a single Arista switch. This configuration must be set to "True" if a pair of switches are configured with MLAG (Multi chassis Link Aggregation) and will be managed by this plugin. If this flag is set to True, both primary_l3_host and secondary_l3_ host fields must be set to the IP addresses of primary and secondary switches.
`use_vrf`	False	This flag dictates if the router is to be associated with a VRF. By default, the router is created/ associated in default VRF. If this flag is set, the router is created/associated with a specific VRF. The name of the router specified at the time of creation (neutron router-create <name>) is used as the VRF name. Hence, the VRF name is not required. Note: VRF support in MLAG configurations was added in the Queens release Note: Please be aware of VRF scale limitations for the Arista switches in your environment Note: VRF support for Newton and older OpenStack releases is not compatible with EOS 4.23.0F and newer releases
`sync_interval`	180 (seconds)	This identical to sync_interval specified for ML2 configuration. This is used to sync L3 configuration between OpenStack Neutron and Arista switches performing routing functionality.
`enable_cleanup`	False	Enables the cleanup of unused VLANs, VRFs and SVIs on EOS L3 hosts in the sync worker. If enabled, ensure that all non-openstack VLANs are added to 'protected_vlans to ensure that they are not removed by the sync worker.
`protected_vlans`	None	List of vlans or <vlan_min>:<vlan_max> ranges that should never be cleaned up by the L3 sync worker. This applies to both VLANs and SVIs. This setting is only relevant if enable_cleanup is True
`vrf_default_route`	False	When this parameter is set to True, the L3 plugin will create a default route when a router is connected to an external network. The created default route will use the gateway IP for the subnet defined in the external network as the next hop.This parameter is valid only when use_vrf is enabled

Restart OpenStack Neutron

Please restart the OpenStack Neutron server on all nodes after completing the configuration steps above and passing the `ml2_conf_arista.ini` file as a `--config-file` parameter to OpenStack Neutron.

Router IP selection for VARP Configuration

When mlag_config is specified in `/etc/neutron/plugins/ml2/ml2_conf_arista.ini` the router IP address for each instance of the router is automatically selected by the Arista L3 Plugin.

The router instance on the primary switch (`primary_l3_host`) gets the highest IP address for a given subnet and the router instance on the secondary switch (`secondary_l3_host`) gets the second highest IP address.

For example, for IPv4 subnet 10.10.10.0/24, the router instance on primary switch is assigned with address of 10.10.10.254 and secondary switch is assigned with 10.10.10.253.

Appendix A: References

Arista OpenStack Neutron drivers and plugins can be found at:

https://github.com/openstack/networking-arista

Instructions on installing CVX can be found at:

https://www.arista.com/en/cg-cv/cloudvision-chapter-2-cloudvision-exchange-cvx

Instructions on CVX High Availability (HA) can be found at:

https://eos.arista.com/eos-4-15-4f/cvx-ha/

Additional information about CloudVision and services available on CVX can be found at:

https://eos.arista.com/category/cvx/

https://eos.arista.com/tag/cvx/

Configuring Bare Metal Provisioning networks:

http://docs.openstack.org/developer/ironic/deploy/multitenancy.html

Open vSwitch ovs-vsctl Documentation:

http://openvswitch.org/support/dist-docs/ovs-vsctl.8.html

Configuring Neutron Trunk ports:

https://github.com/openstack/neutron/blob/master/doc/source/admin/config-trunking.rst

Appendix B: Troubleshooting

The section below contains troubleshooting steps for diagnosing issues in an Arista backed OpenStack deployment. For additional assistance, please contact Arista TAC.

Troubleshooting initial setup

The steps below are intended for troubleshooting initial setup steps.

CVX - TOR Connectivity

For troubleshooting connectivity between Arista CVX and Arista Top of Rack switches, first ensure that the CVX server is enabled on the CVX node(s) by executing the following command and ensuring that the status is "enabled." If it is not enabled, please see Step 3 - Arista CVX Setup.

```
CVX01#show cvx
CVX Server
  Status: Enabled
  Heartbeat interval: 30.0
  Heartbeat timeout: 300.0
```

If the Arista CVX server is enabled, verify that CVX can connect to all the TOR switches:

```
CVX01#show cvx connections
Switch 00:1c:73:00:ae:a0
  Hostname: TOR1
  Status: up
  Last heartbeat sent: 20899.7651362
  Last heartbeat received: 20907.1354178
```

If the TOR switch is not listed, verify the steps in Step 2 - Arista TOR Switch Configuration and Step 3 - Arista CVX Setup..

Also, verify that appropriate services are enabled and running. Execute the following command and ensure that Arista CVX and OpenStack are enabled.

```
CVX01#show cvx service
CVX
  Status: Enabled
  Supported versions: 1
  Switch 00:1c:73:00:ae:a0
    Status: Enabled
OpenStack
  Status: Enabled
  Supported versions: 1
  Switch 00:1c:73:00:ae:a0
    Status: Enabled
    Negotiated version: 1
```

Topology

If the network topology shows incorrect information, verify LLDP operation on each TOR switch. For example, the following command lists each compute hypervisor and adjacent switch connected to the TOR switch:

```
TOR2#show lldp neighbors
Last table change time   : 0:00:05 ago
Number of table inserts  : 6
Number of table deletes  : 0
Number of table drops    : 0
Number of table age-outs : 0

Port       Neighbor Device ID            Neighbor Port ID          TTL
Et1        os-clrl                       eth5                      180
Et8        os-comp1                      eth3                      180
Et9        os-comp2                      eth4                      180
Et17       TOR1                          Ethernet9                 120
```

Verify LLDP is running on the TOR switch

The Arista solution requires LLDP to be run on each TOR switch. Though Arista switches run LLDP by default, the following example displays the output seen if LLDP is disabled.

```
TOR1#show lldp
% LLDP is not enabled
```

Enable LLDP on the TOR switch

If LLDP is not running on the TOR switch, it can be enabled with the following command.

```
TOR1(config)#lldp run
```

Additionally, perform the troubleshooting steps described in CVX-TOR Connectivity.

Duplicate hosts in topology

If all verification steps above have been performed and hosts are showing up as duplicates, this can be caused by any one of the following conditions:

1. LLDPD as well as LADVD are both enabled and running (or any other combination of multiple daemons that speak LLDP are running). Please ensure only a single LLDP daemon is running.
2. More than one interface from a given host is connected to switch. This is by design and is not an issue. If the desire is to see a single interface, turn off the LLDPD/LADVD on the other interfaces.

Missing hosts in topology

This can happen for the following reasons:

1. Physical connectivity - please verify that the missing host has physical connectivity to the switch.
2. LLDPD/LADVD is not properly configured on a given host or is enabled on the wrong interface of the host - for example host interface eth3 is connected to the TOR switch, but LLDP is enabled on interface eth4.

Neutron - CVX Connectivity

Once OpenStack Neutron is configured to run the Arista drivers, Neutron server logs can help verify that the Arista ML2 driver is able to connect to the active CVX instance. There should be a log entry indicating that the Arista driver is syncing its state with CVX: spacing.

```
2013-10-11 18:24:49.151 9419 INFO neutron.plugins.ml2.drivers.mech_arista.mechanism_arista [-]
Syncing Neutron <-> EOS
2013-10-11 18:24:49.151 9419 INFO neutron.plugins.ml2.drivers.mech_arista.mechanism_arista [-]
Executing command on Arista EOS: ['enable', 'configure', 'management openstack', 'region
RegionOne'
0/ user neutron password', 'exit', 'exit']
```

If the log entry above is not present, verify that the steps in Step 4 - Configure Neutron to load the Arista mechanism driver and that CVX is reachable from the OpenStack controller node running OpenStack Neutron.

Also, examine the trace or logs for the Neutron server and verify that the Arista driver has been loaded and there have been no failures.

VMs are unable to communicate

If all the above steps work correctly, but there are still noticing issues with VMs being unable to communicate successfully, verify the Open vSwitch (OVS) or Linux bridge configuration.

The following is an example which shows the OVS bridge configuration on compute hypervisor host os-comp2. It shows that interface eth4 is added to the OVS physical bridge br-eth4, which is what the integration bridge is connected to. This should be the same interface which shows up in the topology connected to the TOR switch:

```
os-comp2@os-comp2:~$ sudo ovs-vsctl show
23ebdf64-9414-4789-8a00-83f77f0a1c84
    Bridge br-tun
        Port br-tun
            Interface br-tun
                type: internal
        Port patch-int
            Interface patch-int
                type: patch
                options: {peer=patch-tun}
```

```
Bridge "br-eth4"
        Port "phy-br-eth4"
            Interface "phy-br-eth4"
        Port "eth4"
            Interface "eth4"
        Port "br-eth4"
            Interface "br-eth4"
                type: internal
    Bridge br-int
        Port "qvo78334931-99"
            tag: 113
            Interface "qvo78334931-99"
        Port "qvo81d550bb-6d"
            tag: 113
            Interface "qvo81d550bb-6d"
        Port "qvo2131fa66-92"
            tag: 114
            Interface "qvo2131fa66-92"
        Port "int-br-eth4"
            Interface "int-br-eth4"
        Port "qvob603dbb0-9d"
            tag: 113
            Interface "qvob603dbb0-9d"
        Port "qvoa8b768e1-48"
            tag: 114
            Interface "qvoa8b768e1-48"
        Port br-int
            Interface br-int
                type: internal
    ovs_version: "1.4.0+build0"
```

For details on the ovs controller commands, use `'sudo ovs-vsctl --help'` or refer to the Open vSwitch documentation for more information.

VM and Tenant Names are Unknown or not updated

The Arista OpenStack Agent periodically communicates with the OpenStack Keystone service to update the names of VMs and Tenants. In order to identify the OpenStack Keystone service endpoint, the OpenStack agent attempts to use information the Arista ML2 driver sends to Arista CVX, which in turn obtains it from configuration files specified in Step 4 - Configure OpenStack Neutron. However, it is possible to override those values, please see optional configuration in Step 3 - Arista CVX Setup.

By default, this communication takes place every 6 hours as it is anticipated that these names do not change on a regular basis. This interval is configurable by executing the following command:

```
CVX01(config-cvx)#name-resolution interval <value in seconds>
```

If a name is changed and is not updated, it is possible to force name resolution by executing the following command:

```
CVX01(config-cvx)#name-resolution force
```

Note: If no tenant name is configured, it is displayed as "Unknown"

Steps to troubleshoot name resolution issues

Please use following steps to troubleshoot and fix name resolution issues:

Step 1 - Verify that CVX has valid URL to reach the OpenStack Keystone endpoint

Assuming there is no configured OpenStack Keystone URL, the following steps will help determine if Arista CVX has a valid OpenStack Keystone URL.

On Arista CVX, execute "`show openstack regions`" and inspect the value of "`Authentication URL`" and ensure that it is reachable. If the value is the same as shown below, name resolution will not succeed. If it's a valid URL, and reachable from CVX, proceed to step B below.

```
CVX01()#show openstack regions
   Region           Service        Authentication URL            User
--------------- -------------- -------------------------------- -------
   RegionOne        keystone       http://127.0.0.1:35357        neutron
```

Note1: In the above example, the URL is pointing to localhost, which is not correct. Please update the OpenStack Neutron configuration files to reflect the correct URL. Be sure to restart the OpenStack Neutron service after making this change.

Note2: In the Pike release of networking-arista (2017.2.0), the arista ML2 driver no longer registers keystone authentication endpoint with CVX. The user is required to configure the authentication URL when running EOS-4.18.x releases. In the absence of the keystone authentication endpoint the 'show openstack regions' displays no output.

```
Example of configuration in RegionOne:

switch(config-cvx-openstack-RegionOne)#keystone auth-url http://1.2.3.4 port 35357 version v3
```

Step 2 - Valid Keystone URL, but names are still showing unknown

After executing the step above and verifying that:

- A valid Keystone URL is present in the configuration file
- Verified that the OpenStack Keystone endpoint is reachable from CVX

If names are still showing unknown, please execute the steps described in Step 3 - Arista CVX Setup.

Removing a region from CVX

The following command may be used to manually force the cleanup of the entire configuration for a given OpenStack controller. This will delete every tenant, network, and VM for a given region.

```
CVX01(config)#cvx
CVX01(config-cvx)#service openstack
CVX01(config-cvx-openstack)#no  region <region name>
```

Appendix C: Release Notes

The section below contains both OpenStack integration notes as well as EOS specific release notes.

OpenStack integration release notes

The following section lists features added per OpenStack release.

Queens

- Improved efficiency and reliability of CVX synchronization
- Added support for baremetal trunk ports
- Added support for MLAG + VRFs in the L3 Plugin
- Added support for Neutron HA routers and legacy routers

Pike

- Added support VLAN aware VMs via Neutron Trunk Ports

Ocata

- Miscellaneous bug fixes

Newton

- Added support for Neutron-Ironic integration
- Added support for configuring security groups (access control lists) on bare metal ports
- Added support for hierarchical port binding
- Added support for dedicated OpenStack endpoint on CVX

Mitaka

- Added support for a new type driver: Arista VLAN type driver. This type driver allows the operator to control the range of VLANs used for tenant networks from the CVX
- Added support for the OpenStack Neutron Distributed Virtual Router by automatically provisioning the switch port to carry all VLANs corresponding to subnets reachable by the VM
- Added support for project Astara which simplifies deployment of OpenStack Neutron's network services by bundling a number of L3-L7 network services like routing, load balancers and firewalls into a virtual machine
- Added support for creation of unattached ports that can be bound to an instance at a later point in time
- Miscellaneous bug fixes

Liberty

- Added support for CVX High Availability (HA)
- Added support for Neutron Router High Availability (HA)
- Further enhancements to the Arista ML2 driver sync mechanism
- Completed transition to the networking-arista repository
- Use python requests library instead of jsonrpclib for communication between the Arista ML2 driver and CVX as concurrent requests using jsonrpclib would result in intermittent SSL errors.

Kilo

- Added support for Neutron's shared networks which are created by admin and are visible to all tenants. Tenants can use these networks to create inter-tenant VM connectivity without using routing.
- Added support for OpenStack Neutron Controller High Availability
- Supported community decision to move vendor drivers out of the OpenStack Neutron tree and into a vendor repository (networking-arista: https://github.com/stackforge/networking-arista).
- Support for OpenStack Neutron Router High Availability has been subsequently backported to this release.

Juno

- Added support for the Arista Layer 3 Service Plugin, which automates provisioning of L3 features on Arista switches. In response to API calls to create/delete a router and add/remove interfaces, appropriate SVIs (Switch Virtual Interfaces) are created on respective switches.

IceHouse

- Enhanced the Arista ML2 driver re-sync mechanism: In the event of a CVX reboot (or cold restart), it needs to re-synchronize with neutron. This enhancement, reduces the time taken to do so and requires no user intervention.
- Enhanced eAPI models: The API between the Arista ML2 driver and CVX has been enhanced.
- Enhanced devstack the Arista ML2 driver can be installed in a single step along with OpenStack Neutron. Additionally, no more patches are required when installing the Arista driver.

Note: Newer OpenStack and networking-arista releases include all features introduced in older versions.

OpenStack specific Arista EOS release notes

The following list lists features added per Arista EOS release.

4.23.2F

- Added support for EVPN VxLAN control planes

4.21.0F

- Added support for OpenStack Queens
- Improved stability of OpenStack JSON API

4.17.2FX-OpenStack

- Added support for Neutron-Ironic integration (bare metal provisioning)
- Added support for configuring security groups on bare metal ports
- Added support for DVR
- Added support for hierarchical port binding
- Added support for Keystone v3
- Added support for configuring Keystone URL on EOS

4.15.4F

- Added support for CVX clustering and high availability. On a CVX instance failure, this reduces the time before neutron can continue to provision new resources as a standby instance can take over.

4.14.5F

- Added support for automatic VLAN to VNI mapping from CVX. With this enhancement, CVX can be configured to automatically map a VLAN associated with a tenant network with a VNI in order to provision a L3 fabric without incurring the encap/decap penalty in software.
- Added support for CVX Graceful Restart which prevents reprogramming VLANs associated with tenant network on Arista switches in the time duration between CVX restarting and resyncing with Neutron.

Note: Newer EOS releases include all features introduced in older versions.

Santa Clara—Corporate Headquarters
5453 Great America Parkway,
Santa Clara, CA 95054

Phone: +1-408-547-5500
Fax: +1-408-538-8920
Email: info@arista.com

Ireland—International Headquarters
3130 Atlantic Avenue
Westpark Business Campus
Shannon, Co. Clare
Ireland

Vancouver—R&D Office
9200 Glenlyon Pkwy, Unit 300
Burnaby, British Columbia
Canada V5J 5J8

San Francisco—R&D and Sales Office 1390
Market Street, Suite 800
San Francisco, CA 94102

India—R&D Office
Global Tech Park, Tower A, 11th Floor
Marathahalli Outer Ring Road
Devarabeesanahalli Village, Varthur Hobli
Bangalore, India 560103

Singapore—APAC Administrative Office
9 Temasek Boulevard
#29-01, Suntec Tower Two
Singapore 038989

Nashua—R&D Office
10 Tara Boulevard
Nashua, NH 03062