ARISTA

CloudEOS and vEOS Router Configuration Guide

Arista Networks

www.arista.com

Arista CloudEOS and vEOS Router version 4.24.0.3FX

DOC-03496-07

Headquarters	Support	Sales
5453 Great America Parkway, Santa Clara, CA 95054 Santa Clara, CA 95054 USA +1-408 547-5500	+1-408 547-5502 +1-866 476-0000	+1-408 547-5501 +1-866 497-0000
https://www.arista.com	*support@arista.com*	*sales@arista.com*

Contents

Chapter 1

Overview

Arista CloudEOS and vEOS Router is supported on Amazon Web Service (AWS), Microsoft Azure and Google Cloud Platform, and on-premises deployment.

CloudEOS and vEOS Router

Arista CloudEOS and vEOS Router is a new platform release of EOS that is supported on Amazon Web Service (AWS), Microsoft Azure and other public clouds. It is also supported on customer equipment running Linux and VMware hypervisors. By bringing advanced network telemetry and secure IPSec VPN connectivity in a software-only package, CloudEOS and vEOS Router provides a consistent, secure and universal approach to hybrid cloud networking for any virtualized cloud deployment. Use cases for CloudEOS and vEOS Router include Secure Multi Cloud Connectivity, Interconnecting VPCs/VNets in the Public Cloud, Multi-site VPN aggregation and Network Function Virtualization.

Note: Arista CloudEOS Router is a new product with additional capabilities, it replaces the existing vEOS router. User can now upgrade existing vEOS router deployment to CloudEOS router following the information provided in the chapter Upgrade/Downgrade. CloudEOS Router and vEOS Router can be used interchangeably in this guide.

Chapter 2

License Management

This section describes the procedure for managing CloudEOS and vEOS license files.

2.1 Pay-As-You-Go (PAYG) in Cloud

This section of the document provides a high level overview about verifying the Pay-as-you-go (PAYG) instance installed on the CloudEOS and vEOS Router products on various supported public platform.

Overview

Pay-as-you-go (PAYG) is a software consumption model supported by various public cloud provider to charge the consumer based on the usage. Other software consumption model on public cloud provider is Bring-your-own License(BYOL). Each vendor who publish their product on public cloud imposes a license requirement for the real usage of their product in which case, the consumer needs to get the BYOL from the vendor in order to use the product in the public cloud.

One of the major benefits of the PAYG method is that there are no wasted resources and consumer only pays for the services procured rather than provisioning for a fixed amount of resources that may or may not be used. Another advantage of PAYG is that, consumers can quickly deploy the product on the public cloud without the need to contact the vendor for license. Normally public cloud provider distinguish each published product by vendor with a unique ID. This unique ID is stored in the cloud provider metadata server. Vendor product should check for the unique ID to distinguish its products from BYOL and PAYG, and allow consumers to use without the requirement of license from vendor.

- License Verification
- Troubleshooting

2.1.1 License Verification

The following commands are used to verify if an SFE and IPsec licenses are installed in PAYG mode for CloudEOS and vEOS.

Note: The **show license** command does not show licenses installed through PAYG feature.

Example show output for SFE.

If SFE license is not installed the following output is displayed -

```
switch# show platform sfe licensing

Licensing Information
---------------------
 License TC created: no
 Number of throttled interfaces: 0
```

If SFE license is installed the following output is displayed -

```
switch# show platform sfe licensing

Licensing Information
```

```
---------------------
License TC created: yes
Number of throttled interfaces: 1
Interfaces throttled:
Ethernet1: 80 Mbps
```

Example show output for IPsec

If IPsec is not installed the following output is displayed.

```
switch# show ip sec connection
! No valid IPsec license found. IPsec is disabled.
```

If IPsec is installed the following output is displayed.

```
switch# show ip sec connection
Tunnel      Source     Dest     Status       Uptime      Input      Output
  Rekey    Time
Tunnel63   1.0.0.1   1.0.0.2   Established 22 minutes    0 bytes    0
 bytes    34      minutes
If no valid certificate is installed, it displays configured IPsec
 connections.
```

2.1.2 Troubleshooting

The following **$curl** command is used to verify the if an AWS / Azure instance is an PAYG instance. This command is executed under **Bash** mode.

- PAYG support for AWS
- PAYG support for Azure
- PAYG support for GCP

2.1.2.1 PAYG support for AWS

The step shown in the example below is used to verify if an AWS instance is an PAYG instance. AWS customers can verify the product code of their PAYG instance by querying instance identity document from their running CloudEOS and vEOS Router instance.

- To retrieve the instance identity document, use the following command from your running instance:

```
[switch]$ curl http://169.254.169.254/latest/dynamic/instance-identity/
document
{
  "accountId" : "083837402522",
  "architecture" : "x86_64",
  "availabilityZone" : "us-west-1b",
  "billingProducts" : null,
  "devpayProductCodes" : null,
  "marketplaceProductCodes" : [ "cdcwmm26cap8fqlnkwuqte405" ],
  "imageId" : "ami-017900c328c2edfbe",
  "instanceId" : "i-058ebba29bd475e8b",
  "instanceType" : "c5.xlarge",
  "kernelId" : null,
  "pendingTime" : "2020-05-01T06:53:42Z",
  "privateIp" : "11.0.4.101",
  "ramdiskId" : null,
  "region" : "us-west-1",
  "version" : "2017-09-30"
}
```

2.1.2.2 PAYG support for Azure

The step shown in the example below is used to verify if an Azure instance is an PAYG instance.

Example metadata showing the SKU:

```
[switch]$ curl -H Metadata:true "http://169.254.169.254/metadata/
instance/compute?api-version=2017-08-01"
{"location":"westus",
"name":"adhip-test",
"offer":"cloudeos-router-payg",
"osType":"Linux",
"placementGroupId":"",
"platformFaultDomain":"0",
"platformUpdateDomain":"0",
"publisher":"arista-networks",
"resourceGroupName":"adhip2",
"sku":"cloudeos-4_23_0-payg",
"subscriptionId":"ba0583bb-4130-4d7b-bfe4-0c7597857323",
"tags":"","version":"4.23.0",
"vmId":"c23a7526-44c5-43af-bcf5-8b2419105393",
"vmSize":"Standard_D4_v3"
$
```

2.1.2.3 PAYG support for GCP

The Arista CloudEOS instance needs network connectivity and DNS resolution to use the GCP metadata server "metadata.google.internal" for various services including license validation. Normally the CloudEOS automatically picks up and configures the default route and DNS server(GCP default DNS server: 169.254.169.254) through DHCP during the initial instance bringup. However, to make sure the instance is able to access the DNS server and reach GCP metadata server properly, use the below CLI command as well as the license ID matches 3403635045915687054 for the PAYG image.

```
cloudEos#bash curl http://metadata.google.internal/computeMetadata/v1/i
nstance/licenses/0/id -H "Metadata-Flavor:Google"
3403635045915687054
```

Note: If you are using your own DNS server and/or DHCP server, please make sure that the above commands work properly by setting up the proper DNS resolution/routes.

The following Cloud EOS commands helps in licensing to bypass the DNS/network connectivity issues in case of issues due to custom DHCP/DNS setup:

```
cloudeos-router-payg-router-vm# ip host metadata.google.internal
 169.254.169.254
cloudeos-router-payg-router-vmr# ip route 169.254.169.254/32 Ethernet1
 <default_vpc_router>
```

where **<default_vpc_router>** is the second address in the primary IP range for the subnet in which Ethernet1 resides. For example, default_vpc_router is 10.1.2.1 in 10.1.2.0/24 subnet belonging to Ethernet1 in the google cloud.

However, note that, other features which needs access to the cloud provider web APIs like CloudHA, may still have issues with your own DNS/DHCP setup unless carefully planned. If you are using your own DNS/DHCP servers, please see details at https://cloud.google.com/compute/docs/internal-dns.

2.2 Bring-Your-Own-License (BYOL) in Cloud and On-Prem

License files for CloudEOS and vEOS

CloudEOS and vEOS license files are available to unlock performance limitations and enable IPSec.

Installing License Files

License files are files that are imported via the CLI. Contact your local SE for assistance in obtaining a license. Use the **`license import`** command to download a license file. Save the file to `/mnt/flash/` or a server. For example purposes, the licenses below are non-functional.

```
switch#license import flash:vEOSLic-1.json
switch#license import flash:IPSecLic-1.json
```

License files may also be imported via http. The following example illustrates the structure of the licence files import.

```
http:some-url/license.json
```

Verifying Installed License Files

Use the **`show license`** command to display details regarding the active licenses and device-specific information needed for licensing. For example purposes, the licenses below are non-functional.

```
switch#show license
Customer name: Arista Test Customer
System Serial number: 6FF552005130CB93A1048182A0FE585C
System MAC address: 5254.0062.ab2e
Domain name: Unknown
Platform: CloudEOS-KVM
License feature: IPSec
License parameter: None
Count: 1
Start: 2018-01-31 00:43:31
Expiration: 2026-12-30 16:00:00
Active: yes

License feature: CloudEOS - Virtualized EOS
Throughput: Not Throttled
Count: 1
Start: 2018-01-31 00:42:48
Expiration: 2026-12-30 16:00:00
Active: yes
```

Update License Files (Optional)

Use the **`license update`** command forces the system to evaluate the license files already present in the license store.

```
switch#license update
```

Obtaining and Installing Soft Expiry

Users can obtain license files from Arista that extend the time for which the customer can use a certain feature without any limitations. The license for the feature is considered expired, but the feature continues to work until the grace period as mentioned in the license file lapses.

For example, with a license file such as the one below, customer can continue to use vEOS without any limitations for ten days beyond expiry date.

```
{
        "LicenseFileVersion": "1.0",
        "CustomerName": "Arista Test Customer",
        "LicenseSerialNumber": "ARISTA-TEST-DAYSPAST1",
        "Signature": {
                "SigningCertPEM": "-----BEGIN CERTIFICATE-----7brkfssZDr
RIatxKEkv6Oc
\nh4kXO2mvvMJxQDf7VvGXEC3fSRURLwPz//6JMx942iOKsES8ZT9nT2q9MxJXfInn
\n3EcKGmPWKQR4n2qH
fmq6sfk2eFBUYIrZBm9RUbVbyLZLCOv2KxJ7FFZ9LV1jp5An\nAyHLJUMQqqw/kvUUvUq1bI/
PtEOlNc9Ndt
/3yeh+HByzIw8/f+gjKkUjQpVncuqS\nkFotBPNNj/LjbQD40R/tJ0z/8sPXCGJuo4mE9s/
MwnWmkAHxpZyC
ccMBlNp3LkJk\nFHcsVb36Vclv5XWDe5AxU+0sQjEB4LGP7nYo8wjjvSZIpYXRiAmDRGuAGi/
W/W3F\n6hEQ
661JK4KPJvoQsMqYaO/TkZPIXEAdgEDkmj0=\n-----END CERTIFICATE-----\n",
                "Hash": "f076d2cac1eac2a8261915e0b2ce4cb547e9c9
8bda070d001140daf3c3bd3694",
                "Signature": "304502201ca6fab964d8a3aade43d306232fcf
52b9503fc22f4552
d58fb5a95e1b9e13e6022100dff97ad4f37389b55887f0ec06c9ef29d55a7
5e668e4da654deaf8037633a9bd"
        },
        "Features": {
                "vEOS": [
                        {
                                "Count": 1,
                                "Value": "",
                                "Valid": {
                                        "NotBefore": "2000-01-01T0
0:00:00Z",
                                        "NotAfter": "2001-01-01T0
0:00:00Z"
                                },
                                "BehaviorModifier": {
                                        "DaysAllowedPastExpiration": 10
                                }
                        }
                ]
        },
        "BindingInfo": {
                "SystemMAC": "",
                "DomainAddress": "",
                "SerialNumber": "2BC6A772072B04BED43DCCF8777F036F"
        }
}



--
```

Additional Licensing Show Commands

The following CLIs can be used to verify if a license file is valid, when it expires, what license files are installed and any relevant information regarding a license. The **`show license`** commands do not list features that are unlocked by external license files or means.

Show License Files

Use the **`show license files`** command to display all information related to the active licenses installed. For example purposes, the licenses below are non-functional.

```
switch#show license files

License name:  2017.11.02.08.23.23.053684_IPSecLic-1yr.json
Contents:
{
    "BindingInfo": {
        "DomainAddress": "",
        "SerialNumber": "C3F3580316A92EE8D97DB70C967EAAA4",
        "SystemMAC": "02:9c:a8:a5:51:5a"
    },
    "CustomerName": "Arista Test",
    "Features": {
        "IPSec": [
            {
                "Count": 1,
                "Valid": {
                    "NotAfter": "2018-12-31T00:00:00Z",
                    "NotBefore": "2017-11-02T15:21:22Z"
                },
                "Value": ""
            }
        ]
    },
   (truncated)
}

License name:  2017.11.03.12.27.24.016515_vEOSLic-1234.json
Contents:
{
    "BindingInfo": {
        "DomainAddress": "",
        "SerialNumber": "C3F3580316A92EE8D97DB70C967EAAA4",
        "SystemMAC": ""
    },
    "CustomerName": "Arista Test",
    "Features": {
        "CloudEOS": [
            {
                "Count": 1,
                "Valid": {
                    "NotAfter": "2018-12-31T00:00:00Z",
                    "NotBefore": "2017-11-02T00:00:00Z"
                },
                "Value": ""
            }
        ]
    },
    "LicenseFileVersion": "1.0",
    (truncated)
END CERTIFICATE-----\n"
```

show license expired

The **`show license expired`** command will display the same as the **`show license`** command, but only displays expired license files.

```
switch#show license expired
System Serial number:  2BC6A772072B04BED43DCCF8777F036F
System MAC address:    06:1b:8a:48:8d:0c
Domain name:           Unknown

License feature:  IPSec
 License parameter:  None
 Count:              1
 Start:              2017-10-05 21:49:13
 Expiration:         2017-10-09 17:00:00
 Active:             expired


License feature:  CloudEOS  -  Virtualized EOS
 License parameter:  None
 Count:              1
 Start:              2017-10-05 21:47:34
 Expiration:         2017-10-09 17:00:00
 Active:             expired
```

show license all

The **`show license all`** command will display all license files that are active, expired or license files that have not yet been activated.

```
switch#show license all
System Serial number:  2BC6A772072B04BED43DCCF8777F036F
System MAC address:    06:1b:8a:48:8d:0c
Domain name:           Unknown

License feature:  IPSec
 License parameter:  None
 Count:              1
 Start:              2017-12-30 16:00:00
 Expiration:         2018-12-30 16:00:00
 Active:             in future

 License parameter:  None
 Count:              1
 Start:              2017-09-18 13:56:45
 Expiration:         2017-12-30 16:00:00
 Active:             yes

 License parameter:  None
 Count:              1
 Start:              2017-10-05 21:49:13
 Expiration:         2017-10-09 17:00:00
 Active:             expired


License feature:  CloudEOS  -  Virtualized EOS
 License parameter:  None
 Count:              1
 Start:              2017-10-08 17:00:00
 Expiration:         2017-12-30 16:00:00
 Active:             yes

 License parameter:  None
```

```
Count:              1
Start:              2017-12-30 16:00:00
Expiration:         2018-12-30 16:00:00
Active:             in future

License parameter:  None
Count:              1
Start:              2017-10-05 21:47:34
Expiration:         2017-10-09 17:00:00
Active:             expired
```

Chapter 3

Using CloudEOS and vEOS Router on the AWS Platform

The CloudEOS and vEOS Router, based on the Arista EOS, runs as a virtual machine instance on AWS EC2. Use the CloudEOS and vEOS Router to create the various types of virtual machine router instances for AWS deployment, for example, gateway routers and transit routers.

3.1 CloudEOS and vEOS Router Image Updates

The process to update CloudEOS and vEOS Router images is the standard update process used for EOS images.

For details on the steps to use, refer to the ***Arista EOS User Manual*** (see https://www.arista.com/en/support/product-documentation).

3.2 Amazon Machine Image (AMI) Specifications

The AMI provided by Arista utilizes the architecture, type of root device, virtualization type, and interface type required to configure the CloudEOS and vEOS Router for a robust AWS deployment.

The specifications of the Arista AMI are:

- **Architecture:** x86_64
- **Virtualization type:** HVM
- **Root Device Type:** EBS
- **Network Interface type:** SR-IOV, ENA (Elastic Network Adapter)

3.3 Methods for Launching CloudEOS and vEOS Router Instances

The CloudEOS and vEOS Router supports the use of various methods for launching router instances needed in a typical AWS deployment.

The supported methods are:

- Launching CloudEOS and vEOS Router Instances Using AWS CloudFormation
- Launching CloudEOS and vEOS Router Instances Using EC2 AWS Marketplace
- Network Configuration Tasks for CloudEOS and vEOS Router Instances
- Using User-data for Configuration of Entities and CloudEOS and vEOS Router Instances

3.3.1 Launching CloudEOS and vEOS Router Instances Using AWS CloudFormation

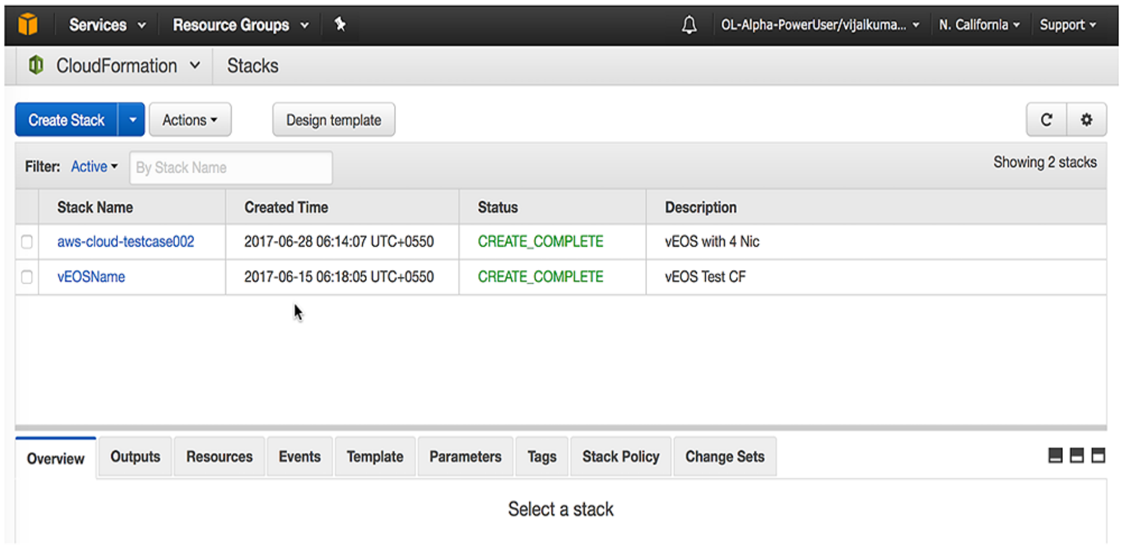
Using AWS CloudFormation to launch CloudEOS and vEOS Router instances involves creating a CloudFormation stack to use to launch the instance. The created stack provides the base configuration for the instance. As part of this task, select a stack template, which defines the base configuration of the instance.

Make sure to select the stack template that provides the resources required for the instances that are launching. Templates can be obtained from https://github.com/aristanetworks. For more information about AWS CloudFormation stacks and using stack templates, refer to the AWS documentation (see https://aws.amazon.com/documentation/cloudformation/).

Complete these steps to launch CloudEOS and vEOS Router instances using AWS CloudFormation.

1. Log in to the Amazon Management Console.
2. Choose **Services** > **CloudFormation**.

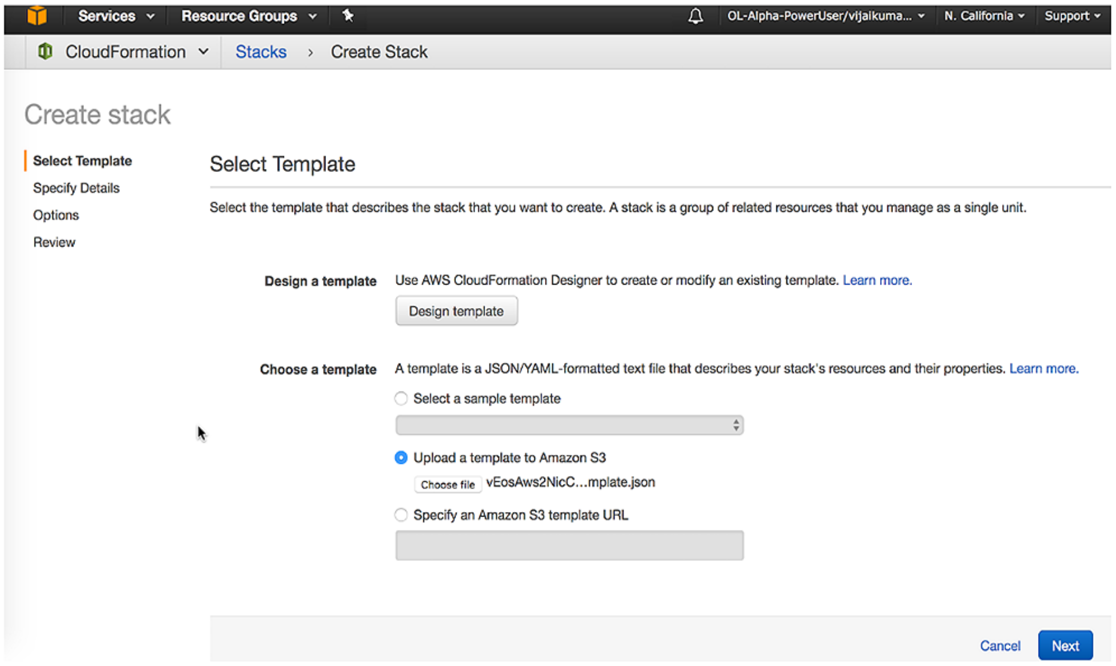
   The CloudFormation page appears showing the current stacks available to use.



Stack Name	Created Time	Status	Description
aws-cloud-testcase002	2017-06-28 06:14:07 UTC+0550	CREATE_COMPLETE	vEOS with 4 Nic
vEOSName	2017-06-15 06:18:05 UTC+0550	CREATE_COMPLETE	vEOS Test CF

3. Click on the **Create Stack** button.

   The page refreshes to show the templates that are available to use to create a new stack.



Select Template

Select the template that describes the stack that you want to create. A stack is a group of related resources that you manage as a single unit.

Design a template: Use AWS CloudFormation Designer to create or modify an existing template. Learn more.

Choose a template: A template is a JSON/YAML-formatted text file that describes your stack's resources and their properties. Learn more.

Select a sample template

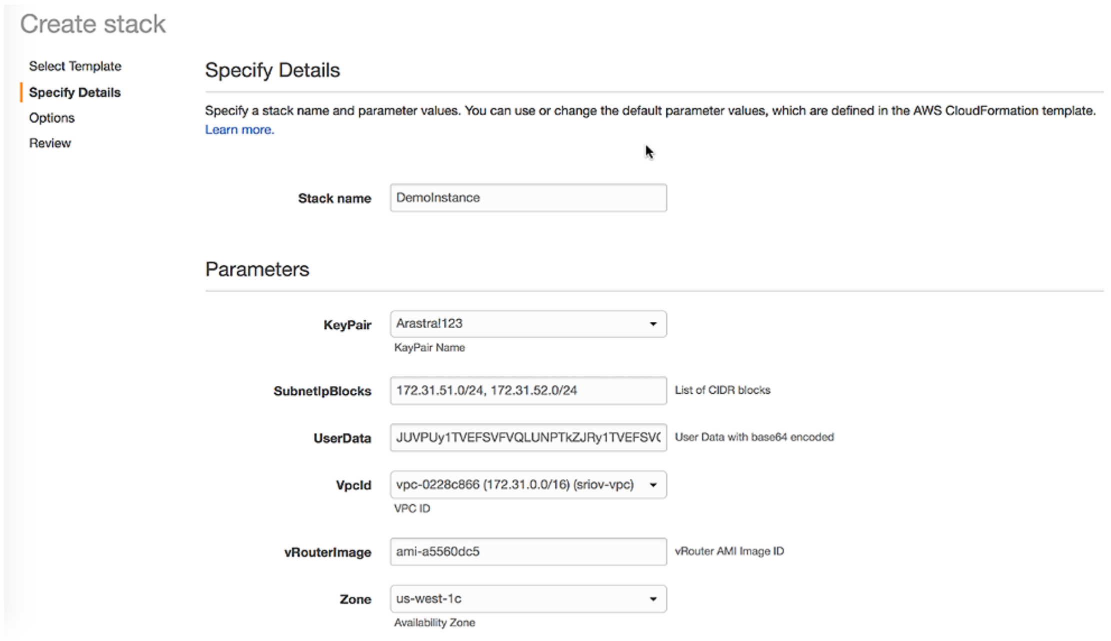
[✓ Upload a template to Amazon S3] Choose file vEosAws2NicC...mplate.json

Specify an Amazon S3 template URL

4. Select a **nic** template for upload, and then click on the **Next** button.

Note: Templates can be found in the docs directory. Press **Select** to choose the desired AMI.

The page refreshes showing the options for specifying the details for the stack.


Create stack

Select Template
Specify Details
Options
Review

Specify Details

Specify a stack name and parameter values. You can use or change the default parameter values, which are defined in the AWS CloudFormation template. Learn more.

Stack name: DemoInstance

Parameters

KeyPair: Arastra!123 (KayPair Name)

SubnetIpBlocks: 172.31.51.0/24, 172.31.52.0/24 (List of CIDR blocks)

UserData: JUVPUy1TVEFSVFVQLUNPTkZJRy1TVEFSV((User Data with base64 encoded)

VpcId: vpc-0228c866 (172.31.0.0/16) (sriov-vpc) (VPC ID)

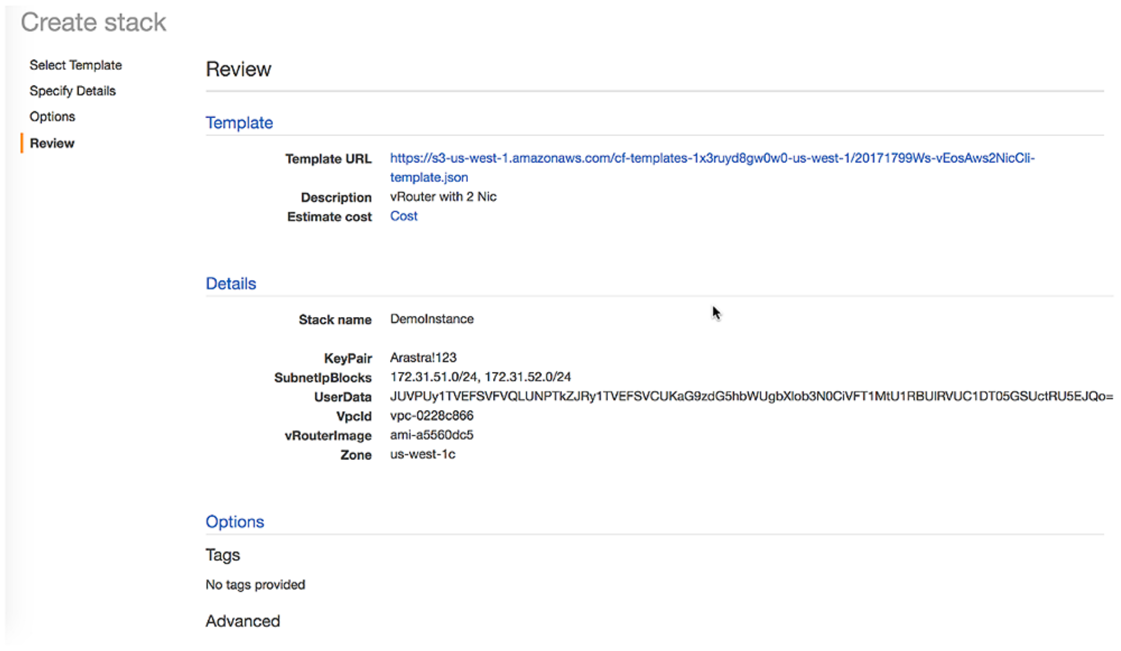
vRouterImage: ami-a5560dc5 (vRouter AMI Image ID)

Zone: us-west-1c (Availability Zone)

5. Enter the Stack Name, Subnet IP Block for each interface, VPC ID, KeyPair Name, UserData in base64 format, AMI ID. (To convert UserData from text to base64 format, use a base64 command on MacOS or Linux machine.)

```
# base64
%EOS-STARTUP-CONFIG-START%
hostname myhost
%EOS-STARTUP-CONFIG-END%
<Press CTRL+D>
JUVPUy1TVEFSVFVQLUNPTkZJRy1TVEFSVCUKaG9zdG5hbWUgbXlob3N0CiVFT1MtU
1RBU1RVUC1DT05GSUctRU5EJQo=
```

6. Review the details and make changes if needed.
7. Click the **Create** button to create the stack.



Create stack

- Select Template
- Specify Details
- Options
- Review

Review

Template

Template URL	https://s3-us-west-1.amazonaws.com/cf-templates-1x3ruyd8gw0w0-us-west-1/20171799Ws-vEosAws2NicCli-template.json
Description	vRouter with 2 Nic
Estimate cost	Cost

Details

Stack name	DemoInstance
KeyPair	Arastra!123
SubnetIpBlocks	172.31.51.0/24, 172.31.52.0/24
UserData	JUVPUy1TVEFSVFVQLUNPTkZJRy1TVEFSVCUKaG9zdG5hbWUgbXlob3N0CiVFT1MtU1RBUlRVUC1DT05GSUctRU5EJQo=
VpcId	vpc-0228c866
vRouterImage	ami-a5560dc5
Zone	us-west-1c

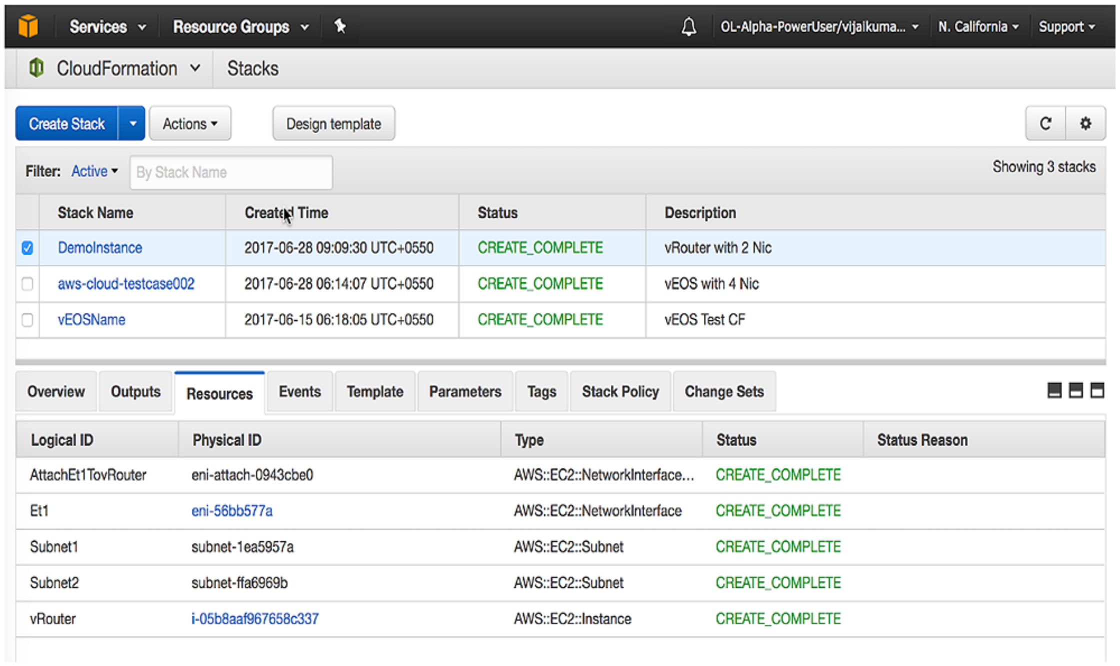
Options

Tags

No tags provided

Advanced

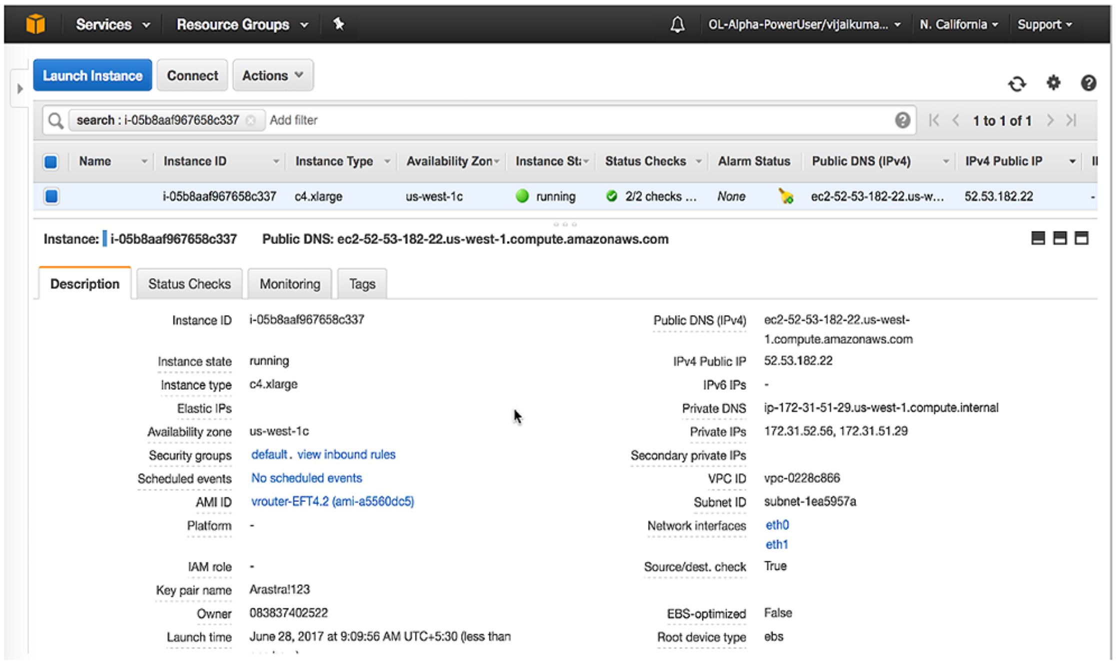
8. Wait for the stack creation to complete. Resources created as part of the stack creation process can be viewed in the Resource tab.



Stack Name	Created Time	Status	Description
DemoInstance	2017-06-28 09:09:30 UTC+0550	CREATE_COMPLETE	vRouter with 2 Nic
aws-cloud-testcase002	2017-06-28 06:14:07 UTC+0550	CREATE_COMPLETE	vEOS with 4 Nic
vEOSName	2017-06-15 06:18:05 UTC+0550	CREATE_COMPLETE	vEOS Test CF

Logical ID	Physical ID	Type	Status	Status Reason
AttachEt1TovRouter	eni-attach-0943cbe0	AWS::EC2::NetworkInterface...	CREATE_COMPLETE	
Et1	eni-56bb577a	AWS::EC2::NetworkInterface	CREATE_COMPLETE	
Subnet1	subnet-1ea5957a	AWS::EC2::Subnet	CREATE_COMPLETE	
Subnet2	subnet-ffa6969b	AWS::EC2::Subnet	CREATE_COMPLETE	
vRouter	i-05b8aaf967658c337	AWS::EC2::Instance	CREATE_COMPLETE	

9. Click on the CloudEOS and vEOS Router instance ID to view the status of CloudEOS and vEOS Router instance. The instance ID is shown in the Physical ID column of the Resources tab.



Name	Instance ID	Instance Type	Availability Zon	Instance St	Status Checks	Alarm Status	Public DNS (IPv4)	IPv4 Public IP
	i-05b8aaf967658c337	c4.xlarge	us-west-1c	running	2/2 checks ...	None	ec2-52-53-182-22.us-w...	52.53.182.22

Instance: i-05b8aaf967658c337 Public DNS: ec2-52-53-182-22.us-west-1.compute.amazonaws.com

Instance ID	i-05b8aaf967658c337	Public DNS (IPv4)	ec2-52-53-182-22.us-west-1.compute.amazonaws.com
Instance state	running	IPv4 Public IP	52.53.182.22
Instance type	c4.xlarge	IPv6 IPs	-
Elastic IPs		Private DNS	ip-172-31-51-29.us-west-1.compute.internal
Availability zone	us-west-1c	Private IPs	172.31.52.56, 172.31.51.29
Security groups	default. view inbound rules	Secondary private IPs	
Scheduled events	No scheduled events	VPC ID	vpc-0228c866
AMI ID	vrouter-EFT4.2 (ami-a5560dc5)	Subnet ID	subnet-1ea5957a
Platform	-	Network interfaces	eth0 eth1
IAM role	-	Source/dest. check	True
Key pair name	Arastra!123		
Owner	083837402522	EBS-optimized	False
Launch time	June 28, 2017 at 9:09:56 AM UTC+5:30 (less than	Root device type	ebs

Recommended Usage

AWS cannot auto-assign a public IPv4 address if an EC2 instance is launched or started from the stopped state with multiple network interfaces attached to it. In such cases, the user cannot connect to the instance over IPv4 unless an Elastic IP address is assigned to the primary network interface (eth0). If the user does not want to associate an Elastic IP address with the CloudEOS and vEOS Router instance, then it is recommended to attach any additional interface only when the instance is in running state and never to stop and start your instance from thereon. The user may reboot the instance either from AWS console or from within CloudEOS and vEOS Router using the CLI or bash commands because the instance reboot does not cause the public IPv4 address to be released as opposed to instance stop. To associate Elastic IP address to your instance or primary network interface, refer to https://docs.aws.amazon.com/AWSEC2/latest/UserGuide/elastic-ip-addresses-eip.html

3.3.2 Launching CloudEOS and vEOS Router Instances Using EC2 AWS Marketplace

Launching CloudEOS and vEOS Router instances using the EC2 AWS Marketplace gives the ability to create and configure CloudEOS and vEOS Router instances in the VPCs of your AWS deployment. This method utilizes Amazon Machine Images (AMIs) to configure the operating system of the instance. Obtain the AMI needed for the instance from the AWS Marketplace. This task involves creating an EC2 key pair, selecting the AMI to configure the operating system of the instance, selecting the instance type, and if needed, configuring advanced details (options) for the instance.

Available Options

During this configuration procedure, choose to configure some options to take advantage of certain features. These optional configuration items are:

- **Assigning an IAM role to the instance**

  To enable AWS services on the instance (for example, AWS CloudWatch logs) assign an IAM role to the instance during this procedure. Assign an IAM role to the instance by:

  - Selecting an existing IAM role.

- Creating a new IAM role (an option is provided as part of the procedure to create a new IAM role).

  Refer to the following AWS documentation for details about creating EC2 key pairs and creating IAM roles:
- Creating EC2 key pairs (https://docs.aws.amazon.com/AWSEC2/latest/UserGuide/ec2-key-pairs.html).
- Creating an IAM role (https://docs.aws.amazon.com/AmazonCloudWatch/latest/logs/QuickStartEC2Instance.html).

- **Using instance user-data to configure the instance**

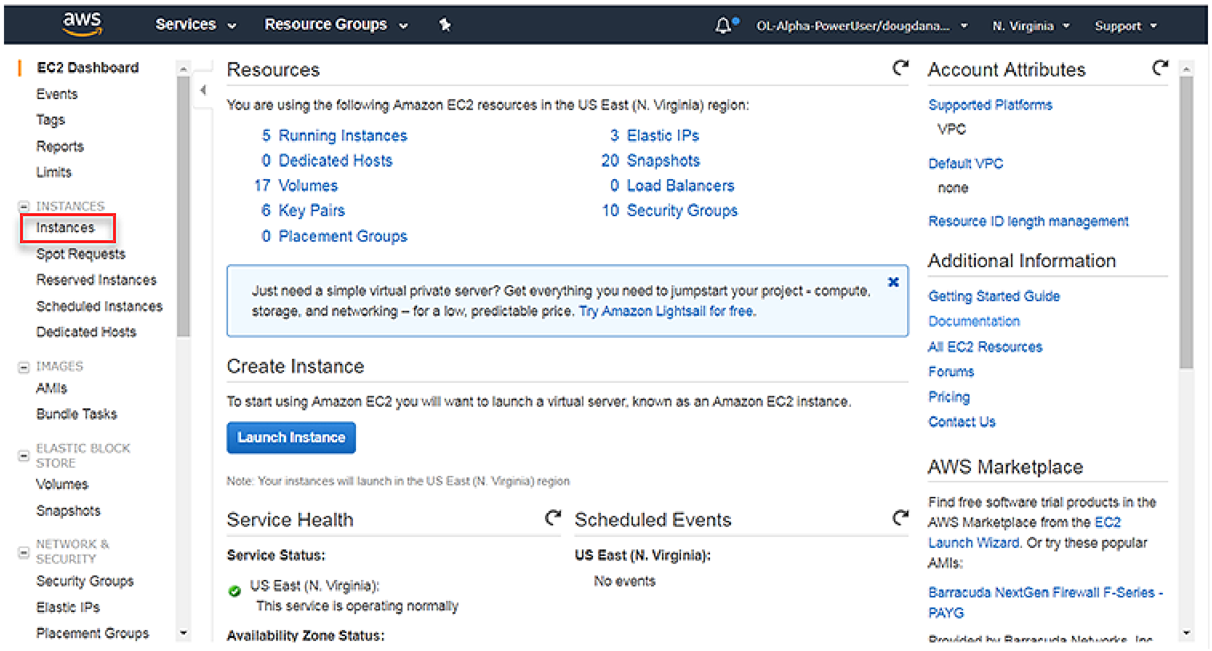
  CloudEOS and vEOS supports the use of CloudEOS and vEOS Router instance user-data to configure CloudEOS and vEOS Router instances at launch. This involves uploading instance user-data to the instance by way of the Advanced Details dialog. There is an option of copying and pasting a configuration into the dialog or attaching a configuration file.

  For details on composing user data for CloudEOS and vEOS Router, see Using User-data for Configuration of Entities and CloudEOS and vEOS Router Instances.

Complete the following steps to launch a CloudEOS and vEOS Router instances.

1. Log in to the Amazon Management Console.
2. Create an EC2 key pair and download the *.pem* file that contains the private key. (The *.pem* file may download automatically.)

   To create an EC2 pair, go to https://docs.aws.amazon.com/AWSEC2/latest/UserGuide/ec2-key-pairs.html.
3. Go to the EC2 Dashboard.



Resources

You are using the following Amazon EC2 resources in the US East (N. Virginia) region:

5 Running Instances	3 Elastic IPs
0 Dedicated Hosts	20 Snapshots
17 Volumes	0 Load Balancers
6 Key Pairs	10 Security Groups
0 Placement Groups	

Just need a simple virtual private server? Get everything you need to jumpstart your project - compute, storage, and networking – for a low, predictable price. Try Amazon Lightsail for free.

Create Instance

To start using Amazon EC2 you will want to launch a virtual server, known as an Amazon EC2 instance.

Note: Your instances will launch in the US East (N. Virginia) region

Service Health

Service Status:

US East (N. Virginia):
This service is operating normally

Availability Zone Status:

Scheduled Events

US East (N. Virginia):

No events

Account Attributes

Supported Platforms
VPC

Default VPC
none

Resource ID length management

Additional Information

Getting Started Guide
Documentation
All EC2 Resources
Forums
Pricing
Contact Us

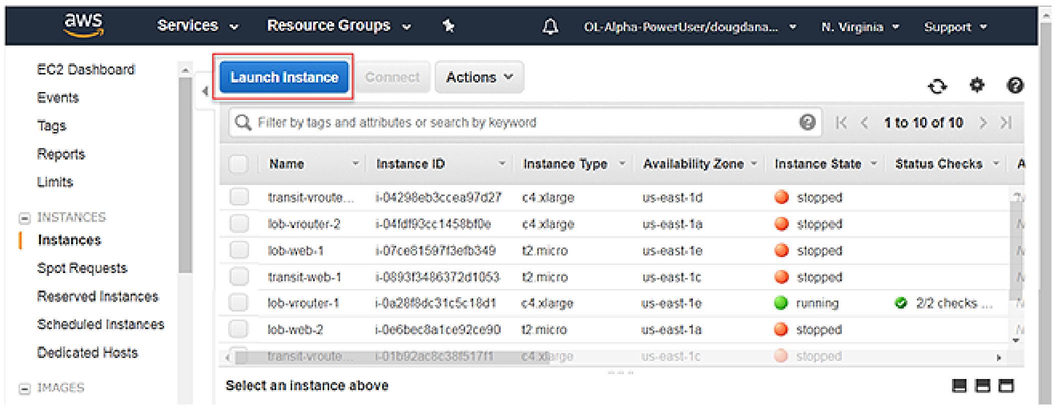
AWS Marketplace

Find free software trial products in the AWS Marketplace from the EC2 Launch Wizard. Or try these popular AMIs:

Barracuda NextGen Firewall F-Series - PAYG

4. From the EC2 Dashboard, click **Instances** in the left pane.

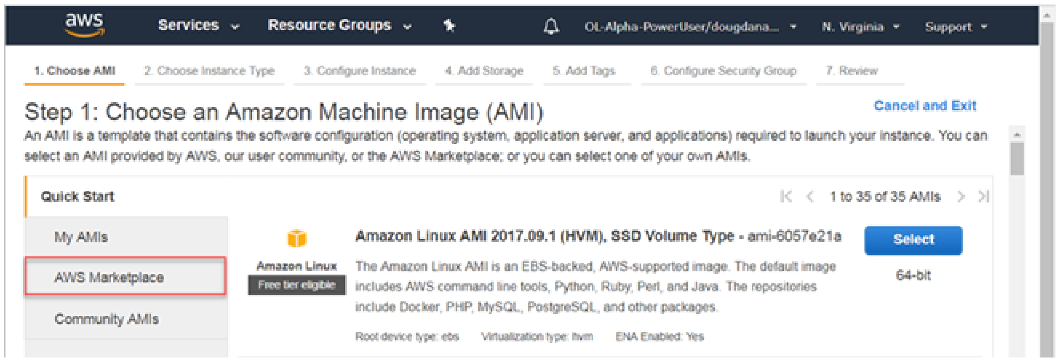
   The Launch Instance page appears.



Name	Instance ID	Instance Type	Availability Zone	Instance State	Status Checks
transit-vroute...	i-04298eb3ccea97d27	c4.xlarge	us-east-1d	stopped	
lob-vrouter-2	i-04fdf93cc1458bf0e	c4.xlarge	us-east-1a	stopped	
lob-web-1	i-07ce81597f3efb349	t2.micro	us-east-1e	stopped	
transit-web-1	i-0893f3486372d1053	t2.micro	us-east-1c	stopped	
lob-vrouter-1	i-0a28f8dc31c5c18d1	c4.xlarge	us-east-1e	running	2/2 checks ...
lob-web-2	i-0e6bec8a1ce92ce90	t2.micro	us-east-1a	stopped	
transit-vroute...	i-01b92ac8c38f517f1	c4.xlarge	us-east-1c	stopped	

5. Click on the **Launch Instance** button.

   The page appears for you to select an AMI.



Step 1: Choose an Amazon Machine Image (AMI)

An AMI is a template that contains the software configuration (operating system, application server, and applications) required to launch your instance. You can select an AMI provided by AWS, our user community, or the AWS Marketplace; or you can select one of your own AMIs.

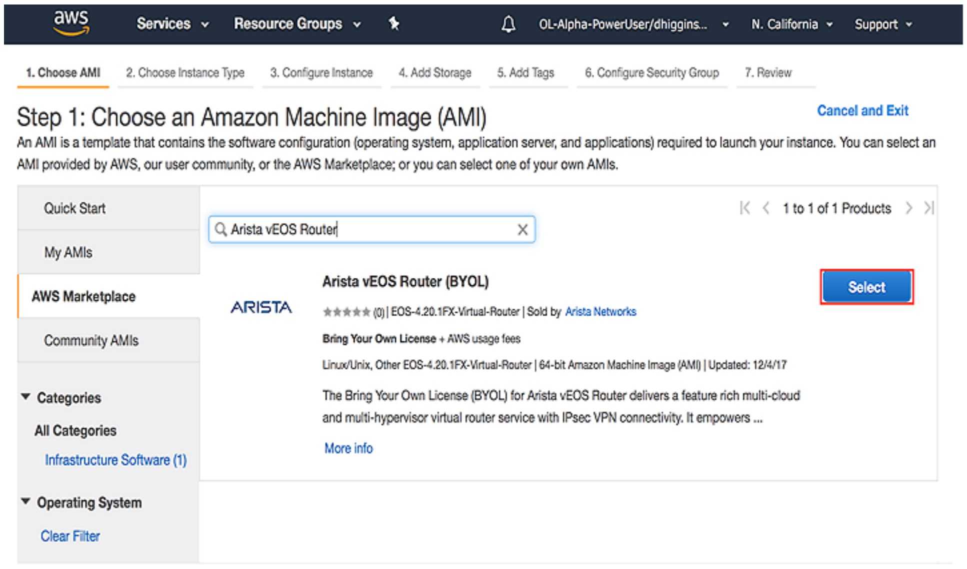
Amazon Linux AMI 2017.09.1 (HVM), SSD Volume Type - ami-6057e21a

The Amazon Linux AMI is an EBS-backed, AWS-supported image. The default image includes AWS command line tools, Python, Ruby, Perl, and Java. The repositories include Docker, PHP, MySQL, PostgreSQL, and other packages.

Root device type: ebs Virtualization type: hvm ENA Enabled: Yes

6. Click on **AWS Marketplace** in the left pane.

   Search for **Arista CloudEOS and vEOS Router** in the search field to bring up the available CloudEOS and vEOS AMIs to use. Select the appropriate AMI for launching.



Step 1: Choose an Amazon Machine Image (AMI)

An AMI is a template that contains the software configuration (operating system, application server, and applications) required to launch your instance. You can select an AMI provided by AWS, our user community, or the AWS Marketplace; or you can select one of your own AMIs.

Arista vEOS Router (BYOL)

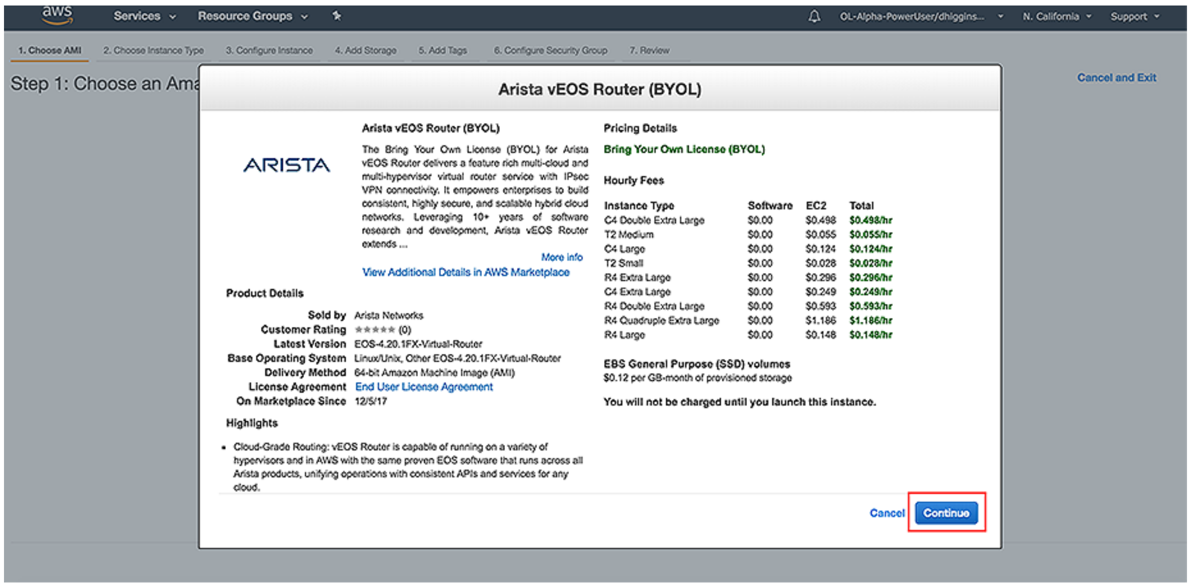
★★★★★ (0) | EOS-4.20.1FX-Virtual-Router | Sold by Arista Networks

Bring Your Own License + AWS usage fees

Linux/Unix, Other EOS-4.20.1FX-Virtual-Router | 64-bit Amazon Machine Image (AMI) | Updated: 12/4/17

The Bring Your Own License (BYOL) for Arista vEOS Router delivers a feature rich multi-cloud and multi-hypervisor virtual router service with IPsec VPN connectivity. It empowers ...

7. A screen appears showing the user highlights, pricing details and instance types available. Press the Continue button to advance.



Step 1: Choose an Ama

Arista vEOS Router (BYOL)

Arista vEOS Router (BYOL)

The Bring Your Own License (BYOL) for Arista vEOS Router delivers a feature rich multi-cloud and multi-hypervisor virtual router service with IPsec VPN connectivity. It empowers enterprises to build consistent, highly secure, and scalable hybrid cloud networks. Leveraging 10+ years of software research and development, Arista vEOS Router extends ...

More info

View Additional Details in AWS Marketplace

Product Details

Sold by	Arista Networks
Customer Rating	★★★★★ (0)
Latest Version	EOS-4.20.1FX-Virtual-Router
Base Operating System	Linux/Unix, Other EOS-4.20.1FX-Virtual-Router
Delivery Method	64-bit Amazon Machine Image (AMI)
License Agreement	End User License Agreement
On Marketplace Since	12/5/17

Highlights

- Cloud-Grade Routing: vEOS Router is capable of running on a variety of hypervisors and in AWS with the same proven EOS software that runs across all Arista products, unifying operations with consistent APIs and services for any cloud.

Pricing Details

Bring Your Own License (BYOL)

Hourly Fees

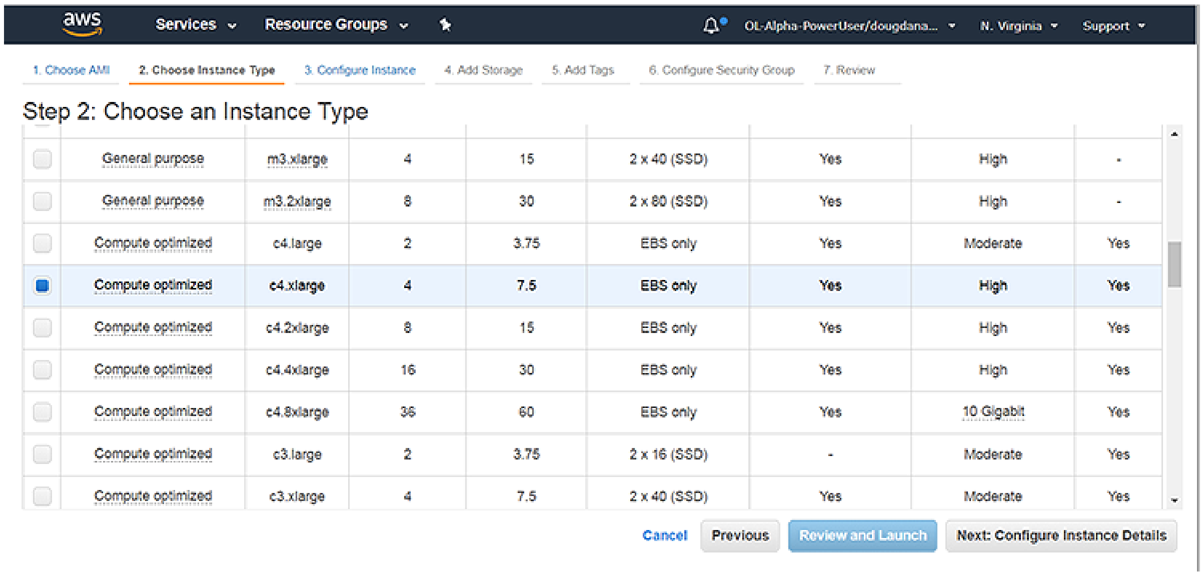
Instance Type	Software	EC2	Total
C4 Double Extra Large	$0.00	$0.498	$0.498/hr
T2 Medium	$0.00	$0.055	$0.055/hr
C4 Large	$0.00	$0.124	$0.124/hr
T2 Small	$0.00	$0.028	$0.028/hr
R4 Extra Large	$0.00	$0.296	$0.296/hr
C4 Extra Large	$0.00	$0.249	$0.249/hr
R4 Double Extra Large	$0.00	$0.593	$0.593/hr
R4 Quadruple Extra Large	$0.00	$1.186	$1.186/hr
R4 Large	$0.00	$0.148	$0.148/hr

EBS General Purpose (SSD) volumes
$0.12 per GB-month of provisioned storage

You will not be charged until you launch this instance.

8. Click in the left pane.

The Choose an Instance Type page appears.



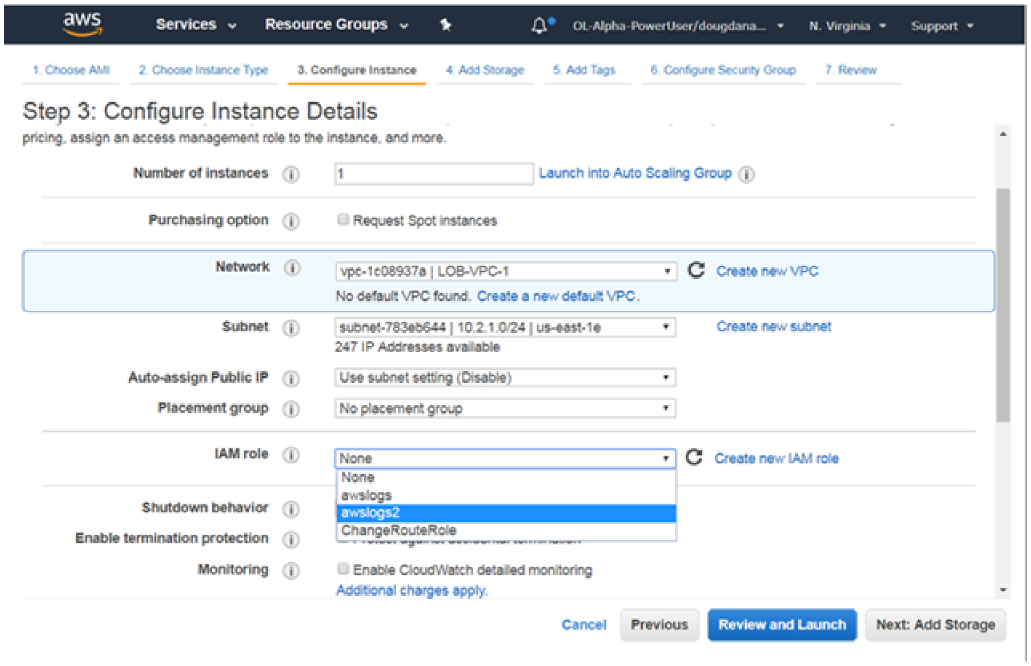
Step 2: Choose an Instance Type

☐	General purpose	m3.xlarge	4	15	2 x 40 (SSD)	Yes	High	-
☐	General purpose	m3.2xlarge	8	30	2 x 80 (SSD)	Yes	High	-
☐	Compute optimized	c4.large	2	3.75	EBS only	Yes	Moderate	Yes
■	Compute optimized	c4.xlarge	4	7.5	EBS only	Yes	High	Yes
☐	Compute optimized	c4.2xlarge	8	15	EBS only	Yes	High	Yes
☐	Compute optimized	c4.4xlarge	16	30	EBS only	Yes	High	Yes
☐	Compute optimized	c4.8xlarge	36	60	EBS only	Yes	10 Gigabit	Yes
☐	Compute optimized	c3.large	2	3.75	2 x 16 (SSD)	-	Moderate	Yes
☐	Compute optimized	c3.xlarge	4	7.5	2 x 40 (SSD)	Yes	Moderate	Yes

9. Select an instance type that meets the requirements for the CloudEOS and vEOS Router instance.

10. Click on the **Next: Configure Instance Details** button (lower right part of the page).

The Configure Instance Details page appears.



Step 3: Configure Instance Details

pricing, assign an access management role to the instance, and more.

Number of instances	1 — Launch into Auto Scaling Group
Purchasing option	Request Spot instances
Network	vpc-1c08937a | LOB-VPC-1 — Create new VPC No default VPC found. Create a new default VPC.
Subnet	subnet-783eb644 | 10.2.1.0/24 | us-east-1e — Create new subnet 247 IP Addresses available
Auto-assign Public IP	Use subnet setting (Disable)
Placement group	No placement group
IAM role	None — Create new IAM role None awslogs awslogs2 ChangeRouteRole
Shutdown behavior	
Enable termination protection	
Monitoring	Enable CloudWatch detailed monitoring Additional charges apply.

11. (Optional) Create a new IAM role or select an existing IAM role. (This is required to enable AWS services on the instance, for example, AWS CloudWatch logs.)

12. (Optional) To configure advanced details for the instance, scroll down to the bottom of the page and click on the **Advanced Details** button.

The Advanced Details dialog appears. You use the dialog to upload user-data to configure the instance.

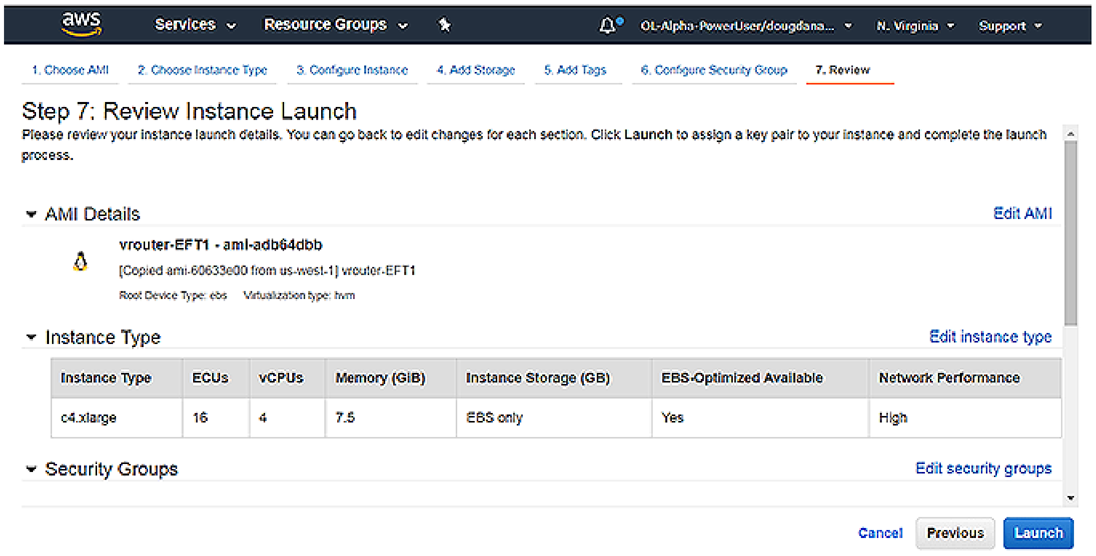
Do one of the following to configure the instance using user-data:

- Choose the **Text** option, and then copy-and-paste **startup-config** in the text box.
- Attach the configuration as a file by **clicking on the file**, and then choose the configuration file to be uploaded.

For details on composing user data for CloudEOS and vEOS Router, see Using User-data for Configuration of Entities and CloudEOS and vEOS Router Instances.

13. From the Configure Instance Details page, click the **Review and Launch** button.

The Review Instance Launch page appears.



Step 7: Review Instance Launch

Please review your instance launch details. You can go back to edit changes for each section. Click Launch to assign a key pair to your instance and complete the launch process.

AMI Details — Edit AMI

vrouter-EFT1 - ami-adb64dbb

[Copied ami-60633e00 from us-west-1] vrouter-EFT1

Root Device Type: ebs Virtualization type: hvm

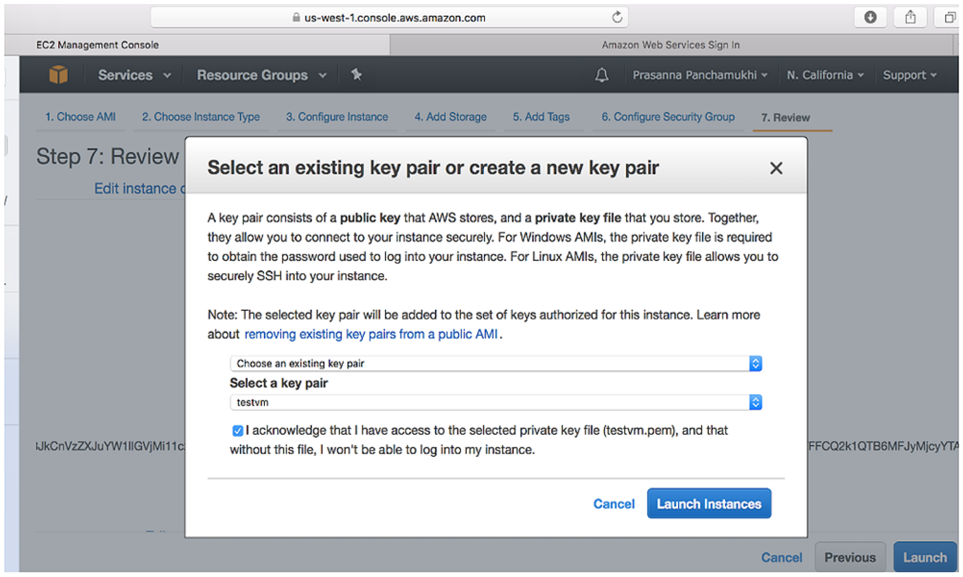
Instance Type — Edit instance type

Instance Type	ECUs	vCPUs	Memory (GiB)	Instance Storage (GB)	EBS-Optimized Available	Network Performance
c4.xlarge	16	4	7.5	EBS only	Yes	High

Security Groups — Edit security groups

14. Click on the **Launch** button.

    A dialog appears for selecting a key pair.



Select an existing key pair or create a new key pair

A key pair consists of a **public key** that AWS stores, and a **private key file** that you store. Together, they allow you to connect to your instance securely. For Windows AMIs, the private key file is required to obtain the password used to log into your instance. For Linux AMIs, the private key file allows you to securely SSH into your instance.

Note: The selected key pair will be added to the set of keys authorized for this instance. Learn more about removing existing key pairs from a public AMI.

Choose an existing key pair

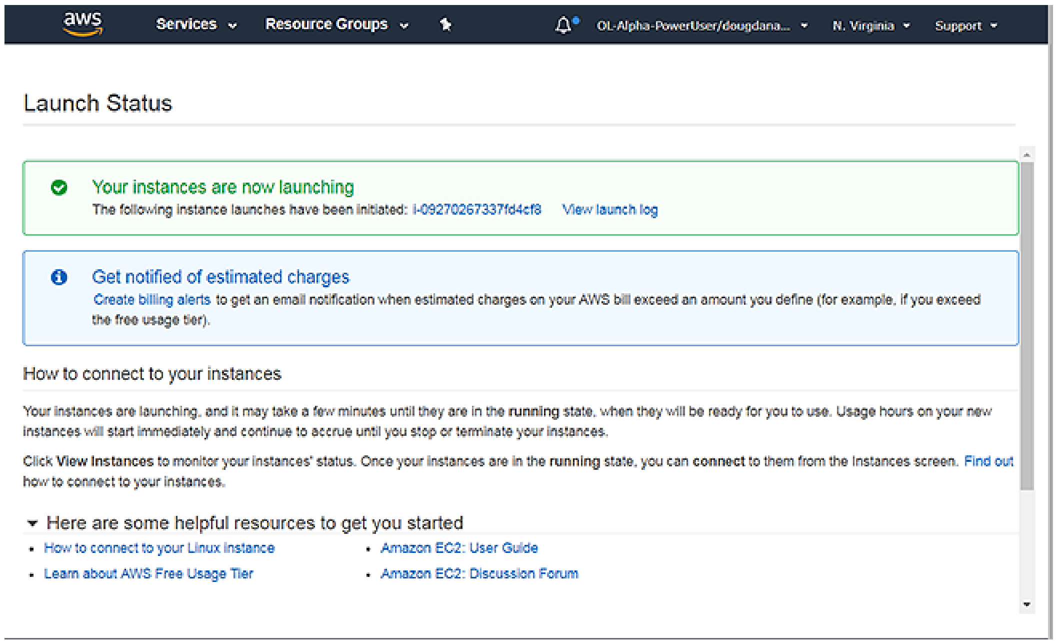
Select a key pair

testvm

☑ I acknowledge that I have access to the selected private key file (testvm.pem), and that without this file, I won't be able to log into my instance.

15. Using the **Select a key pair** menu, select the key pair created earlier in the procedure. In this example, the key pair is named "systest."
16. Select the acknowledgment (near the bottom of the dialog), and then click on the **Launch Instances** button.

    The Launch Status page appears showing the status of the instance.



Launch Status

Your instances are now launching

The following instance launches have been initiated: i-09270267337fd4cf8 View launch log

Get notified of estimated charges

Create billing alerts to get an email notification when estimated charges on your AWS bill exceed an amount you define (for example, if you exceed the free usage tier).

How to connect to your instances

Your instances are launching, and it may take a few minutes until they are in the **running** state, when they will be ready for you to use. Usage hours on your new instances will start immediately and continue to accrue until you stop or terminate your instances.

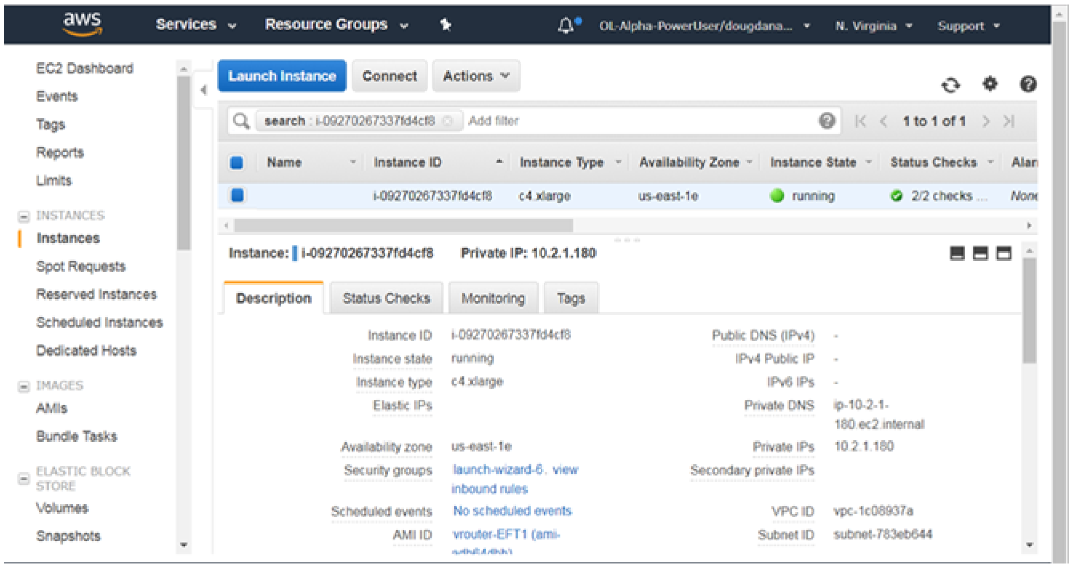
Click **View Instances** to monitor your instances' status. Once your instances are in the **running** state, you can **connect** to them from the Instances screen. Find out how to connect to your instances.

Here are some helpful resources to get you started

- How to connect to your Linux instance
- Learn about AWS Free Usage Tier
- Amazon EC2: User Guide
- Amazon EC2: Discussion Forum

17. Click on the blue link to the instance to view details about the instance. (The link is in the "Your instances are now launching" box near the top of the page.)

    The page shows the details for the instance.



search : i-09270267337fd4cf8

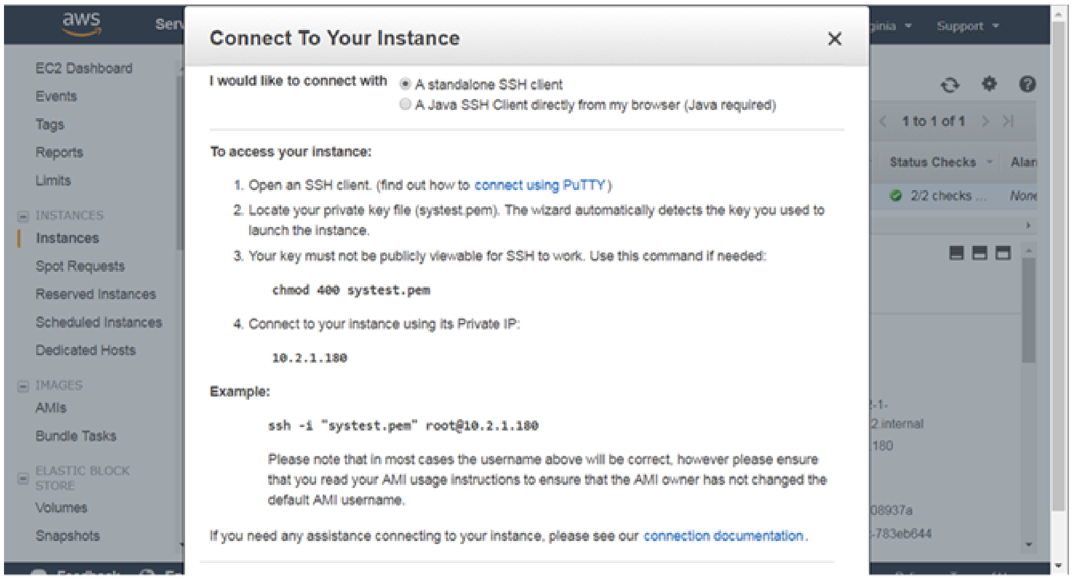
Name	Instance ID	Instance Type	Availability Zone	Instance State	Status Checks	Alarm
	i-09270267337fd4cf8	c4.xlarge	us-east-1e	running	2/2 checks ...	None

Instance: i-09270267337fd4cf8 Private IP: 10.2.1.180

Instance ID	i-09270267337fd4cf8	Public DNS (IPv4)	-
Instance state	running	IPv4 Public IP	-
Instance type	c4.xlarge	IPv6 IPs	-
Elastic IPs		Private DNS	ip-10-2-1-180.ec2.internal
Availability zone	us-east-1e	Private IPs	10.2.1.180
Security groups	launch-wizard-6. view inbound rules	Secondary private IPs	
Scheduled events	No scheduled events	VPC ID	vpc-1c08937a
AMI ID	vrouter-EFT1 (ami-	Subnet ID	subnet-783eb644

18. Make sure the Instance State shows **running**. Wait for the status to update to **running**.
19. (Optional) To use the existing subnet and security group for the instance, record the subnet and security group. This information is required when configuring the network interfaces to be attached to the instance.
20. (Optional) Click on the **Connect** button near the top of the page.

    The Connect to Your Instance dialog appears.



I would like to connect with
(● A standalone SSH client) ○ A Java SSH Client directly from my browser (Java required)

To access your instance:

1. Open an SSH client. (find out how to connect using PuTTY)
2. Locate your private key file (systest.pem). The wizard automatically detects the key you used to launch the instance.
3. Your key must not be publicly viewable for SSH to work. Use this command if needed:

```
chmod 400 systest.pem
```

4. Connect to your instance using its Private IP:

```
10.2.1.180
```

Example:

```
ssh -i "systest.pem" root@10.2.1.180
```

Please note that in most cases the username above will be correct, however please ensure that you read your AMI usage instructions to ensure that the AMI owner has not changed the default AMI username.

If you need any assistance connecting to your instance, please see our connection documentation.

21. Connect to the instance using the public or private IP address of the instance. The correct syntax is: **`ssh -i <privateKey.pem> ec2-user@10.2.1.180`**

    Example:

```
#ssh -i <privateKey.pem> ec2-user@10.2.1.180
```

Complete the networking tasks for the CloudEOS and vEOS Router instances in the gateway topology (see Network Configuration Tasks for CloudEOS and vEOS Router Instances).

- **Configuring the AWS CloudWatch Logs Agent**

3.3.2.1 Configuring the AWS CloudWatch Logs Agent

The AWS CloudWatch Logs Agent is the mechanism that publishes CloudEOS and vEOS Router logs to AWS CloudWatch. Configuring the AWS CloudWatch Logs Agent ensures that the CloudEOS and vEOS Router logs published to AWS CloudWatch conform to the selected requirements. The AWS CloudWatch Logs Agent is packaged with the *awslogs.swix* CloudEOS and vEOS extension, which is installed and enabled by default when the CloudEOS and vEOS Router instances launch through the AWS Marketplace.

Refer to the "AWS CloudWatch Quick Start Guide" to make sure that the CloudEOS and vEOS Router instance has the right credentials for logging in to AWS.

Note: To manually install or uninstall the ***awslogs.swix*** CloudEOS and vEOS extension, see https://eos.arista.com/packaging-and-installing-eos-extensions/. To obtain the ***awslogs.swix*** CloudEOS and vEOS extension, contact Arista TAC if required.

Where to find CloudEOS and vEOS Router logs

The location where CloudEOS and vEOS Router logs are published to depends on the AWS CloudWatch Logs configuration. By default, the logs are located under **CloudWatch**, "**log group**, name ***CloudEOS and vEOSlogs***.

Modifying AWS log configuration

Modify the AWS log configuration by:

- Editing configuration files under the **`/mnt/flash/awslogs/`** directory.
- Passing instance user-data. Make sure to use the correct start and end markers, which are:

```
%AWSLOGS-CONFIG-START%
 #configuration here
 %AWSLOGS-CONFIG-END%
 %AWS-PROXY-START%
 #configuration here
 %AWS-PROXY-END%
```

Note: Restart awslogs using ***sudo systemctl restart awslogs*** under bash. The reconfiguration does not take effect until awslogs restarts.

CloudEOS and vEOS Router log filenames

By default, the hostname of the CloudEOS and vEOS Router instance is the filename of all CloudEOS and vEOS Router logs for that instance.

3.3.3 Network Configuration Tasks for CloudEOS and vEOS Router Instances

Complete additional configuration tasks to ensure that the CloudEOS and vEOS Router instances launched have the required networking configuration. The configuration tasks include creating the additional network interfaces required by the topology, attaching the new interfaces to CloudEOS and vEOS Router instances, and configuring the route table of the AWS Specific Cloud Router.

- Creating the Additional Network Interfaces
- Attaching the New Network Interfaces to Instances
- Configuring the Route Table of the AWS Router

3.3.3.1 Creating the Additional Network Interfaces

Creating the additional network interfaces required for the topology ensures that there are interfaces available to attach to CloudEOS and vEOS Router instances. When creating the new network interfaces, there is the option of using the subnet and security groups that were automatically assigned to the instance, or specify a different subnet and security groups for the instance.

Pre-requisites:

To use the existing subnet and security group for the CloudEOS and vEOS Router instance, make sure to have the following information:

- Subnet ID
- Names of the security groups

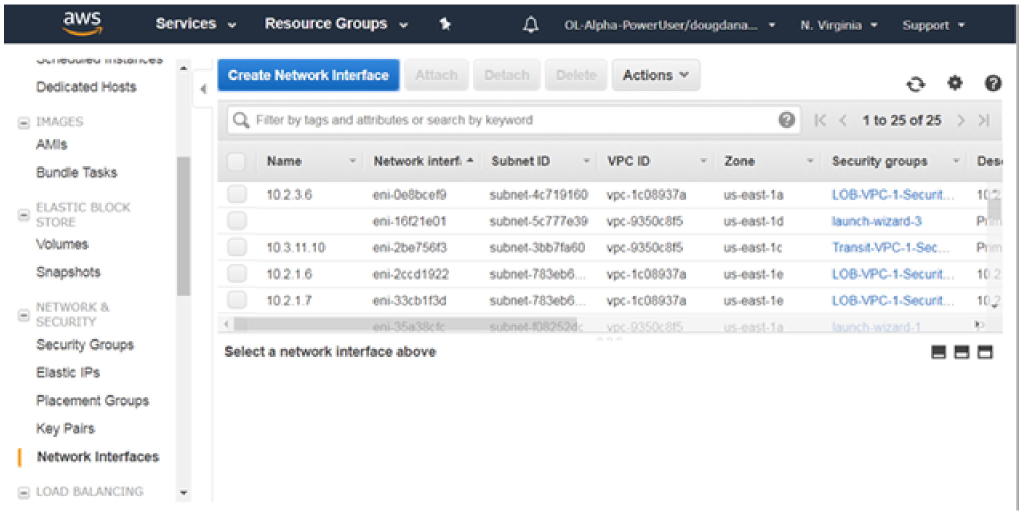
Obtain this information by viewing the instance details.

Procedure

Complete these steps to create network interfaces.

1. Go to the EC2 Dashboard.
2. In the NETWORK & SECURITY menu on the left part of the page, select **Network Interfaces**.

   The page refreshes to show all of the current network interfaces.

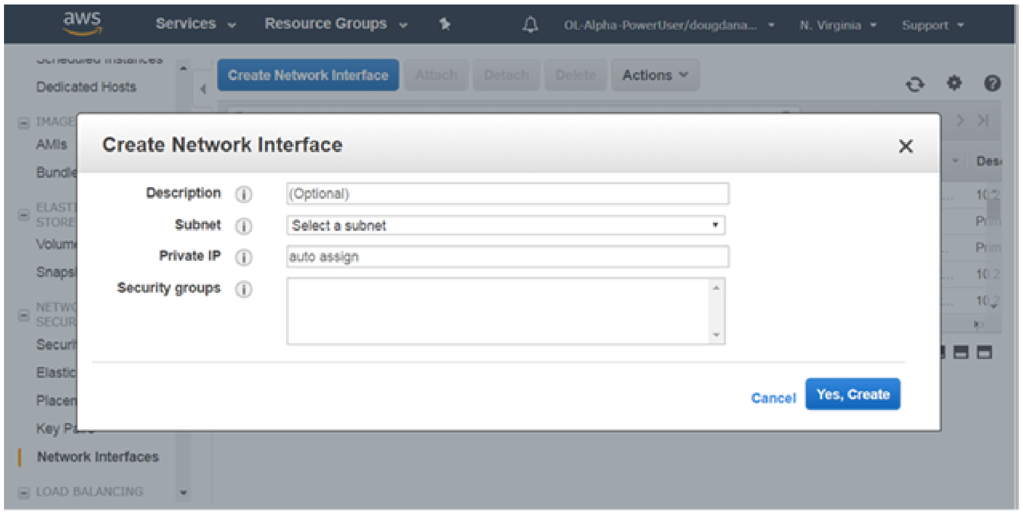


Name	Network interf.	Subnet ID	VPC ID	Zone	Security groups	Des
10.2.3.6	eni-0e8bcef9	subnet-4c719160	vpc-1c08937a	us-east-1a	LOB-VPC-1-Securit...	10.2
	eni-16f21e01	subnet-5c777e39	vpc-9350c8f5	us-east-1d	launch-wizard-3	Pri
10.3.11.10	eni-2be756f3	subnet-3bb7fa60	vpc-9350c8f5	us-east-1c	Transit-VPC-1-Sec...	Prim
10.2.1.6	eni-2ccd1922	subnet-783eb6...	vpc-1c08937a	us-east-1e	LOB-VPC-1-Securit...	10.2
10.2.1.7	eni-33cb1f3d	subnet-783eb6...	vpc-1c08937a	us-east-1e	LOB-VPC-1-Securit...	10.2
	eni-35a38cfc	subnet-f08252dc	vpc-9350c8f5	us-east-1a	launch-wizard-1	

Select a network interface above

3. Select the **Create Network Interface** button.

   The **Create Network Interface** dialog appears.



Create Network Interface

Description	(Optional)
Subnet	Select a subnet
Private IP	auto assign
Security groups	

Cancel   Yes, Create

4. Do the following:

   a. Enter a **description** for the network interface.

   b. Select the **subnet** for the network interface. (This can be the existing subnet for the CloudEOS and vEOS Router instance or a different subnet.)

   c. Type the **names of the security groups** for the network interface. (Specify the existing security groups for the CloudEOS and vEOS Router instance, or different security groups.)

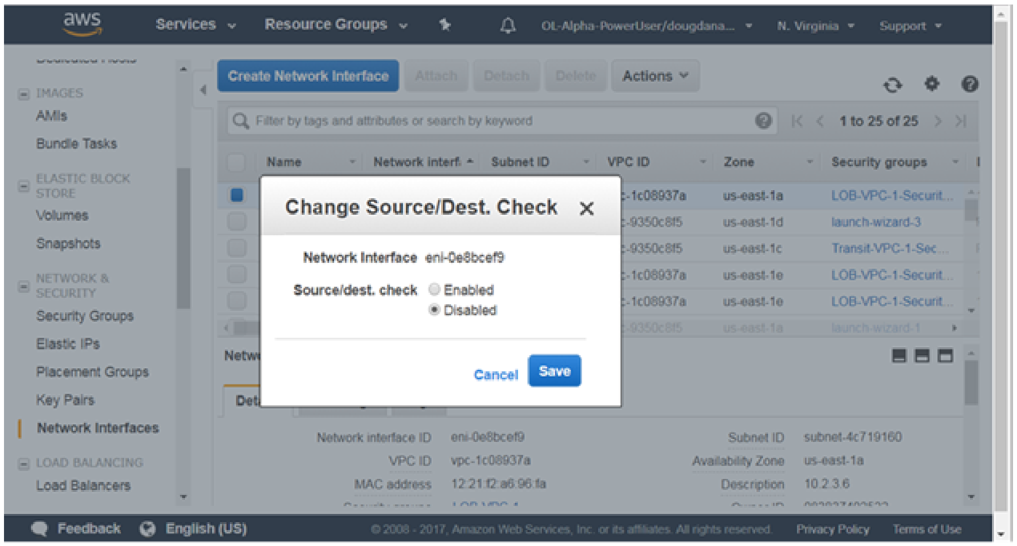
5. Select the **Yes, Create** button.

The new network interface is added to the list of interfaces on the page.

6. Repeat steps 3 through 5 to create additional interfaces as needed.
7. For each network interface created, complete steps **a** and **b**:

   a. Select the interface, then choose **Actions** > **Change Source/Dest Check**.

      The **Change Source/Dest Check** dialog appears showing the selected name of the network interface.



Change Source/Dest. Check

Network Interface eni-0e8bcef9

Source/dest. check Enabled [✓ Disabled]

Network interface ID eni-0e8bcef9
VPC ID vpc-1c08937a
MAC address 12:21:f2:a6:96:fa
Subnet ID subnet-4c719160
Availability Zone us-east-1a
Description 10.2.3.6

   b. Select the **Disabled** option, then click on the **Save** button.

Attach the new network interface to a CloudEOS and vEOS Router instance (see Attaching the New Network Interfaces to Instances).

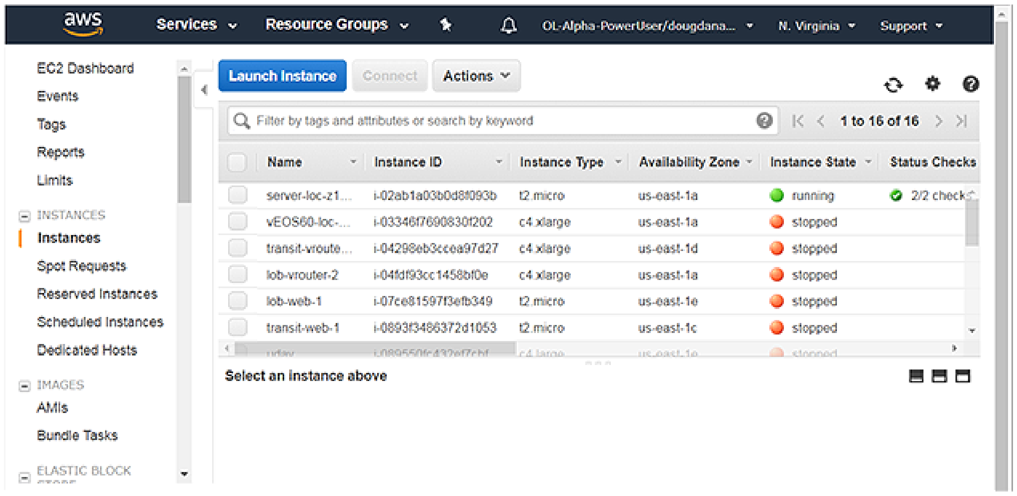
3.3.3.2 Attaching the New Network Interfaces to Instances

Attaching the new network interfaces to CloudEOS and vEOS Router instances is the second networking configuration task. This task involves selecting the new network interfaces created in the previous procedure and then attaching the interfaces to CloudEOS and vEOS Router instances.

Complete these steps to attach the new network interfaces to CloudEOS and vEOS Router instances.

1. Go to the EC2 Dashboard.
2. Open the INSTANCES menu on the left side of the page, then click **Instances**.

   The page lists all of the current network interfaces.

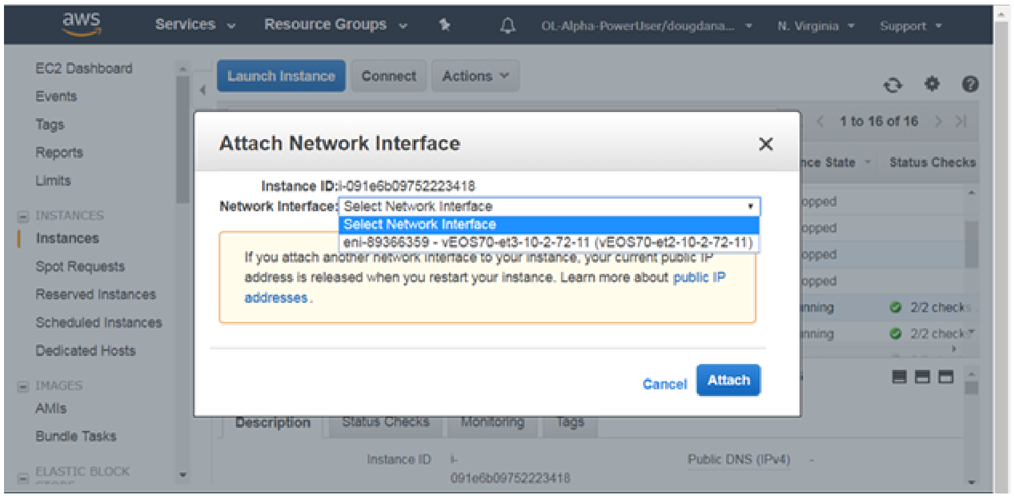


Name	Instance ID	Instance Type	Availability Zone	Instance State	Status Checks
server-loc-z1...	i-02ab1a03b0d8f093b	t2.micro	us-east-1a	running	2/2 check
vEOS60-loc-...	i-03346f7690830f202	c4.xlarge	us-east-1a	stopped	
transit-vroute...	i-04298eb3ccea97d27	c4.xlarge	us-east-1d	stopped	
lob-vrouter-2	i-04fdf93cc1458bf0e	c4.xlarge	us-east-1a	stopped	
lob-web-1	i-07ce81597f3efb349	t2.micro	us-east-1e	stopped	
transit-web-1	i-0893f3486372d1053	t2.micro	us-east-1c	stopped	

Select an instance above

3. Select the **CloudEOS and vEOS Router instance** to attach a newly created network interface.
4. Choose **Actions** > **Networking** > **Attach Network Interface**.

   The Attach Network Interface dialog appears.



Instance ID: i-091e6b09752223418

Network Interface: Select Network Interface

Select Network Interface

eni-89366359 - vEOS70-et3-10-2-72-11 (vEOS70-et2-10-2-72-11)

If you attach another network interface to your instance, your current public IP address is released when you restart your instance. Learn more about public IP addresses.

5. Using the **Network Interface** menu, select the new network interface created to attach to the instance.
6. Click the **Attach** button.
7. Use the **show interfaces** command on the CloudEOS and vEOS Router instance to view the new network interfaces created.

   Example

```
CloudEOS and switch#show interfaces
Ethernet1 is up, line protocol is up (connected)
   Hardware is Ethernet, address is 0235.4079.d2a8 (bia 0235.4079.d2a8)
   Ethernet mtu 8973 bytes, BW 10000000 kbit
   Full-duplex, 10Gb/s, auto negotiation: off, uni-link: n/a
   Up 20 minutes, 42 seconds
   [...]
Ethernet2 is up, line protocol is up (connected)
   Hardware is Ethernet, address is 0287.4ba7.1f88 (bia 0287.4ba7.1f88)
   Ethernet mtu 8973 bytes, BW 10000000 kbit
   Full-duplex, 10Gb/s, auto negotiation: off, uni-link: n/a
   Up 20 minutes, 42 seconds
```

8. Repeat steps 1 through 7 as needed to attach new network interfaces to instances.

Configure the route table of the AWS Router (see Configuring the Route Table of the AWS Router).

3.3.3.3 Configuring the Route Table of the AWS Router

To take advantage of the advanced services provided by CloudEOS and vEOS, configure the route table of the AWS Router so that traffic is forwarded from the AWS Router to CloudEOS and vEOS Router instances. This task involves logging into the AWS Router and modifying route table entries for the CloudEOS and vEOS Router instances to which you want traffic forwarded.

Complete these steps to configure the route table of the AWS router.

1. Log in to the AWS Router.
2. Select the network interface that is attached to a CloudEOS and vEOS Router instance.
3. Obtain the Subnet ID and the route table ID that corresponds to the subnet in which the CloudEOS and vEOS Router instance resides.

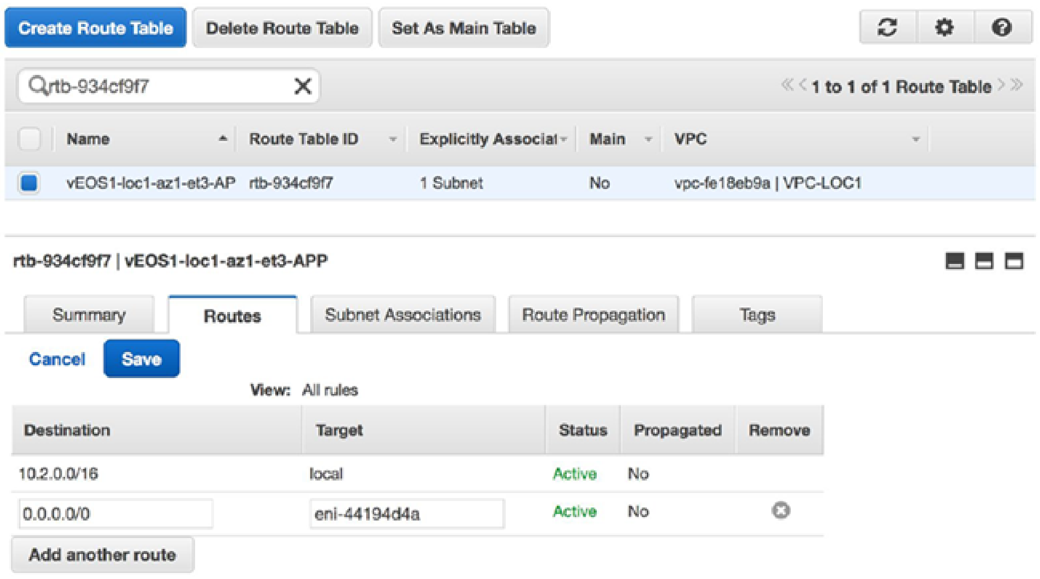
   Example:

   Subnet ID (subnet-1c68b744).

   Route table ID (rtb-934cf9f7).
4. Edit the route table entry so that it points to the corresponding interface of the CloudEOS and vEOS Router in that subnet.

   Example

   To reach any subnet other than 10.2.0.0/24, enter the **Target** to be the network interface ID of the locally connected interface of the CloudEOS and vEOS Router.



Name	Route Table ID	Explicitly Associat	Main	VPC
vEOS1-loc1-az1-et3-AP	rtb-934cf9f7	1 Subnet	No	vpc-fe18eb9a | VPC-LOC1

rtb-934cf9f7 | vEOS1-loc1-az1-et3-APP

View: All rules

Destination	Target	Status	Propagated	Remove
10.2.0.0/16	local	Active	No	
0.0.0.0/0	eni-44194d4a	Active	No	

5. (Optional) Repeat steps 2 through 4 to modify route table entries for additional CloudEOS and vEOS Router instances.

Configure the AWS CloudWatch Logs Agent (see Configuring the AWS CloudWatch Logs Agent). Configuring the Agent ensures that the CloudEOS and vEOS Router logs publish to AWS.

3.3.4 Using User-data for Configuration of Entities and CloudEOS and vEOS Router Instances

CloudEOS and vEOS supports configuration of startup-configuration, AWS CloudWatch, and Cloud HA through the use of user-data. Because user-data can be used to pass in configurations; administrators can take advantage of this feature to quickly configure CloudEOS and vEOS Router instances, AWS CloudWatch, and Cloud HA.

Note: It is recommended to test CloudEOS and vEOS Router configurations on a CloudEOS and vEOS Router or EOS device before using them to deploy a new CloudEOS and vEOS Router.

Requirements for Uploading User-data

To ensure that the user-data is accepted on upload, make sure the user-data meets the following requirements:

- The configuration must be separated by start and end markers.
- Markers are required at the beginning of the line.
- You must upload either text or configuration files (these are the types of files supported by CloudEOS and vEOS Router).

EOS configuration for all interfaces can be passed in during deployment. The configuration takes effect as new interfaces attach to the CloudEOS and vEOS Router.

List of Start and End Markers to Use

This table lists the start and end markers to use when configuring the EOS, AWS, Cloudwatch, and Cloud HA entities. For each specific entity, the configuration file and the location (file path) of the configuration file are given.

Entity / Configuration File / Use	Markers	File Path
Entity: EOS File: EOS CLI configuration file Use: Configure CloudEOS and vEOS Router	`%EOS-STARTUP-CONFIG-START%` `%EOS-STARTUP-CONFIG-END%`	N/A
Entity: AWS Logs File: `aws.conf` Use: Set up AWS region	`%AWS-CONFIG-START%` `%AWS-CONFIG-END%`	`/mnt/flash/awslogs/aws.conf`
Entity: AWS Logs File: `awslogs.conf` Use: Configure logging parameters	`%AWSLOGS-CONFIG-START%` `%AWSLOGS-CONFIG-END%`	`/mnt/flash/awslogs/awsconf.conf`
Entity: AWS Logs File: `proxy.conf` Use: Configure proxy settings	`%AWS-PROXY-START%` `%AWS-PROXY-END%`	`/mnt/flash/awslogs/proxy.conf`
Entity: Cloud HA File: `cloud_ha_config.json` Use: Configure CloudEOS and vEOS Router for High Availability	`%CLOUDHA-CONFIG-START%` `%CLOUDHA-CONFIG-END%`	`/mnt/flash/cloud_ha_config.json`

- Sample Instance User-data

3.3.4.1 Sample Instance User-data

The following sample user-data contains lines to startup the instance and to configure various entities.

The sample contains lines to configure:

- AWS CloudWatch logs (for the us-east-1 region)
- AWS logging parameters
- AWS proxy settings

Sample

```
%EOS-STARTUP-CONFIG-START%
! EOS startup config
hostname my-veos
username admin nopassword
username admin sshkey file flash:key.pub
%EOS-STARTUP-CONFIG-END%
%AWS-CONFIG-START%
[plugins]
cwlogs = cwlogs
[default]
region = us-east-1
%AWS-CONFIG-END%

%AWSLOGS-CONFIG-START%
[general]
state_file = /var/awslogs/state/agent-state
[/var/log/messages]
datetime_format = %b %d %H:%M:%S
file = /var/log/messages
buffer_duration = 5000
log_group_name = veoslogs
log_stream_name = {hostname}
initial_position = start_of_file
%AWSLOGS-CONFIG-END%

%AWS-PROXY-START%
HTTP_PROXY=http://<your_proxy>:<your_proxy_port>
HTTPS_PROXY=http://<your_proxy>:<your_proxy_port>
NO_PROXY=169.254.169.254
%AWS-PROXY-END%
```

Chapter 4

Using the CloudEOS and vEOS Router on Microsoft Azure

The CloudEOS and vEOS Router, which is based on the Arista EOS, runs as a virtual machine instance on Azure. Use the CloudEOS and vEOS Router to create the various types of virtual machine router instances you need for your Azure deployment. For example, gateway routers and transit routers.

- CloudEOS and vEOS Router Image Updates
- Launching CloudEOS and vEOS Router Azure Instance
- vEOS Router Startup-Configuration using Instance Custom-Data
- Troubleshooting Instance
- Resources

4.1 CloudEOS and vEOS Router Image Updates

The process you use to update CloudEOS and vEOS Router images is the standard update process used for EOS images.

For details on the steps to use, refer to the Arista *EOS User Manual* (see https://www.arista.com/en/support/product-documentation).

4.2 Launching CloudEOS and vEOS Router Azure Instance

There are two methods which can be used to launch a CloudEOS and vEOS Router instance.

Below is a summary of each method.

- **Portal Marketplace** This method launches an instance using the Azure Portal Marketplace UI.
- **Azure CLI 2.0:** This method launches an instance using a custom template through the Azure CLI 2.0. The primary advantage of a CLI deployment is the ability to include custom-data and customize your deployment.

Do not deploy the same template twice into a single resource group, because this creates name conflicts. To deploy multiple instances into the same resource group, modify the template, so all resources are renamed, and all IP addresses are unique.

- Creating an Instance using the Portal Marketplace
- Creating an Instance under Azure CLI 2.0
- Logging into Instance

4.2.1 Creating an Instance using the Portal Marketplace

To create an instance using the Portal Marketplace, complete the following steps.

1. In the Azure portal, select the green '+' button in the top left of the screen.
2. In the search bar, type "Arista" and press enter.

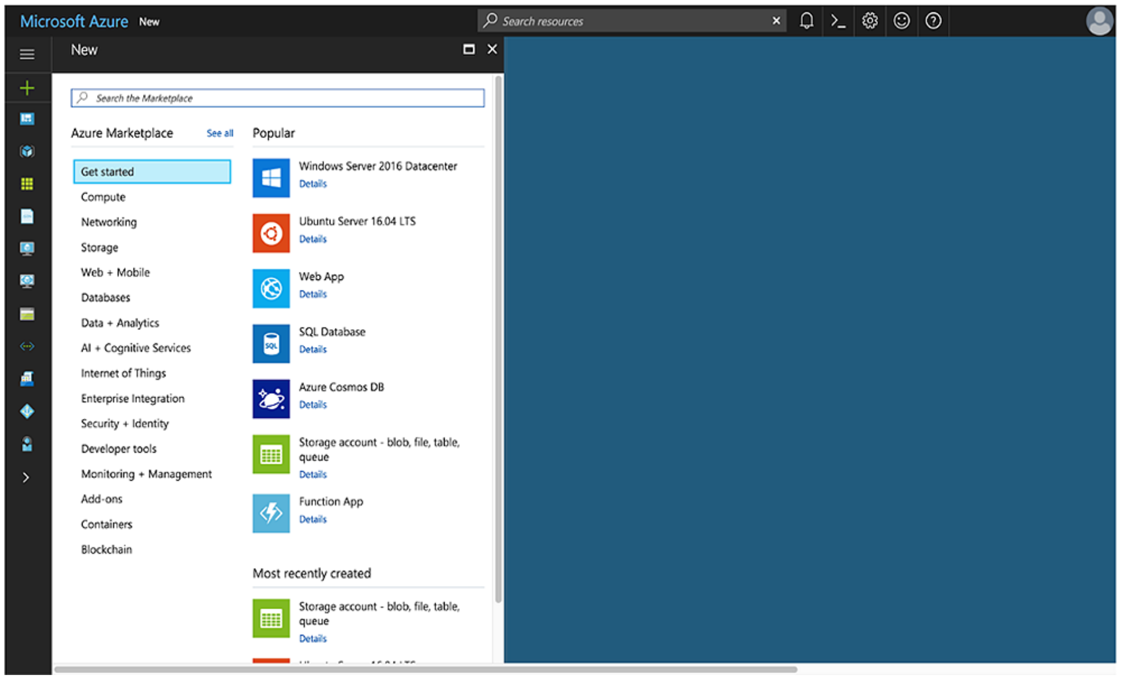
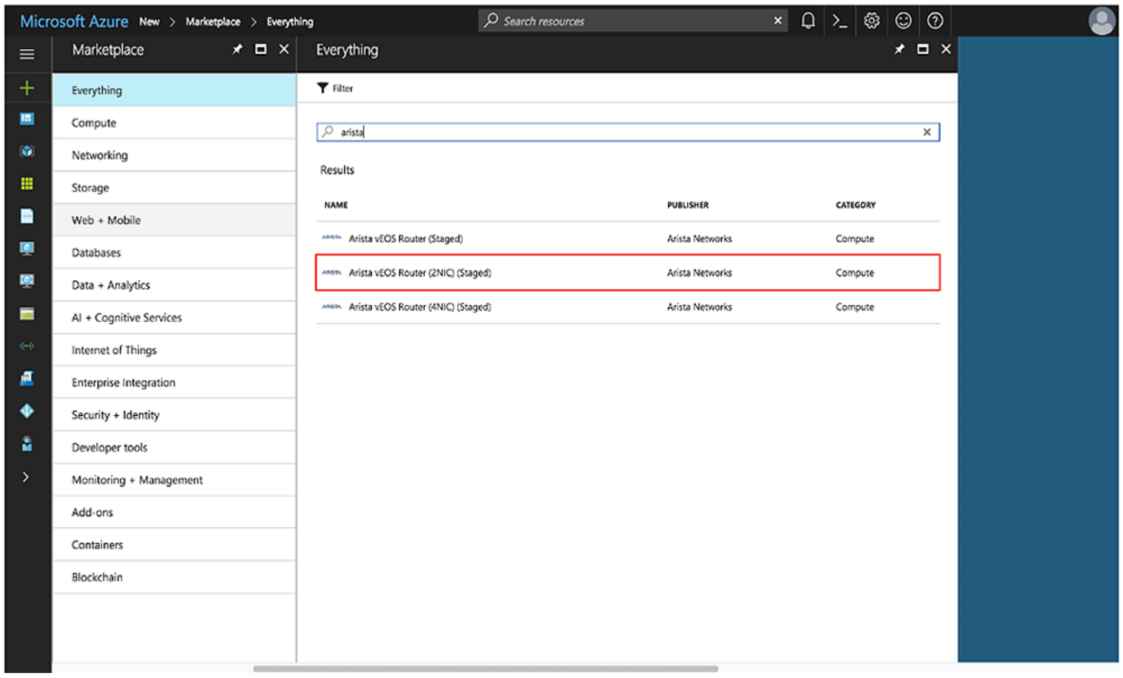


Figure 1: Type '"Arista"

3. Select the Arista offer you are interested in.

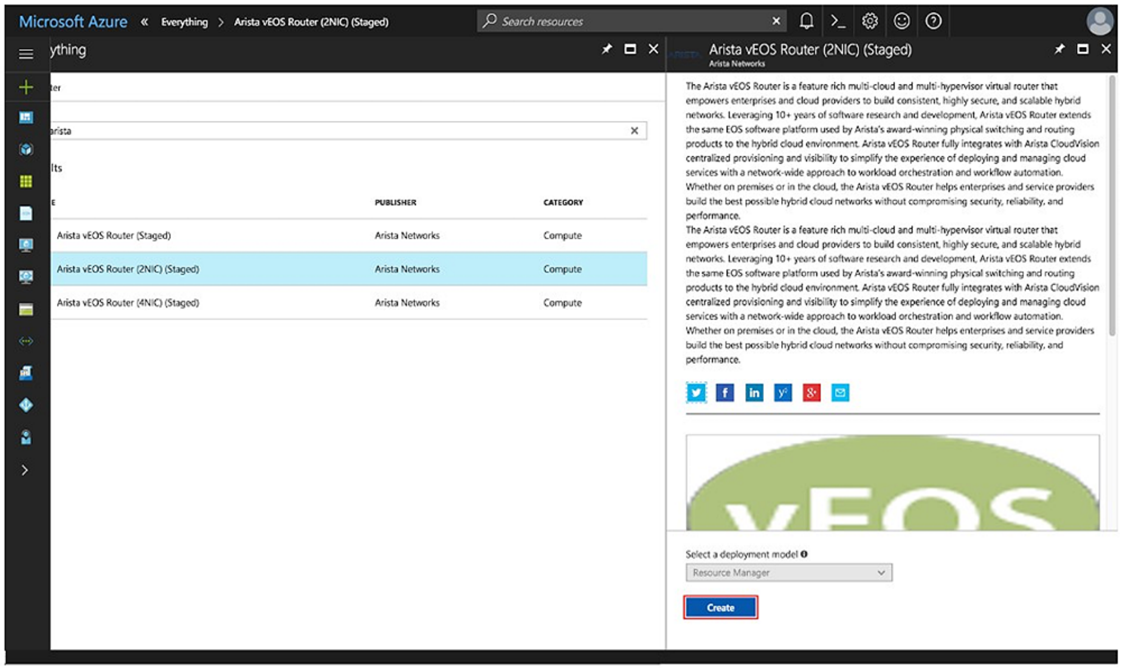


Results

NAME	PUBLISHER	CATEGORY
Arista vEOS Router (Staged)	Arista Networks	Compute
Arista vEOS Router (2NIC) (Staged)	Arista Networks	Compute
Arista vEOS Router (4NIC) (Staged)	Arista Networks	Compute

Figure 2: Arista selection

4. Select "Create".


Microsoft Azure
Everything
Arista vEOS Router (2NIC) (Staged)
Search resources
ything
Arista vEOS Router (2NIC) (Staged)
Arista Networks

	PUBLISHER	CATEGORY
Arista vEOS Router (Staged)	Arista Networks	Compute
Arista vEOS Router (2NIC) (Staged)	Arista Networks	Compute
Arista vEOS Router (4NIC) (Staged)	Arista Networks	Compute

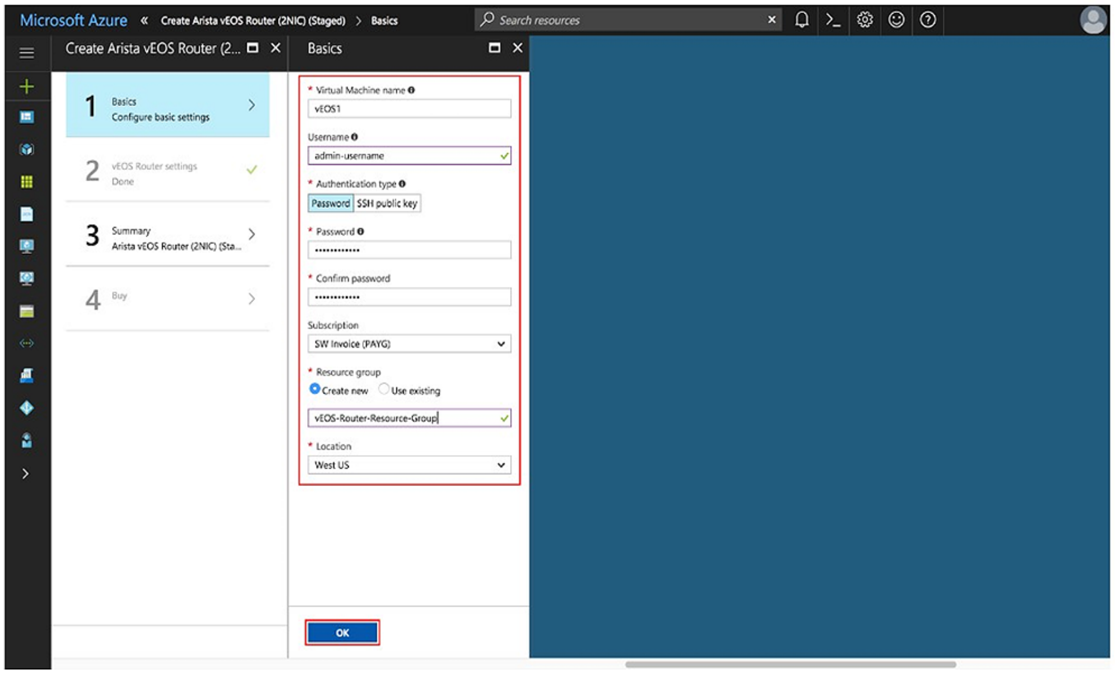
The Arista vEOS Router is a feature rich multi-cloud and multi-hypervisor virtual router that empowers enterprises and cloud providers to build consistent, highly secure, and scalable hybrid networks. Leveraging 10+ years of software research and development, Arista vEOS Router extends the same EOS software platform used by Arista's award-winning physical switching and routing products to the hybrid cloud environment. Arista vEOS Router fully integrates with Arista CloudVision centralized provisioning and visibility to simplify the experience of deploying and managing cloud services with a network-wide approach to workload orchestration and workflow automation. Whether on premises or in the cloud, the Arista vEOS Router helps enterprises and service providers build the best possible hybrid cloud networks without compromising security, reliability, and performance.

The Arista vEOS Router is a feature rich multi-cloud and multi-hypervisor virtual router that empowers enterprises and cloud providers to build consistent, highly secure, and scalable hybrid networks. Leveraging 10+ years of software research and development, Arista vEOS Router extends the same EOS software platform used by Arista's award-winning physical switching and routing products to the hybrid cloud environment. Arista vEOS Router fully integrates with Arista CloudVision centralized provisioning and visibility to simplify the experience of deploying and managing cloud services with a network-wide approach to workload orchestration and workflow automation. Whether on premises or in the cloud, the Arista vEOS Router helps enterprises and service providers build the best possible hybrid cloud networks without compromising security, reliability, and performance.

Select a deployment model: Resource Manager

Figure 3: Select "Create"

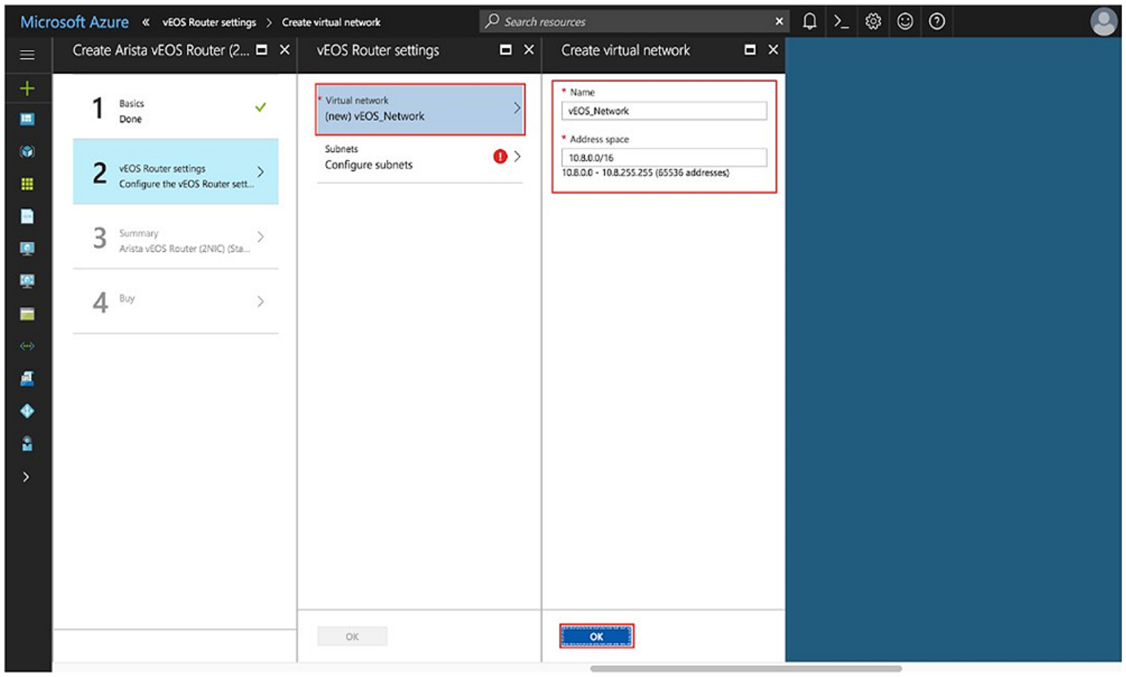
5. Fill out the required information and press "OK".



* Virtual Machine name: vEOS1
Username: admin-username
* Authentication type: [✓ Password] SSH public key
* Password: ••••••••••••
* Confirm password: ••••••••••••
Subscription: SW Invoice (PAYG)
* Resource group: [✓ Create new] Use existing
vEOS-Router-Resource-Group
* Location: West US

Figure 4: Required information

6. Configure the VNet and press "OK".


Microsoft Azure « vEOS Router settings > Create virtual network
Search resources
Create Arista vEOS Router (2...
vEOS Router settings
Create virtual network
1 Basics Done
2 vEOS Router settings Configure the vEOS Router sett...
3 Summary Arista vEOS Router (2NIC) (Sta...
4 Buy
OK

* Virtual network
(new) vEOS_Network

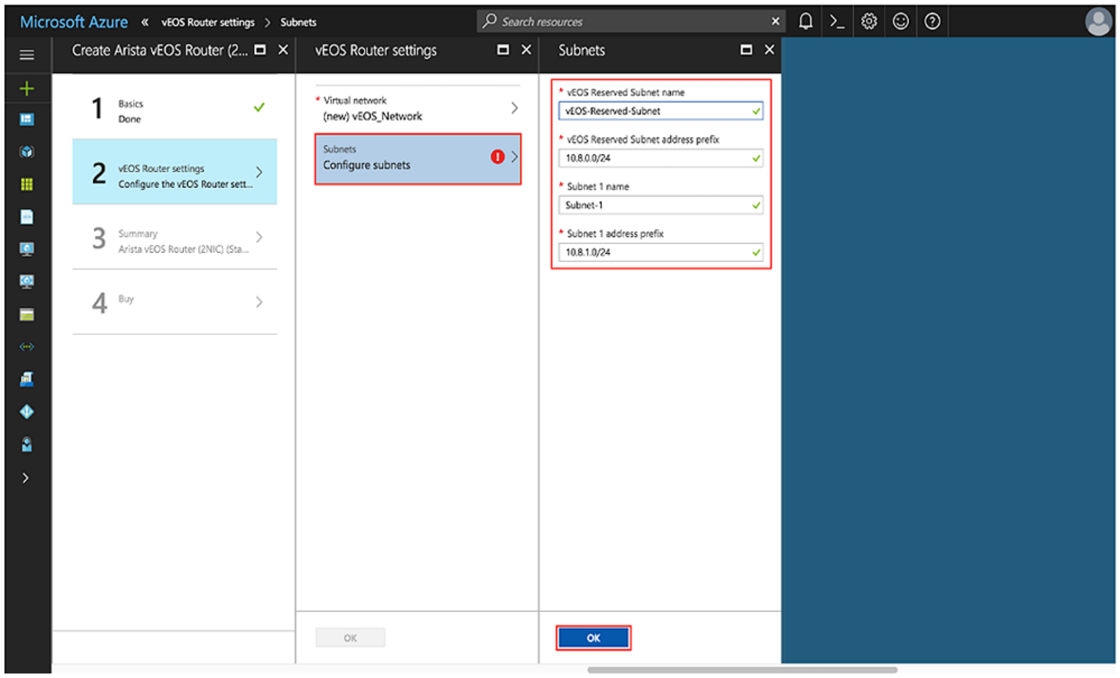
Subnets
Configure subnets

* Name
vEOS_Network

* Address space
10.8.0.0/16
10.8.0.0 - 10.8.255.255 (65536 addresses)

Figure 5: Configuring the VNet

7. Configure the subnets and press "OK".


Microsoft Azure « vEOS Router settings > Subnets
Search resources
Create Arista vEOS Router (2...
vEOS Router settings
Subnets
1 Basics Done
2 vEOS Router settings Configure the vEOS Router sett...
3 Summary Arista vEOS Router (2NIC) (Sta...
4 Buy
OK

* Virtual network
(new) vEOS_Network

Subnets
Configure subnets

* vEOS Reserved Subnet name
vEOS-Reserved-Subnet

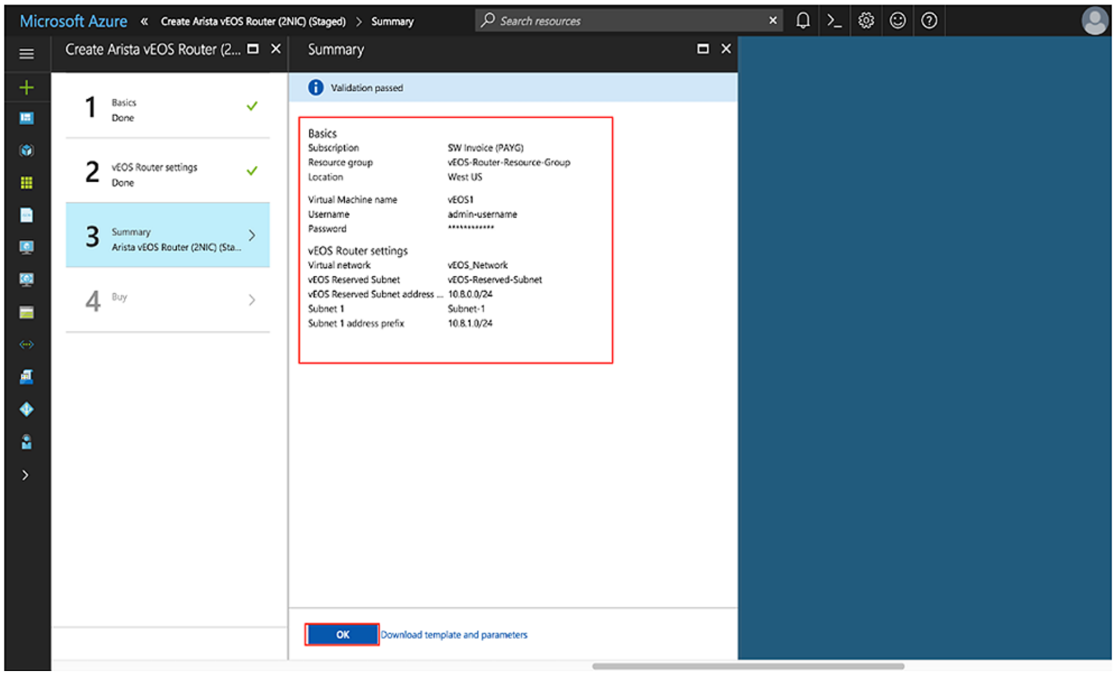
* vEOS Reserved Subnet address prefix
10.8.0.0/24

* Subnet 1 name
Subnet-1

* Subnet 1 address prefix
10.8.1.0/24

Figure 6: Configuring the subnets

8. Verify the information is correct and press "OK".


Microsoft Azure
Create Arista vEOS Router (2NIC) (Staged) > Summary
Search resources
Create Arista vEOS Router (2...
Summary
1 Basics Done
2 vEOS Router settings Done
3 Summary Arista vEOS Router (2NIC) (Sta...
4 Buy
OK
Download template and parameters

Validation passed

Basics

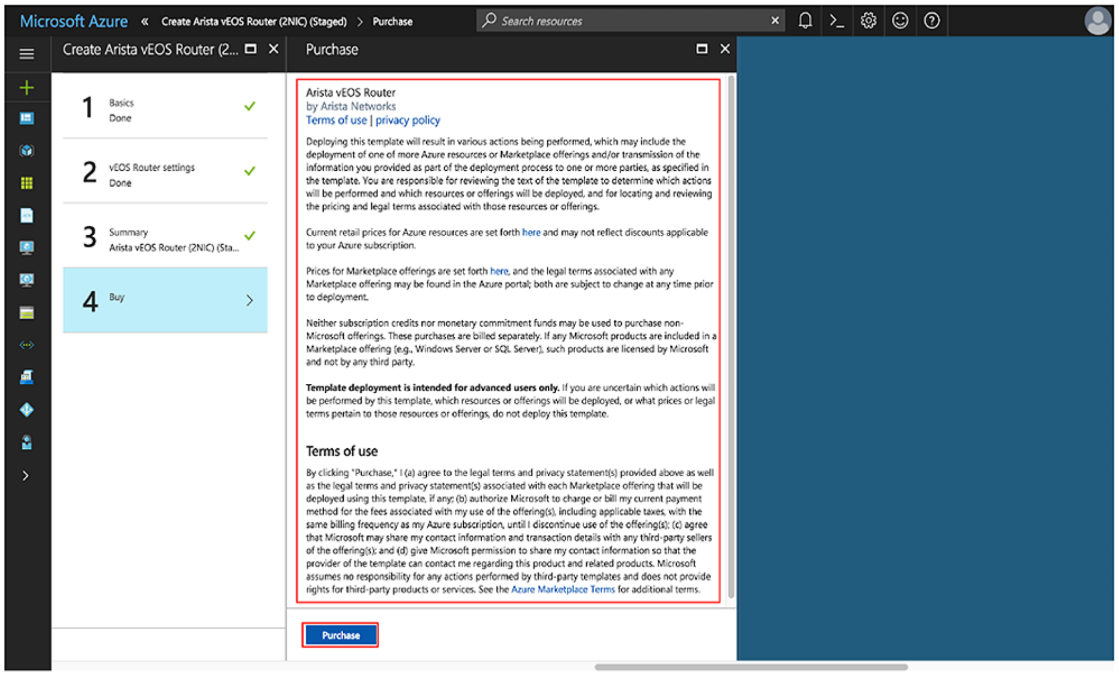
Subscription	SW Invoice (PAYG)
Resource group	vEOS-Router-Resource-Group
Location	West US
Virtual Machine name	vEOS1
Username	admin-username
Password	************

vEOS Router settings

Virtual network	vEOS_Network
vEOS Reserved Subnet	vEOS-Reserved-Subnet
vEOS Reserved Subnet address ...	10.8.0.0/24
Subnet 1	Subnet-1
Subnet 1 address prefix	10.8.1.0/24

Figure 7: Verification

9. Read the Terms and Conditions, then press "Purchase".


Microsoft Azure
Create Arista vEOS Router (2NIC) (Staged) > Purchase
Search resources
Create Arista vEOS Router (2...
Purchase
1 Basics Done
2 vEOS Router settings Done
3 Summary Arista vEOS Router (2NIC) (Sta...
4 Buy
Purchase

Arista vEOS Router
by Arista Networks
Terms of use | privacy policy

Deploying this template will result in various actions being performed, which may include the deployment of one of more Azure resources or Marketplace offerings and/or transmission of the information you provided as part of the deployment process to one or more parties, as specified in the template. You are responsible for reviewing the text of the template to determine which actions will be performed and which resources or offerings will be deployed, and for locating and reviewing the pricing and legal terms associated with those resources or offerings.

Current retail prices for Azure resources are set forth here and may not reflect discounts applicable to your Azure subscription.

Prices for Marketplace offerings are set forth here, and the legal terms associated with any Marketplace offering may be found in the Azure portal; both are subject to change at any time prior to deployment.

Neither subscription credits nor monetary commitment funds may be used to purchase non-Microsoft offerings. These purchases are billed separately. If any Microsoft products are included in a Marketplace offering (e.g., Windows Server or SQL Server), such products are licensed by Microsoft and not by any third party.

Template deployment is intended for advanced users only. If you are uncertain which actions will be performed by this template, which resources or offerings will be deployed, or what prices or legal terms pertain to those resources or offerings, do not deploy this template.

Terms of use

By clicking "Purchase," I (a) agree to the legal terms and privacy statement(s) provided above as well as the legal terms and privacy statement(s) associated with each Marketplace offering that will be deployed using this template, if any; (b) authorize Microsoft to charge or bill my current payment method for the fees associated with my use of the offering(s), including applicable taxes, with the same billing frequency as my Azure subscription, until I discontinue use of the offering(s); (c) agree that Microsoft may share my contact information and transaction details with any third-party sellers of the offering(s); and (d) give Microsoft permission to share my contact information so that the provider of the template can contact me regarding this product and related products. Microsoft assumes no responsibility for any actions performed by third-party templates and does not provide rights for third-party products or services. See the Azure Marketplace Terms for additional terms.

Figure 8: Terms and Conditions

4.2.2 Creating an Instance under Azure CLI 2.0

To create an instance under Azure CLI 2.0, complete the following steps.

1. Install Azure CLI 2.0 (https://docs.microsoft.com/en-us/cli/azure/install-azure-cli?view=azure-cli-latest).
2. Run **az login** and follow the prompts to authorize the machine.

3. Download the template and parameters files from the GitHub repository. https://github.com/Azure/azure-quickstart-templates
4. Open `<prefix>-parameters.json:`. Locate the `./single_line_json.sh user_data.txt` script.
5. Copy and paste the generated output into the **customData** value field of the JSON parameters file.
6. Use the script as in the following example:

```
#!/usr/bin/bash
cat $1 | python -c 'import json, sys; print( json.dumps( sys.stdin.rea
d() ) )'
```

7. Use the template and parameters JSON files to launch a CloudEOS and vEOS Router instance in Azure using the Azure CLI 2.0.

```
$ az group create --name ExampleGroup --location "Central US"
```

Note: You must use the same location as the storage account where the VHD image is uploaded.

```
$ az group deployment create \
  --name ExampleDeployment \
  --resource-group ExampleGroup \
  --template-file <prefix>-template.json \
  --parameters @<prefix>-parameters.json
```

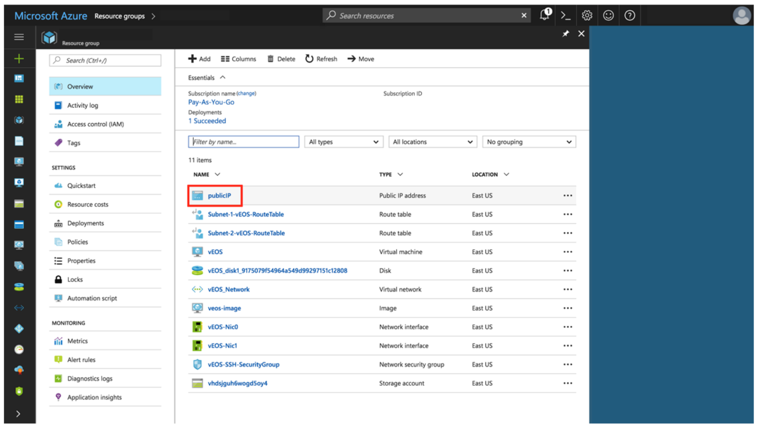
Note: If you are using a newer version of the Azure CLI 2.0, you may encounter a parameter file parsing bug. To fix this, remove the @ symbol before the parameters filename.

4.2.3 Logging into Instance

To log into an instance, complete the following steps.

1. Select the resource group containing your CloudEOS and vEOS Router deployment from the **Resource groups** list.
2. Select the item **publicIP**.

Figure 9: Selecting the PublicIP

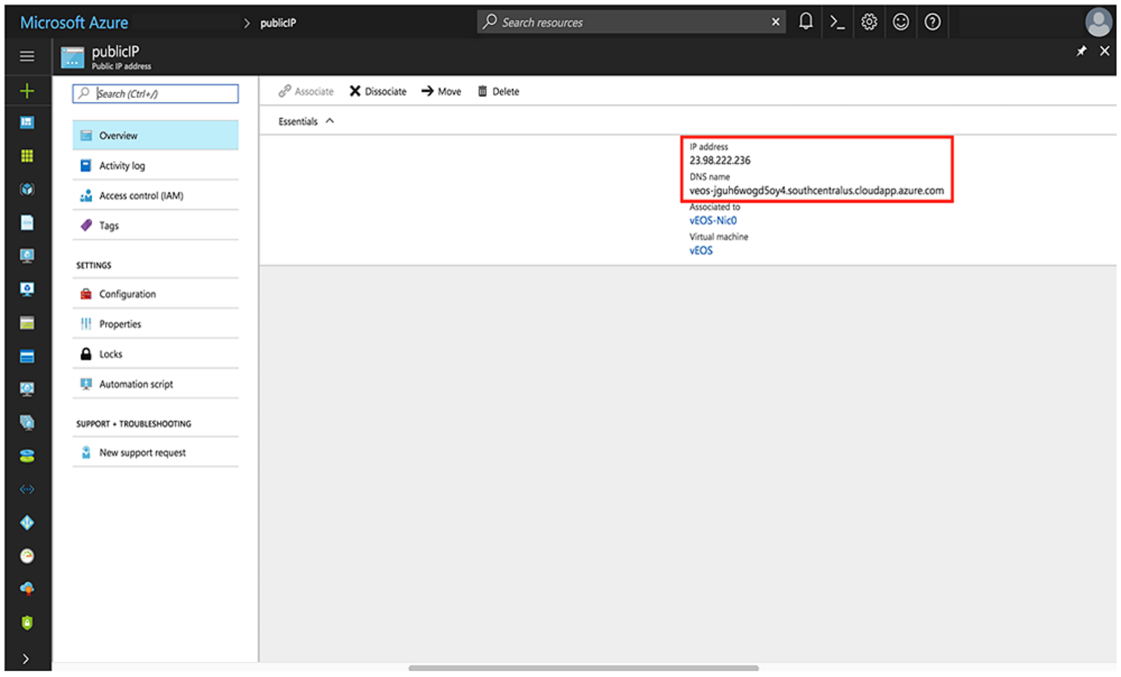


Subscription name (change)
Pay-As-You-Go
Deployments
1 Succeeded
Subscription ID

11 items

NAME	TYPE	LOCATION
publicIP	Public IP address	East US
Subnet-1-vEOS-RouteTable	Route table	East US
Subnet-2-vEOS-RouteTable	Route table	East US
vEOS	Virtual machine	East US
vEOS_disk1_9175079f54964a549d99297151c12808	Disk	East US
vEOS_Network	Virtual network	East US
veos-image	Image	East US
vEOS-Nic0	Network interface	East US
vEOS-Nic1	Network interface	East US
vEOS-SSH-SecurityGroup	Network security group	East US
vhdsjguh6wogd5oy4	Storage account	East US

3. Locate the IP address and DNS name found on the **Overview** page.



IP address
23.98.222.236
DNS name
veos-jguh6wogd5oy4.southcentralus.cloudapp.azure.com
Associated to
vEOS-Nic0
Virtual machine
vEOS

Figure 10: Locating the IP address and DNS

NOTE: If either of these fields is not populated, your instance still deploys. Refresh the page after a couple of minutes.

4. Secure Shell (SSH) to your Virtual Machine (VM) using the IP address or Domain Name Server (DNS) name found in the previous step, using the credentials you gave when you initially setup the VM.

```
bash# ssh myusername@123.123.123.1
Password: *********
```

NOTE: It may take between 5-10 minutes for the instance to become reachable after the deployment starts. Refer to the section Troubleshooting Instance for additional information.

4.3 vEOS Router Startup-Configuration using Instance Custom-Data

Describes launch employing custom-data information.

During the initial launching of the vEOS Router Instance, Azure provides a feature to upload custom-data. The administrator can upload vEOS Router configuration using custom-data at the time of the launching of the vEOS Router Instance.

Custom-data can be used to pass in configuration for multiple entities. Currently, only the EOS configuration is supported in Azure. This configuration must be separated by start and end markers.

Entity	Markers	File Path
EOS CLI configuration file	`%EOS-STARTUP-CONFIG-START%` `%EOS-STARTUP-CONFIG-END%`	N/A
Cloud HA configuration file	`%CLOUDHA-CONFIG-START%` `%CLOUDHA-CONFIG-END%`	`/mnt/flash/cloud_ha_config.json`

Note, the following regarding the custom-data.

- Markers must be at the beginning of the line.
- The user is expected to have tested the configurations on a live system before using the configurations to deploy the new vEOS Router. Mis-configuration may result in an unrecoverable instance.
- EOS configuration for all interfaces can be passed in during deployment. The configuration takes effect as the new instances attach to the vEOS Router.

- Sample Instance Custom-Data
- Providing Startup-Configuration using Azure Custom-Data

4.3.1 Sample Instance Custom-Data

Illustrates a sample Instance with custom-data.

```
%EOS-STARTUP-CONFIG-START%
! EOS startup config
username admin nopassword
username admin sshkey file flash:key.pub
%EOS-STARTUP-CONFIG-END%
```

4.3.2 Providing Startup-Configuration using Azure Custom-Data

Adding custom-data to an instance.

Currently, custom-data can only be used on instances deployed using the Azure CLI 2.0.

In order to add custom-data to an instance, the custom-data must be provided as a single-line value with `'\n'` delimiting newlines.

Use the `single_line_json.sh` script to convert your custom-data into this format.

```
#!/usr/bin/bash
cat $1 | python -c 'import json, sys; print( json.dumps( sys.stdin.rea
d() ) )'
```

Usage of the script is as follows:

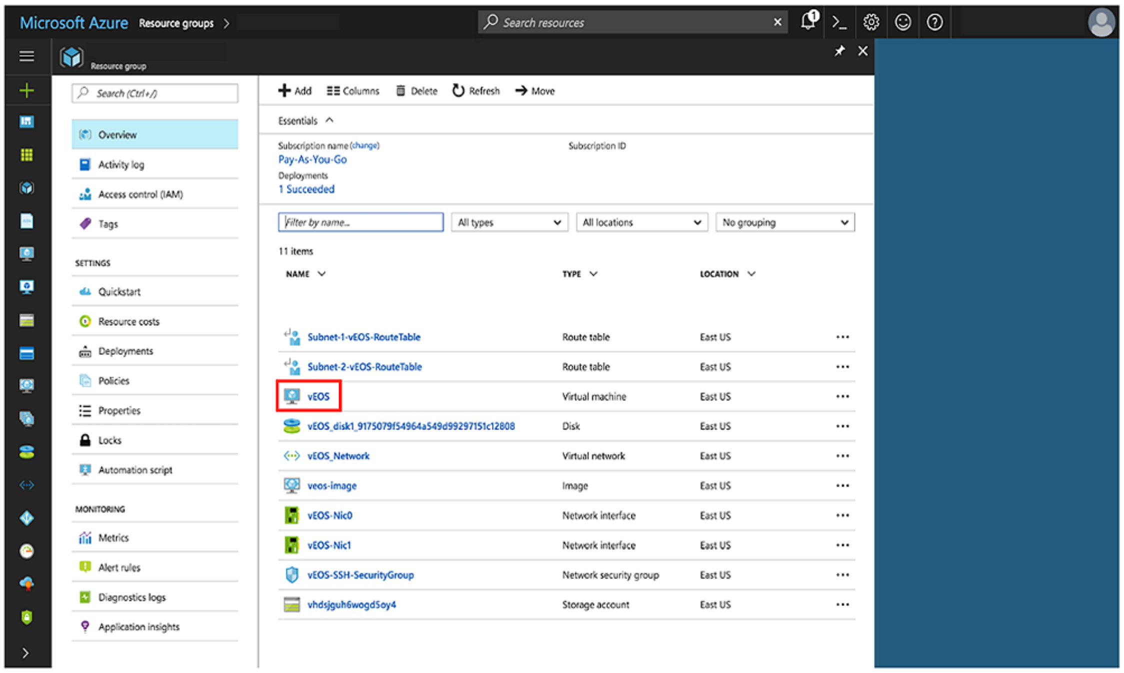
```
./single_line_json.sh user_data.txt
```

Copy and paste the generated output into the **customData** value field of the JSON parameters file.

4.4 Troubleshooting Instance

To troubleshoot the instance, complete the following steps.

1. Select the resource group containing your CloudEOS and vEOS Router deployment from the **Resource groups** list.
2. Select the item **CloudEOS and vEOS Router.**



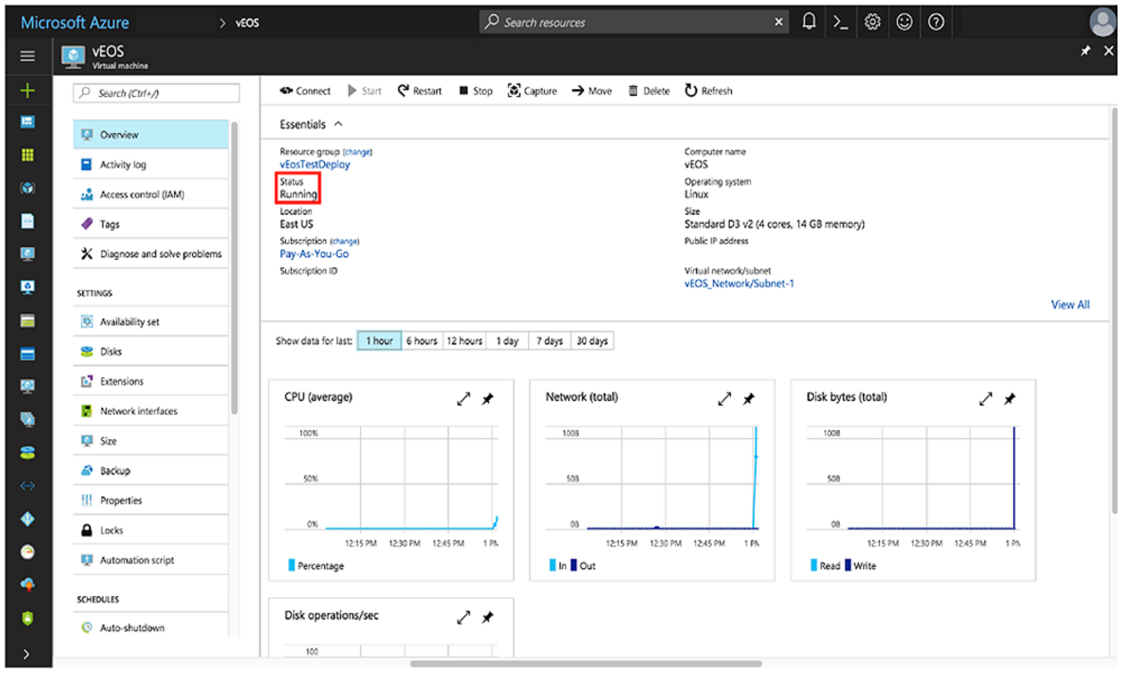
Subscription name (change)
Pay-As-You-Go
Deployments
1 Succeeded
Subscription ID

11 items

NAME	TYPE	LOCATION
Subnet-1-vEOS-RouteTable	Route table	East US
Subnet-2-vEOS-RouteTable	Route table	East US
vEOS	Virtual machine	East US
vEOS_disk1_9175079f54964a549d99297151c12808	Disk	East US
vEOS_Network	Virtual network	East US
veos-image	Image	East US
vEOS-Nic0	Network interface	East US
vEOS-Nic1	Network interface	East US
vEOS-SSH-SecurityGroup	Network security group	East US
vhdsjguh6wogd5oy4	Storage account	East US

Figure 11: Select the CloudEOS and vEOS Router

3. Note the status of the VM. It should either be "Creating", "Starting", or "Running".

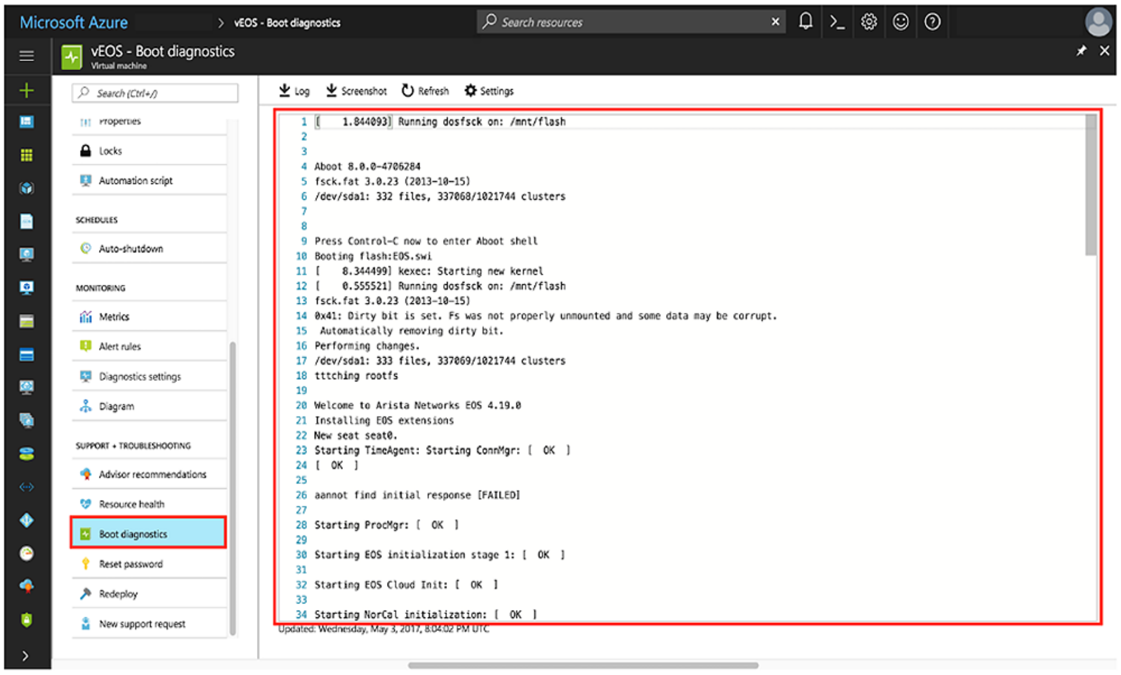


Resource group (change)
vEosTestDeploy
Status
Running
Location
East US
Subscription (change)
Pay-As-You-Go
Subscription ID

Computer name
vEOS
Operating system
Linux
Size
Standard D3 v2 (4 cores, 14 GB memory)
Public IP address
Virtual network/subnet
vEOS_Network/Subnet-1

Figure 12: Status of the VM

4. Check the boot diagnostics for any error messages or warnings.



```
[    1.844093] Running dosfsck on: /mnt/flash


Aboot 8.0.0-4706284
fsck.fat 3.0.23 (2013-10-15)
/dev/sda1: 332 files, 337068/1021744 clusters


Press Control-C now to enter Aboot shell
Booting flash:EOS.swi
[    8.344499] kexec: Starting new kernel
[    0.555521] Running dosfsck on: /mnt/flash
fsck.fat 3.0.23 (2013-10-15)
0x41: Dirty bit is set. Fs was not properly unmounted and some data may be corrupt.
 Automatically removing dirty bit.
Performing changes.
/dev/sda1: 333 files, 337069/1021744 clusters
tttching rootfs

Welcome to Arista Networks EOS 4.19.0
Installing EOS extensions
New seat seat0.
Starting TimeAgent: Starting ConnMgr: [  OK  ]
[  OK  ]

aannot find initial response [FAILED]

Starting ProcMgr: [  OK  ]

Starting EOS initialization stage 1: [  OK  ]

Starting EOS Cloud Init: [  OK  ]

Starting NorCal initialization: [  OK  ]
```

Updated: Wednesday, May 3, 2017, 8:04:02 PM UTC

Figure 13: Error messages and warnings

4.5 Resources

Additional resources.

1. How To: Deploy Azure Virtual Machines With An Azure Resource Manager (ARM) Template - https://www.youtube.com/watch?v=wi74jR0MRLg
2. How To Deploy Resources - https://docs.microsoft.com/en-us/azure/azure-resource-manager/resource-group-template-deploy-cli

Chapter 5

Arista CloudEOS on Google Cloud Platform (GCP)

Arista CloudEOS is now supported on Google Cloud Platform (GCP), as well as other public and private clouds.

- Overview
- Deploying Arista CloudEOS on GCP
- Logging into Arista CloudEOS
- Arista CloudEOS Instance with more than 2 Interfaces

5.1 Overview

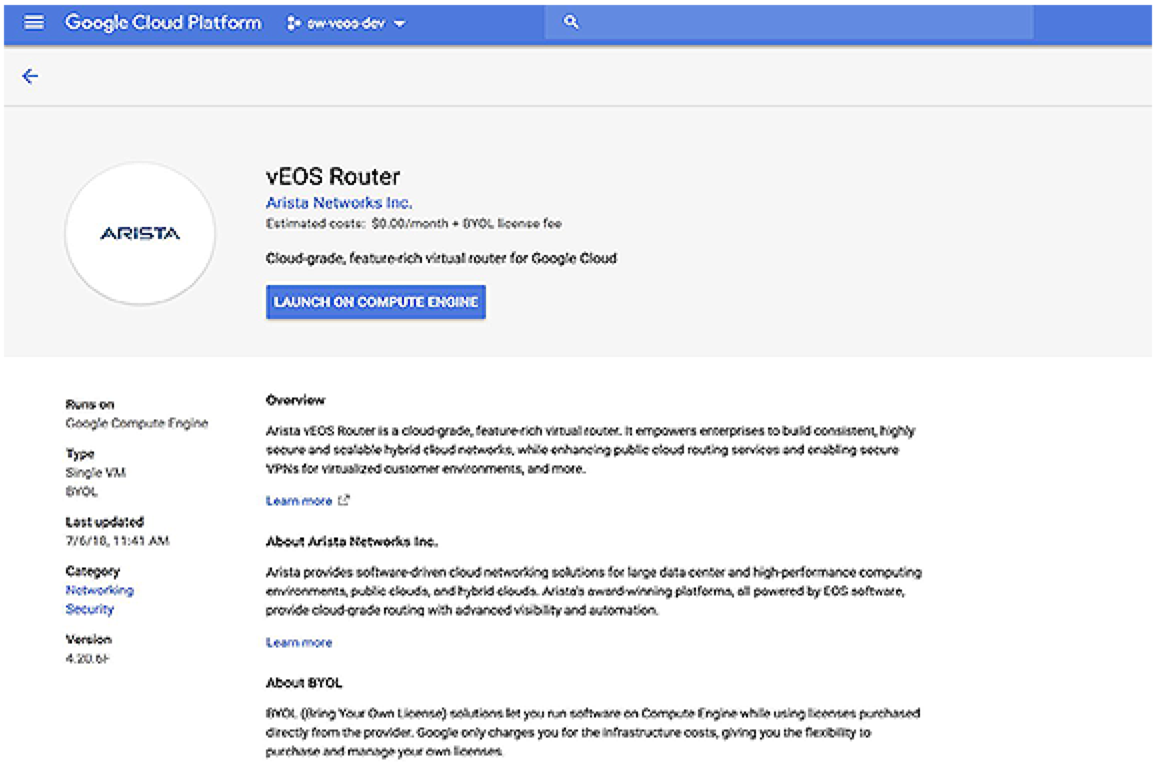
Arista CloudEOS

Arista CloudEOS is a cloud-grade and feature-rich virtual router for Google cloud. This software-only release of EOS software is supported on public clouds, as well as on customer premises equipment running Linux and VMware hypervisors. By bringing advanced network telemetry and secure IPSec VPN connectivity in a software-only package, CloudEOS provides a consistent, secure and universal approach to hybrid cloud networking for any virtualized cloud deployment.

This release of CloudEOS is available as a software subscription in **Google Cloud Launcher** following a BYOL license model. A CloudEOS license activation key must be obtained separately from Arista, which unlocks the platform from a default performance limit of 10 Mbps, and enable the use of IPsec encrypted VPNs.

5.2 Deploying Arista CloudEOS on GCP

1. Locate the CloudEOS listing in the **Google Cloud Launcher**, then select **LAUNCH ON COMPUTE ENGINE**.



vEOS Router

Arista Networks Inc.

Estimated costs: $0.00/month + BYOL license fee

Cloud-grade, feature-rich virtual router for Google Cloud

Runs on
Google Compute Engine

Type
Single VM
BYOL

Last updated
7/6/18, 11:41 AM

Category
Networking
Security

Version
4.20.6F

Overview

Arista vEOS Router is a cloud-grade, feature-rich virtual router. It empowers enterprises to build consistent, highly secure and scalable hybrid cloud networks, while enhancing public cloud routing services and enabling secure VPNs for virtualized customer environments, and more.

Learn more

About Arista Networks Inc.

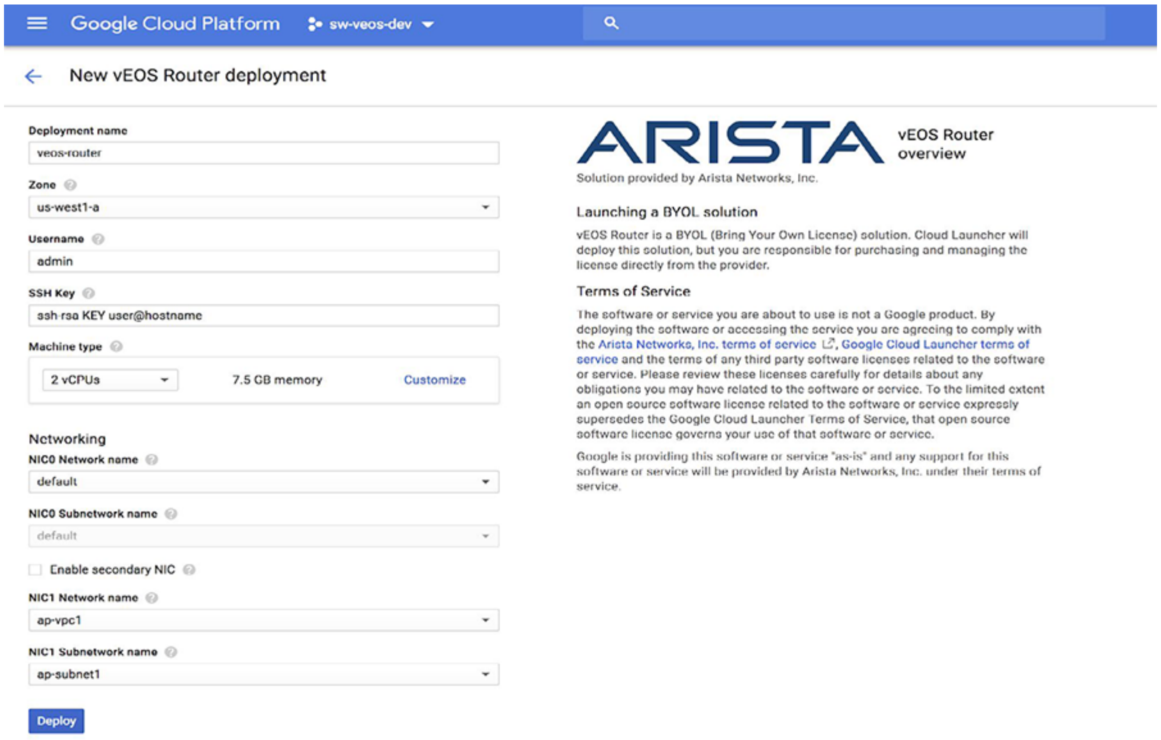
Arista provides software-driven cloud networking solutions for large data center and high-performance computing environments, public clouds, and hybrid clouds. Arista's award-winning platforms, all powered by EOS software, provide cloud-grade routing with advanced visibility and automation.

Learn more

About BYOL

BYOL (Bring Your Own License) solutions let you run software on Compute Engine while using licenses purchased directly from the provider. Google only charges you for the infrastructure costs, giving you the flexibility to purchase and manage your own licenses.

2. Fill out the relevant fields in the deployment screen, then select **Deploy**.



New vEOS Router deployment

Deployment name: veos-router

Zone: us-west1-a

Username: admin

SSH Key: ssh-rsa KEY user@hostname

Machine type: 2 vCPUs 7.5 GB memory Customize

Networking

NIC0 Network name: default

NIC0 Subnetwork name: default

Enable secondary NIC

NIC1 Network name: ap-vpc1

NIC1 Subnetwork name: ap-subnet1

vEOS Router overview

Solution provided by Arista Networks, Inc.

Launching a BYOL solution

vEOS Router is a BYOL (Bring Your Own License) solution. Cloud Launcher will deploy this solution, but you are responsible for purchasing and managing the license directly from the provider.

Terms of Service

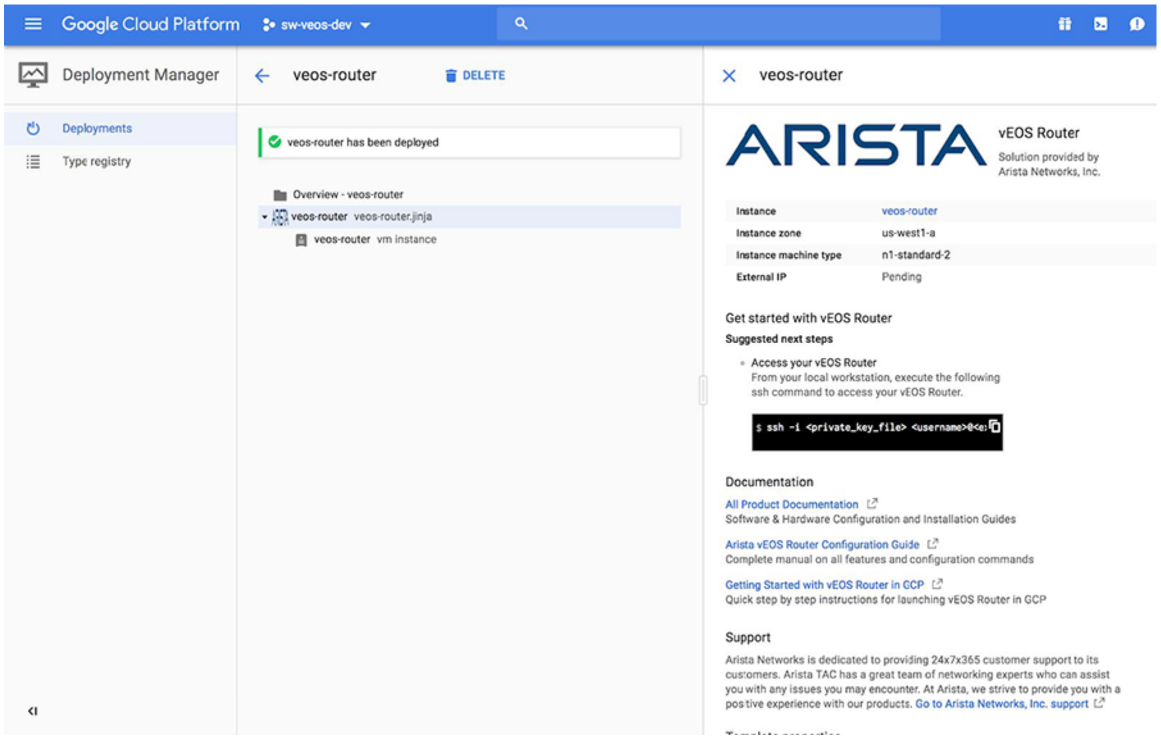
The software or service you are about to use is not a Google product. By deploying the software or accessing the service you are agreeing to comply with the Arista Networks, Inc. terms of service, Google Cloud Launcher terms of service and the terms of any third party software licenses related to the software or service. Please review these licenses carefully for details about any obligations you may have related to the software or service. To the limited extent an open source software license related to the software or service expressly supersedes the Google Cloud Launcher Terms of Service, that open source software license governs your use of that software or service.

Google is providing this software or service "as-is" and any support for this software or service will be provided by Arista Networks, Inc. under their terms of service.

Note: By default the instance is created with a single NIC and the values for **NIC1 Network name** and **NIC1 Subnetwork name** is ignored. To create the instance with the second interface please check **Enable secondary NIC** checkbox.

Note: When adding a SSH public key make sure you paste the key without any extra spaces and newlines.

3. After deployment, you will find the information about your CloudEOS instance in the post deployment screen.



veos-router has been deployed

Overview - veos-router
veos-router veos-router.jinja
veos-router vm instance

ARISTA vEOS Router
Solution provided by Arista Networks, Inc.

Instance	veos-router
Instance zone	us-west1-a
Instance machine type	n1-standard-2
External IP	Pending

Get started with vEOS Router

Suggested next steps

- Access your vEOS Router
  From your local workstation, execute the following ssh command to access your vEOS Router.

```
$ ssh -i <private_key_file> <username>@<e:
```

Documentation

All Product Documentation
Software & Hardware Configuration and Installation Guides

Arista vEOS Router Configuration Guide
Complete manual on all features and configuration commands

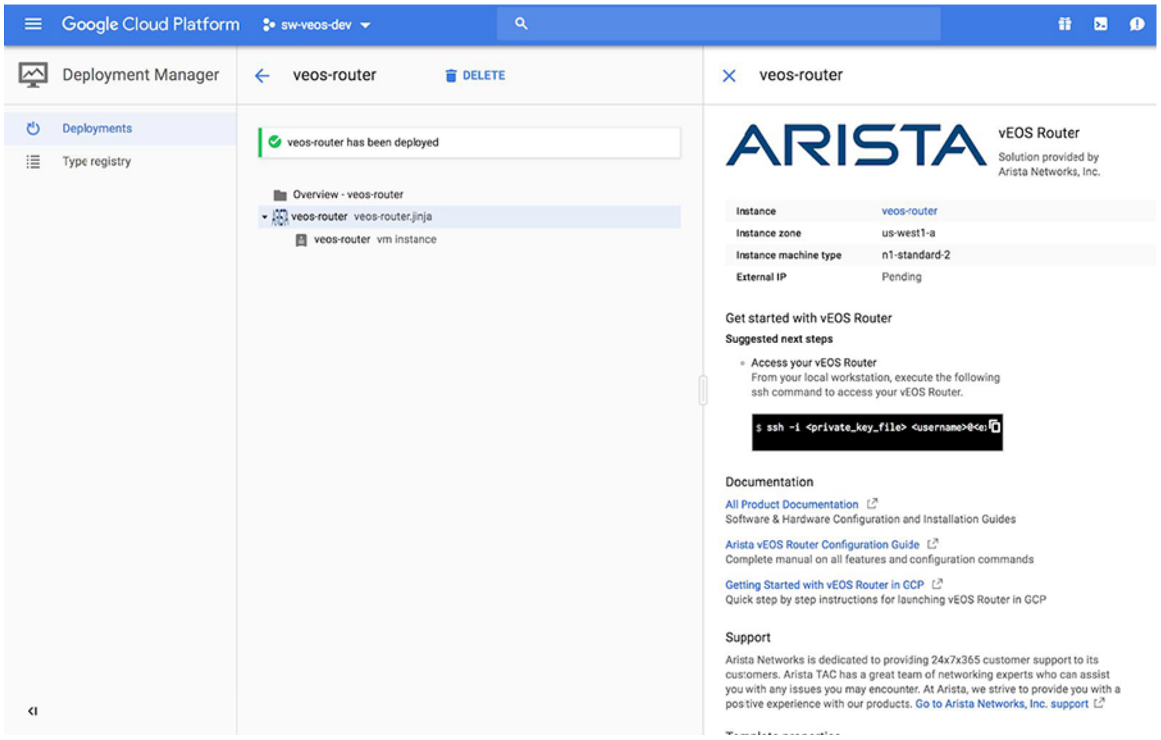
Getting Started with vEOS Router in GCP
Quick step by step instructions for launching vEOS Router in GCP

Support

Arista Networks is dedicated to providing 24x7x365 customer support to its customers. Arista TAC has a great team of networking experts who can assist you with any issues you may encounter. At Arista, we strive to provide you with a positive experience with our products. Go to Arista Networks, Inc. support

5.3 Logging into Arista CloudEOS

1. From the post deployment screen select **vm instance**.



veos-router has been deployed

Overview - veos-router
veos-router veos-router.jinja
veos-router vm instance

ARISTA vEOS Router
Solution provided by Arista Networks, Inc.

Instance veos-router
Instance zone us-west1-a
Instance machine type n1-standard-2
External IP Pending

Get started with vEOS Router

Suggested next steps

- Access your vEOS Router
  From your local workstation, execute the following ssh command to access your vEOS Router.

$ ssh -i <private_key_file> <username>@<e:

Documentation

All Product Documentation
Software & Hardware Configuration and Installation Guides

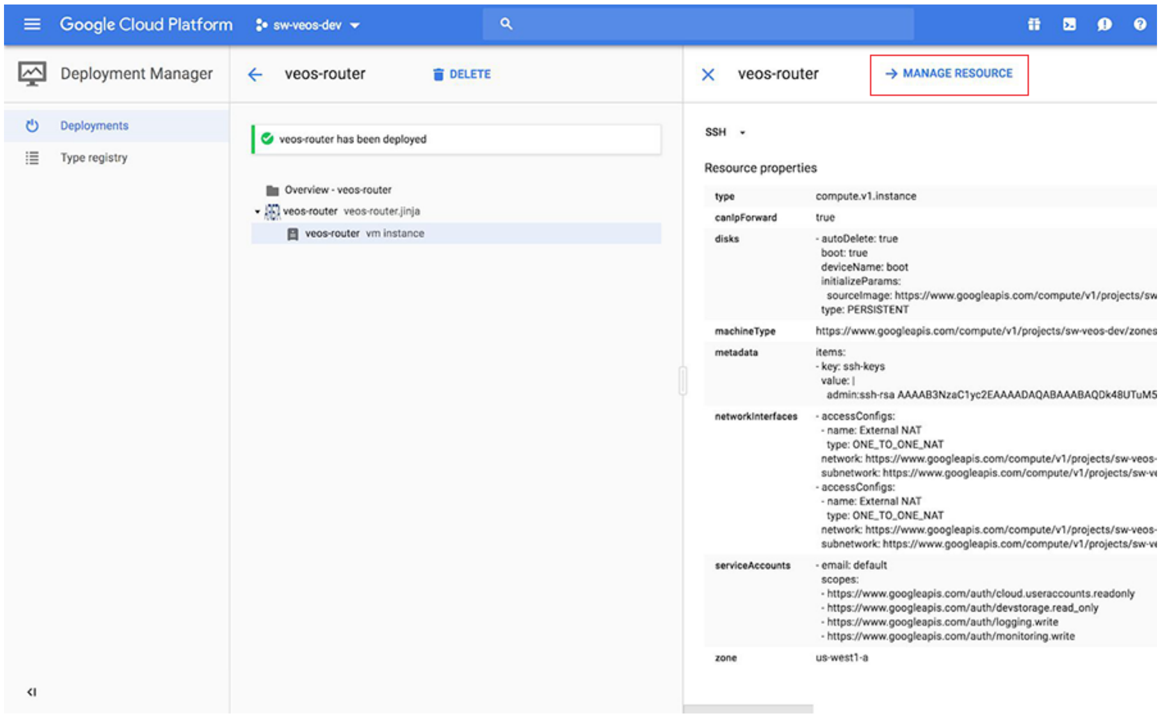
Arista vEOS Router Configuration Guide
Complete manual on all features and configuration commands

Getting Started with vEOS Router in GCP
Quick step by step instructions for launching vEOS Router in GCP

Support

Arista Networks is dedicated to providing 24x7x365 customer support to its customers. Arista TAC has a great team of networking experts who can assist you with any issues you may encounter. At Arista, we strive to provide you with a positive experience with our products. Go to Arista Networks, Inc. support

2. Select **MANAGE RESOURCE**.



veos-router has been deployed

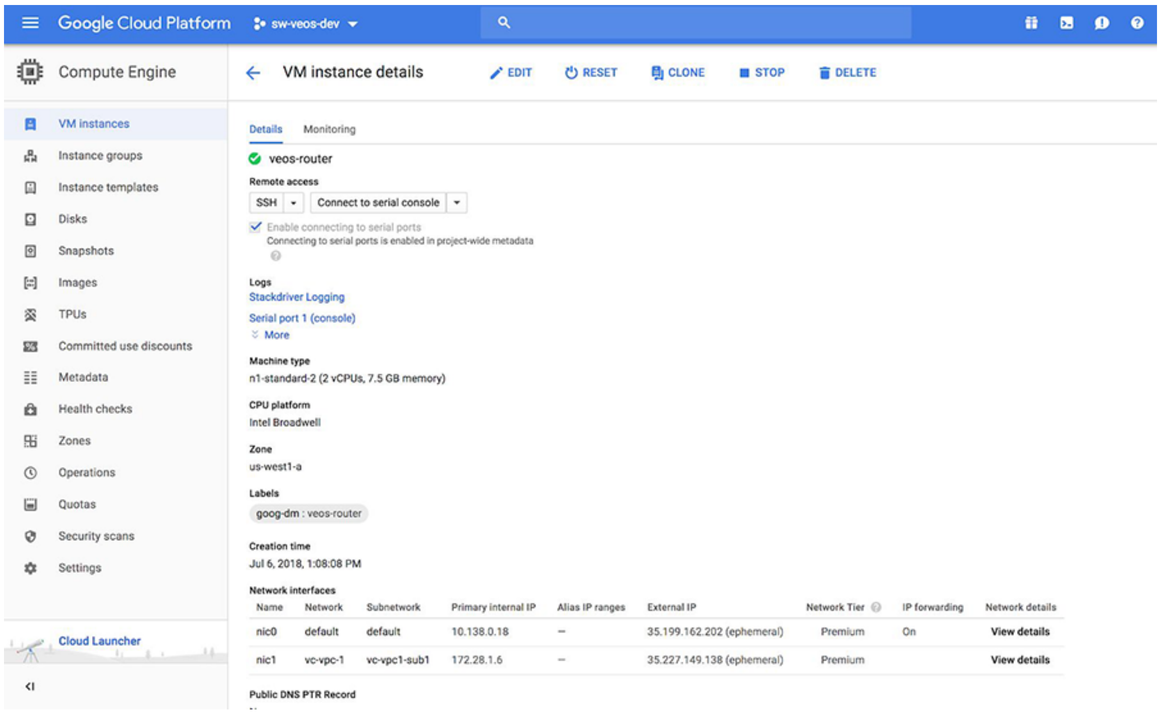
Overview - veos-router
veos-router veos-router.jinja
veos-router vm instance

SSH

Resource properties

type	compute.v1.instance
canIpForward	true
disks	- autoDelete: true boot: true deviceName: boot initializeParams: sourceImage: https://www.googleapis.com/compute/v1/projects/sw type: PERSISTENT
machineType	https://www.googleapis.com/compute/v1/projects/sw-veos-dev/zones
metadata	items: - key: ssh-keys value: | admin:ssh-rsa AAAAB3NzaC1yc2EAAAADAQABAAABAQDk48UTuM5
networkInterfaces	- accessConfigs: - name: External NAT type: ONE_TO_ONE_NAT network: https://www.googleapis.com/compute/v1/projects/sw-veos- subnetwork: https://www.googleapis.com/compute/v1/projects/sw-v - accessConfigs: - name: External NAT type: ONE_TO_ONE_NAT network: https://www.googleapis.com/compute/v1/projects/sw-veos- subnetwork: https://www.googleapis.com/compute/v1/projects/sw-v
serviceAccounts	- email: default scopes: - https://www.googleapis.com/auth/cloud.useraccounts.readonly - https://www.googleapis.com/auth/devstorage.read_only - https://www.googleapis.com/auth/logging.write - https://www.googleapis.com/auth/monitoring.write
zone	us-west1-a

3. Locate the **External IP**.



veos-router

Remote access

SSH | Connect to serial console

✓ Enable connecting to serial ports

Connecting to serial ports is enabled in project-wide metadata

Logs

Stackdriver Logging

Serial port 1 (console)

More

Machine type

n1-standard-2 (2 vCPUs, 7.5 GB memory)

CPU platform

Intel Broadwell

Zone

us-west1-a

Labels

goog-dm : veos-router

Creation time

Jul 6, 2018, 1:08:08 PM

Network interfaces

Name	Network	Subnetwork	Primary internal IP	Alias IP ranges	External IP	Network Tier	IP forwarding	Network details
nic0	default	default	10.138.0.18	–	35.199.162.202 (ephemeral)	Premium	On	View details
nic1	vc-vpc-1	vc-vpc1-sub1	172.28.1.6	–	35.227.149.138 (ephemeral)	Premium		View details

Public DNS PTR Record

4. Log into the instance using the credentials you entered during the deployment:

```
“ssh -i <private_key_file> <username>@<external_ip>”.
```

5.4 Arista CloudEOS Instance with more than 2 Interfaces

You can launch an Arista VM either by launching an instance using template, or by using deployment manager.

- Launching Using Instance Template
- Launching Using Deployment Manager Template

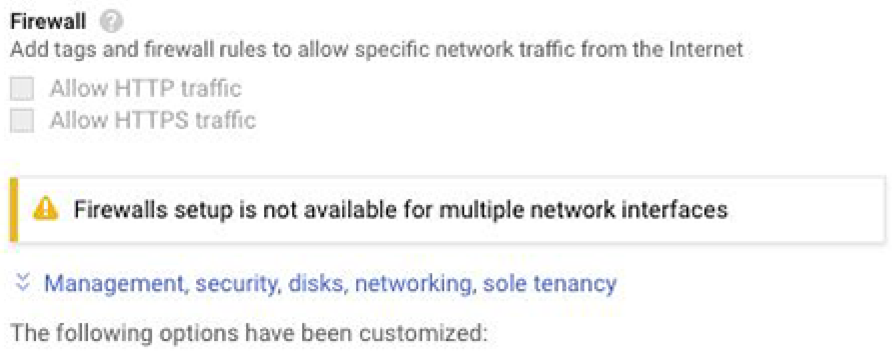
5.4.1 Launching Using Instance Template

To create a Arista vEOS instance on Google cloud with more than 2 interfaces, you need to create an instance template first and then launch the VM from the template. Google cloud has a maximum limit of 8 interfaces per VM Instance. The following steps are to create the VM template, and launch the VM with additional network interfaces.

1. Create network subnets in a specific VPC for each attached interfaces through Google Cloud console.
2. To activate Google Cloud Shell: **Goto Menu > Compute Engine > VM Instances and click on > _** on the top right of the Menu.
3. Create an VM template with additional network interfaces by running the below command in the Google Cloud CLI, as shown in the example below:

```
gcloud compute instance-templates create arista-template-1 \
    --network-interface subnet=default \
    --network-interface subnet=net1-subnet-b,no-address \
    --network-interface subnet=net2-subnet-c,no-address \
    --region us-central1  --machine-type=n1-standard-4\
    --image-project=sw-veos-public \
    --image=https://www.googleapis.com/compute/v1/projects/
            sw-veos-public/global/images/arista-eos-4-21-3f-01-16-2019
```

4. Goto **Menu > Compute Engine > Instance Templates** and refresh to see the **arista-template-1**.
5. Click on the **arista-template-1**, and then, click on **Create VM button** on the top menu.
6. Scroll down and click on **Management, security, disk, networking, sole tenancy** as shown below.


Firewall

Add tags and firewall rules to allow specific network traffic from the Internet

Allow HTTP traffic

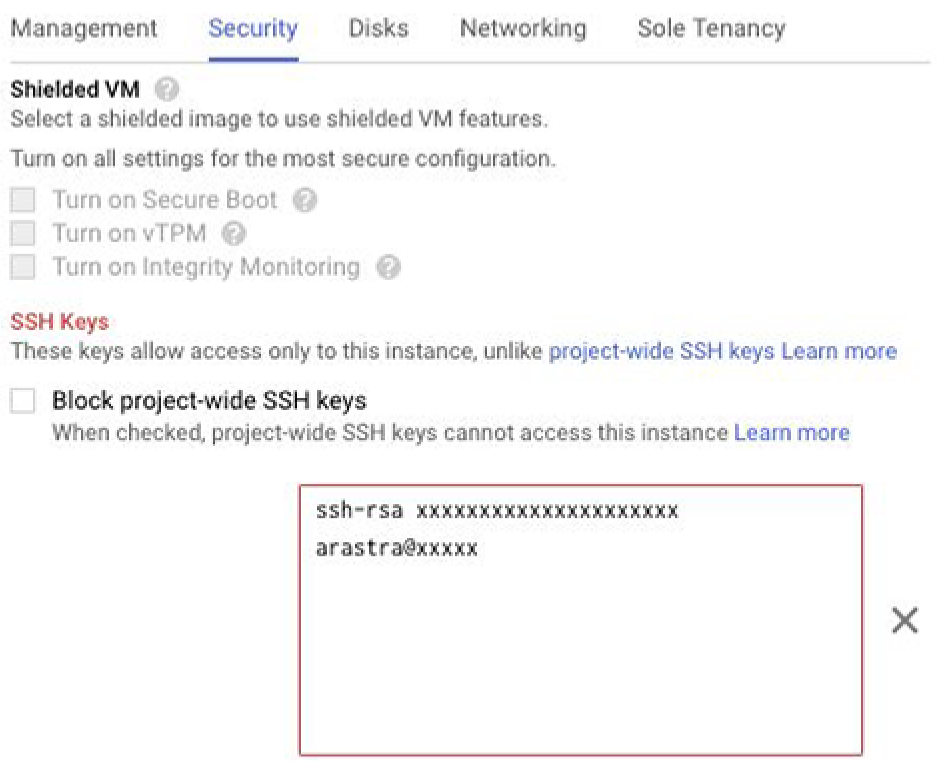
Allow HTTPS traffic

Firewalls setup is not available for multiple network interfaces

Management, security, disks, networking, sole tenancy

The following options have been customized:

7. Click on the **Security** tab, and then copy the SSH key in the text window provided.



Shielded VM

Select a shielded image to use shielded VM features.

Turn on all settings for the most secure configuration.

- [ ] Turn on Secure Boot
- [ ] Turn on vTPM
- [ ] Turn on Integrity Monitoring

SSH Keys

These keys allow access only to this instance, unlike project-wide SSH keys Learn more

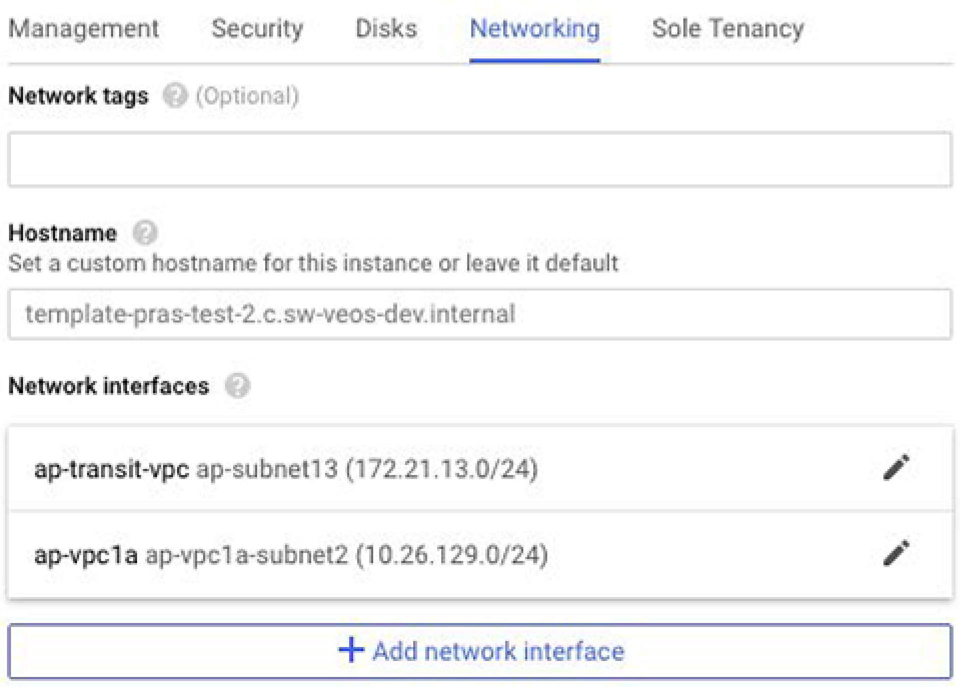
- [ ] Block project-wide SSH keys

  When checked, project-wide SSH keys cannot access this instance Learn more

```
ssh-rsa xxxxxxxxxxxxxxxxxxxxx
arastra@xxxxx
```

8. Click on **Networking** tab, edit the 1st interface configuration to enable IP forwarding and external IP for this interface. Click on **+Add network interface** tab below to add required number of interfaces.

   Note: The number of interfaces are limited based on the machine-type selected. Change the machine type appropriately based on the number of interfaces.



Network tags (Optional)

Hostname

Set a custom hostname for this instance or leave it default

template-pras-test-2.c.sw-veos-dev.internal

Network interfaces

ap-transit-vpc ap-subnet13 (172.21.13.0/24)

ap-vpc1a ap-vpc1a-subnet2 (10.26.129.0/24)

9. Click on the **Create** tab to start Arista CloudEOS with required number of interfaces.

5.4.2 Launching Using Deployment Manager Template

1. To launch instance using deployment manager, cut/paste and store the following deployment sample in a **yaml** file. Please be careful in cut-pasting SSH key as any extra whitespace/newline will not let you log into the instance using SSH key. For example, **mydeployment.yaml** and edit/add fields for your environment such as instance zone, instance type, instance name, networks, subnetworks and so on. You can also add any startup configurations to the config file. The following example adds a user called "testuser" with password "test123" to the launched instance config which is used to login into the instance in addition to the SSH based login.

   Note: Only SSH based GCP authentication is recommended, this is just an example and should not be used in production.

```
-------Deployment Config ----
 resources:

  - name: deploy-arista-three-nic-mgmt

    type: compute.v1.instance

    properties:

      zone: us-east1-b

      machineType: zones/us-east1-b/machineTypes/n1-standard-4

      disks:

      - deviceName: boot

        type: PERSISTENT

        boot: true

        autoDelete: true

        initializeParams:

          sourceImage: https://www.googleapis.com/compute/v1/projects/
sw-veos-public/
                       global/images/arista-cloudeos-4-23-0fx-
10-24-2019
      networkInterfaces:

        - network: global/networks/default

          accessConfigs:

          - name: External NAT

            type: ONE_TO_ONE_NAT


        - network: global/networks/ts-test-vpc-1

          subnetwork: regions/us-east1/subnetworks/sub1

        - network: global/networks/ts-test-vpv-3

          subnetwork: regions/us-east1/subnetworks/ts-test-vpc-3-sub1
      canIpForward: true

      metadata:

        items:

        - key: user-data

          value: |

             %EOS-STARTUP-CONFIG-START%

             username testuser privilege 15 secret test123

             %EOS-STARTUP-CONFIG-END%

        - key: ssh-keys

          value: |

testadmin:ssh-rsa AAAAB3NzaC1yc2EAAAADAQABAAABAQDYP3+wSjsNMwg9l
/yWR/ioRmdMrzSoLTheRQsG3BxDRssdsdshQXcRcw6wAhujKEZaz
T3dPNbABpWK44tJSWSIMWDBer6PIpxME+FCzB3ALaMKdZ9TDU9TiYMGngM5C8
qBfrKixVoPIuRht7NLEosE3b4aNkgi5Fd5aRbCsdkIdZa3KAmPKE2IX
dZqAscccsD5W2Lmhwg7qOXf2JFcGdTwffffMIXh2FMzqDb0vpxbsdubPEN
+A9E6Npj0Q58XiY7roRLTtl1Z9aJtNnQforfD2/OcECBHcrvdj
//rGCPFhT5fVQ6N9tgpGJq/ECxDMDaVG5gLGpmzUrlwgVi7jYu5E8koKOpmtlp
 testadmin@junk.com
```

2. Deploy **mydeployment.yaml:**

```
$gcloud --project <my_project> deployment-manager deployments create --
config mydeployment.yaml
```

3. Get the **IP** address and connect to created instance.

```
$gcloud --project <my_project> compute  instances  describe --zone us-
east1-b deploy-arista-three-nic-mgmt
...check the NAT IP address of first interface ...
$ssh -i <my-ssh-priv-key> testadmin@<instance_nat_ip>
```

Chapter 6

Using the vEOS Router on KVM and ESXi

This chapter describes the system requirements, installation, and configuration procedures for vEOS router on hypervisor.

Server

A server can be either a hardware or software entity.

A ***hardware server*** is the physical computer that executes the virtual machine manager or hypervisor and all the virtual machines, also known as the host machine.

A ***software server*** is the hypervisor or virtual machine manager that hosts and manages the virtual machines. It is also sometimes referred to as the *host*.

VMware ESXi Minimum Server Requirements

x86-64 Server class CPU (32-bit CPUs are not supported) with

- Ethernet NICs must be SR-IOV capable
- BIOS / System Firmware support for SR-IOV
- 8 GB free disk space
- 16 GB RAM
- 4 cores running a minimum 2.4GHz or greater and 16 GB memory
- Intel VT-x and VT-d support

Note: To ensure compatibility, upgrade the ESXi NIC drivers to the latest version provided from VMware.

VMware ESXi SR-IOV based deployment

- Ethernet NICs must be SR-IOV capable
- BIOS / System Firmware support for SR-IOV

KVM Requirements

vEOS is must be deployed on an x86-64 architecture server running KVM hypervisor.

KVM Minimum Server Requirements

8 GB free disk space

16 GB RAM

x86-64 Server class CPU (32-bit CPUs are not supported) with

- Intel VT-x or AMD-V support for CPU Virtualization
- Intel VT-d or AMD-IOMMU support for PCIe passthrough
- Intel AES-NI support
- 4 CPU cores running at 2.4GHz.

KVM SR-IOV Based Deployment

- Ethernet NICs must be SR-IOV capable
- BIOS / System Firmware support for SR-IOV

Supported Topologies

The following scenarios are described in the Hypervisor Chapter

- Launching ESXi using vSphere Web Client
- Launching vEOS on KVM with Linux bridge
- Launching vEOS on KVM with SR-IOV
- Launching vEOS on KVM with PCI-Passthrough

This chapter includes the following sections:

- VMware ESXi Hypervisor
- KVM

6.1 VMware ESXi Hypervisor

Describes the launch sequence for VMware ESXi 6.0 and 6.5.

- Launching VMware ESXi 6.0 and 6.5
- Enabling SR-IOV or PCI Passthrough on ESXi

6.1.1 Launching VMware ESXi 6.0 and 6.5

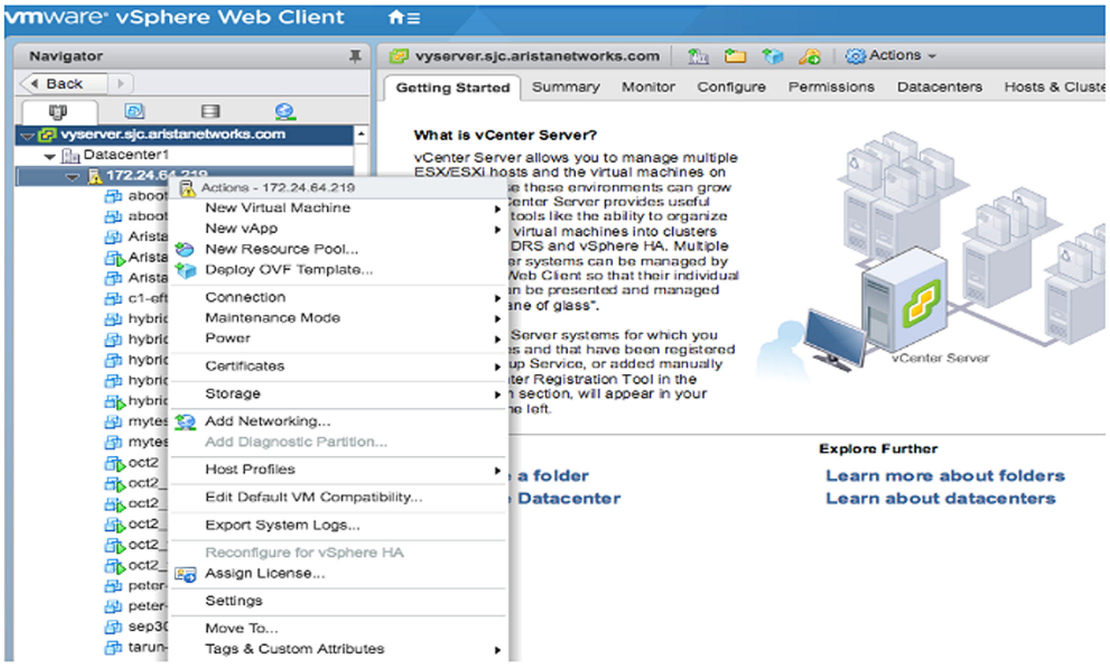
How to launch VMWare ESXi 6 and ESXi 6.5 for vEOS.

There are different ESXi user interfaces for managing the ESXi host, such as the vSphere Web Client and the ESXi Web Client. The following task is required to launch VMware 6.0 and 6.5 and provides a general guideline on the steps involved in deploying virtual machines with an OVF/OVA template.

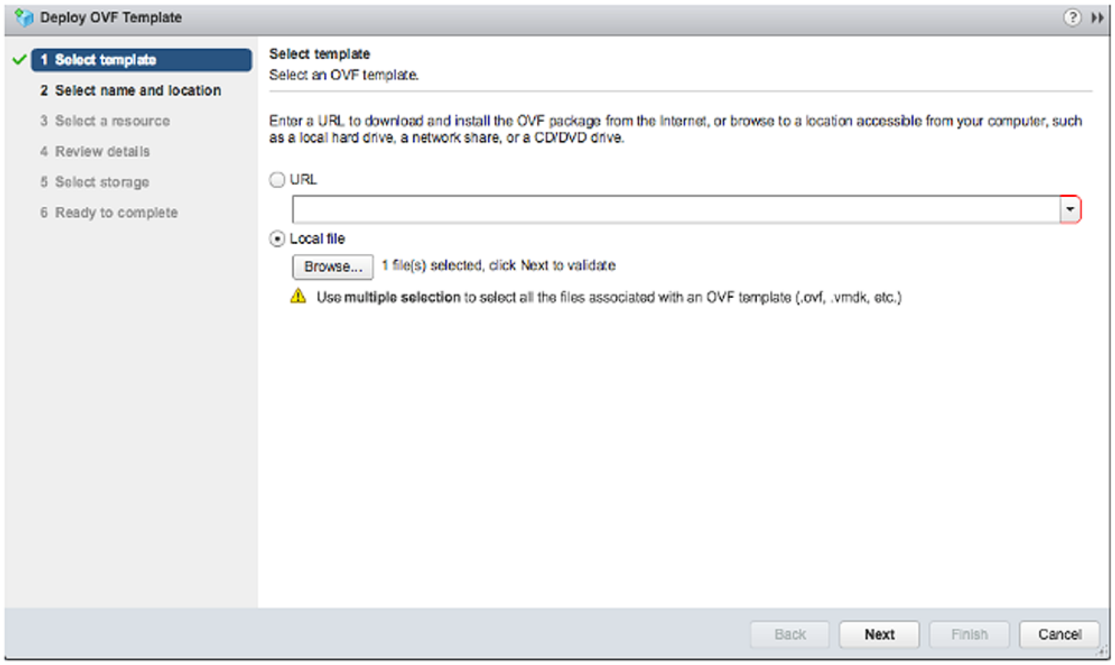
Note: Arista support suggests using only the Vsphere Web client. The ESXi Web Client may have untested issues.

Note: Make sure the VMWare/ESXi Client used for OVA deployment supports the SHA256 hashing algorithm.

1. From the vCenter Server WEB-UI navigator, select **Deploy OVF template**.



2. Select the OVA file from the local machine.



Select template
Select an OVF template.

Enter a URL to download and install the OVF package from the Internet, or browse to a location accessible from your computer, such as a local hard drive, a network share, or a CD/DVD drive.

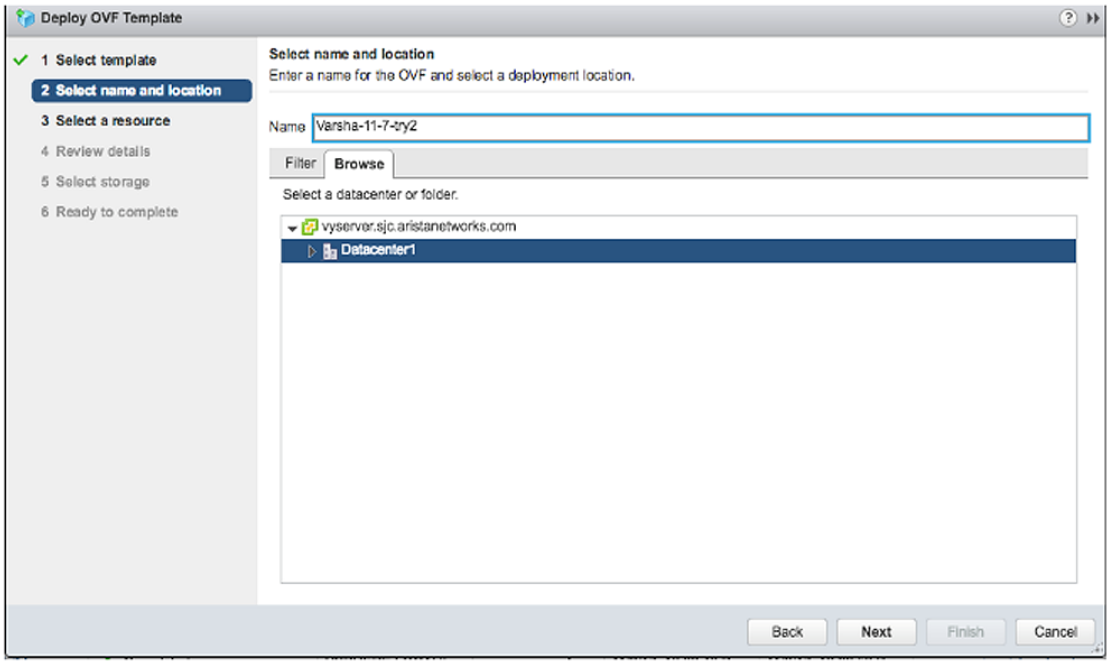
URL

[✓ Local file]

Browse... 1 file(s) selected, click Next to validate

Use **multiple selection** to select all the files associated with an OVF template (.ovf, .vmdk, etc.)

3. Select the name and location for vEOS deployment.



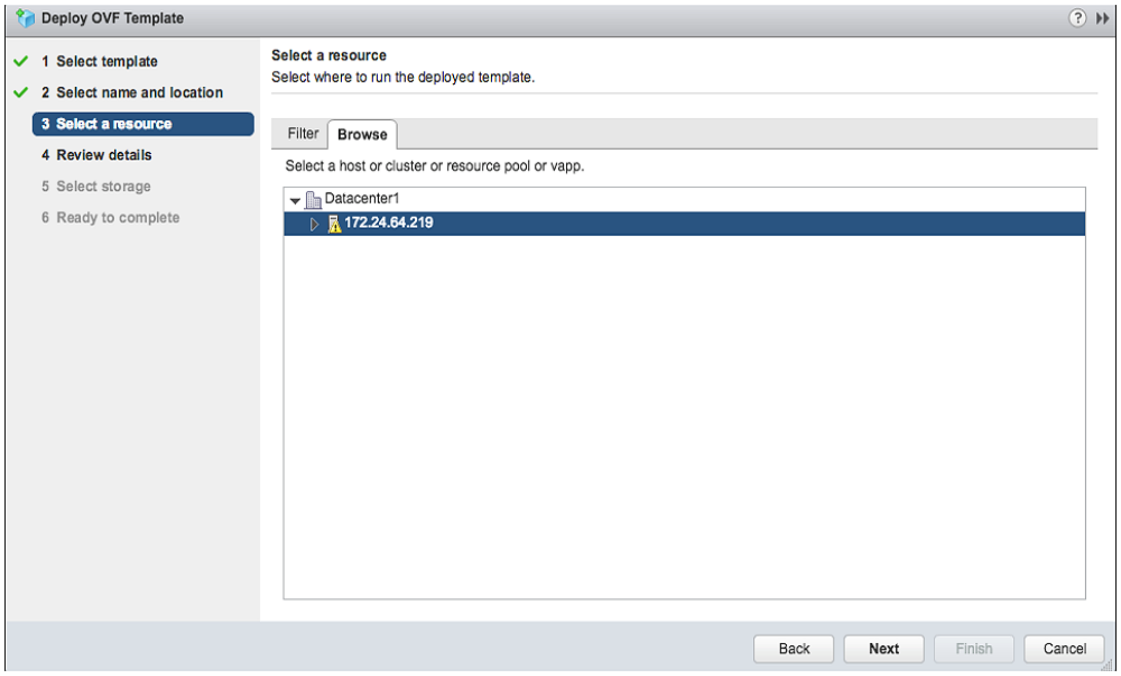
Select name and location
Enter a name for the OVF and select a deployment location.

Name Varsha-11-7-try2

Select a datacenter or folder.

vyserver.sjc.aristanetworks.com
Datacenter1

4. Select the host, cluster, resource pool or VAPP.

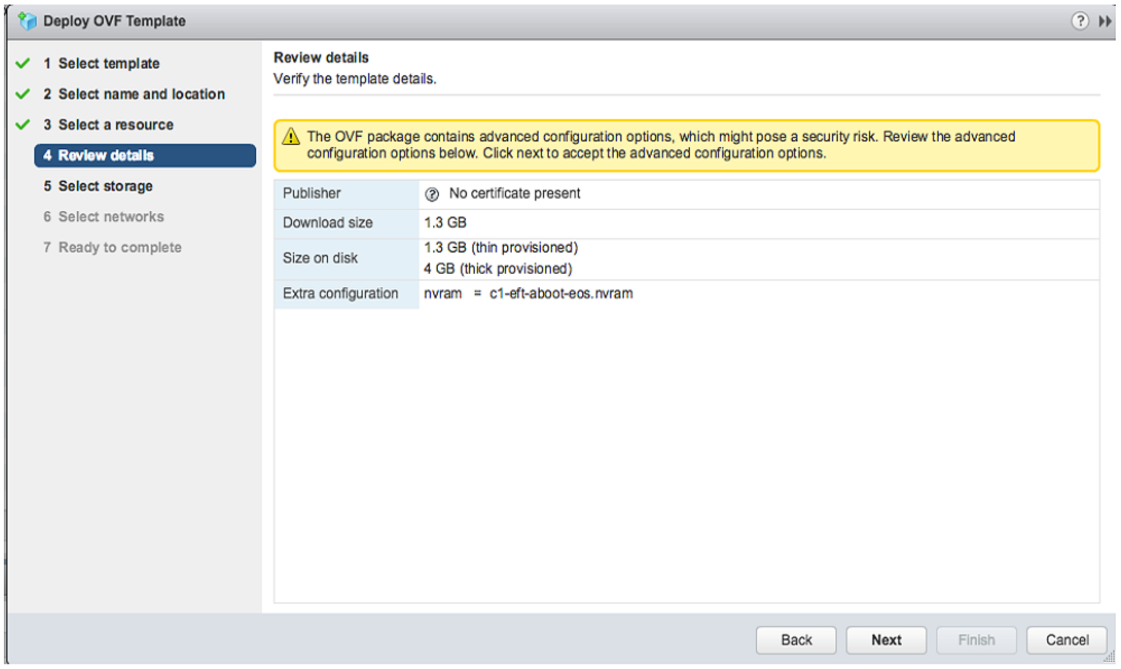


Select a resource

Select where to run the deployed template.

Select a host or cluster or resource pool or vapp.

5. Verify the template details.



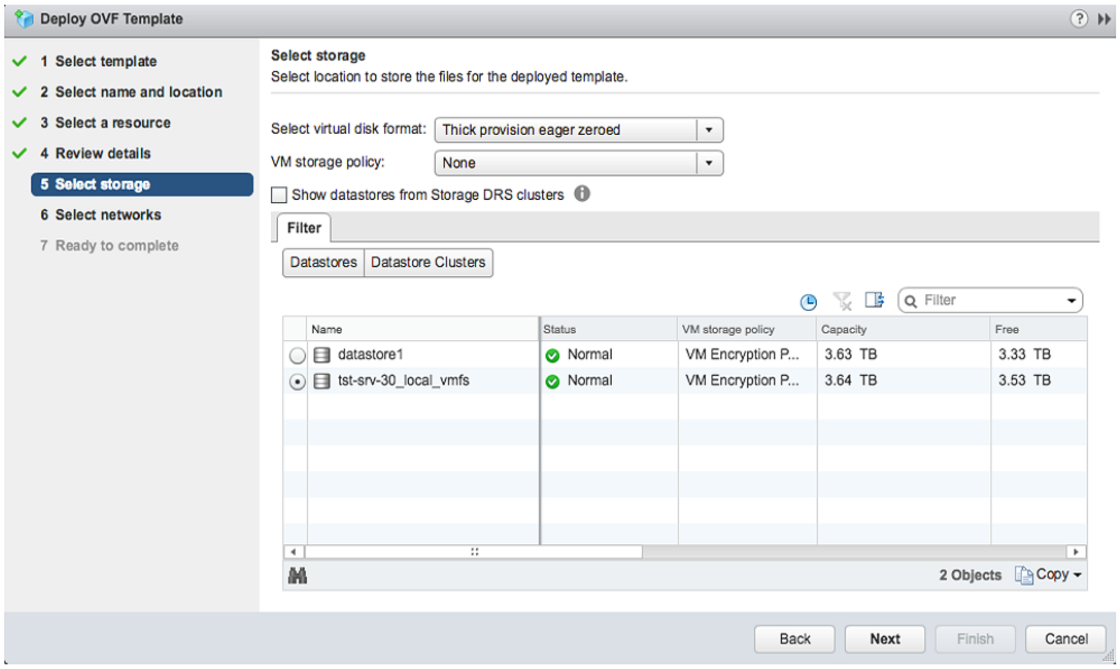
Review details

Verify the template details.

The OVF package contains advanced configuration options, which might pose a security risk. Review the advanced configuration options below. Click next to accept the advanced configuration options.

Publisher	No certificate present
Download size	1.3 GB
Size on disk	1.3 GB (thin provisioned) 4 GB (thick provisioned)
Extra configuration	nvram = c1-eft-aboot-eos.nvram

6. Select **Thick provision eager zeroed** from the datastore.



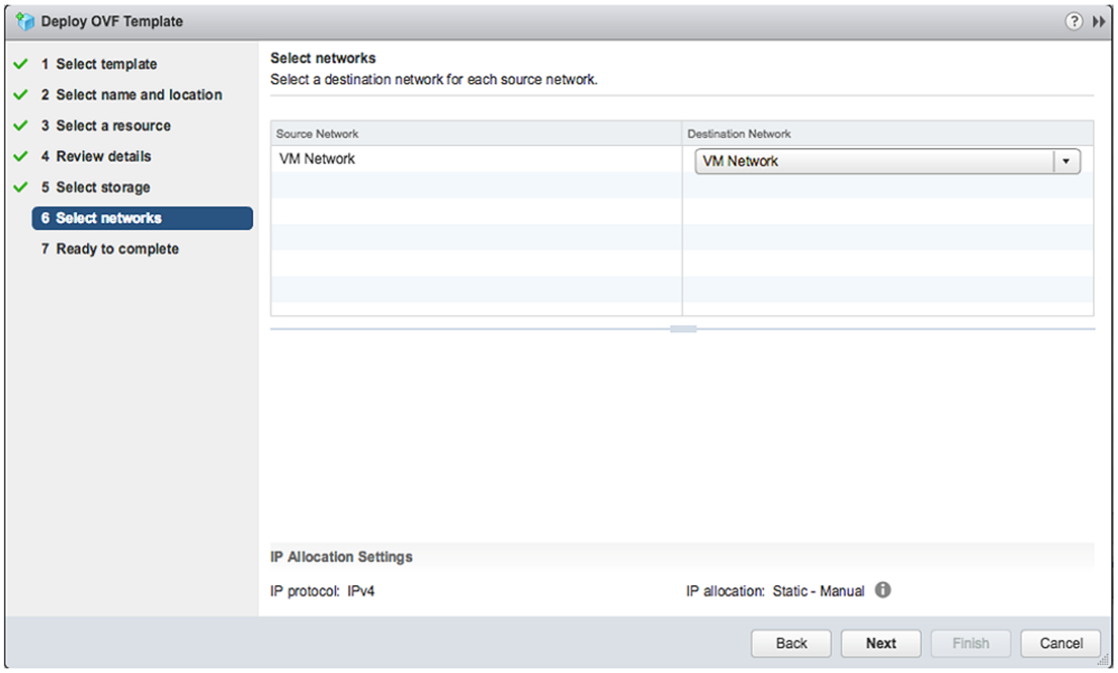
Select storage
Select location to store the files for the deployed template.

Select virtual disk format: Thick provision eager zeroed
VM storage policy: None
Show datastores from Storage DRS clusters

	Name	Status	VM storage policy	Capacity	Free
○	datastore1	Normal	VM Encryption P...	3.63 TB	3.33 TB
●	tst-srv-30_local_vmfs	Normal	VM Encryption P...	3.64 TB	3.53 TB

2 Objects

7. Select the default network.



Select networks
Select a destination network for each source network.

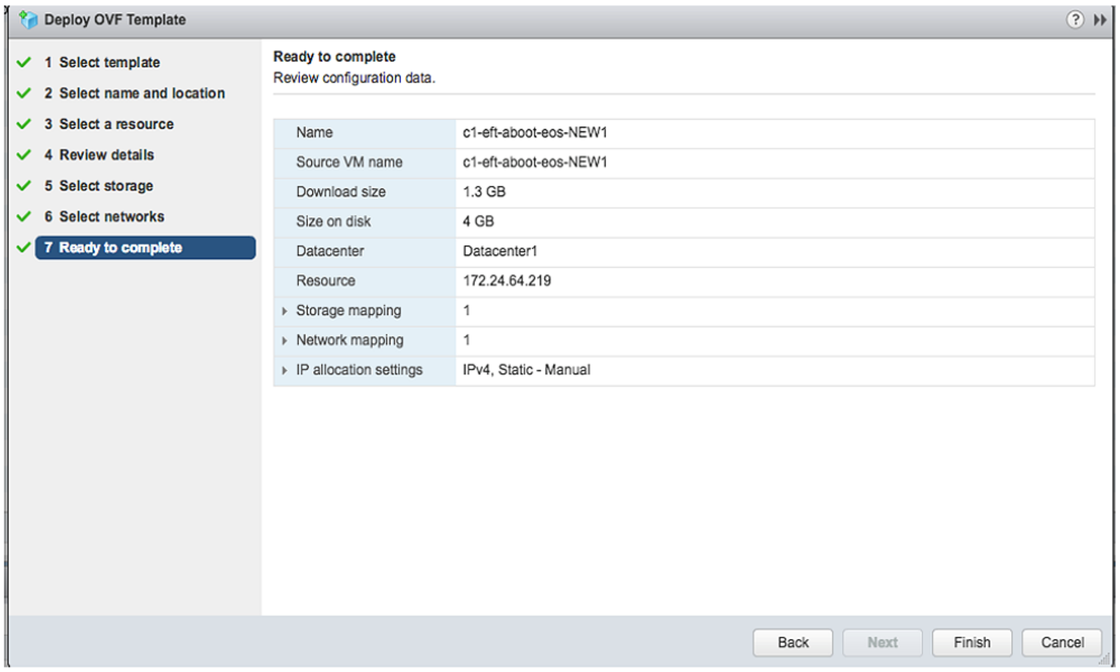
Source Network	Destination Network
VM Network	VM Network

IP Allocation Settings

IP protocol: IPv4

IP allocation: Static - Manual

8. Complete the launch process.

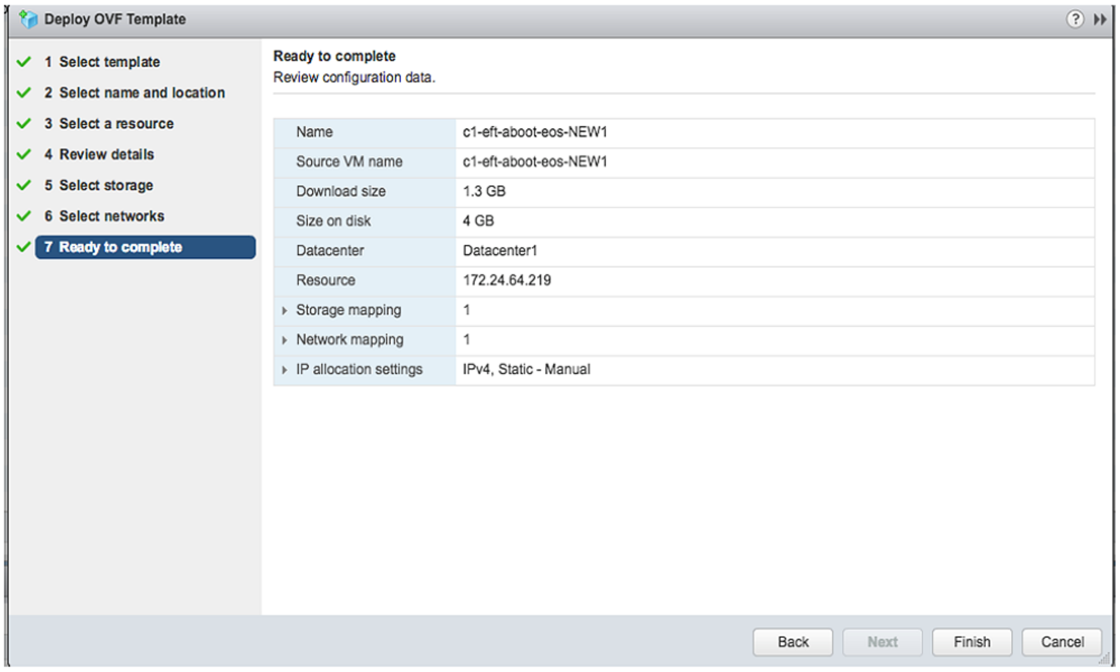

Deploy OVF Template

- ✓ 1 Select template
- ✓ 2 Select name and location
- ✓ 3 Select a resource
- ✓ 4 Review details
- ✓ 5 Select storage
- ✓ 6 Select networks
- ✓ 7 Ready to complete

Ready to complete
Review configuration data.

Name	c1-eft-aboot-eos-NEW1
Source VM name	c1-eft-aboot-eos-NEW1
Download size	1.3 GB
Size on disk	4 GB
Datacenter	Datacenter1
Resource	172.24.64.219
Storage mapping	1
Network mapping	1
IP allocation settings	IPv4, Static - Manual

9. Under the ***Recent Tasks*** tab at the bottom of the page, the progress of deployment displays. Once the deployment is complete, power-on the machine.


Deploy OVF Template

- ✓ 1 Select template
- ✓ 2 Select name and location
- ✓ 3 Select a resource
- ✓ 4 Review details
- ✓ 5 Select storage
- ✓ 6 Select networks
- ✓ 7 Ready to complete

Ready to complete
Review configuration data.

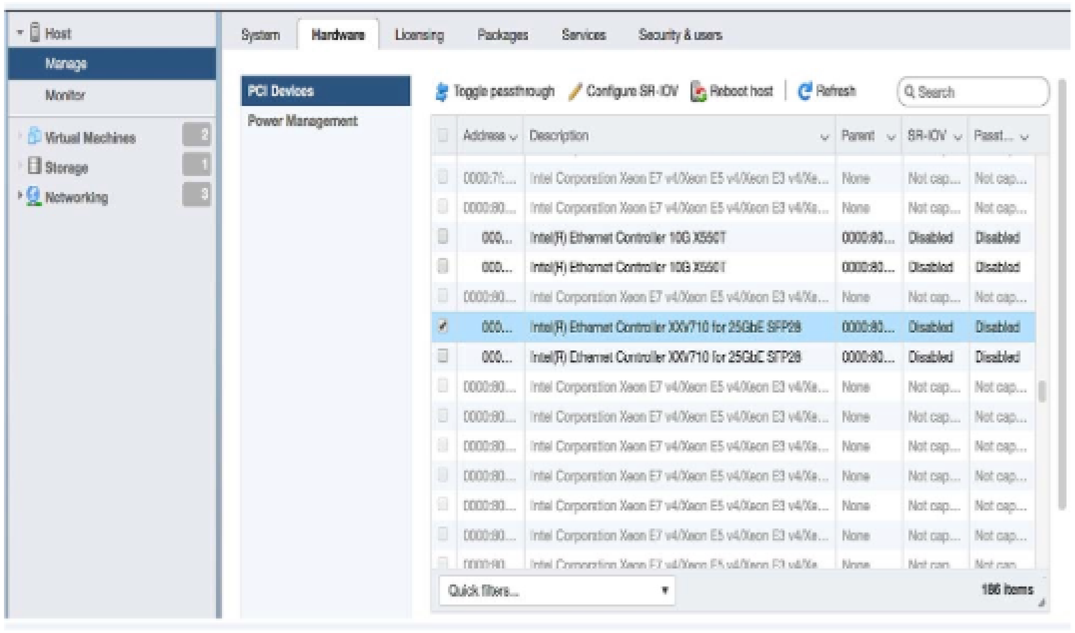
Name	c1-eft-aboot-eos-NEW1
Source VM name	c1-eft-aboot-eos-NEW1
Download size	1.3 GB
Size on disk	4 GB
Datacenter	Datacenter1
Resource	172.24.64.219
Storage mapping	1
Network mapping	1
IP allocation settings	IPv4, Static - Manual

6.1.2 Enabling SR-IOV or PCI Passthrough on ESXi

Describes how to enable single route input/output vitalization (SR-IOV) or PCI passthough on VMware ESXi.

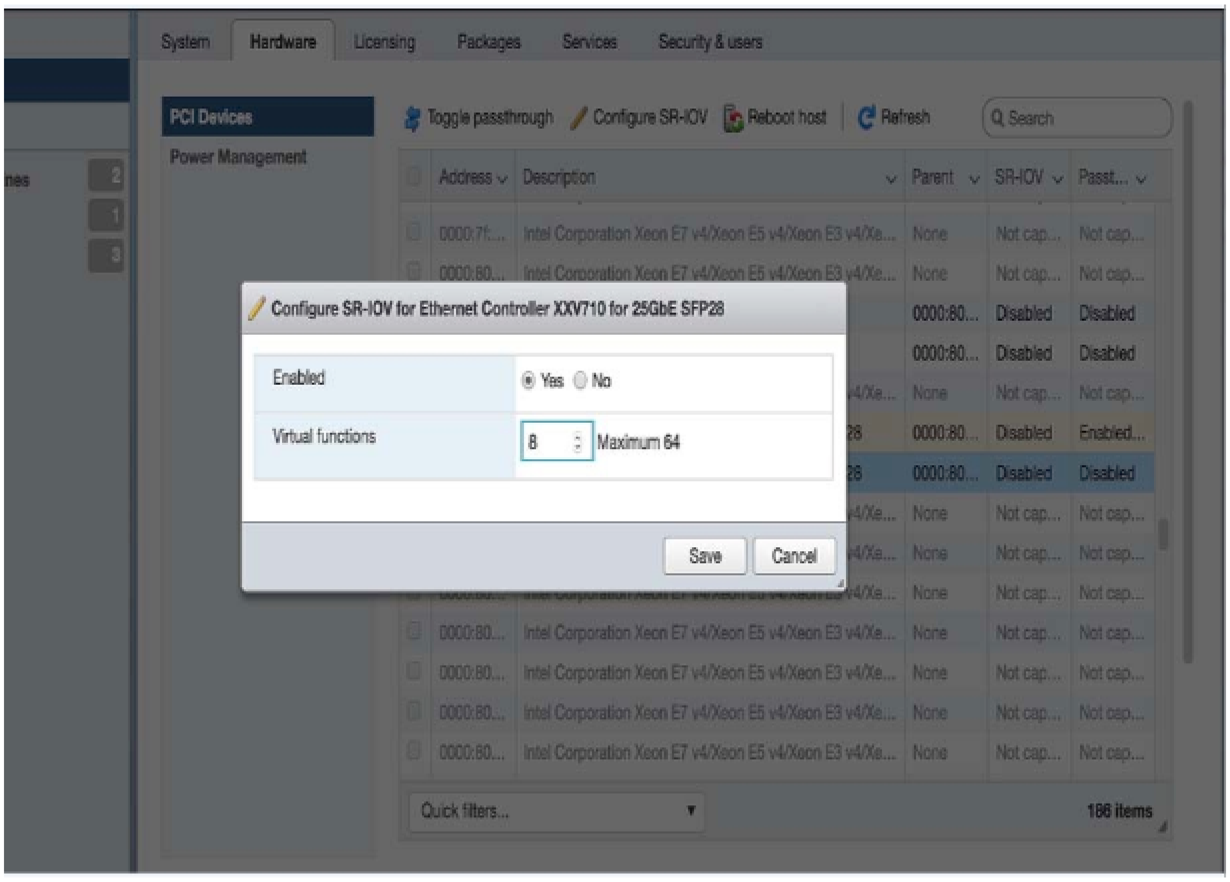
To enable SR-IOV or PCI passthrough on ESXi, complete the following steps.

1. Navigate to the ESXi host's **Manage** , then select the **Hardware** tab.



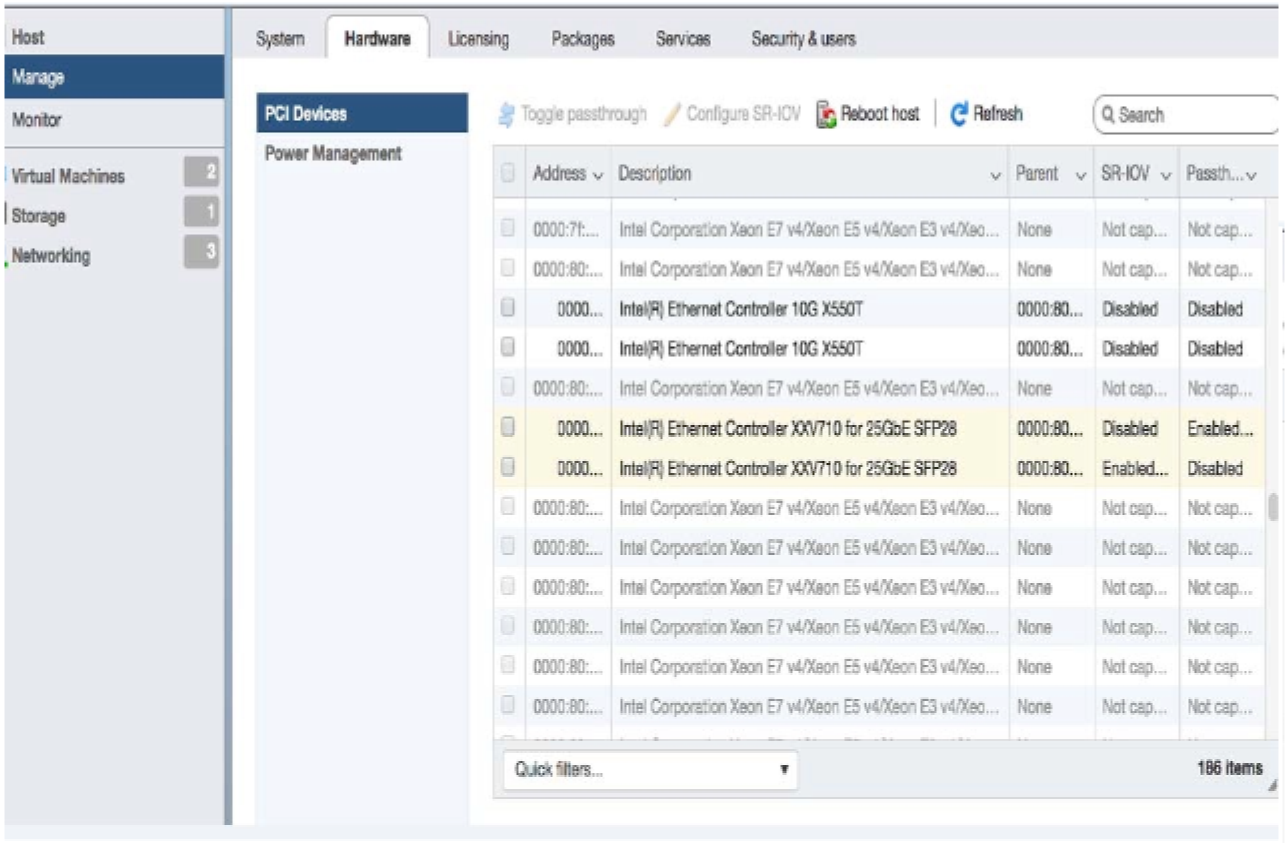
Address	Description	Parent	SR-IOV	Passt...
0000:7f:...	Intel Corporation Xeon E7 v4/Xeon E5 v4/Xeon E3 v4/Xe...	None	Not cap...	Not cap...
0000:80...	Intel Corporation Xeon E7 v4/Xeon E5 v4/Xeon E3 v4/Xe...	None	Not cap...	Not cap...
000...	Intel(R) Ethernet Controller 10G X550T	0000:80...	Disabled	Disabled
000...	Intel(R) Ethernet Controller 10G X550T	0000:80...	Disabled	Disabled
0000:80...	Intel Corporation Xeon E7 v4/Xeon E5 v4/Xeon E3 v4/Xe...	None	Not cap...	Not cap...
000...	Intel(R) Ethernet Controller XXV710 for 25GbE SFP28	0000:80...	Disabled	Disabled
000...	Intel(R) Ethernet Controller XXV710 for 25GbE SFP28	0000:80...	Disabled	Disabled
0000:80...	Intel Corporation Xeon E7 v4/Xeon E5 v4/Xeon E3 v4/Xe...	None	Not cap...	Not cap...
0000:80...	Intel Corporation Xeon E7 v4/Xeon E5 v4/Xeon E3 v4/Xe...	None	Not cap...	Not cap...
0000:80...	Intel Corporation Xeon E7 v4/Xeon E5 v4/Xeon E3 v4/Xe...	None	Not cap...	Not cap...
0000:80...	Intel Corporation Xeon E7 v4/Xeon E5 v4/Xeon E3 v4/Xe...	None	Not cap...	Not cap...
0000:80...	Intel Corporation Xeon E7 v4/Xeon E5 v4/Xeon E3 v4/Xe...	None	Not cap...	Not cap...
0000:80...	Intel Corporation Xeon E7 v4/Xeon E5 v4/Xeon E3 v4/Xe...	None	Not cap...	Not cap...

2. Locate and select your PIC device/NIC.
3. Use either the **Toggle passthrough** or the **Configure SR-IOV** selection to activate the mode.



Configure SR-IOV for Ethernet Controller XXV710 for 25GbE SFP28

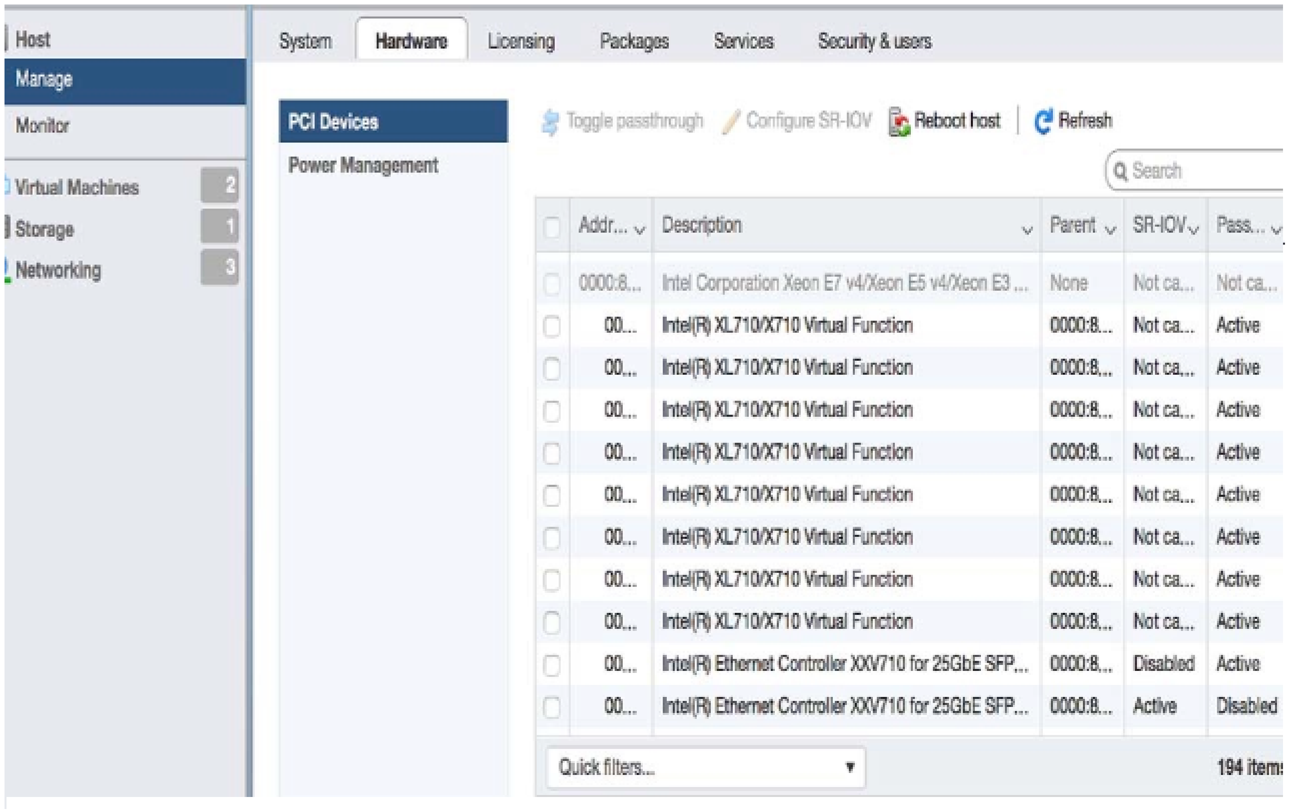
Enabled	[✓ Yes] No
Virtual functions	8 Maximum 64



	Address	Description	Parent	SR-IOV	Passth...
	0000:7f:...	Intel Corporation Xeon E7 v4/Xeon E5 v4/Xeon E3 v4/Xeo...	None	Not cap...	Not cap...
	0000:80:...	Intel Corporation Xeon E7 v4/Xeon E5 v4/Xeon E3 v4/Xeo...	None	Not cap...	Not cap...
	0000...	Intel(R) Ethernet Controller 10G X550T	0000:80...	Disabled	Disabled
	0000...	Intel(R) Ethernet Controller 10G X550T	0000:80...	Disabled	Disabled
	0000:80:...	Intel Corporation Xeon E7 v4/Xeon E5 v4/Xeon E3 v4/Xeo...	None	Not cap...	Not cap...
	0000...	Intel(R) Ethernet Controller XXV710 for 25GbE SFP28	0000:80...	Disabled	Enabled...
	0000...	Intel(R) Ethernet Controller XXV710 for 25GbE SFP28	0000:80...	Enabled...	Disabled
	0000:80:...	Intel Corporation Xeon E7 v4/Xeon E5 v4/Xeon E3 v4/Xeo...	None	Not cap...	Not cap...
	0000:80:...	Intel Corporation Xeon E7 v4/Xeon E5 v4/Xeon E3 v4/Xeo...	None	Not cap...	Not cap...
	0000:80:...	Intel Corporation Xeon E7 v4/Xeon E5 v4/Xeon E3 v4/Xeo...	None	Not cap...	Not cap...
	0000:80:...	Intel Corporation Xeon E7 v4/Xeon E5 v4/Xeon E3 v4/Xeo...	None	Not cap...	Not cap...
	0000:80:...	Intel Corporation Xeon E7 v4/Xeon E5 v4/Xeon E3 v4/Xeo...	None	Not cap...	Not cap...
	0000:80:...	Intel Corporation Xeon E7 v4/Xeon E5 v4/Xeon E3 v4/Xeo...	None	Not cap...	Not cap...

186 items

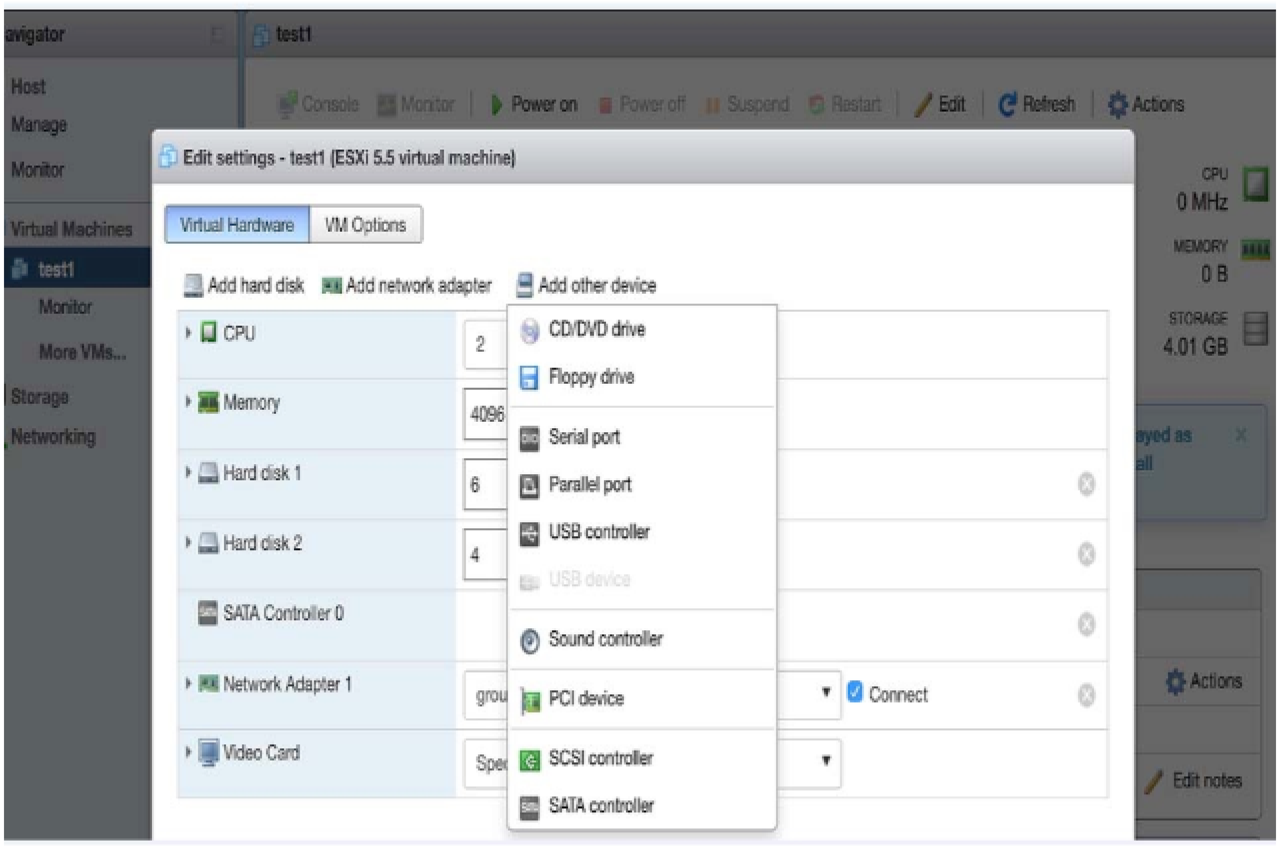
4. Reboot the ESXi host for the configuration to take effect.
5. After reboot, the NIC reflects the changes. For SR-IOV, new virtual function devices (VF) is created.



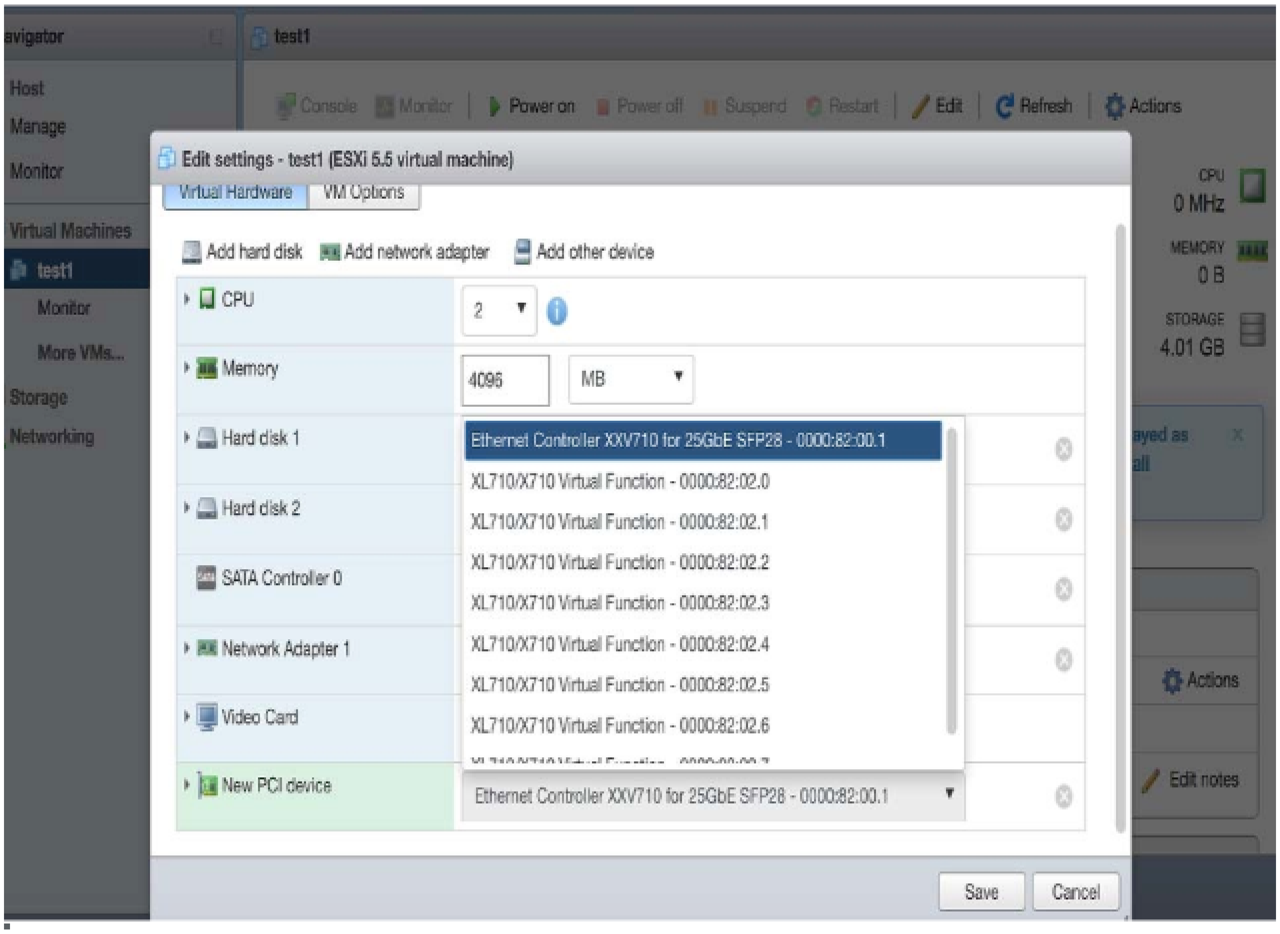
	Addr...	Description	Parent	SR-IOV	Pass...
	0000:8...	Intel Corporation Xeon E7 v4/Xeon E5 v4/Xeon E3 ...	None	Not ca...	Not ca...
	00...	Intel(R) XL710/X710 Virtual Function	0000:8...	Not ca...	Active
	00...	Intel(R) XL710/X710 Virtual Function	0000:8...	Not ca...	Active
	00...	Intel(R) XL710/X710 Virtual Function	0000:8...	Not ca...	Active
	00...	Intel(R) XL710/X710 Virtual Function	0000:8...	Not ca...	Active
	00...	Intel(R) XL710/X710 Virtual Function	0000:8...	Not ca...	Active
	00...	Intel(R) XL710/X710 Virtual Function	0000:8...	Not ca...	Active
	00...	Intel(R) XL710/X710 Virtual Function	0000:8...	Not ca...	Active
	00...	Intel(R) XL710/X710 Virtual Function	0000:8...	Not ca...	Active
	00...	Intel(R) Ethernet Controller XXV710 for 25GbE SFP...	0000:8...	Disabled	Active
	00...	Intel(R) Ethernet Controller XXV710 for 25GbE SFP...	0000:8...	Active	Disabled

194 items

6. Edit the VM and select **Add other device**, then select **PIC Device** to create the **New PIC Device** for the VM.



7. Select the **New PIC Device** to use the SR-IOV VF or PIC Passthrough device.



Edit settings - test1 (ESXi 5.5 virtual machine)

Virtual Hardware | VM Options

Add hard disk | Add network adapter | Add other device

CPU	2
Memory	4096 MB
Hard disk 1	
Hard disk 2	
SATA Controller 0	
Network Adapter 1	
Video Card	
New PCI device	Ethernet Controller XXV710 for 25GbE SFP28 - 0000:82:00.1

Ethernet Controller XXV710 for 25GbE SFP28 - 0000:82:00.1
XL710/X710 Virtual Function - 0000:82:02.0
XL710/X710 Virtual Function - 0000:82:02.1
XL710/X710 Virtual Function - 0000:82:02.2
XL710/X710 Virtual Function - 0000:82:02.3
XL710/X710 Virtual Function - 0000:82:02.4
XL710/X710 Virtual Function - 0000:82:02.5
XL710/X710 Virtual Function - 0000:82:02.6

Save | Cancel

6.2 KVM

This section describes the system requirements, installation and configuration procedures for CloudEOS and vEOS.

Server

A server can be either a hardware or software entity.

A ***hardware server*** is the physical computer that executes the virtual machine manager or hypervisor and all the virtual machines. This is also known as the host machine.

A ***software server*** is the hypervisor or virtual machine manager that hosts and manages the virtual machines. It is also sometimes referred to as the *host*. In this document specifically, the software server is comprised of RedHat Linux with virtualization support (KVM).

6.2.1 System Requirements

Below are the minimum system requirements for using KVM**.**

Minimum Server Requirements

Any VMware supported ESXi server hardware.

Hypervisor support

- RedHat 7x with virtualization support. Please see below for virtualization https://wiki.centos.org/HowTos/KVM.

- **Libvirt** is installed by executing **virsh list** which should return without errors. Python 2.7+ is required to run the installation script vSphere 6.0.

vEOS Virtual Machine

Minimum requirements:

- 2 vCPUs
- 4GB Memory
- 8G Free disk space

Maximum capacities

- 16 vCPUs
- 8 network interfaces

Supported Images

Image Name	File Name	Details
KVM vEOS image	`EOS.qcow2`	Image Hard Disk that contains vEOS. This file can grow as agents in vEOS generates logs/traces, etc.

6.2.2 Using Libvirt to Manage CloudEOS and vEOS VM on KVM

Libvirt is an open source library which provides CloudEOS and vEOS management of Virtual Machines.

Libvirt supports many functions such as creation, update, and deletion and of VMs.

The complete Libvirt command reference can be found at http://libvirt.org/virshcmdref.html

Define a new VM

Define a domain from an XML file, by using the **`virsh define <vm-definition-file.xml >`** command. This defines the domain, but it does not start the domain.

The definition file has vm-name, CPU, memory, network connectivity, and a path to the image. The parameters can be found at https://libvirt.org/formatdomain.html. There is a sample CloudEOS and vEOS file in the example below.

Undefine the Inactive Domain

Undefine the configuration for the inactive domain by using the **`virsh undefine <vm-name>`** and specifying its domain name.

Start VM

Start a previously defined or inactive domain by using the **`virsh start <vm-name>`** command.

Stop VM

Terminate a domain immediately by using the **`virsh destroy <vm-name>`** command

Managing Networks

The XML definition format for networks is defined at https://libvirt.org/formatnetwork.html. These commands are similar to the VM, but with a prefix '**net-**' :

The **`virsh net-define <network-definition-file.xml>`** command.

The **`virsh net-undefine network-name`** command removes an inactive virtual network from the libvirt configuration.

The **`virsh start network-name`** command manually starts a virtual network that is not running.

The **`virsh destroy network-name`** command shuts down a running virtual network.

6.2.3 Launching vEOS in LinuxBridge Mode

Use the script **SetupLinuxBridge.pyc usage python SetupLinuxBridge.pyc** <bridge- name>

Cut and paste the following XML template into a file (veos.xml) and customize the elements that are in **bold** below.

- virsh define <veos define file say veos.xml>
- virsh start <veos-name>
- virsh console <veos-name>

```
<domain type='kvm'>
<!-- veos name, cpu and memory settings -->
<name>kvs1-veos1</name>
<memory unit='MiB'>4096</memory>
<currentMemory unit='MiB'>4096</currentMemory>
<vcpu placement='static'>2</vcpu>
<resource>
<partition>/machine</partition>
</resource>
<cpu mode='host-model'/>
<os>
<type arch='x86_64’>hvm</type>
<boot dev='cdrom'/>
<boot dev='hd'/>
</os>
<features>
<acpi/>
<apic/>
<pae/>
</features>
<clock offset='utc'/>
<on_poweroff>destroy</on_poweroff>
<on_reboot>restart</on_reboot>
<on_crash>restart</on_crash>
<devices>
<emulator>/usr/bin/qemu-system-x86_64</emulator>
<disk type='file' device='disk'>
<driver name='qemu' type='qcow2' cache='directsync'/>
<source file="/path-to-veos-image/EOS.qcow2"/>
<target dev='hda' bus='ide'/>
<alias name='ide0-0-0'/>
<address type='drive' controller='0' bus='0' target='0' unit='0'
</disk>
<disk type='file' device='cdrom'>
<driver name='qemu' type='raw'/>
<source file="/path-to-aboot-image/Aboot-veos-serial.iso"/>
<target dev='hdc' bus='ide'/>

<readonly/>
<alias name='ide0-1-0'/>
<address type='drive' controller='0' bus='1' target='0' unit='0'
</disk>
<controller type='usb' index='0'>
<alias name='usb0'/>
<address type='pci' domain='0x0000' bus='0x00' slot='0x01' function='0
</controller>
<controller type='pci' index='0' model='pci-root'>
<alias name='pci0'/>
</controller>
<controller type='ide' index='0'>
<alias name='ide0'/>
<address type='pci' domain='0x0000' bus='0x00' slot='0x01' funct
```

```
</controller>
<!-- In this case management is connected to linux bridge -->
<interface type='bridge'>
<source bridge='brMgmt'/>
<model type='virtio'/>
<address type='pci' domain='0x0000' bus='0x00' slot='0x03' funct
</interface>
<serial type='pty'>
<source path='/dev/pts/4'/>
<target port='0'/>
<alias name='serial0'/>
<target port='0'/>
<alias name='serial0'/>
</serial>
<console type='pty' tty='/dev/pts/4'>
<source path='/dev/pts/4'/>
<target type='serial' port='0'/>
<alias name='serial0'/>
</console>
<input type='mouse' bus='ps2'/>
<graphics type='vnc' port='5903' autoport='yes' listen='127.0.0.1'
<listen type='address' address='127.0.0.1'/>
</graphics>
<video>
<model type='cirrus' vram='9216' heads='1'/>
<alias name='video0'/>
<address type='pci' domain='0x0000' bus='0x00' slot='0x02' funct
</video>
<memballoon model='virtio'>
<alias name='balloon0'/>
<address type='pci' domain='0x0000' bus='0x00' slot='0x04' funct
</memballoon>
<!-- Has two data ports on different vlans

Cut and paste the more interface elements for more interfaces but
 increment the slot number.
Note that brWAN and brLAN bridges need to be created beforehand -->
<interface type='bridge'>
<source bridge='brWAN'/>
<model type='virtio'/>
<address type='pci' domain='0x0000' bus='0x00' slot='5' function
</interface>
<interface type='bridge'>
<source bridge='brLAN'/>
<model type='virtio'/>
<address type='pci' domain='0x0000' bus='0x00' slot='6' function
</interface>
</devices>
</domain>

<!-- veos name, cpu and memory settings -->
<name>kvs1-veos1</name>
<memory unit='MiB'>4096</memory>
<currentMemory unit='MiB'>4096</currentMemory>
<vcpu placement='static'>2</vcpu>
<resource>
<partition>/machine</partition>
</resource>
<cpu mode='host-model'/>
<os>
<type arch='x86_64’>hvm</type>
<boot dev='cdrom'/>
<boot dev='hd'/>
```

```
</os>
<features>
<acpi/>
<apic/>
<pae/>
</features>
<clock offset='utc'/>
<on_poweroff>destroy</on_poweroff>
<on_reboot>restart</on_reboot>
<on_crash>restart</on_crash>
<devices>
<emulator>/usr/bin/qemu-system-x86_64</emulator>
<disk type='file' device='disk'>
<driver name='qemu' type='qcow2' cache='directsync'/>
<source file="/path-to-veos-image/EOS.qcow2"/>
<target dev='hda' bus='ide'/>
<alias name='ide0-0-0'/>
<address type='drive' controller='0' bus='0' target='0' unit='0'
</disk>
<disk type='file' device='cdrom'>
<driver name='qemu' type='raw'/>
<source file="/path-to-aboot-image/Aboot-veos-serial.iso"/>
<target dev='hdc' bus='ide'/>

<readonly/>
<alias name='ide0-1-0'/>
<address type='drive' controller='0' bus='1' target='0' unit='0'
</disk>
<controller type='usb' index='0'>
<alias name='usb0'/>
<address type='pci' domain='0x0000' bus='0x00' slot='0x01' function='0
</controller>
<controller type='pci' index='0' model='pci-root'>
<alias name='pci0'/>
</controller>
<controller type='ide' index='0'>
<alias name='ide0'/>
<address type='pci' domain='0x0000' bus='0x00' slot='0x01' funct
</controller>
<!-- In this case management is connected to linux bridge -->
<interface type='bridge'>
<source bridge='brMgmt'/>
<model type='virtio'/>
<address type='pci' domain='0x0000' bus='0x00' slot='0x03' funct
</interface>
<serial type='pty'>
<source path='/dev/pts/4'/>
<target port='0'/>
<alias name='serial0'/>
<target port='0'/>
<alias name='serial0'/>
</serial>
<console type='pty' tty='/dev/pts/4'>
<source path='/dev/pts/4'/>
<target type='serial' port='0'/>
<alias name='serial0'/>
</console>
<input type='mouse' bus='ps2'/>
<graphics type='vnc' port='5903' autoport='yes' listen='127.0.0.1'
<listen type='address' address='127.0.0.1'/>
</graphics>
<video>
<model type='cirrus' vram='9216' heads='1'/>
```

```
<alias name='video0'/>
<address type='pci' domain='0x0000' bus='0x00' slot='0x02' funct
</video>
<memballoon model='virtio'>
<alias name='balloon0'/>
<address type='pci' domain='0x0000' bus='0x00' slot='0x04' funct
</memballoon>
<!-- Has two data ports on different vlans

Cut and paste the more interface elements for more interfaces but
 increment the slot number.
Note that brWAN and brLAN bridges need to be created beforehand -->
<interface type='bridge'>
<source bridge='brWAN'/>
<model type='virtio'/>
<address type='pci' domain='0x0000' bus='0x00' slot='5' function
</interface>
<interface type='bridge'>
<source bridge='brLAN'/>
<model type='virtio'/>
<address type='pci' domain='0x0000' bus='0x00' slot='6' function
</interface>
</devices>
</domain>
```

Example Deployment

VIRTIO & Linux Bridging Deployment

vEOS can employ para-virtualized network I/O interfaces, which in Linux KVM is also known as Virtio . Each NIC is connected to a unique underlying Linux layer-2 bridge in the hypervisor which in-turn provides access to an uplink.

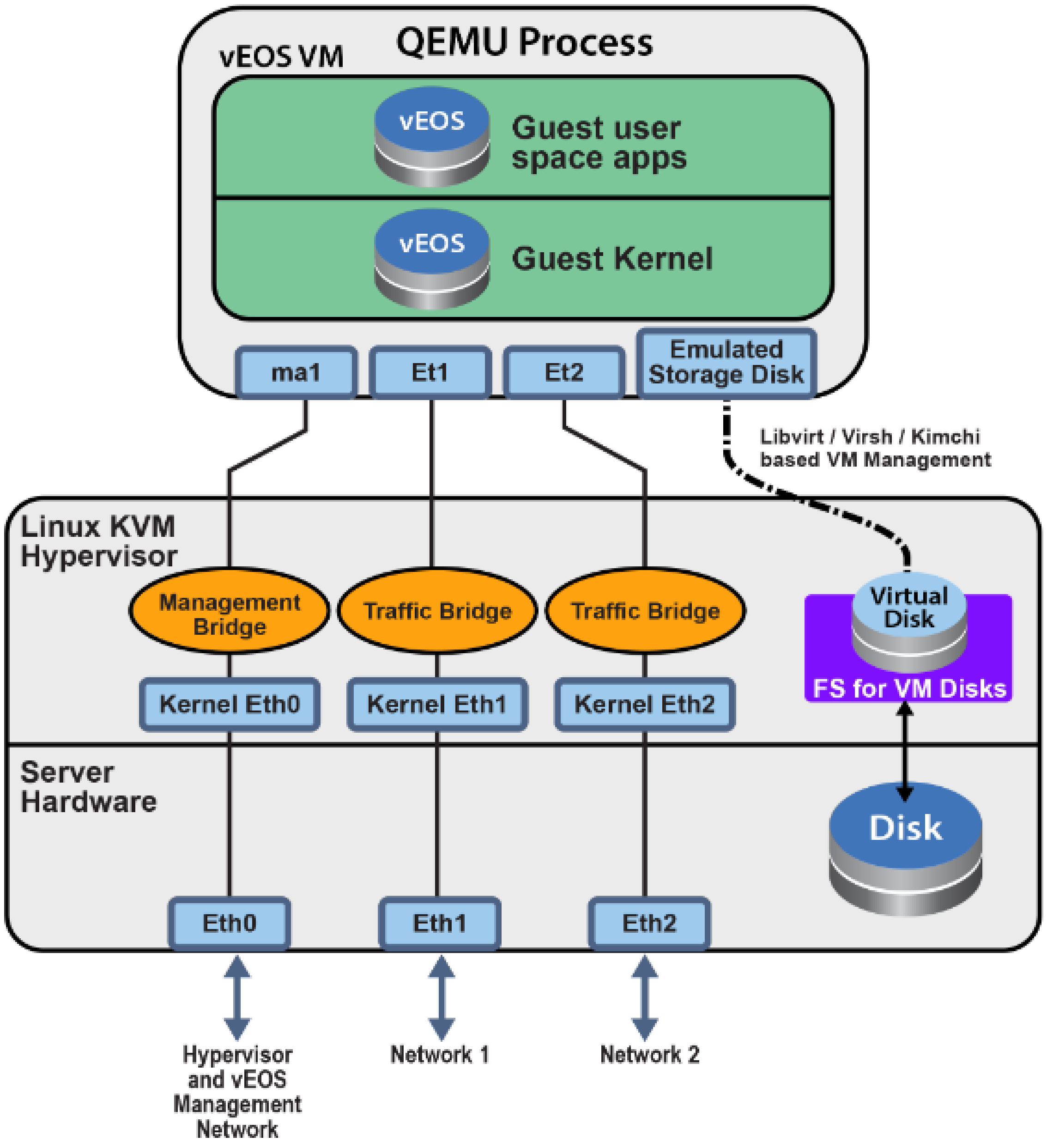
In this example,

- Ethernet1 connects to the physical Ethernet port that connects to the WAN through a LinuxBridge. The Router is configured with a WAN IP address on this port.
- Ethernet2 connects to the physical ethernet port that connects to the LAN through a LinuxBridge.
- Server IP address in the diagram is assumed to be configured on the LAN LinuxBridge device.

Note: Arista recommends using Ethernet1 for WAN and Ethernet2 for LAN. However, any vEOS port can be used.

Figure 14: Linux Bridge and Virtio-based Deployment

Linux Bridge and Virtio-based Deployment



6.2.4 Setting Up the Host for Single Root I/O Virtualization (SR-IOV)

Single Root I/O Virtualization (SR-IOV) allows a single PCIe physical device under a single root port to appear to be multiple physical devices to the hypervisor.

The following tasks are required to set up the host for SR-IOV.

1. Verify the IOMMU Support.

Use the **`virt-host-validate`** Linux command to check IOMMU (input/output memory management unit) support. If it does not "PASS" for IOMMU, check the BIOS setting and kernel settings.

The example below is what should be displayed.

```
[arista@solution]$ virt-host-validate
  QEMU: Checking for device assignment IOMMU support : PASS
  QEMU: Checking if IOMMU is enabled by kernel       : PASS
```

2. Verify the Drivers are Supported.

Ensure the PCI device with SR-IOV capabilities is detected. In the example below, an INTEL **82599** ES network interface card is detected which supports SR-IOV.

Verify the ports and NIC IDs that are in bold in the **`lspci | grep Ethernet`** Linux command output below.

```
# lspci | grep Ethernet
01:00.0 Ethernet controller: Intel Corporation I350 Gigabit Network
 Connection (rev 01)
01:00.1 Ethernet controller: Intel Corporation I350 Gigabit Network
 Connection (rev 01)
01:00.2 Ethernet controller: Intel Corporation I350 Gigabit Network
 Connection (rev 01)
01:00.3 Ethernet controller: Intel Corporation I350 Gigabit Network
 Connection (rev 01)
81:00.0 Ethernet controller: Intel Corporation I350 Gigabit Network
 Connection (rev 01)
81:00.1 Ethernet controller: Intel Corporation I350 Gigabit Network
 Connection (rev 01)
81:00.2 Ethernet controller: Intel Corporation I350 Gigabit Network
 Connection (rev 01)
81:00.3 Ethernet controller: Intel Corporation I350 Gigabit Network
 Connection (rev 01)
82:00.0 Ethernet controller: Intel Corporation 82599ES 10-Gigabit SFI/SFP
+ Network Connection (rev 01)
82:00.1 Ethernet controller: Intel Corporation 82599ES 10-Gigabit SFI/SFP
+ Network Connection (rev 01)
83:00.0 Ethernet controller: Intel Corporation 82599ES 10-Gigabit SFI/SFP
+ Network Connection (rev 01)
83:00.1 Ethernet controller: Intel Corporation 82599ES 10-Gigabit SFI/SFP
+ Network Connection (rev 01)
```

3. Verify the driver kernel is active.

After confirming the device support, the driver kernel module should load automatically by the kernel. To verify the driver kernel is active, use the **`lsmod | grep igb`** Linux command.

```
[root@kvmsolution]# lsmod | grep igb
igb                   197328  0
ptp                    19231  2 igb,ixgbe
dca                    15130  2 igb,ixgbe
i2c_algo_bit           13413  2 ast,igb
i2c_core               40756  6 ast,drm,igb,i2c_i801,drm_kms_helper,i2c
_algo_bit
```

4. Activate Virtual Functions (VFs).

The maximum number of supported virtual functions depends on the type of card. To activate the VFs use `[arista@localhost]$ /sys/class/net/<Device_Name>/device/sriov_numvfs` or the method shown in the example below, it shows that the PF identifier ***82:00.0*** supports a total of **63** VFs.

Example

```
[arista@localhost]$ cat/sys/bus/pci/devices/0000\:82\:00.0/sriov_totalvf
s 63
```

To activate the seven VFs per PFs and make them persistent after reboot, add the line options `igb max_vfs=7` in `ixgbe.conf` and the `sriov.conf` files in `/etc/modprobe.d`

Use the **`rmmod ixgbe`** and **`modprobe ixgbe`** Linux commands to unload and reload the module.

5. Verify the VFs are detected.

Verify the VFs are detected by using the **`lspci | grep Ethernet`** Linux command. For the two identifiers ***82:00.0*** and ***82:00.1***, 14 VFs are detected.

```
# lspci | grep Ethernet
82:00.0 Ethernet controller: Intel Corporation 82599ES 10-Gigabit SFI/SFP
+ Network Connection (rev 01)
82:00.1 Ethernet controller: Intel Corporation 82599ES 10-Gigabit SFI/SFP
+ Network Connection (rev 01)
82:10.0 Ethernet controller: Intel Corporation 82599 Ethernet Controller
 Virtual Function (rev 01)
82:10.1 Ethernet controller: Intel Corporation 82599 Ethernet Controller
 Virtual Function (rev 01)
82:10.2 Ethernet controller: Intel Corporation 82599 Ethernet Controller
 Virtual Function (rev 01)
82:10.3 Ethernet controller: Intel Corporation 82599 Ethernet Controller
 Virtual Function (rev 01)
82:10.4 Ethernet controller: Intel Corporation 82599 Ethernet Controller
 Virtual Function (rev 01)
82:10.5 Ethernet controller: Intel Corporation 82599 Ethernet Controller
 Virtual Function (rev 01)
82:10.6 Ethernet controller: Intel Corporation 82599 Ethernet Controller
 Virtual Function (rev 01)
82:10.7 Ethernet controller: Intel Corporation 82599 Ethernet Controller
 Virtual Function (rev 01)
82:11.0 Ethernet controller: Intel Corporation 82599 Ethernet Controller
 Virtual Function (rev 01)
82:11.1 Ethernet controller: Intel Corporation 82599 Ethernet Controller
 Virtual Function (rev 01)
82:11.2 Ethernet controller: Intel Corporation 82599 Ethernet Controller
 Virtual Function (rev 01)
82:11.3 Ethernet controller: Intel Corporation 82599 Ethernet Controller
 Virtual Function (rev 01)
82:11.4 Ethernet controller: Intel Corporation 82599 Ethernet Controller
 Virtual Function (rev 01)
82:11.5 Ethernet controller: Intel Corporation 82599 Ethernet Controller
 Virtual Function (rev 01)
```

6. Locate the serial numbers for the PFs and VRFs.

Locate the serial numbers for the PFs and VFs. The Linux **`virsh nodedev-list | grep 82`** command below displays the serial number for identifiers ***82:00.0*** and ***82:00.1***. The first two numbers are the serial numbers for the PFs and the remaining are the serial numbers for the VFs.

```
# virsh nodedev-list | grep 82
pci_0000_82_00_0
pci_0000_82_00_1
pci_0000_82_10_0
pci_0000_82_10_1
pci_0000_82_10_2
pci_0000_82_10_3
pci_0000_82_10_4
```

```
pci_0000_82_10_5
pci_0000_82_10_6
pci_0000_82_10_7
pci_0000_82_11_0
pci_0000_82_11_1
pci_0000_82_11_2
pci_0000_82_11_3
pci_0000_82_11_4
pci_0000_82_11_5
```

7. Select the serial number of the VF.

Select the serial number of the VF that will attach to the VM (vEOS). Using the Linux **`virsh nodedev-dumpxml <serial number>`** command, locate the bus, slot, and function parameters. For example, serial number: ***pci_0000_82_11_1*** displays the following details.

```
# virsh nodedev-dumpxml  pci_0000_82_11_1
<device>
  <name>pci_0000_82_11_1</name>
  <path>/sys/devices/pci0000:80/0000:80:02.0/0000:82:11.1</path>
  <parent>computer</parent>
  <driver>
    <name>ixgbevf</name>
  </driver>
  <capability type='pci'>
    <domain>0</domain>
    <bus>130</bus>
    <slot>17</slot>
    <function>1</function>
    <product id='0x10ed'>82599 Ethernet Controller Virtual Function</pro
duct>
    <vendor id='0x8086'>Intel Corporation</vendor>
    <capability type='phys_function'>
      <address domain='0x0000' bus='0x82' slot='0x00' function='0x1'/>
    </capability>
    <iommuGroup number='71'>
      <address domain='0x0000' bus='0x82' slot='0x11' function='0x1'/>
    </iommuGroup>
    <numa node='1'/>
    <pci-express>
      <link validity='cap' port='0' width='0'/>
      <link validity='sta' width='0'/>
    </pci-express>
  </capability>
</device>
```

8. Create a new Interface.

Shutdown the vEOS VM if it is already running. Open the XML file for the specific vEOS VM for editing using the Linux command **`virsh edit <vm-name>`**. In the interface section, create a new interface by adding the details as shown below. The bus, slot, and function values are in the hexadecimal format of the decimal values found in step 7.

```
<interface type='hostdev' managed='yes'>
      <source>
        <address type='pci' domain='0x0000' bus='0x82' slot='0x11'
 function='0x1'/>
      </source>
</interface>
```

9. Start the vEOS VM. Verify there is an added interface on the VM. Using the command **`ethtool -i et9`** to verify that the driver for the added interface is ***ixgbevf*** .

```
switch(config)#show interface status
Port   Name    Status        Vlan      Duplex Speed  Type      Flags
       Et9     notconnect    routed    unconf        unconf    10/100/1000
       Ma1     connected     routed    a-full        a-1G      10/100/1000

[admin@vEOS]$ ethtool -i et9
driver: ixgbevf
version: 2.12.1-k
firmware-version:
bus-info: 0000:00:0c.0
supports-statistics: yes
supports-test: yes
supports-eeprom-access: no
supports-register-dump: yes
supports-priv-flags: no
```

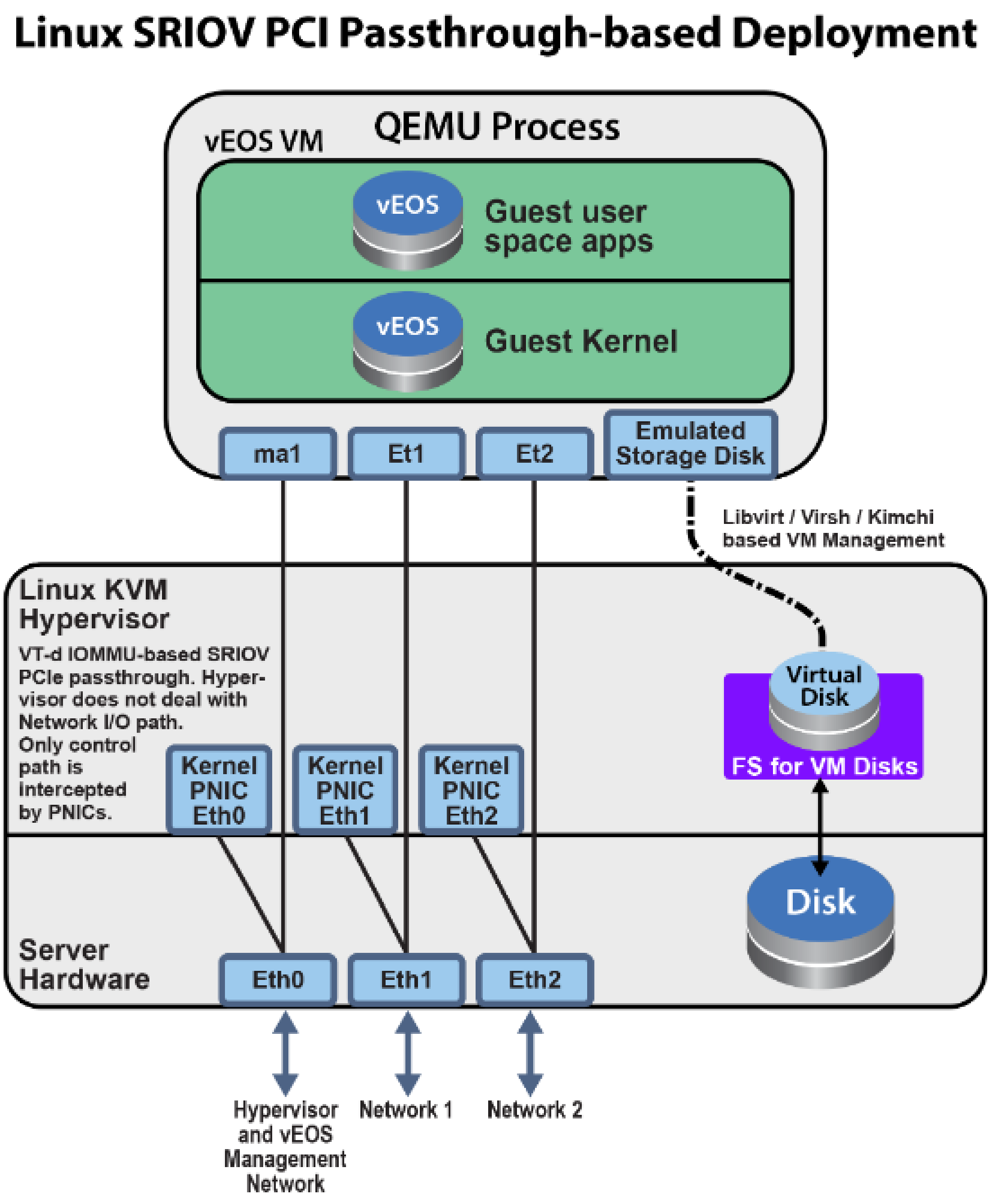
Launching SR-IOV

vEOS can also use PCIE SRI-OV I/O interfaces. Each SRI-OV NIC is passed-through to the VM such that network I/O does not hit the hypervisor. In this model, the hypervisor and multiple VMs can share the same NIC card.

SR-IOV has the following advantages over LinuxBridge:

- Higher Performance ~ 2x.
- Better latency and jitter characteristics.
- vEOS directly receives physical port state indications from the virtual device.
- Using SR-IOV virtualize the NIC.
- The NICs have a built-in bridge to do basic bridging.
- Avoids software handling of the packets in the kernel.

Figure 15: Linux SRIOV PCI Passthough-based Deployment


Linux SRIOV PCI Passthrough-based Deployment
vEOS VM
QEMU Process
vEOS
Guest user space apps
vEOS
Guest Kernel
ma1
Et1
Et2
Emulated Storage Disk
Libvirt / Virsh / Kimchi based VM Management
Linux KVM Hypervisor
VT-d IOMMU-based SRIOV PCIe passthrough. Hypervisor does not deal with Network I/O path.
Only control path is intercepted by PNICs.
Kernel PNIC Eth0
Kernel PNIC Eth1
Kernel PNIC Eth2
Virtual Disk
FS for VM Disks
Disk
Server Hardware
Eth0
Eth1
Eth2
Hypervisor and vEOS Management Network
Network 1
Network 2

6.2.5 Setting Up the Host and Launching PCI Pass-through

Set up a networking device to use PCI pass-through.

When sharing resources are not efficient, or packets are consumed by a virtualized switch before reaching the VM (vEOS), implementing PCI Pass-through for NIC provides dedicated and non-filtered network resources to the VM.

1. Identify Available Physical Functions.

Similar to the SR-IOV, identify an available physical function (a NIC in this scenario) and its identifier. Use the **`lspci | grep Ethernet`** Linux command to display the available physical functions.

In this example, 82:00.0 Ethernet controller: Intel Corporation 82599ES 10-Gigabit SFI/SFP+ Network Connection is the physical function and 82:00.0 is the device identification code.

```
# lspci | grep Ethernet
01:00.0 Ethernet controller: Intel Corporation I350 Gigabit Network
 Connection (rev 01)
01:00.1 Ethernet controller: Intel Corporation I350 Gigabit Network
 Connection (rev 01)
01:00.2 Ethernet controller: Intel Corporation I350 Gigabit Network
 Connection (rev 01)
01:00.3 Ethernet controller: Intel Corporation I350 Gigabit Network
 Connection (rev 01)
81:00.0 Ethernet controller: Intel Corporation I350 Gigabit Network
 Connection (rev 01)
81:00.1 Ethernet controller: Intel Corporation I350 Gigabit Network
 Connection (rev 01)
81:00.2 Ethernet controller: Intel Corporation I350 Gigabit Network
 Connection (rev 01)
81:00.3 Ethernet controller: Intel Corporation I350 Gigabit Network
 Connection (rev 01)
82:00.0 Ethernet controller: Intel Corporation 82599ES 10-Gigabit SFI/SFP
+ Network Connection (rev 01)
82:00.1 Ethernet controller: Intel Corporation 82599ES 10-Gigabit SFI/SFP
+ Network Connection (rev 01)
83:00.0 Ethernet controller: Intel Corporation 82599ES 10-Gigabit SFI/SFP
+ Network Connection (rev 01)
83:00.1 Ethernet controller: Intel Corporation 82599ES 10-Gigabit SFI/SFP
+ Network Connection (rev 01)
```

2. Verify Available Physical Functions.

Verify the available physical functions by using the **`virsh`** Linux commands.

```
[arista@solution]$ virsh nodedev-list | grep 82_00_0
pci_0000_82_00_0
[arista@solution]$ virsh nodedev-dumpxml pci_0000_82_00_0
<device>
  <name>pci_0000_82_00_0</name>
  <path>/sys/devices/pci0000:80/0000:80:02.0/0000:82:00.0</path>
  <parent>pci_0000_80_02_0</parent>
  <driver>
    <name>vfio-pci</name>
  </driver>
  <capability type='pci'>
    <domain>0</domain>
    <bus>130</bus>
    <slot>0</slot>
    <function>0</function>
    <product id='0x10fb'>82599ES 10-Gigabit SFI/SFP+ Network Connection</
product>
    <vendor id='0x8086'>Intel Corporation</vendor>
    <capability type='virt_functions' maxCount='64'/>
```

In this example, the domain is 0 (Hex **domain=0x0**), the bus is 130 (Hex **bus=0x82**), the slot is 0 (Hex **slot=0x0**), and function is 0 (Hex **function=0x0**).

With the domain, bus, slot, and function information, construct the device entry and add it into the VMs XML configuration.

```
<devices>
  ...
    <hostdev mode='subsystem' type='pci' managed='yes'>
```

```
      <source>
        <address domain='0x0000' bus='0x82' slot='0x00' function='0x0'/>
      </source>
    </hostdev>
```

3. Verify the NIC was detected by the VM.

When starting the VM (vEOS in this case), the VM should detect NIC.

```
switch#bash

Arista Networks EOS shell

[admin@veos1 ~]$ lspci | grep Ethernet
00:03.0 Ethernet controller: Intel Corporation 82599EB 10-Gigabit SFI/SFP
+ Network Connection (rev 01)
00:05.0 Ethernet controller: Red Hat, Inc Virtio network device
[admin@veos ~]$
```

4. Verify Driver Requirements.

If the NIC is supported by the vEOS and any other driver requirements are met, the corresponding ethernet interfaces are available to use on the vEOS. Use the **`show interface`** command to display the available vEOS Ethernet interfaces.

```
switch#show interface  status
Port   Name   Status       Vlan     Duplex Speed  Type   Flags
       Et1    connected    routed   full          10G    10/100/1000

       Ma1    connected    routed   a-full        a-1G   10/100/1000

switch#bash
bash-4.3# ethtool -i et1
driver: ixgbe
version: 4.2.1-k
firmware-version: 0x18b30001
bus-info: 0000:00:03.0
supports-statistics: yes
supports-test: yes
supports-eeprom-access: yes
supports-register-dump: yes
supports-priv-flags: no
```

Example Deployment

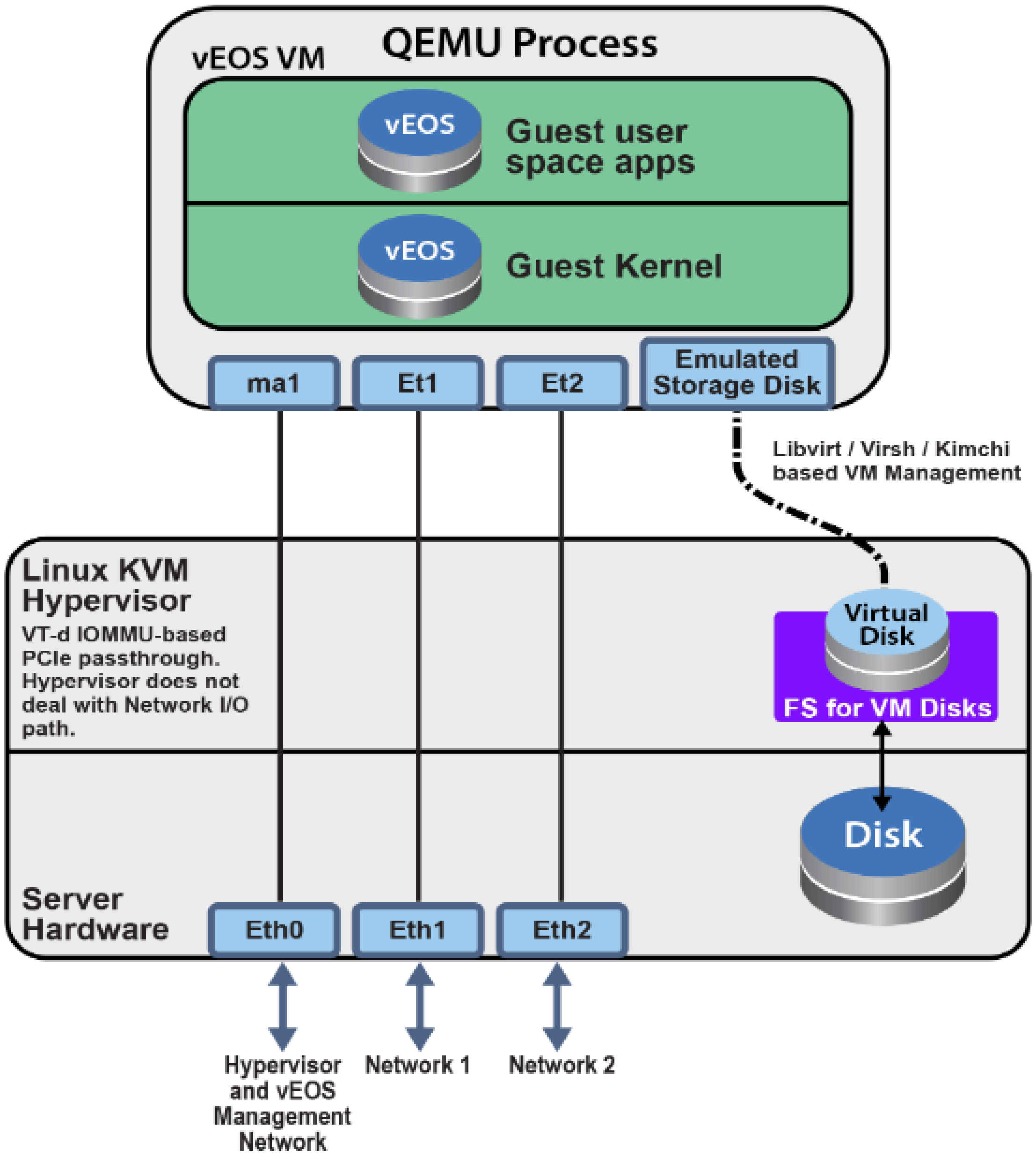
vEOS can use passthrough I/O interfaces where the network I/O does not hit the hypervisor. In this model, the VM owns the entire network card, thus fully bypassing the hypervisor.

Setting up SR-IOV is initially more involved. Arista recommends starting out with LinuxBridge.

- SR-IOV has the following advantages over LinuxBridge Higher Performance ~ 2x
- Better latency and jitter characteristics
- vEOS directly receives physical port state indications from the virtual device.

Figure 16: Linux PCI Passthrough-based Deployment

Linux PCI Passthrough-based Deployment



Chapter 7

Using the CloudEOS and vEOS Router on Arista Appliance (DCA-200-vEOS)

This sections describes the CloudEOS and vEOS router setup and configuration of the CloudEOS and vEOS Router Appliance.

- Overview
- Appliance Setup
- CloudEOS and vEOS Router Installation
- Accessing CloudEOS and vEOS
- Troubleshooting
- Supported Transceivers
- Limitations

7.1 Overview

The Appliance is used to host CloudEOS and vEOS Router virtual machines. The appliance uses KVM Hypervisor and has two 2x10G SFP+ NIC cards for data traffic, and two 1G ports for management traffic. The CloudEOS and vEOS VM instance is specified with resources such as number of CPU cores, memory and interfaces depending upon the customer network deployment model as well as desired performance. The CloudEOS and vEOS launcher script **dca-200- vEOS-setup-vm.py** is used to quickly launch new CloudEOS and vEOS instances with the right resources and setup the network interfaces.

- Hardware
- Interfaces

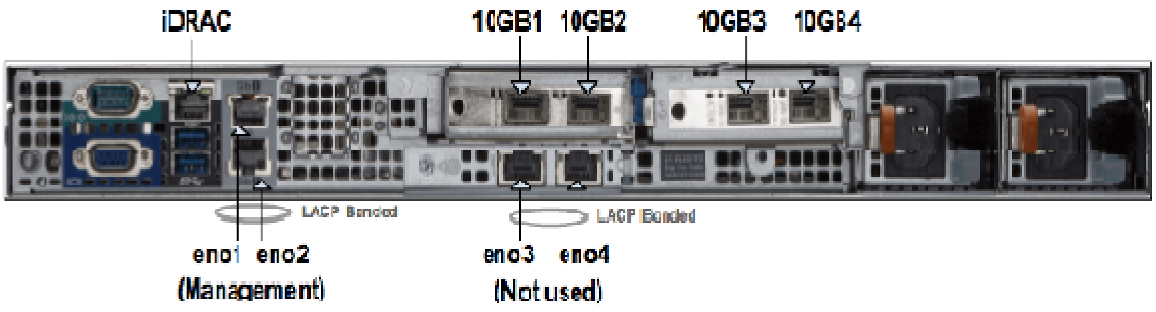
7.1.1 Hardware

The CloudEOS and vEOS Router appliance has the following hardware configuration:

- 2 sockets with 10 CPU cores each. The server is configured for optimal CloudEOS and vEOS Router performance. For example Hyperthreading has been turned off.
- 64 GB memory.
- Two 2 x 10G SFP+ NIC cards for a total of 4 10G NIC ports for data traffic.
- 2 x 1G ports (eno1 and eno2) for management.
- 2 x 1G ports (eno3 and eno4) NOT used by CloudEOS and vEOS launcher scripts.
- iDRAC

7.1.2 Interfaces

The below figure shows all the interfaces on the appliance.



7.1.2.1 Management Interfaces

As shown in the above figure, the appliance has 4 physical 1G ports --- eno1/2/3/4. eno1 and eno2 are aggregated to a bonded interface device0 in 802.3ad mode. So they need to be connected to one or more network devices supporting Link Aggregation Control Protocol (LACP). Bonded interface device0 is connected to a Linux bridge named devicebr internally. CloudEOS and vEOS launcher script will setup CloudEOS and vEOS Router with connecting their management interfaces to devicebr. eno3 and eno4 are aggregated to bonded interface cluster0 and cluster0 is connected to Linux bridge clusterbr in the same way. However, they are not used for CloudEOS and vEOS Router setup.

7.1.2.2 Data Traffic Interfaces

As shown in the above figure, the appliance has 4 physical 10G ports --- 10GB1/2/3/4 those are configured in SR-IOV mode. Each port is partitioned into 32 SR-IOV Virtual Functions to provide a total of 128 virtual interfaces for CloudEOS and vEOS instances on the appliance. You may optionally configure a VLAN to be used for each virtual interface. The VLAN configuration allows separation of broadcast domain for traffic in and out of each physical port. The VLAN tag handling is done by SRIOV NIC and it is transparent to the CloudEOS and vEOS Router. Please note that for performance reasons, the CloudEOS and vEOS launcher script creates CloudEOS and vEOS Router with all of its CPU cores and memory from the same NUMA node. Therefore, all required CPU resources for a CloudEOS and vEOS Router need to be available on one socket. If required, resources for launching a new CloudEOS and vEOS Router are split across two sockets, CloudEOS and vEOS launcher will not be able to launch the CloudEOS and vEOS Router. In such scenario, user may need to reconfigure the existing VMs and/or reduce resource requirements of the new VM to fit within a NUMA node.

7.2 Appliance Setup

1. Setup Management Connections
   - Connect iDRAC (IPMI) to a network device.
   - Connect eno1 and eno2 to a network device supporting LACP.
   - Make sure the DHCP server is setup properly.
   - After appliance boot up, iDRAC and devicebr (the management bridge interface) will get it's DHCP assigned IP addresses.
   - For DHCP based IP address setup, refer to ***section 2.5.1 - DHCP Based IP Address Setup*** in Arista CloudVision Appliance Quick Start Guide.
   - For Manual based IP address, refer to ***section 2.5.2 - Manual IP Address Setup*** in Arista CloudVision Appliance Quick Start Guide. After configuring IP addresses manually for management interfaces, reboot the appliance instead of just restarting the network-service, because the appliance setup scripts for **DCA-200-vEOS** takes effect only during the reboot.
2. There are multiple ways to connect to the appliance:
   - SSH to devicebr DHCP address.

- Web access to iDRAC *https://<hostname or IP of iDRAC* using Google Chrome or any other web browser.
- Use the terminals connected to VGA and other peripherals if DHCP addresses of management interfaces are not known.

3. Login to the appliance using username:root (password:arista). Change appliance username/ password appropriately as needed, by referring to ***Chapter 3 - Accessing CloudVision Appliance*** in Arista CloudVision Appliance Quick Start Guide.

7.3 CloudEOS and vEOS Router Installation

CloudEOS and vEOS launcher script and CloudEOS and vEOS image are on the appliance under **/ data/tools/**. Before launching CloudEOS and vEOS Routers, you can decide on how many CloudEOS and vEOS instances are needed and how should these be configured.

- Supported vEOS Router Configurations
- Launching/Removing/Query CloudEOS and vEOS Router

7.3.1 Supported vEOS Router Configurations

Resource	**Supported configurations**	**Notes**
CPUs	2, 4 and 8 cores	All cores of a vEOS Router need to be on the same socket. vEOS Launcher checks if there's enough available cores on the same socket.
Memory	4, 8, 16 GB	Memory is dependant on features and scale. Typically 4 core, 8 core instances are deployed with 8GB and 2 core with 4GB.
Interfaces	<=16	Each interface is a virtual ethernet interface connected to one of the four 10G ports

7.3.2 Launching/Removing/Query CloudEOS and vEOS Router

In the above figure in "Interfaces" section shows the rear view of the appliance and ethernet ports (10GB1/2/3/4) the CloudEOS and vEOS launcher script references. The ethernet ports in the CloudEOS and vEOS Router are virtual ethernet ports connected to one of the 10GB1/2/3/4 ports. VLANs are configured on each interface when installing CloudEOS and vEOS. The VLAN tagging is done by the SRIOV NICs. Note, that the connected networking devices need to have the same VLANs configured on the trunk port.

The appliance are shipped with a version of CloudEOS and vEOS Router image which is found in **/data/tools**. If you want to install the latest CloudEOS and vEOS Router image download desired **CloudEOS.qcow2** version from Arista.com to the appliance to another directory.

The CloudEOS and vEOS launcher is a python script named **dca-200-veos-setup-vm.py** which is found in **/data/tools** directory as shown.

```
Router# ./dca-200-veos-setup-vm.py --help
usage: dca-200-veos-setup-vm.py [-h] [-n VMNAME] [-m IMAGE] [-d]
                                [-i [INTERFACE [INTERFACE ...]]] [-s
 MEMORY]
```

```
                                [-c CORES] [-r [REMOVE [REMOVE ...]]] [-
q]

Create/Remove VEOS instances

optional arguments:
  -h, --help            show this help message and exit
  -n VMNAME, --name VMNAME
                        Name of the VEOS VM
  -m IMAGE, --image IMAGE
                        Qcow2 image name to use for launching the VM
  -d, --debug           Print detailed debug info
  -i [INTERFACE [INTERFACE ...]], --interface [INTERFACE [INTERFACE ...]]
                        Interfaces and optional vlans/mac. The interfaces
 must
                        be listed in guest interfaces order. The
 interfase can
                        be specified either in PCI address format (using
 lspci
                        command) Or 10GB1/2/3/4. For example: '-i
                        10GB1,vlan=10 10GB2 10GB3,vlan=40' or '-i
                        3b:10.2,vlan=50 3b:10.3,vlan=10 af:10.2 af:10.3'
  -s MEMORY, --memory MEMORY
                        Memory in Gbytes to assign to VM. Default is 4 Gb
  -c CORES, --cores CORES
                        Number of Cores to assign to VM. Default is 4
 cores
  -r [REMOVE [REMOVE ...]], --remove [REMOVE [REMOVE ...]]
                        Remove VMs
  -q, --query           Query info about configured VMs
```

Example

Below is an example of commands used to launch a vEOS VMs with core count of 4 (default), 4GB memory (default), and with 4 ethernet interfaces.

```
Router# ./dca-200-veos-setup-vm.py -n veos-router1 -m /tmp/CloudEOS.qcow2
 -i 10GB2,vlan=50 10GB1,vlan=10 10GB3,vlan=100 af:10.0,vlan=200
Extracting info for existing VMs: ['']
Total count is: 20, reserved for hypervisor: 2, Total Available: 18
Used CPU count is 0, Free cores 18
intfList is: ['10GB2,vlan=50', '10GB1,vlan=10', '10GB3,vlan=100',
 'af:10.0,vlan=200']
Used CPU count is 0, Free cores 18
Free core set on Node 0 : [2, 4, 6, 8, 10, 12, 14, 16, 18]
CPU core used are:  [2, 4, 6, 8]
Using PCI interfaces for new VM veos-router1:
    ('veos-router1', 'et1') --> 10GB2 PCI address: 3b:10.0 vlan 50 mac
 None
    ('veos-router1', 'et2') --> 10GB1 PCI address: 3b:10.1 vlan 10 mac
 None
    ('veos-router1', 'et3') --> 10GB3 PCI address: af:10.1 vlan 100 mac
 None
    ('veos-router1', 'et4') --> 10GB4 PCI address: af:10.0 vlan 200 mac
 None
```

The following observations are from the above example:

- Without specifying number of cores, VM will be created with 4 cores by default. vEOS launcher picks core 2,4,6,8 on NUMA node0 for veos-router1.
- A different image “/tmp/CloudEOS.qcow2” than the default vEOS Router image (/data/tools/CloudEOS.qcow2) is specified

- Interfaces MUST be specified in VM interface order (the physical 10GB port eth1, eth2.. in VM) in either 10GBx or PCI address format. In the above example we used both 10GBx format as well as PCI address format to specify 4 interfaces. The interfaces are configured on different VLANs
- Launcher script will print out the guest interface mapping to host 10GB interfaces.

If an error occurs while creating a new VM using vEOS launcher, then refer to the Troubleshooting section of the chapter for more information.

The **dca-200-veos-setup-vm.py** script is used to remove the running VMs. The example below shows how to remove two existing VMs.

```
Router# ./dca-200-veos-setup-vm.py -r veos-router1 veos-router2
Cleaning up VM:  veos-router1
Cleaning up VM:  veos-router2
```

Besides launching/removing functionality, **dca-200-veos-setup-vm.py** script also provides a query command to print the current status of the running VMs. The output includes

- List of running VMs
- Mapping of running VM interfaces to host interfaces
- Mapping of VM CPUs to host CPUs

Below is an example output:

```
Router# ./dca-200-veos-setup-vm.py -q
Extracting info for existing VMs: ['veos-router1', 'veos-router2']
Total count is: 20, reserved for hypervisor: 2, Total Available: 18
Used CPU count is 8, Free cores 10
VM veos-router1 :
  interfaces:
    et1 --> 10GB2 PCI address: 3b:10.0 vlan 50 mac 52:54:00:d4:f4:46
    et2 --> 10GB1 PCI address: 3b:10.1 vlan 10 mac 52:54:00:d8:a9:50
    et3 --> 10GB3 PCI address: af:10.1 vlan 100 mac 52:54:00:0c:0a:15
    et4 --> 10GB4 PCI address: af:10.0 vlan 200 mac 52:54:00:20:4a:67
  CPU Core Mapping:
    0 --> 2
    1 --> 4
    2 --> 6
    3 --> 8
VM veos-router2 :
  interfaces:
    et1 --> 10GB4 PCI address: af:10.2 vlan 50 mac 52:54:00:bb:ab:f1
    et2 --> 10GB3 PCI address: af:10.3 vlan 10 mac 52:54:00:58:f7:2b
  CPU Core Mapping:
    0 --> 10
    1 --> 12
    2 --> 14
    3 --> 16
Available free cores:  3 5 7 9 11 13 15 17 18 19
```

7.4 Accessing CloudEOS and vEOS

For virtual console access there are two options:

1. On the appliance **virsh console <vm_name>**
2. Launch browser based VM management tool **kimchi** to edit/view/console-access of the VMs at **https://<management_ip>:8001**.

7.5 Troubleshooting

- PCI Addresses for Virtual Functions
- CloudEOS and vEOS Launcher Debugging Functionalities
- Appliance Setup Debugging

7.5.1 PCI Addresses for Virtual Functions

The following commands are used to find the bus number for the physical port:

```
[root@cv ~]# ethtool -i 10GB1 | grep bus
bus-info: 0000:3b:00.1
[root@cv ~]# ethtool -i 10GB2 | grep bus
bus-info: 0000:3b:00.0
[root@cv ~]# ethtool -i 10GB3 | grep bus
bus-info: 0000:af:00.1
[root@cv ~]# ethtool -i 10GB4 | grep bus
bus-info: 0000:af:00.0
```

Command to get PCI bus information for all ethernet physical and virtual functions:

```
[root@cv ~]# lspci | grep Ethernet
04:00.0 Ethernet controller: Broadcom Inc. and subsidiaries NetXtreme
 BCM5720 Gigabit Ethernet PCIe
04:00.1 Ethernet controller: Broadcom Inc. and subsidiaries NetXtreme
 BCM5720 Gigabit Ethernet PCIe
3b:00.0 Ethernet controller: Intel Corporation Ethernet 10G 2P X520
 Adapter (rev 01)⇐ 10GB2 PF
3b:00.1 Ethernet controller: Intel Corporation Ethernet 10G 2P X520
 Adapter (rev 01)⇐ 10GB1 PF
3b:10.0 Ethernet controller: Intel Corporation 82599 Ethernet Controller
 Virtual Function (rev 01)⇐ 10GB2 VF
3b:10.1 Ethernet controller: Intel Corporation 82599 Ethernet Controller
 Virtual Function (rev 01)⇐ 10GB1 VF
3b:10.2 Ethernet controller: Intel Corporation 82599 Ethernet Controller
 Virtual Function (rev 01)
3b:10.3 Ethernet controller: Intel Corporation 82599 Ethernet Controller
 Virtual Function (rev 01)
...
3b:17.6 Ethernet controller: Intel Corporation 82599 Ethernet Controller
 Virtual Function (rev 01)
3b:17.7 Ethernet controller: Intel Corporation 82599 Ethernet Controller
 Virtual Function (rev 01)
5e:00.0 Ethernet controller: Broadcom Inc. and subsidiaries NetXtreme
 BCM5720 Gigabit Ethernet PCIe
5e:00.1 Ethernet controller: Broadcom Inc. and subsidiaries NetXtreme
 BCM5720 Gigabit Ethernet PCIe
af:00.0 Ethernet controller: Intel Corporation Ethernet 10G 2P X520
 Adapter (rev 01)⇐ 10GB4 PF
af:00.1 Ethernet controller: Intel Corporation Ethernet 10G 2P X520
 Adapter (rev 01)⇐ 10GB3 PF
af:10.0 Ethernet controller: Intel Corporation 82599 Ethernet Controller
 Virtual Function (rev 01)⇐ 10GB4 VF
af:10.1 Ethernet controller: Intel Corporation 82599 Ethernet Controller
 Virtual Function (rev 01)⇐ 10GB3 VF
af:10.2 Ethernet controller: Intel Corporation 82599 Ethernet Controller
 Virtual Function (rev 01)
af:10.3 Ethernet controller: Intel Corporation 82599 Ethernet Controller
 Virtual Function (rev 01)
...
```

```
af:17.6 Ethernet controller: Intel Corporation 82599 Ethernet Controller
 Virtual Function (rev 01)
af:17.7 Ethernet controller: Intel Corporation 82599 Ethernet Controller
 Virtual Function (rev 01)
```

Command for VFs and Parent interface mapping (used for output above):

```
[root@cv ~]# virsh nodedev-list | grep 3b_00_0 ⇐ PF PCI
pci_0000_3b_00_0
[root@cv ~]# virsh nodedev-dumpxml pci_0000_3b_00_0
<device>
  <name>pci_0000_3b_00_0</name>
  <path>/sys/devices/pci0000:3a/0000:3a:00.0/0000:3b:00.0</path>
  <parent>pci_0000_3a_00_0</parent>
  <driver>
    <name>ixgbe</name>
  </driver>
  <capability type='pci'>
    <domain>0</domain>
    <bus>59</bus>
    <slot>0</slot>
    <function>0</function>
    <product id='0x154d'>Ethernet 10G 2P X520 Adapter</product>
    <vendor id='0x8086'>Intel Corporation</vendor>
    <capability type='virt_functions' maxCount='63'>
      <address domain='0x0000' bus='0x3b' slot='0x10' function='0x0'/> ⇐
 VF PCI
      <address domain='0x0000' bus='0x3b' slot='0x10' function='0x2'/>
      <address domain='0x0000' bus='0x3b' slot='0x10' function='0x4'/>
      <address domain='0x0000' bus='0x3b' slot='0x10' function='0x6'/>
      <address domain='0x0000' bus='0x3b' slot='0x11' function='0x0'/>
      <address domain='0x0000' bus='0x3b' slot='0x11' function='0x2'/>
      <address domain='0x0000' bus='0x3b' slot='0x11' function='0x4'/>
      <address domain='0x0000' bus='0x3b' slot='0x11' function='0x6'/>
      <address domain='0x0000' bus='0x3b' slot='0x12' function='0x0'/>
      <address domain='0x0000' bus='0x3b' slot='0x12' function='0x2'/>
      <address domain='0x0000' bus='0x3b' slot='0x12' function='0x4'/>
      <address domain='0x0000' bus='0x3b' slot='0x12' function='0x6'/>
      <address domain='0x0000' bus='0x3b' slot='0x13' function='0x0'/>
      <address domain='0x0000' bus='0x3b' slot='0x13' function='0x2'/>
      <address domain='0x0000' bus='0x3b' slot='0x13' function='0x4'/>
      <address domain='0x0000' bus='0x3b' slot='0x13' function='0x6'/>
      <address domain='0x0000' bus='0x3b' slot='0x14' function='0x0'/>
      <address domain='0x0000' bus='0x3b' slot='0x14' function='0x2'/>
      <address domain='0x0000' bus='0x3b' slot='0x14' function='0x4'/>
      <address domain='0x0000' bus='0x3b' slot='0x14' function='0x6'/>
      <address domain='0x0000' bus='0x3b' slot='0x15' function='0x0'/>
      <address domain='0x0000' bus='0x3b' slot='0x15' function='0x2'/>
      <address domain='0x0000' bus='0x3b' slot='0x15' function='0x4'/>
      <address domain='0x0000' bus='0x3b' slot='0x15' function='0x6'/>
      <address domain='0x0000' bus='0x3b' slot='0x16' function='0x0'/>
      <address domain='0x0000' bus='0x3b' slot='0x16' function='0x2'/>
      <address domain='0x0000' bus='0x3b' slot='0x16' function='0x4'/>
      <address domain='0x0000' bus='0x3b' slot='0x16' function='0x6'/>
      <address domain='0x0000' bus='0x3b' slot='0x17' function='0x0'/>
      <address domain='0x0000' bus='0x3b' slot='0x17' function='0x2'/>
      <address domain='0x0000' bus='0x3b' slot='0x17' function='0x4'/>
      <address domain='0x0000' bus='0x3b' slot='0x17' function='0x6'/>
    </capability>
    <iommuGroup number='30'>
      <address domain='0x0000' bus='0x3b' slot='0x00' function='0x0'/>
    </iommuGroup>
    <numa node='0'/>
    <pci-express>
      <link validity='cap' port='0' speed='5' width='8'/>
      <link validity='sta' speed='5' width='8'/>
    </pci-express>
  </capability>
</device>
```

7.5.2 CloudEOS and vEOS Launcher Debugging Functionalities

CloudEOS and vEOS launcher script provides the below mechanisms helping with debugging:

- CloudEOS and vEOS launcher error messages
- CloudEOS and vEOS launcher query command

The below two examples are of error messages that a customer may see during creating vEOS instances:

```
[root@cv /data/tools]# ./dca-200-veos-setup-vm.py -n veos-router2 -m ./
CloudEOS.qcow2 -i af:10.0,vlan=50 10GB3,vlan=10
Extracting info for existing VMs: ['veos-router1']
Total count is: 20, reserved for hypervisor: 2, Total Available: 18
Used CPU count is 4, Free cores 14
intfList is: ['af:10.0,vlan=50', '10GB3,vlan=10']
Error: Interface af:10.0 is already assigned to VM veos-router1
```

Above error message points out the specified interface is already used by other VM. User can use a query command **dca-200-veos-setup-vm.py -q** to list all the interfaces used by the existing VMs and start using available interfaces to create new VMs. Refer to **Launching/Removing/Query CloudEOS and vEOS Router** section of the chapter for more query command information.

```
[root@cv /data/tools]# ./dca-200-veos-setup-vm.py -n veos-router5 -m ./
CloudEOS.qcow2 -i 10GB4,vlan=20 10GB1,vlan=30
Extracting info for existing VMs: ['veos-router1', 'veos-router2', 'veos-
router3', 'veos-router4']
Total count is: 20, reserved for hypervisor: 2, Total Available: 18
Used CPU count is 16, Free cores 2
intfList is: ['10GB4,vlan=20', '10GB1,vlan=30']
Free core set on Node 0 : [18]
Free core set on Node 1 : [19]
Not enough CPU cores available on any NUMA node to allocate 4 cores.
```

The above example shows when user tries to create a VM with 4 cores (by default), an error message points out there's not enough CPU cores available on any NUMA. It also prints out current free cores on each NUMA node (core 18 on node0 and core 19 on node1). User may choose to reduce the number of cores for new instance or reprovision existing VMs to fit new VM in.

7.5.3 Appliance Setup Debugging

When the appliance is shipped to customer, it is in a setup ready stage. Ideally customers need not worry about the scripts mentioned in this section. However, if there was some issues setting up the appliance, user may choose to run below test scripts under **/data/imaging/ directory** to verify the settings:

- dca-200-veos-test.sh
- dca-200-veos-test-nics.py

/data/imaging/dca-200-veos-test.sh checks things like hyperthreading, interface MTU and etc.

Below output from the script means appliance is properly setup:

```
[root@cv /data/imaging]# ./dca-200-veos-test.sh
```

```
Device '10GB1' successfully disconnected.
Connection successfully activated (D-Bus active path: /org/freedesktop/
NetworkManager/ActiveConnection/19)
Device '10GB2' successfully disconnected.
Connection successfully activated (D-Bus active path: /org/freedesktop/
NetworkManager/ActiveConnection/20)
Device '10GB3' successfully disconnected.
Connection successfully activated (D-Bus active path: /org/freedesktop/
NetworkManager/ActiveConnection/21)
Device '10GB4' successfully disconnected.
Connection successfully activated (D-Bus active path: /org/freedesktop/
NetworkManager/ActiveConnection/22)
Appliance is correctly set for VEOS use
```

Note: /data/imaging/dca-200-veos-test.sh will bring host interfaces down and up again, which may impact running VMs traffic on the host.

/data/imaging/dca-200-veos-test-nics.py creates 4 VMs and sends traffic among them to test NICs are setup properly for creating new VMs and sending traffic.

Below output from the script means appliance NICs are in good stage:

```
[root@cv /data/imaging]# ./dca-200-veos-test-nics.py -a -i /data/tools/C
loudEOS.qcow2
Cleaning up VMs:  ['autoDut1']
Cleaning up VMs:  ['autoDut2']
Cleaning up VMs:  ['autoDut3']
Cleaning up VMs:  ['autoDut4']
Creating instance autoDut1
Creating instance autoDut2
Creating instance autoDut3
Creating instance autoDut4
Starting Traffic test on created VMs
Running Tests on  autoDut1
Running Tests on  autoDut2
Running Tests on  autoDut3
Running Tests on  autoDut4
All Tests finished Successfully
Cleaning up VMs:  ['autoDut1']
Cleaning up VMs:  ['autoDut2']
Cleaning up VMs:  ['autoDut3']
Cleaning up VMs:  ['autoDut4']
```

Note: To make **/data/imaging/dca-200-veos-test-nics.py** test properly, appliance interfaces have to be connected in a way that 10GB1 connects to 10GB3, 10GB2 connects to 10GB4 and there's no existing VMs on the appliance.

If there are error messages shown from the above two test scripts, user can run **/data/imaging/dca-200-veos-setup.sh** to re-setup the appliance. Note, The **/data/imaging/dca-200-veos-setup.sh** will reconfigure the interfaces and other parameters on the appliance and reboot the VMs, which may affect the running VMs to working properly and stop.

The appliance shipped should be in good condition, and quality checked stage, where setup and test scripts have already been run. So it is **NOT** recommended for customer to run **/data/imaging/dca-200-veos-setup.sh** without contacting Arista support.

7.6 Supported Transceivers

The 10G ethernet ports are tested using the following Arista transceivers:

- -10G-SR

- -10G-SRL
- -10G-LR
- -10G-AOC
- -10G-CR

7.7 Limitations

- There can be a maximum 32 virtual interfaces (virtual functions) on a physical interface.
- The optimization for NUMA may reduce the VMs that can be hosted on the appliance. You may need to reprovision existing VMs to leverage all the resources in case of resource fragmentation.
- DCA-200-vEOS starts supporting CVA upgrade from 2.1.2 to afterward releases.

Chapter 8

Upgrade/Downgrade

8.1 CloudEOS and vEOS Upgrade/Downgrade

The CloudEOS and vEOS images are upgraded/downgraded just like any other Arista switch by copying the desired **CloudEOS.swi** file to /mnt/flash. Configure the **boot system flash:CloudEOS.swi**, and then reload the VM from the Arista CLI. For more information, refer https://www.arista.com/en/um-eos/eos-upgrades-and-downgrades. Note, while upgrading your existing vEOS Router image to CloudEOS Router image, use **.swi** as the file extension.

Note: Starting from EOS 4.23, please do not upgrade CloudEOS / vEOS router using hardware EOS SWI or vice versa.

8.2 Appliance Upgrade

The Appliance itself can be upgraded apart from the CloudEOS and vEOS VMs. Refer to Appendix E - **Tools to Manage and Update Images** section in Arista DCA-200-CloudEOS and vEOS guide for steps to upgrade CVA. During CVA upgrade process, all DCA-200-vEOS scripts under /data/imaging/ and /data/tool/ directories are also upgraded. There will be a newly created directory named as current CVA version under both /data/imaging/ and /data/tool/ (For example, /data/imaging/2.1.2/ and /data/imaging/2.1.2/). Older version of the scripts are moved down to that version directory. Newer version of the scripts are copied to /data/imaging/ and /data/tool/ directly.

After a system reboot from the last step of upgrade, the CloudEOS and vEOS instances becomes up and runs automatically after appliance is up again. Allow 20 minutes for the application running in CloudEOS and vEOS instances to be accessible again.

Note: The DCA-200-vEOS supports CVA upgrade from 2.1.2 to afterward releases.

Chapter 9

Cloud High Availability

Amazon Web Services cloud and Microsoft Azure cloud resources are hosted in multiple locations worldwide. These locations are composed of Regions and Availability Zones. Each Region is a separate geographic area and each Region has multiple, isolated locations known as Availability Zones.

In the cloud, resources can be deployed across different regions or multiple locations within a region for fault tolerance reasons. AWS Availability Zones and Azure Availability Sets (or Fault Domains; Azure currently supports different resource groupings within a physical datacenter) are examples of cloud high availability offerings. When deploying CloudEOS and vEOS Routers to enhance your cloud's network capability, deploy the CloudEOS and vEOS Routers as a high availability pair using the CloudEOS and vEOS Cloud High Availability feature that fits your cloud's high availability design.

The ***Cloud High Availability (Cloud HA)*** feature adds support to make the CloudEOS and vEOS Router deployment more resilient to various failure scenarios in the cloud, such as:

- CloudEOS and vEOS Router instance goes down due to underlying cloud infrastructure issues.
- CloudEOS and vEOS Router instance is unable to forward traffic due to connectivity issues in the cloud infrastructure.
- CloudEOS and vEOS Router experiences an internal issue leading to unavailability.

CloudEOS and vEOS Router HA pair with Cloud HA is an active-active deployment model for different cloud high availability design in a region. Each CloudEOS and vEOS Router in an HA pair provides enhanced routing capabilities as the gateway (or next-hop router for certain destinations) for the subnets to which the CloudEOS and vEOS routers connect. The two CloudEOS and vEOS Router peers monitor the liveliness of each other by using Bidirectional Forwarding Detection (BFD) between the router interfaces. In case of the cloud infrastructure issues or CloudEOS and vEOS router failure, the active CloudEOS and vEOS router takes over as the gateway or next-hop for the subnets that were connected to the peer router through cloud-specific API calls that modify the corresponding cloud route table(s) according to pre-configured information.

This chapter includes the following sections:

9.1 Cloud HA Topology

This diagram shows an example of a vEOS Router Cloud HA implementation.

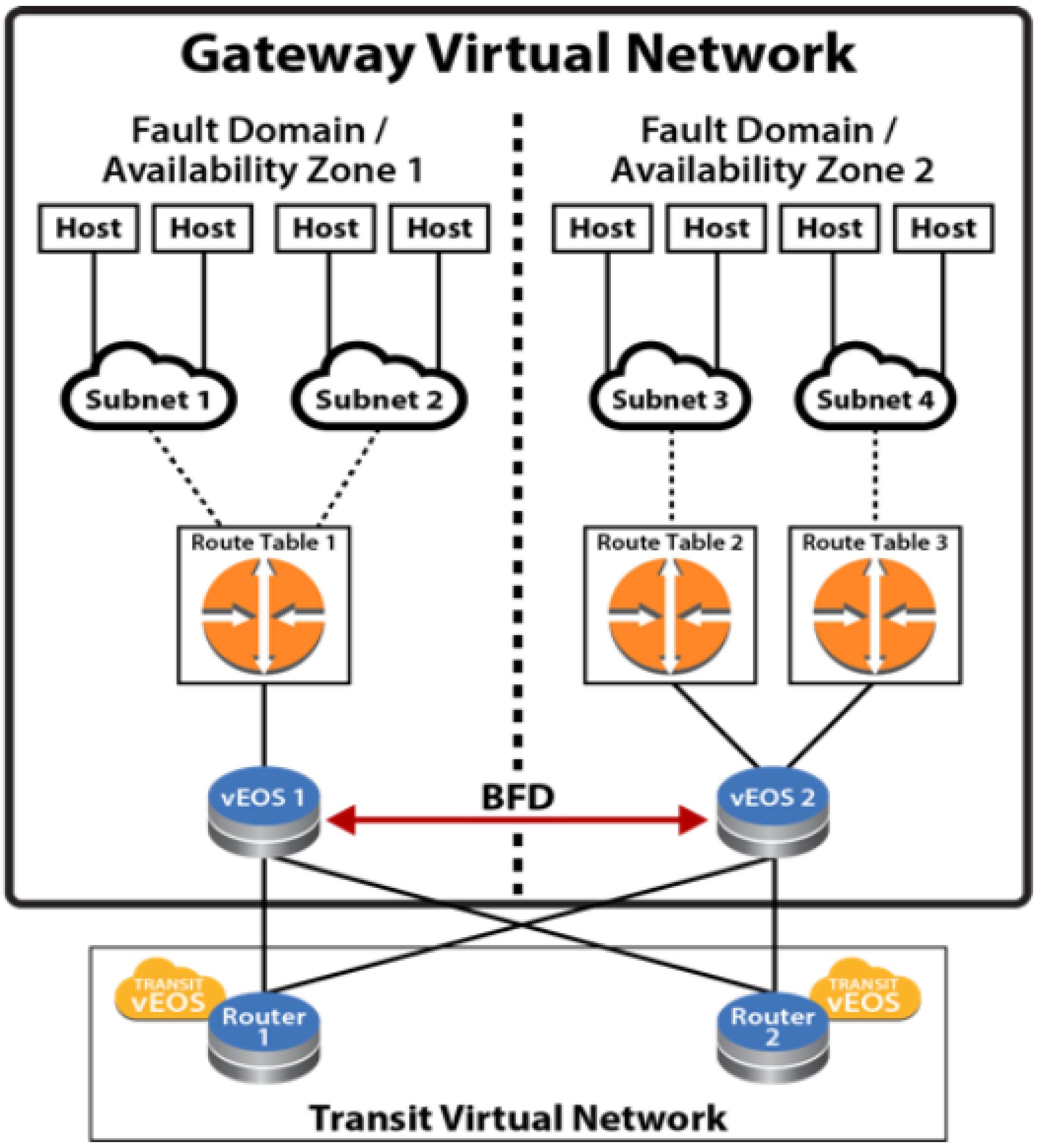


Figure 17: Cloud high availability network topology with vEOS router instances

In the diagram above, a virtual network is a collection of resources that are in the same cloud region. Within this virtual network, the resources, including vEOS routers, deploy into two cloud high availability zones (Availability Zones for AWS and Fault Domain for Azure) for fault tolerance reasons.

Note: For ease of discussion, we will use availability zone 1 and 2 to reference the high availability design in different clouds going forward.

Within each availability zone, the hosts/VMs and vEOS interfaces are connected to their corresponding subnets when the network is operating normally. Each subnet associates to a route table within the cloud infrastructure. Static routes are configured in the cloud route tables so the traffic from the hosts/VMs are routed to vEOS Routers in the corresponding availability zone as gateway or next-hop to reach certain destinations. For example, configure a default route (0.0.0.0/0) in the cloud route table with the next-hop as vEOS Router's cloud interface ID or IP (varies depending on the cloud). The routing policy or protocol, such as BGP, on the vEOS Routers, are user configurable based on user's network design.

The two vEOS Routers in the diagram above are configured with the Cloud HA feature as HA peers. The Cloud HA on the vEOS routers would establish a BFD peering session between the two devices through ethernet or tunnel interfaces.

When BFD connectivity loss is detected by the active vEOS router, the existing routes in the backup route table in the cloud would be updated through cloud-specific API to use the active vEOS router as the next-hop. For example, if ***vEOS 2*** detected BFD connectivity loss with its peer, ***vEOS 2*** would update the routes in ***Route Table 1*** so traffic from hosts in ***Subnet 1*** and ***Subnet 2*** for ***vEOS 1*** would be forwarded to next-hop ID or IP owned by ***vEOS 2***. Traffic from the hosts in availability zone 1 would first be forwarded to the corresponding subnet gateways in the cloud. After that, the subnet gateways in the cloud would forward the traffic toward the new next-hop interface ID or IP that exist on ***vEOS 2***. When ***vEOS 2*** received the traffic, it would forward the traffic on according to its routing table.

What about traffic going toward the hosts in availability zone 1 while connectivity to ***vEOS 1*** is down? When connectivity to ***vEOS 1*** is down, hosts behind ***Subnet 1*** and ***Subnet 2*** become unreachable to the other part of the network (routes being withdrawn by routing protocols like BGP). Since ***Subnet 1*** and ***Subnet 2*** are not directly connected to ***vEOS 2***, a routing strategy for the two subnets as "backup" on ***vEOS 2*** is to be considered as part of your network design. A typical design would be to use static routes for the subnets connected to the peer vEOS router and point them toward the cloud subnet gateways of the active vEOS router (for example, static route for peer subnet ***10.1.1.0/24*** would be configured on the active vEOS router as ***ip route10.1.1.0/24 10.2.1.1 255*** where ***10.2.1.1*** is the gateway/next-hop for one of the ethernet interfaces) with a high administrative distance value (least preferred). The static routes would be redistributed or advertised when the original routes with better administrative distance are withdrawn or removed by dynamic routing protocol (such as BGP).

When BFD peering session is restored to UP state upon recovery, each active vEOS router would restore its locally controlled route table entries (per user configuration) to point to itself as primary gateway again.

9.2 Configuring the Cloud Proxy

Optional proxies can be configured if used in a deployment. The configuration is applicable for any cloud type. All web traffic for the underlying restful APIs for the Cloud provider SDK will use the configured proxies. Multiple proxies can be configured but only one can be used at any given time from the Cloud High-Availability configuration.

```
switch(config)#
switch(config)#cloud proxy test
switch(config-cloud-proxy-test)#
```

The following example con#gures the cloud proxy IP, port, username, and password for HTTP.

```
switch(config)#
switch(config)#cloud proxy test
switch(config-cloud-proxy-test)#http 1.2.3.4 1234 username test password
 7 075E731F1A
switch(config-cloud-proxy-test)#
```

9.3 Configuring the Cloud Provider

The following describes configurations required for Cloud HA on different types of clouds.

- Cloud Configuration
- Configuring BFD

- Configuring the Recovery Time
- Cloud Provider Helpful Tips

9.3.1 Cloud Configuration

To have access to the cloud services, the CloudEOS and vEOS Router must be provided with credentials. Additionally, a proxy may be configured for the connection to the cloud services to go through.

AWS Specific Cloud

Complete the following tasks to configure AWS Specific Cloud services.

- Configure Credentials
- Access to AWS Specific Cloud API Server
- If CloudEOS and vEOS is associated with a public IP address, no special configuration is required.
- If CloudEOS and vEOS is not associated with an public IP address, either use AWS Private Link or Proxy configuration

Configure Credentials

In the AWS Specific Cloud configuration, a region must be specified. It is recommended to authorize the CloudEOS and vEOS Router by assigning it an IAM role, but an explicit credential can also be specified.

- IAM Role Configuration - No credentials. See Cloud Provider Helpful Tips for additional information.
- Explicit Credential Configuration

AWS Specific Cloud IAM Role Configuration

The IAM role should be configured on the AWS Specific as shown below. This is the recommended configuration.

- "Trust Relationships" has "ec2.amazonaws.com" as trusted entities.
- "Policy" with "Permissions" for the network related EC2 actions.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "ec2:AssociateRouteTable",
                "ec2:CreateRoute",
                "ec2:CreateRouteTable",
                "ec2:DeleteRoute",
                "ec2:DeleteRouteTable",
                "ec2:DescribeRouteTables",
                "ec2:DescribeVpcs",
                "ec2:ReplaceRoute",
                "ec2:DisassociateRouteTable",
                "ec2:ReplaceRouteTableAssociation",
                "ec2:DescribeNetworkInterfaces",
                "ec2:DescribeInstances",
                "ec2:DescribeSubnets"
            ],
            "Resource": "*"
        }
    ]
}
```

This is applicable only when running in AWS cloud environment and configures various aspects of Cloud HA feature to interact with AWS web services.

Note: The **access-key-id** and **secret access-key** commands are either both configured or both are omitted. If omitted, the Cloud HA Agent will try to use AWS IAM role for security tokens to access and control AWS route tables. Verify the IAM role for the CloudEOS and vEOS router Virtual Machine(VM) is configured properly on the AWS cloud. Refer to AWS documentation to configure IAM role.

```
switch(config)#
switch(config)#cloud provider aws
switch(config-cloud-aws)#access-key 0 ATPAILIL5E982IPT7P3R
switch(config-cloud-aws)#secret access-key 0 M0RRUtAA8I8wYxJB8
switch(config-cloud-aws)#region us-west-1
switch(config-cloud-aws)#proxy test
```

Configure the **backup-gateway**, **primary-gateway**, **Route Table ID(rtb**) and **local interface** for AWS.

The Route Table ID specifies for AWS the backup-gateway and primary gateway, then the destination selects the individual route within the route table to control. The **local-cloud-interface** then points to the interface ID ***eni-867caa86*** (from AWS perspective) of the vEOS router that the traffic should be directed.

```
switch(config)#cloud high-availability
switch(config-cloud-ha)#peer veos2
switch(config-cloud-ha-peer-veos2)#aws
switch(config-cloud-ha-peer-veos2-aws)#backup-gateway rtb-40b72d24
0.0.0.0/0 local-cloud-interface eni-867caa86
switch(config-cloud-ha-peer-veos2-aws)#primary-gateway rtb-2843124c
0.0.0.0/0 local-cloud-interface eni-867caa86
```

Explicit Credential Configuration

The explicit credential should be configured as shown below.

```
switch(config)#cloud provider aws
switch(config-cloud-aws)#region us-west-1
switch(config-cloud-aws)#access-key 0 MYEXAMPLESECRETKEY
switch(config-cloud-aws)#secret access-key 0 MYEXAMPLESECRETKEY
switch(config-cloud-aws)#exit
switch(config-cloud)#exit
```

Azure

There are two authorization models that can be used in Azure: SDK Auth Credentials and Active Directory Credentials. SDK Auth Credentials are the recommended authorization model.

- ***SDK Auth Credentials***

  To generate SDK Auth Credentials, use the `sdk authentication credential-file flash:startup-config` command in the ***config-cloud-azure*** configuration mode.

  ```
  switch(config)#cloud provider azure
  switch(config-cloud-azure)#sdk authentication credential-file
  flash:startup-config
  ```

- ***Active Directory Credentials***

  The following example places the vEOS router into the ***config-cloud-azure*** configuration mode and sets the active directory credentials.

  ```
  switch(config)#cloud provider azure
  ```

```
switch(config-cloud-azure)#active-directory credential
email subscription-id ef16892c-aa46-4aba-ae9a-d4fhsb1c612c
```

9.3.2 Configuring BFD

To configure the BFD link between the HA pair of CloudEOS and vEOS Routers that is used to detect peer failure, the peer IP address and local BFD source interface must be provided. The following example configures Tunnel 2 as a single hop for the source interface for BFD.

```
switch(config)#cloud high-availability
switch(config-cloud-ha)#peer veos2
switch(config-cloud-ha-peer-veos2)#bfd source-interface tunnel 2 single-
hop
```

9.3.3 Configuring the Recovery Time

The **recovery wait-time** command in the ***cloud-ha*** configuration sub-mode configures the amount of time to take back control of local route tables after failure recovery. The following example shows the wait time is configured to 90 seconds.

```
switch(config-cloud-ha-peer-veos2)#recovery wait-time 90
```

9.3.4 Cloud Provider Helpful Tips

The following are needed for Cloud High Availability but are not part of the CloudEOS and vEOS configuration on the CloudEOS and vEOS Router. These may change or can be another way to achieve the same effect without changing the CloudEOS and vEOS Router.

AWS VPN Specific Cloud PrivateLink

AWS VPN Specific Cloud PrivateLink allows a private (no public IP address) CloudEOS and vEOS instance to access services offered by AWS (without using proxy).

The interface VPC endpoints enables a private CloudEOS and vEOS instance to connect to AWS VPN Specific Cloud PrivateLink.

To configure Interface VPC Endpoints:

1. Open the Amazon VPC console and choose **Endpoints** in the navigation panel.
2. Select **Create Endpoint.**
3. Choose the **AWS Services** and select service name **com.amazonaws.<your-region>.ec2.**
4. Choose the VPC and the subnets in each availability zone for the Interface VPC endpoints.
5. Enable private DNS name and set security group accordingly.
6. Select **Create Endpoint**.

Once the Endpoint(s) is created, the EC2 API IP associated with the domain-name will be updated to the endpoint IP.

Additional interface VPC endpoints information can be found at: https://docs.aws.amazon.com/AmazonVPC/latest/UserGuide/vpce-interface.html

9.4 Configuring Cloud High Availability

To enable the Cloud HA and its parameters, use the following configurations.

Enable Cloud High Availability

The **cloud high-availability** command places the CloudEOS and vEOS in the ***cloud-ha*** configuration mode. This example enables cloud high-availability and configures the peer **veos2**.

```
switch(config)#cloud high-availability
switch(config-cloud-ha)#no shutdown
switch(config-cloud-ha)#peer veos2
switch(config-cloud-ha-peer-veos2)#
```

- AWS Specific for High Availability
- Azure Specific for High Availability
- GCP Specific for High Availability

9.4.1 AWS Specific for High Availability

This is applicable only when running in AWS cloud environment and configures various aspects of Cloud HA feature to interact with AWS web services.

Note: The **access-key-id** and **secret access-key** commands are either both configured or both are omitted. If omitted, the Cloud HA Agent will try to use AWS IAM role for security tokens to access and control AWS route tables. Verify the IAM role for the CloudEOS and vEOS router Virtual Machine(VM) is configured properly on the AWS cloud. Refer to AWS documentation to configure IAM role.

```
switch(config)#
switch(config)#cloud provider aws
switch(config-cloud-aws)#access-key 0 ATPAILIL5E982IPT7P3R
switch(config-cloud-aws)#secret access-key 0 M0RRUtAA8I8wYxJB8
switch(config-cloud-aws)#region us-west-1
switch(config-cloud-aws)#proxy test
```

Configure the **backup-gateway**, **primary-gateway**, **Route Table ID(rtb**) and **local interface** for AWS.

The Route Table ID specifies for AWS the backup-gateway and primary gateway, then the destination selects the individual route within the route table to control. The **local-cloud-interface** then points to the interface ID ***eni-867caa86*** (from AWS perspective) of the vEOS router that the traffic should be directed.

AWS

```
switch(config)#cloud high-availability
switch(config-cloud-ha)#peer veos2
switch(config-cloud-ha-peer-veos2)#aws
switch(config-cloud-ha-peer-veos2-aws)#backup-gateway rtb-40b72d24
0.0.0.0/0 local-cloud-interface eni-867caa86
switch(config-cloud-ha-peer-veos2-aws)#primary-gateway rtb-2843124c
0.0.0.0/0 local-cloud-interface eni-867caa86
```

9.4.2 Azure Specific for High Availability

Azure

There are two authorization models that can be used in Azure: SDK Auth Credentials and Active Directory Credentials. SDK Auth Credentials are the recommended authorization model.

- ***SDK Auth Credentials***

  To generate SDK Auth Credentials, use the `sdk authentication credential-file flash:startup-config` command in the *config-cloud-azure* configuration mode.

  ```
  switch(config)#cloud provider azure
  switch(config-cloud-azure)# sdk authentication credential-file
  ```

```
flash:startup-config
```

- ***Active Directory Credentials***

  The following example places the vEOS router into the *config-cloud-azure* configuration mode and sets the active directory credentials.

  ```
  switch(config)#cloud provider azure
  switch(config-cloud-azure)#active-directory credential
  email subscription-id ef16892c-aa46-4aba-ae9a-d4fhsb1c612c
  ```

Configure the **backup-gateway**, **primary-gateway**, **Route Table ID (rtb)**, **resource-group** and **next-hop** for Azure

The **resource group** specified is the one which contains the route table referenced beneath it. The **nextHopIp** is the IP of the vEOS Router interface that traffic should be directed.

Azure

```
switch(config)#cloud high-availability
switch(config-cloud-ha)#peer veos2
switch(config-cloud-ha-peer-veos2)#azure
switch(config-cloud-ha-peer-veos2-azure)#backup-gateway Subnet-2-vEOS-
RouteTable 0.0.0.0/0 10.1.1.4 resource-group my_resource_group_64f86970
ffe24ab
```

9.4.3 GCP Specific for High Availability

In GCP specific cloud configuration, you must specify a Project. The GCP cloud uses either the Default Credentials, or a Service Account mode for authorization.

- **Default Credentials:** Cloud HA uses the Application Default Credentials for authorization and no configuration is required. Setup a GCP instance with a service account with the requisite permissions. For more information on creating an instance with a service account or changing the service account of an existing instance, refer: https://cloud.google.com/compute/docs/access/create-enable-service-accounts-for-instances
- **Service Account:** Specify a service account file. For more information on creating a service account file, refer:https://cloud.google.com/iam/docs/creating-managing-service-account-keys#creating_service_account_keys

Sample service account file:

```
{
  "type": "service_account",
  "project_id": "project-id",
  "private_key_id": "key-id",
  "private_key": "-----BEGIN PRIVATE KEY-----\nprivate-key\n-----END
 PRIVATE KEY-----\n",
  "client_email": "service-account-email",
  "client_id": "client-id",
  "auth_uri": "https://accounts.google.com/o/oauth2/auth",
  "token_uri": "https://accounts.google.com/o/oauth2/token",
  "auth_provider_x509_cert_url": "https://www.googleapis.com/oauth2/v1/c
erts",
  "client_x509_cert_url": "https://www.googleapis.com/robot/v1/metadata/
x509/service-account-email"
}
```

For Default Credentials or Service Account authorization model, the role associated with the CloudEOS instance/service account must have the following permissions:

- compute.networks.get
- compute.networks.updatePolicy
- compute.projects.get
- compute.routes.create
- compute.routes.delete
- compute.routes.get
- compute.routes.list

Example

- Cloud HA GCP configuration example.

```
cloudEos(config)#
cloudEos(config)#cloud provider gcp
cloudEos(config-cloud-gcp)#project gcp-project-name
cloudEos(config-cloud-gcp)#service-account file flash:.gcp_se
rvice_account.json
cloudEos(config-cloud-gcp)#proxy test
```

Specify a destination prefix for Cloud HA routes on GCP to select the individual routes that we wish to control. Since route table is not present in GCP, you can specify an optional tag for each route to simulate the route table.

Example

```
cloudEos(config)#cloud high-availability
cloudEos(config-cloud-ha)#peer veos2
cloudEos(config-cloud-ha-peer-veos2)#gcp
cloudEos(config-cloud-ha-peer-veos2-gcp)#backup-gateway 0.0.0.0/0 tag
 tag2
cloudEos(config-cloud-ha-peer-veos2-gcp)#primary-gateway 0.0.0.0/0 tag
 tag1
```

Limitations

- For Cloud HA on AWS/Azure, pre-create the HA routes before configuring these routes on a CloudEOS instance. For GCP, the HA routes are created automatically by Cloud HA, and thus no need to pre-create any HA routes to avoid route conflicts.
- For GCP, though the HA routes are automatically created by Cloud HA, they do not get deleted automatically from GCP when removed from the Cloud HA configuration from CloudEOS instance. Hence, you need to delete these routes from GCP manually.
- Cloud HA on GCP allows adding routes only in the VPC network corresponding to the first interface (nic0) of the CloudEOS instance.
- Since routes in GCP are added at a per VPC level, specify the tags used to simulate route tables when adding a Cloud HA route using the primary-gateway / backup-gateway command. Use the same tag to apply manually on the Cloud HA routes to a subset of the instances in the VPC network.

9.4.3.1 CloudEOS HA GCP provider commands

Global

- cloud provider gcp

Interface (GCP)

- project
- service-account

Cloud HA GCP Configuration

- primary-gateway
- backup-gateway

Show Commands

- show cloud high-availability routes
- show cloud provider gcp

9.4.3.1.1 cloud provider gcp

The **`cloud provider gcp`** command places the switch in cloud-provider-gcp configuration mode, and allows you to configure cloud provider gcp command parameters. The **`exit`** command returns the CloudEOS to global configuration mode.

Command Mode

Global Configuration

Command Syntax

`cloud provider gcp`

Example

- These commands places the cloud provider for GCP into the configuration mode.

```
cloudEos#config
cloudEos(config)#cloud provider gcp
cloudEos(config-cloud-gcp)#
```

- The **exit** command returns to the global configuration mode.

```
cloudEos(config-cloud-gcp)#exit
cloudEos(config)#
```

9.4.3.1.2 project

The **`project`** command specifies the GCP project name. The **`no project`** command removes the configuration from the CloudEOS running-config.

Command Mode

Global Cloud Provider GCP Configuration

Command Syntax

`project` *name*

`no project` *name*

Parameter

- *name* Specifies the selected GCP project name.

Example

- These commands configures the GCP project.

```
cloudEos(config)#cloud provider gcp
cloudEos(config-cloud-gcp)#project test-project
```

- The **no project** command removes the GCP project.

```
cloudEos(config)#cloud provider gcp
cloudEos(config-cloud-gcp)#no project test-project
```

9.4.3.1.3 service-account

The **`service-account`** specifies the service account file when using the Service Account authorization model. The **`no service-account`** command removes the configuration from the CloudEOS running-config.

Command Mode

Global Cloud Provider GCP Configuration

Command Syntax

`service-account` file *sa-file*

`no service-account` file *sa-file*

Parameter

- *sa-file* Specifies the path to the service account file.

Examples

- These commands configures the Service Account file used.

```
cloudEos(config)#cloud provider gcp
cloudEos(config-cloud-gcp)#service-account file flash:.gcp_se
rvice_account.json
cloudEos(config-cloud-gcp)#
```

- The **no service-account** command removes the Service Account file configuration.

```
cloudEos(config)#cloud provider gcp
cloudEos(config-cloud-gcp)#no service-account file flash:.gcp_se
rvice_account.json
cloudEos(config-cloud-gcp)#
```

9.4.3.1.4 primary-gateway

The **`primary-gateway`** command in the cloud-ha submode adds a primary high availability route for GCP. The **`no primary-gateway`** command removes the route configuration from the CloudEOS and vEOS running-config.

Command Mode

Cloud HA GCP Configuration Submode

Command Syntax

`primary-gateway` *dest-prefix* [*tag rt-tag*]

`no primary-gateway` *dest-prefix* [*tag rt-tag*]

Parameter

- *dest-prefix* Specifies the destination IP prefix.
- *rt-tag* Specifies the route tag.

Examples

- These commands configures a primary high availability route.

```
cloudEos(config)#cloud high-availability
cloudEos(config-cloud-ha)#peer veos2
cloudEos(config-cloud-ha-peer-veos2)#gcp
cloudEos(config-cloud-ha-peer-veos2-azure)#primary-gateway 10.1.0.0/16
 tag tag1
```

- The **no primary-gateway** command removes the primary high availability route configuration.

```
cloudEos(config)#cloud high-availability
cloudEos(config-cloud-ha)#peer veos2
cloudEos(config-cloud-ha-peer-veos2)#gcp
cloudEos(config-cloud-ha-peer-veos2-azure)#no primary-gateway
 10.1.0.0/16 tag tag1
```

9.4.3.1.5 backup-gateway

The `backup-gateway` command in the cloud-ha submode adds a backup high availability route for GCP. The `no backup-gateway` command removes the route configuration from the CloudEOS and vEOS running-config.

Command Mode

Cloud HA GCP Configuration Submode

Command Syntax

`backup-gateway` *dest-prefix* [*tag rt-tag*]

`no backup-gateway` *dest-prefix* [*tag rt-tag*]

Parameter

- *dest-prefix* Specifies the destination IP prefix.
- *rt-tag* Specifies the route tag.

Examples

- These commands configures a backup high availability route.

```
cloudEos(config)#cloud high-availability
cloudEos(config-cloud-ha)#peer veos2
cloudEos(config-cloud-ha-peer-veos2)#gcp
cloudEos(config-cloud-ha-peer-veos2-azure)#backup-gateway 10.1.0.0/16
 tag tag2
```

- The **no primary-gateway** command removes the backup high availability route configuration.

```
cloudEos(config)#cloud high-availability
cloudEos(config-cloud-ha)#peer veos2
cloudEos(config-cloud-ha-peer-veos2)#gcp
cloudEos(config-cloud-ha-peer-veos2-azure)#no backup-gateway
 10.1.0.0/16 tag tag2
```

9.4.3.1.6 show cloud high-availability routes

The `show cloud high-availability routes` command displays the configured local or peer route table, destination IP address and local Next Hop Interface.

Command Mode

EXEC

Command Syntax

`show cloud high-availability routes`

Example

- The `show cloud high-availability routes` command displays high availability routes information.

```
cloudEos(config)#show cloud high-availability routes
```

```
 Peer   Route Type    Destination   Tag
------ ------------ -------------- ------
 peer   primary      21.2.3.14/24
 peer   primary        1.2.3.4/21   tag1
 peer   backup         1.2.3.4/21   tag2
 peer   backup        11.2.3.4/21   az1
```

9.4.3.1.7 show cloud provider gcp

The **`show cloud provider gcp`** command displays cloud provider information for the GCP platform.

Command Mode

EXEC

Command Syntax

`show cloud provider gcp`

Examples

- The **`show cloud provider gcp`** command displays the GCP cloud configuration for Default Credentials authorization model. .

```
cloudEos(config)#show cloud provider gcp
Project: project-123
Authentication mode: default credentials
Service account file:
Proxy: myProxyName
```

- The following example displays the GCP cloud configuration for Service Account authorization model.

```
cloudEos(config)#show cloud provider gcp
Project: project-123
Authentication mode: service account
Service account file: flash:.gcp_service_account.json
Proxy: myProxyName
```

9.5 Full Configurations

AWS VPN Specific Cloud Full Configuration

The following AWS configuration is valid for use with the IAM role.

```
cloud provider aws
 region us-west-1
!
cloud high-availability
 no shutdown
 !
 peer veos2
  aws
   backup-gateway  rtb-40b72d24 0.0.0.0/0 local-cloud-interface
 eni-26cb1d27
   backup-gateway  rtb-17b32973 0.0.0.0/0 local-cloud-interface
 eni-1589e714
   backup-gateway  rtb-54503330 0.0.0.0/0 local-cloud-interface
 eni-56cf1957
   primary-gateway rtb-a4be24c0 0.0.0.0/0 local-cloud-interface
 eni-26cb1d27
```

```
   primary-gateway rtb-40b72d24 0.0.0.0/0 local-cloud-interface
 eni-56cf1957
   primary-gateway rtb-63b02a07 0.0.0.0/0 local-cloud-interface
 eni-1589e714
  peer address 10.2.201.149
  recovery wait-time 5
  bfd source-interface Ethernet1
!
```

Azure Full Configuration

The following Azure configuration is valid for the MSI.

```
cloud high-availability
 no shutdown
 !
 peer veos2
 azure
  backup-gateway Subnet-2-vEOS-RouteTable 0.0.0.0/0 10.1.2.4 resource-
group CloudHaAzure
  backup-gateway Subnet-2-vEOS-RouteTable 10.1.0.0/16 10.1.2.4 resource-
group CloudHaAzure
  backup-gateway Subnet-3-vEOS-RouteTable 10.1.0.0/16 10.1.3.4 resource-
group CloudHaAzure
  backup-gateway Subnet-3-vEOS-RouteTable 0.0.0.0/0 10.1.3.4 resource-
group CloudHaAzure
  primary-gateway Subnet-1-vEOS-RouteTable 10.1.0.0/16 10.1.1.4 resource-
group CloudHaAzure
  primary-gateway Subnet-1-vEOS-RouteTable 0.0.0.0/0 10.1.1.4 resource-
group CloudHaAzure

 peer address 10.1.0.5
 recovery wait-time 10
 bfd source-interface Ethernet1
```

Cloud HA GCP Full Configuration

- The following GCP configuration uses default credentials.

  CloudEOS - 1

  ```
  cloud provider gcp
   project gcp-project-name
  !
  cloud high-availability
   no shutdown
   !
   peer cloudEOS2
    gcp
     backup-gateway 0.0.0.0/0 tag tag1
     backup-gateway 0.0.0.0/0 tag tag2
     primary-gateway 0.0.0.0/0 tag tag3
     primary-gateway 0.0.0.0/0 tag tag4
    peer address 10.3.0.59
    recovery wait-time 5
    bfd source-interface Ethernet1
  !
  ```

- CloudEOS - 2

  ```
  cloud provider gcp
   project gcp-project-name
  !
  ```

```
cloud high-availability
 no shutdown
 !
 peer cloudEOS1
  gcp
   backup-gateway 0.0.0.0/0 tag tag3
   backup-gateway 0.0.0.0/0 tag tag4
   primary-gateway 0.0.0.0/0 tag tag1
   primary-gateway 0.0.0.0/0 tag tag2
  peer address 10.3.0.59
  recovery wait-time 5
  bfd source-interface Ethernet1
!
```

9.6 General Troubleshooting Tips

If the Cloud HA feature is not working as expected, follow these tips for debugging.

- Make sure that the network connectivity is there and DNS server is setup correctly for this feature to work.
- If using Proxy and IAM role under AWS, make sure that the HTTP traffic (TCP port 80) is not proxied to allow for temporarily security credentials to be retrieved by CloudEOS and vEOS instance.
- Make sure to use a corresponding BFD source interface on the peer CloudEOS and vEOS instance. This makes sure that the BFD traffic ingress and egress are on the same interface on each instance.
- For an AWS Specific Cloud, if the IAM role does not work, Arista recommends temporarily using **access-key id** and **secret access key** with enough permissions to make sure the rest of the Cloud HA configuration is fine until you debug IAM role policy.

9.7 Caveats and Limitations

- This feature was introduced in EOS release 4.20.5F which uses ***/mnt/flash/cloud_ha_config.json*** file for Cloud HA configuration without any CLI support. Starting from release 4.20.5.A1 onwards, Cloud HA feature supports CLI based configuration only. Deployments using JSON based configurations are not supported and will not work when the image is upgraded or downgraded. To upgrade image, the administrator must configure Cloud HA feature manually by converting the JSON configuration to equivalent CLI configuration. Downgrading will work as long as the older JSON file is still present in ***/mnt/flash*** directory.
- Only a single **resource-group** is supported across all routing entries for Azure under Cloud specific config HA configuration.
- The Cloud HA feature currently supports only a single peer.
- The AWS IAM role or Azure MSI needs to be configured properly using cloud provider's management tools and should give sufficient permissions to CloudEOS and vEOS instance to access and update route table entries.
- The CloudEOS and vEOS instance should have connectivity to the cloud provider's web services. The access can also be via proxy or using feature like AWS private-link.
- The recovery **wait-time** should not be configured less than 10 seconds to avoid unnecessary route flapping when experiencing periodic instabilities.
- The Cloud HA feature completely validates all the provided cloud configuration to make sure it is consistent and has all required permissions. However, the administrator should not change the provider's network configuration afterwards to avoid any issues during fail-over.

- When there are BFD connectivity issues between the two CloudEOS and vEOS peers, each instance will take over the other's traffic. This cross traffic forwarding on provider's network should not have any adverse affect and still work as active-active even though both of the instance will report as fail-over. After the network connectivity is resolved, the traffic pattern reverts to the normal active-active mode.
- The user can adjust the BFD specific parameters for the session used by Cloud HA feature using normal BFD commands such as multiplier, tx/rx intervals, etc. The Cloud HA fail-over and traffic takeover time is directly correlated with BFD failure detection time. However, when using an overly aggressive BFD, the failover time may incur higher overhead as well may result in greater instability during traffic bursts. Arista recommends using the use default BFD interval which is currently 300 msec with a multiplier of 3.
- The **bfd source-interface** used in Cloud HA configuration should not belong and/or routable via the route-tables controlled by the CloudEOS and vEOS router instance itself to avoid traffic looping issues.
- If the Cloud HA is in an invalid configuration state due to erroneous/mismatched configuration in the provider's cloud, the administrator has to force update the Cloud HA configuration (for example, by shut/no shut under Cloud HA mode) after updating the provider's cloud configuration. In other words, by itself, the Cloud HA feature will not retry the back-end configuration check if it is found to be invalid at the time of configuration.

9.8 Cloud High Availability Commands

Global

- cloud high availability shutdown
- cloud high availability peer

Interface

- backup-gateway
- bfd-source_interface
- peer *peerName*
- primary-gateway (Azure Submode)
- recovery-wait-time

Cloud Provider Commands

Global

- cloud provider azure
- cloud provider aws
- proxy

Interface (Azure)

- active-directory credential email subscription-id

Interface (AWS)

- access-key-id
- region
- secret access-key

Cloud Proxy Commands

Global

- cloud proxy

Interface

- http
- https
- proxy

Show Commands

EXEC

- show cloud high-availability
- show cloud high-availability routes
- show cloud provider aws
- show cloud provider azure
- show cloud proxy

9.8.1 Cloud High Availability CLIs

The Cloud High Availability CLIs are divided into three separate configuration modes:

• **Cloud Proxy** - For proxy related configuration such as http and https.

• **Cloud Provider** - For cloud provider specific configuration such as region, credential, and proxy name.

• **Cloud High-Availability** - For configurations such as route, next-hop, BFD source interface, and peer.

9.8.1.1 access-key-id (CloudEOS and vEOS-AWS)

The cloud provider AWS command places the CloudEOS and vEOS in cloud-provider-aws configuration mode. This configuration mode allows user to configure **`cloud provider aws access-key-id`** command parameters. The **`no access-key-id`** command removes the configuration from the CloudEOS and vEOS ***running-config***. The **`exit`** command returns the CloudEOS and vEOS to global configuration mode.

Note: Supported only on AWS platform.

Command Mode

Cloud Provider AWS Configuration

Command Syntax

`access-key-id` **Password_Type**

`no access-key-id` **Password_Type**

Parameters

Password_Type

- **0** *access-key-id* The password is a clear-text string. Equivalent to no parameter.
- **7** *encrypted_key* The password is an encrypted string.
- Text

Example

The following example configures the AWS access key to encrypted.

```
switch(config)#cloud provider aws
switch(config-cloud-aws)#access-key 0 565656 test
```

Example

The following example removes the AWS access key and returns the vEOS to Global configuration mode.

```
switch(config-cloud-aws)#access-key 0 565656 test
switch(config-cloud-aws)#no access-key 0 565656 test
switch(config)#
```

Example

The following example returns the vEOS to Global configuration mode.

```
switch(config-cloud-aws)#access-key 0 565656 test
switch(config-cloud-aws)#exit
switch(config)#
```

9.8.1.2 active-directory credential email subscription-id (CloudEOS and vEOS-Azure)

The **`active-directory credential email subscription-id`** command configures Azure's **cloud provider azure active-directory credential** parameters. The **`no active-directory`** command removes the configuration from the CloudEOS and vEOS ***running-config***. The **`exit`** command returns the CloudEOS and vEOS to global configuration mode.

Note: Supported only on Azure platform.

Command Mode

Cloud Provider Azure Configuration

Command Syntax

`active-directory credential email subscription-id` *ID*

`no active-directory credential email subscription-id`

Parameters

- *ID* Defines the active directory subscription ID.

Example

The following example places the cloud provider for Azure into the configuration mode.

```
switch(config)#cloud provider azure
switch(config-cloud-azure)#active-directory credential email
 subscription-id
```

Example

```
switch(config)#cloud provider azure
switch(config-cloud-azure)#active-directory credential email
 subscription-id
```

9.8.1.3 azure (CloudEOS and vEOS - Azure)

The **`azure`** command in the **cloud-ha-peer** configuration sub-mode, accessible through the **cloud-ha** configuration mode, allows the user to configure cloud high-availability peer related parameters. The **`exit`**command returns the CloudEOS and vEOS to the to the **cloud-ha-peer** configuration mode.

Note: Supported on Azure platform only.

Command Mode

Global Cloud High Availability Peer Configuration Submode

Command Syntax

azure

Example

The following example configures the peer related information for Azure.

```
switch(config)#cloud high-availability
switch(config-cloud-ha)#peer p
switch(config-cloud-ha-peer-veos2)#azure
switch(config-cloud-ha-peer-veos2-azure)#
```

Example

The following example returns the CloudEOS and vEOS to the ***cloud-ha*** configuration mode.

```
switch(config-cloud-ha-peer-veos2-azure)#exit
switch(config-cloud-ha-peer-veos2)#
```

9.8.1.4 backup-gateway (CloudEOS and vEOS - Azure)

The **cloud high-availability** command in the **cloud-ha** submode assigns the backup gateway parameters for the Azure high availability peered cloud. The **no backup-gateway** command removes the configuration from the CloudEOS and vEOS **running-config**. The **exit** command returns the CloudEOS and vEOS to global configuration mode.

Command Mode

Cloud HA azure configuration submode

Command Syntax

backup-gateway [Azure Rt_Info] resource-group [*Name*]

no backup-gateway

Parameters

- Azure Rt_Info
  - *azure-rt-name* The azure route name.
  - *dest-ip-address/mask* The destination IP address.
  - *local-ip-address* The local IP address.
- resource-group
  - *Name* Azure resource group name.

Example

The following example configures the parameters for the Azure high availability peered cloud.

```
switch(config)#cloud high-availability
switch(config-cloud-ha)#peer veos2
switch(config-cloud-ha-peer-veos2)#azure
switch(config-cloud-ha-peer-veos2-azure)#backup-gateway Rt1 10.10.1.1/10
 1.1.1.1 resource-group test
```

Example

The following example removes the backup-gateway parameters for the Azure high availability peered cloud.

```
switch(config-cloud-ha-peer-veos2-azure)#no backup-gateway Rt1
 10.10.1.1/10
switch(config-cloud-ha-peer-veos2-azure)#
```

9.8.1.5 bfd source-interface (CloudEOS and vEOS)

The `bfd source-interface` command in the **cloud-ha** configuration submode configures BFD source interface parameters for the high availability peer. The `no bfd source-interface` command removed the BFD configurations from the CloudEOS and vEOS ***running-config.***

Command Mode

Global Cloud HA peer configuration mode

Command Syntax

`bfd source-interface` [Interface_Type] single-hop

`no bfd source-interface`

Parameters

- Interface_Type
  - *Ethernet* Ethernet Port number <1-4>.
  - *Loopback* Loopback interface <0-1000>.
  - *Tunnel* Tunnel interface <0-255>.
- Single-hop Single hop BFD . Default is multi-hop.

Example

The following example configures Ethernet 1 as the source interface for BFD and multi-hop set as the default .

```
switch(config)#cloud high-availability
switch(config-cloud-ha)#peer veos2
switch(config-cloud-ha-peer-veos2)#bfd source-interface ethernet 1
```

Example

The following example configures Tunnel 2 as a single hop the source interface for BFD.

```
switch(config)#cloud high-availability
switch(config-cloud-ha)#peer veos2
switch(config-cloud-ha-peer-veos2)#bfd source-interface tunnel 2 single-
hop
```

Example

The following example removes the BFD configuration.

```
switch(config-cloud-ha-peer-veos2)#no bfd source-interface
```

9.8.1.6 cloud high-availability peer

Configures one peer at a time into high availability.

Command Mode

Global Cloud HA configuration submode

Command Syntax

`config-cloud-ha-peer` ***<peer name>***

Example

The following example configures the peer and places it in the cloud high availability configuration mode.

```
switch(config)#cloud high-availability peer peer1
 switch(config-cloud-ha-peer-peer1)#
```

9.8.1.7 cloud high-availability shutdown (CloudEOS and vEOS)

The **`shutdown`** command in the **cloud-ha** configuration mode disables High Availability for virtual EOS instances running in the cloud environment.

Command Mode

Cloud High Availability configuration

Command Syntax

`shutdown`

Example

The following example configures the peer and places it in the cloud high availability configuration mode.

```
switch(config)#cloud high-availability
switch(config-cloud-ha)#shutdown
```

9.8.1.8 cloud provider aws (CloudEOS and vEOS)

The **`cloud provider aws`** command places the CloudEOS and vEOS in **cloud-provider-aws** configuration mode. This configuration mode allows user to configure **`cloud provider aws`** command parameters. The **`exit`** command returns the CloudEOS and vEOS to global configuration mode.

Note: Supported on AWS platform only.

Command Mode

Global Configuration

Command Syntax

`cloud provider aws`

Example

The following example places the cloud provider for AWS into the configuration mode.

```
switch#config
switch(config)#cloud provider aws
switch(config-cloud-aws)#
```

Example

The following example returns to the global configuration mode.

```
switch(config-cloud-aws)#exit
switch(config)#
```

9.8.1.9 cloud provider azure (CloudEOS and vEOS)

The **`cloud provider azure`** command places the CloudEOS and vEOS in **cloud-provider-azure** configuration mode. This configuration mode allows user to configure **`cloud provider azure`**command parameters. The **`exit`** command returns the CloudEOS and vEOS to global configuration mode.

Note: Enabled for Azure platform only.

Command Mode

Global Configuration

Command Syntax

`cloud provider azure`

Example

The following example places the cloud provider for Azure into the configuration mode.

```
switch(config)#cloud provider azure
switch(config-cloud-azure)#
```

9.8.1.10 cloud proxy (CloudEOS and vEOS)

The **`cloud proxy`** command places the CloudEOS and vEOS in **cloud-proxy** configuration mode. This configuration mode allows user to configure the **`cloud proxy`** command parameters. The **`no cloud proxy`** command disables the named proxy and returns the CloudEOS and vEOS to global configuration mode.

Command mode

Global Configuration

Command Syntax

`cloud proxy` ***proxy_name***

`no cloud proxy` ***proxy_name***

Parameters

proxy_name The proxy name to configure.

Example

The following example configures the cloud proxy configuration setting for "test".

```
switch(config)#
switch(config)#cloud proxy test
switch(config-cloud-proxy-test)#
```

Example

This command disables the cloud proxy named "test" and returns the vEOS to global configuration mode.

```
switch(config-cloud-proxy-test)# no cloud proxy test
switch(config)#
```

9.8.1.11 http (CloudEOS and vEOS)

The **http** command in the **cloud-proxy** configuration submode configures the IP, port, username, and password parameters. The **no http** command removes the configured cloud proxy information for HTTP from the ***running-config*** and returns the CloudEOS and vEOS to the global configuration mode.

Command mode

Global Cloud Proxy Configuration

Command Syntax

http [*proxy_IP_port*] [*username*] [*password*]

no http [*proxy_IP_port*] [*username*] [*password*]

Parameters

- *proxy_IP_port* Port number to be used for the HTTP server. Options include:
  - *proxy-ip* IP address used for the HTTPs proxy. Dotted decimal location.
  - *proxy_port* HTTPS proxy port. Value ranges from 1 to 65535.
- *username* Name string.
- *password* Password string.
  - **0** *cleartext-passwd* Indicates the cleartext password is in clear text. Equivalent to the **no** parameter case.
  - **7** *encrypted_passwd* Indicates encrypted password is md5 encrypted.

Example

The following example configures the cloud proxy IP, port and username and password for HTTP.

```
switch(config)#cloud proxy test
switch(config-cloud-proxy-test)# http 1.2.3.4 1234 username test password
 7 075E731F1A
switch(config-cloud-proxy-test)#
```

Example

The following example removes the configured cloud proxy information for HTTP from the ***running-config***.

```
switch(config-cloud-proxy-test)# no http 1.2.3.4 1234 username test
 password 7 075E731F1A
switch(config-cloud-proxy-test)#
```

9.8.1.12 https (CloudEOS and vEOS)

The **https** command in the command in the **cloud-proxy** configuration submode configures the IP, port, username and password parameters. The **no https** command removes the configured cloud proxy information for HTTPS from the **running-config** and returns the CloudEOS and vEOS to global configuration mode.

Command mode

Global Cloud Proxy Configuration

Command Syntax

https [[*proxy_IP_port*] [*username*]] [*password*]

no https [[*proxy_IP_port*] [*username*]] [*password*]

Parameters

- *proxy_IP_port* Port number to be used for the HTTP server. Options include:

- *proxy-ip* IP address used for the HTTPs proxy. Dotted decimal location.
- *proxy_port* HTTPS proxy port. Value ranges from 1 to 65535.

- *username* Name string.
- *password* Password string.
  - **0** *cleartext-passwd* Indicates the cleartext password is in clear text. Equivalent to the **no** parameter case.
  - **7** *encrypted_passwd* Indicates encrypted password is md5 encrypted.

Example

The following example configures the cloud proxy IP and port for HTTPS.

```
switch(config)#
switch(config)#cloud proxy test
switch(config-cloud-proxy-test)#https 10.3.255.155 8888
```

Example

The following example removes the configured cloud proxy HTTPS information from the ***running-config***.

```
switch(config-cloud-proxy-test)#no https 10.3.255.155 8888
switch(config-cloud-proxy-test)#
```

9.8.1.13 peer (CloudEOS and vEOS)

The **`peer`** command in the **cloud-ha** configuration mode identifies which peer to configure by name. The **`peer`** command in the **cloud-ha** configuration submode configures the cloud high-availability resource group peer related parameters. The **`no peer`** command removes the configuration from the CloudEOS and vEOS ***running-config***. The **`exit`**command returns the CloudEOS and vEOS to the **cloud-ha** configuration mode.

Command Mode

Cloud High Availability Configuration

Cloud High Availability Configuration Submode

Command Syntax

`peer` ***ip-address***

`no peer` ***ip-address***

Parameters

- *IP-address* The peer IP address.

Example

The following example configures the cloud high availability peer.

```
switch(config-cloud-ha)#peer veos2
switch(config-cloud-ha-peer-veos2)#
```

Example

The following example configures the peer IP address as 10.10.10.149.

```
switch(config)#cloud high-availability
switch(config-cloud-ha)#peer veos2
switch(config-cloud-ha-peer-veos2)#peer 10.10.10.149
```

Example

The following example removes the peer IP address from the vEOS ***running-config***.

```
switch(config)#cloud high-availability
switch(config-cloud-ha)#peer veos2
switch(config-cloud-ha-peer-veos2)#no peer 10.10.10.149
```

9.8.1.14 primary gateway (CloudEOS and vEOS - Azure)

The **`primary-gateway`** command in the **cloud-ha** submode assigns the primary gateway parameters for the Azure high availability peered cloud. The **`no primary-gateway`** command removes the configuration from the CloudEOS and vEOS **running-config.**

Note: Supported on Azure platform only.

Command Mode

Cloud HA Azure Configuration Submode

Command Syntax

`primary-gateway` **[Azure Rt_Info] resource-group [*Name*]**

`no primary-gateway` **[Azure Rt_Info]**

Parameters

- Azure Rt_Info
  - *azure-rt-name* The azure route name.
  - *dest-ip-address/mask* The destination IP address.
  - *local-ip-address* The local IP address.
- resource-group
  - *Name* Azure resource group name.

Example

The following example configures the parameters for the Azure high availability peered cloud.

```
switch(config)#cloud high-availability
switch(config-cloud-ha)#peer veos2
switch(config-cloud-ha-peer-veos2)#azure
switch(config-cloud-ha-peer-veos2-azure)#primary-gateway Rt1 10.10.1.1/10
 1.1.1.1 resource-group test
```

Example

The following example removes the primary-gateway parameters for the Azure high availability peered cloud.

```
switch(config-cloud-ha-peer-veos2-azure)#no primary-gateway Rt1
 10.10.1.1/10
switch(config-cloud-ha-peer-veos2-azure)#
```

9.8.1.15 proxy (CloudEOS and vEOS)

The **proxy** command configures the cloud provider aws proxy. The **no proxy** command removes the configuration from the **running-config**. The **exit** command returns the CloudEOS and vEOS to global configuration mode.

Note: Supported on AWS platform only.

Command Mode

Global Cloud AWS Configuration

Command Syntax

proxy ***<proxy_name>***

no proxy ***<proxy_name>***

Parameters

- *proxy_name* Proxy name to configure.

Example

The following example configures the Azure cloud proxy named "test".

```
switch(config)#cloud provider aws
switch(config-cloud-aws)#proxy test
```

9.8.1.16 recovery (cloud HA peer)

The **recovery wait-time** command in the **cloud-ha-peer** configuration submode defines the amount of time, in seconds, to take control of the local route tables after failure recovery.

Command Mode

Cloud HA peer configuration

Command Syntax

recovery wait-time ***<time-in-secs>***

no recovery wait-time ***<time-in-secs>***

Parameters

- *time-in-secs* The defined amount of time to take back control of local route tables after failure recovery. Default is 30 seconds.

Example

The following example configures the recovery wait time to 90 seconds.

```
switch(config)#cloud ha
switch(config-cloud-ha)#p1
switch(config-cloud-ha-p1)#recovery wait-time 90
```

9.8.1.17 recovery wait-time (CloudEOS and vEOS)

The **recovery wait-time** command in the **cloud-ha** configuration sub-mode allows takes back control of local route tables after failure recovery. The **no recovery wait-time** command removes the configuration from the CloudEOS and vEOS **running-config**. Default is set at 30 seconds.

Command Mode

Cloud High Availability peer configuration

Command Syntax

`recovery wait-time` ***<period>***

`no recovery wait-time` ***<period>***

`default recovery wait-time` ***<period>***

Parameters

- *period* The defined amount of time to take back control of local route tables after failure recovery. Default is 30 seconds.

Example

The following example shows the wait time is configured to 90 seconds.

```
switch(config-cloud-ha-peer1)#recovery wait-time 90
```

Example

The following example removes the configured the wait time.

```
switch(config-cloud-ha-peer1)#no recovery wait-time
```

Example

The following example configures the wait time to the default of 30 seconds.

```
switch(config-cloud-ha-peer1)#default recovery wait-time
```

9.8.1.18 region (CloudEOS and vEOS - AWS)

The **`cloud provider aws`** command places the CloudEOS and vEOS in **cloud-provider-aws** configuration mode. This configuration mode allows user to configure AWS **`cloud provider region`** command parameters. The **`no region`** command removes the configuration from the CloudEOS and vEOS **running-config**. The **`exit`** command returns the CloudEOS and vEOS to global configuration mode.

Note: Supported on AWS platform only.

Command Mode

Global Cloud Provider AWS Configuration

Command Syntax

`region` ***aws-region***

`no region` ***aws-region***

Parameters

- *aws-region* Specifies the selected region.

Example

The following example configures the cloud provider AWS region.

```
switch(config)#cloud provider aws
switch(config-cloud-aws)#region us-west-1
switch(config-cloud-aws)#
```

Example

The following example removes the cloud provider AWS region.

```
switch(config)#cloud provider aws
switch(config-cloud-aws)#no region us-west-1
switch(config-cloud-aws)#
```

9.8.1.19 secret-access_key (CloudEOS and vEOS - AWS)

The **`cloud provider aws`** command places the CloudEOS and vEOS in **cloud-provider-aws** configuration mode. This configuration mode allows user to configure **`cloud provider aws secret access-key`** command parameters. The **`no secret access-key`** command removes the configuration from the CloudEOS and vEOS **running-config**. The **`exit`** command returns the CloudEOS and vEOS to global configuration mode.

Note: Supported on AWS platform only.

Command Mode

Global Cloud Provider AWS configuration

Command Syntax

`secret access-key` ***password_type***

`no secret access-key` ***password_type***

Parameters

- **0** *access-key-id* The password is a clear-text string. Equivalent to no parameter.
- **7** *encrypted_key* The password is an encrypted string.
- Text

Example

The following example configures the AWS secret access key.

```
switch(config)#cloud provider aws
switch(config-cloud-aws)#secret access-key 0 565656 test
switch(config-cloud-aws)#
```

Example

The following example removes the secret access key from the CloudEOS and vEOS running-config.

```
switch(config-cloud-aws)#no secret access-key 0 565656 test
switch(config-cloud-aws)#
```

Example

The following example returns the vEOS to Global configuration mode.

```
switch(config-cloud-aws)#secret access-key 0 565656 test
switch(config-cloud-aws)#exit
switch(config)#
```

9.8.1.20 show cloud high-availability (CloudEOS and vEOS)

The **`show cloud high-availability`** command displays the high availability configured settings.

Command Mode

EXEC

Command Syntax

`show cloud high-availability`

Example

This command displays details and status of the cloud high-availability configuration.

```
switch#show cloud high-availability
Cloud HA Configuration:
Peer address                       : 10.2.201.149
Source interface                   : Ethernet1
Enabled                            : True
Failover recovery time             : 5
Status                             : valid
State                              : ready
Last failover time                 : never
Last recovery time                 : never
Last config validation start time : 0:26:08 ago
Last config validation end time   : 0:26:06 ago
Failovers                          : 0
```

9.8.1.21 show cloud high-availability routes

The **`show cloud high-availability routes`** command displays the configured local or peer route table, destination IP address and local Next Hop Interface.

Command Mode

EXEC

Command Syntax

`show cloud high-availability routes`

Example

The example below displays high availability routes information.

```
switch(config)#show cloud high-availability routes
Peer Route  Type Route  ID Destination Next Hop   Interface
----------- ----------- -------------- ---------- ------------
veos6       primary     rtb-1dc75679   0.0.0.0/0  eni-e61d95e7
veos6       primary     rtb-a29617c6   0.0.0.0/0  eni-69109868
veos6       primary     rtb-acc756c8   0.0.0.0/0  eni-7f1d957e
veos6       backup      rtb-43c65727   0.0.0.0/0  eni-7f1d957e
veos6       backup      rtb-71c65715   0.0.0.0/0  eni-e61d95e7
veos6       backup      rtb-aca223c8   0.0.0.0/0  eni-69109868
```

9.8.1.22 show cloud provider aws (CloudEOS and vEOS - AWS)

The **`show cloud provider aws`** command displays cloud provider information for the AWS platform.

Command Mode

EXEC

Command Syntax

`show cloud provider aws`

Example

The following example displays the AWS cloud configuration.

```
switch#show cloud provider aws
Cloud AWS Configuration
Region                    : us-west-1
Access key ID             :
Access secret key         :
Proxy                     : test
```

Example

The following example displays the primary and backup gateway information for the AWS cloud provider.

```
switch#show run section cloud
cloud provider aws
 us-west-1
 proxy test
!
cloud high-availability
 no shutdown
 !
 peer vEOS12
  aws
   backup-gateway  rtb-40b72d24 0.0.0.0/0 local-cloud-interface
 eni-26cb1d27
   backup-gateway  rtb-17b32973 0.0.0.0/0 local-cloud-interface
 eni-1589e714
   backup-gateway  rtb-54503330 0.0.0.0/0 local-cloud-interface
 eni-56cf1957
   primary-gateway rtb-a4be24c0 0.0.0.0/0 local-cloud-interface
 eni-26cb1d27
   primary-gateway rtb-e64b2882 0.0.0.0/0 local-cloud-interface
 eni-56cf1957
   primary-gateway rtb-63b02a07 0.0.0.0/0 local-cloud-interface
 eni-1589e714
  peer address 10.2.201.149
 recovery wait-time 5
 bfd source-interface Ethernet1
!
cloud proxy test
 https 10.3.255.155 8888
```

9.8.1.23 show cloud provider azure (CloudEOS and vEOS - Azure)

The **`show cloud provider azure`** command displays Azure cloud provider information.

Command Mode

EXEC

Command Syntax

`show cloud provider azure`

Example

The following example displays the Azure cloud configuration.

```
switch#show cloud provider azure
Cloud Azure Configuration:
Active credentials           : SDK authentication credential file
SDK auth credentials file    : flash:
Proxy name                   :
```

```
Active directory credentials :
```

9.8.1.24 show cloud proxy

The **`show cloud proxy`** command displays cloud proxy information.

Command Mode

EXEC

Command Syntax

`show cloud proxy` **[<*proxy_name*>]**

Parameters

- *proxy_name* Identifies the selected proxy by name.

Example

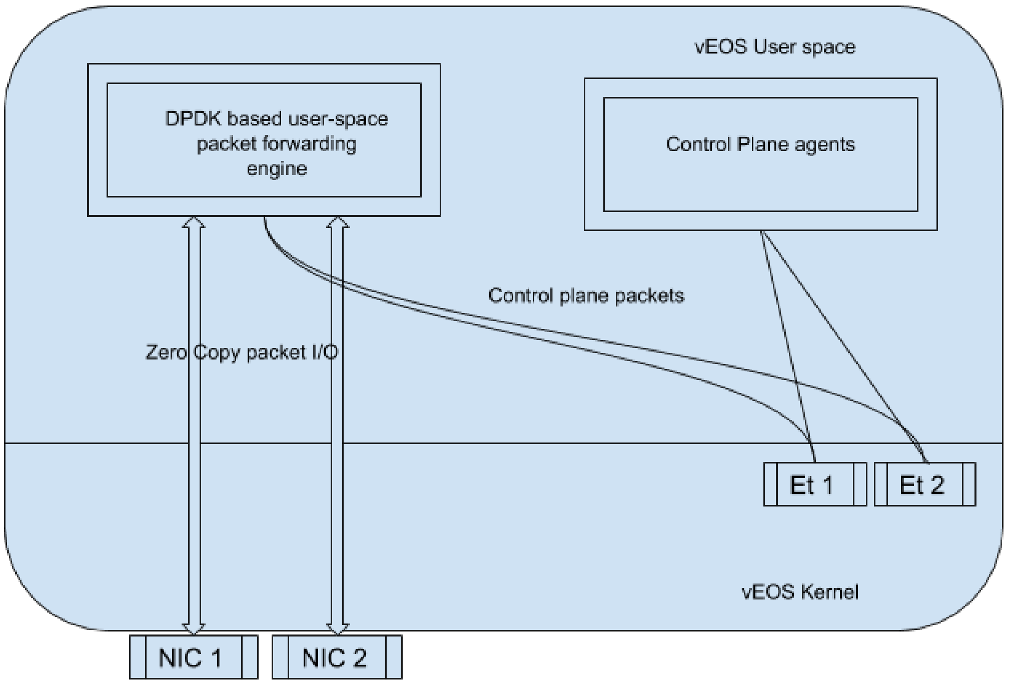
```
#show cloud proxy [<proxy_name>]
Proxy Name : MyProxyName
Http proxy : 1.2.3.4:8080
Https proxy : 1.2.3.4:4443
Proxy user : proxyuser1
Proxy password : obfuscatedpassword
```

Chapter 10

DPDK Mode

This chapter will focus on new DPDK based vEOS, while highlighting the changes from the existing kernel based vEOS, as needed.

The Arista vEOS Router is a cloud-grade, feature-rich, multi-cloud and multi-hypervisor virtual router that empowers enterprises and cloud providers to build consistent, highly secure and scalable multi-cloud networks.



The vEOS Router can run in two modes : DPDK (high performance) and kernel, each with its own set of supported features. From vEOS-Router-4.23.0FX new installations of vEOS router from the hypervisor image or on public cloud by default run in high performance DPDK mode. Going forward all the new features and development will be in the DPDK mode.

Both flavors of vEOS perform all the packet forwarding operations in software, but use different software components for the same. DPDK based vEOS which use DPDK to perform packet forwarding operations is much more efficient. Note, that while DPDK mode has more number of features than the kernel mode, there are certain features that are available only in the kernel mode like Zone Based Segmentation (ZSS) and sflow.

Note: We publish both 64 bit and 32 bit mode we recommend users to use 64 bit mode. Customers who upgrade from 32 bit to 64 bit mode need to be aware that the 64bit mode requires 30 percent more memory than 32 bit mode. In general, 64 bit mode can scale better and has higher datapath performance. The public cloud images are only available in 64 bit mode.

This chapter includes the following sections:

- Platform Compatibilitys
- Hardware Resource Requirements
- Switching to DPDK Mode

- CloudEOS and vEOS-DPDK Mode Verification
- vEOS vCPU Core Allocation
- Monitoring Datapath CPU utilization in DPDK Mode
- General Troubleshooting

10.1 Platform Compatibility

The following platforms supports vEOS-DPDK mode.

- VMWare ESXi 6.0+
  - Supported NICs
    - VMWare vmxnet3 (para-virtualized)
    - Intel x520/82599 PCIe passthrough and SRIOV mode
- Linux / KVM
  - RHEL/CentOS 7.0+
  - Ubuntu 18.04+
    - Supported NICs
      - Virtio-net (para-virtualized)
      - Intel x520/82599 PCIe passthrough and SRIOV mode

10.2 Hardware Resource Requirements

The vEOS Router DPDK mode depends on the features available in modern CPUs, and thus it is important that vEOS VMs are deployed on only certain types of server platforms in order to meet performance benchmarks. Ideal server configuration should be similar to the following.

- Intel E5-26XX v4 or later class of CPU
  - > 2.2 GHz core frequency for best performance
  - Minimum 2 CPU cores reserved for vEOS
    - At-least 2 additional cores will be used by hypervisor software
- - Hyper-threading disabled for best performance
  - non-NUMA or ensure vEOS resources are from single NUMA node
  - Power saving and frequency scaling features disabled in BIOS/Firmware/Hypervisor
- High speed hard-drive (~7200 RPM) or SSD and minimum 8 GB of free disk space.
- At-least 4 GB for RAM reserved for vEOS.
  - 4 GB RAM can only support 3 VRFs
  - 8 GB required for supporting up-to 8 VRFs.
  - Server should have more memory for use by hypervisor software.

In addition to the server requirements, user need to configure the hypervisor to provide memory and CPU reservation for vEOS VMs to ensure optimum performance is achieved. In addition to this, map each vCPU used by vEOS VM to a unique physical CPU core. These configuration and settings are based on the type of hypervisor used, and information is usually documented in configuration guide(s) provided by the hypervisor vendor.

10.3 Switching to DPDK Mode

All the new installations of CloudEOS and vEOS router from the hypervisor image or public cloud by default run in high performance DPDK mode from CloudEOS and vEOS-Router-4.23.0FX.

If you have an existing instance that runs in the kernel mode that you want to switch to DPDK mode, please consider the following:

- Make sure you are not enabled features that DPDK mode does not support.
- Memory requirements of DPDK mode are listed below under “Hardware resource requirements”. DPDK requires more memory than the kernel mode.
- When upgrading an existing instance make sure the instance has sufficient memory to run in DPDK mode.
- DPDK mode runs the datapath cores in poll mode and therefore it runs the CPU at 100%.

To perform following actions, please ensure that the system is configured to allow bash access, at-least temporarily and the perform the following actions on CloudEOS and vEOS CLI prompt.

```
switch#conf t
switch(config)#bash sudo su -
Arista Networks EOS shell
-bash-4.3# cat /mnt/flash/veos-config
# Use 'MODE' to set the forwarding plane for vEOS. If 'MODE' is set
 multiple times
# the last configuration takes effect.
# 'MODE=linux' runs vEOS with linux forwarding plane
MODE=linux
# 'MODE=sfe' runs vEOS with DPDK forwarding plane
#MODE=sfe
```

Now, please use a text editor to modify this file by commenting out **MODE=linux** and un-commenting **MODE=sfe**. After modification verify the changes and then save the file. The file should look like as shown.

```
-bash-4.3# cat /mnt/flash/veos-config
# Use 'MODE' to set the forwarding plane for vEOS. If 'MODE' is set
 multiple times
# the last configuration takes effect.
# 'MODE=linux' runs vEOS with linux forwarding plane
#MODE=linux
# 'MODE=sfe' runs vEOS with DPDK forwarding plane
MODE=sfe
-bash-4.3# exit
logout
switch(config)# reload
After reload,  vEOS Router will boot up in DPDK mode.
```

Upgrading CloudEOS and vEOS-Kernel to CloudEOS and vEOS-DPDK

Upgrade an existing CloudEOS and vEOS installation from kernel mode to DPDK mode, by copying **EOS.swi to /mnt/flash** , and then follow procedure defined above in Installing CloudEOS and vEOS-DPDK, to switch to DPDK mode. Please note, this process requires a system reload.

10.4 CloudEOS and vEOS-DPDK Mode Verification

To check if CloudEOS and vEOS is running in DPDK mode, verify if the “sfe” agent is running using the following command.

```
switch#show agent sfe ping
show agent sfe ping
Agent Name   Last Ping     Max Ping          Max Ping Response Seen  Last
 Ping Response Seen
---------------------- -------------       ------------
 -----------------------                   ------------------------
Sfe                    1.571 ms     2209.819 ms   2019-11-15 11:14:05
       2019-12-12 15:02:48
```

A system in DPDK mode uses 100% of CPU cycles for each datapath vCPU. This is normal and expected. To ensure that packet forwarding tasks, which are CPU intensive, do not starve control plane and management operations, EOS dedicates CPU cores for control/management functions.

Linux “top” command followed by typing “1” when “top” is running is used to get detailed CPU utilization. The below output shows “top” results for a CloudEOS and vEOS with 2 cores. Depending on the version either “Sfe” or “bessd” will show using the 100% of the datapath core.

```
vEOS-CLI(config)#bash top -n 1
Tasks: 236 total,   1 running, 235 sleeping,   0 stopped,   0 zombie
%Cpu0  :  1.6 us,  0.7 sy,  0.0 ni, 95.1 id,  0.0 wa,  2.6 hi,  0.0 si,
 0.0 st
%Cpu1  :100.0 us,  0.0 sy,  0.0 ni,  0.0 id,  0.0 wa,  0.0 hi,  0.0 si,
 0.0 st
KiB Mem:   8122156 total,  4642632 used,  3479524 free,   255624 buffers
KiB Swap:        0 total,        0 used,        0 free,  1857744 cached

  PID USER       PR  NI  VIRT  RES  SHR S  %CPU %MEM    TIME+  COMMAND
 3355 root       20   0 2186m 239m 201m S 100.1  3.0  39262:38 Sfe [or
 bessd]
 2544 root       20   0  375m  58m  38m S   0.3  0.7 192:36.08 ProcMgr-
worker
 2705 root       20   0  403m 180m 142m S   0.3  2.3 135:53.49 Sysdb
 3102 root       20   0  379m 111m  95m S   0.3  1.4   8:06.90 Ira
 3119 root       20   0  373m  86m  70m S   0.3  1.1  24:34.26 StpTxRx
```

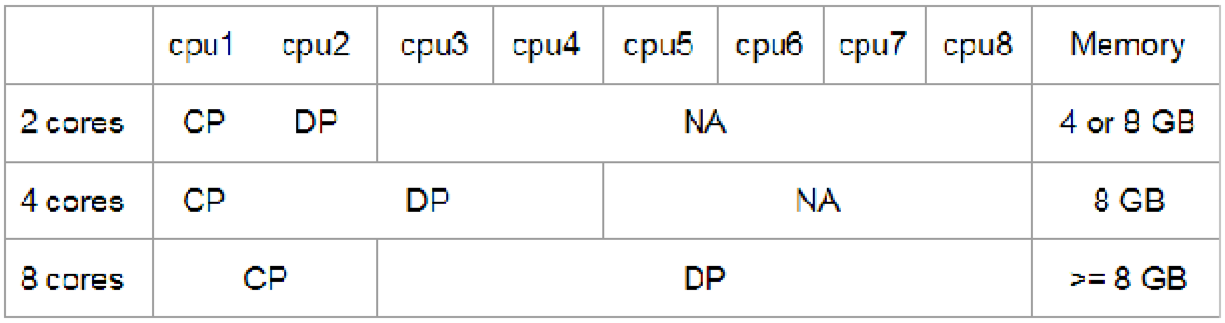
10.5 vEOS vCPU Core Allocation

The vEOS-DPDK VM is supported only on a few standard CPU configurations. Such as:

- 2-core version
- 4-core version
- 8-core version

Depending on the CPU core count, a certain number of CPU cores are reserved for control and management operations, and the remaining are reserved for packet forwarding and data-plane operations.

- On 2-core and 4-core VMs, a single core is reserved for management and control plane and remaining core(s) is/are reserved for packet-forwarding or data-plane operations.
- On 8-core VMs, two cores are reserved for control/management functions and the remaining are reserved for data-plane functions.



	cpu1	cpu2	cpu3	cpu4	cpu5	cpu6	cpu7	cpu8	Memory
2 cores	CP	DP	NA						4 or 8 GB
4 cores	CP	DP			NA				8 GB
8 cores	CP		DP						>= 8 GB

10.6 Monitoring Datapath CPU utilization in DPDK Mode

Since DPDK runs in poll mode, it always shows the CPU utilization is 100% utilized. To verify this use **sfe platform** command and see what percentage of CPU is used for packet processing.

```
switch#platform sfe bessctl
shType "help" for more information.
localhost:10514 $ show busy
   Worker ID      Busy       PPS
           0         0         0
           1         0         0
           2         0         0
localhost:10514 $ show busy
   Worker ID      Busy       PPS
           0         0         0
           1         0         0
           2         0         0
```

In addition to this, a syslog message is logged if the CPU utilization is over 80% for 60 seconds, all the while doing useful packet processing.

```
2019-10-28 23:44:11.776909  8908 Log
0 %SFE-4-CORE_SUSTAINED_BUSY: Software Data Plane Forwarding service
 is experiencing heavy load as it has been80 percent busy over last 60
 seconds
2019-10-29 15:54:53.394284  8908 Log
0 %SFE-4-CORE_SUSTAINED_BUSY: Software Data Plane Forwarding service
 is experiencing heavy load as it has been80 percent busy over last 60
 seconds
```

Note: That these features are for capacity planning and are intended to be used _after_ high CPU alarms are turned off in the hypervisor. Sfe/DPDK runs at 100 % always, and if the alarms are turned off, then this is the way to distinguish between CPU usage due to useful packet processing task and idle spin.

10.7 General Troubleshooting

To find information about packet forwarding agent, please check the following log file.

```
vEOS-CLI(config)# bash cat /var/log/agents/bessd.INFO
```

In case of errors, another log file would be generated by the system and can be accessed by using the following command.

```
vEOS-CLI(config)# bash cat /var/log/agents/bessd.FATAL
```

In addition to the aforementioned log file(s), syslog and EOS show-tech are also a valuable source of troubleshooting information.

Chapter 11

IPsec Support

The vEOS Router provides robust support for the use of IPsec to establish and maintain IPsec tunnels for secure or encrypted communications between virtual router peer instances as well as virtual peer instances to non-virtual routers.

The vEOS Router supports the use of IPsec to:

- Secure the communications between vEOS Router instances.
- Secure the communications between vEOS Router instances and third party virtual router instances.

Note: For the latest information on the types of virtual routers that can share IPsec tunnels with vEOS Router, see the vEOS Router Release Notes.

- Supported Tunnel Types

  The vEOS Router supports the use of two basic types of IPsec tunnels. The tunnel types are determined based on the encapsulation mode.
- Requirements when Behind a NAT

  The vEOS Router supports the use of NAT-Traversal to communicate with the remote peer virtual router. To ensure that the tunnel configuration between the vEOS Router and peer router is successful, make sure that vEOS Router tunnel configuration meets the requirements for using NAT.

  Note: NAT-Traversal for IPsec is not supported for DCS-7020SRG.
- Using IPsec on CloudEOS and vEOS Router Instances

  The vEOS Router enables you to establish and maintain GRE-over-IPsec and VTI IPsec tunnels for secure or encrypted communications between peer vEOS Router instances.
- Using IPsec on CloudEOS and vEOS and Third Party Devices

  The vEOS Router enables you to establish and maintain IPsec tunnels for secure or encrypted communications between vEOS Router instances and third party peer router instances.
- CloudEOS IPsec Connectivity to Azure Virtual Network Gateway

11.1 Supported Tunnel Types

The CloudEOS and vEOS Router supports the use of two basic types of IPsec tunnels. The tunnel types are determined based on the encapsulation mode.

The supported tunnel types are:

GRE-over-IPsec

- In GRE-over-IPsec encapsulation mode, the application payload is first encapsulated within a GRE packet. IPsec then encrypts the GRE packet, which results in the packet being encapsulated and encrypted by the IPsec header.
- Select this encapsulation type by specifying **tunnel mode gre** for the tunnel interface to which the IPsec profile is applied. This ensures that the packets forwarded on the interface are encrypted.
- When using GRE-over-IPsec encapsulation mode, both IPsec mode options are supported (select either **transport** or **tunnel**).

VTI IPsec

- In VTI encapsulation mode, the application payload is directly encapsulated and encrypted by the IPsec header.
- Select this encapsulation type by specifying **tunnel mode ipsec** for the tunnel interface to which the IPsec profile is applied. This ensures that the packets forwarded on the interface are encrypted.
- When using VTI encapsulation mode, set the IPsec mode to **tunnel**. The **transport** option under the IPsec mode has no effect.

11.2 Requirements when Behind a NAT

The CloudEOS and vEOS Router supports the use of NAT-Traversal to communicate with the remote peer behind a NAT. Configure the tunnel source with the outgoing interface IP address on the router.

Flow Parallelization

To achieve high throughput over an IPsec connection, enable the IPsec flow parallelization feature. When the feature is enabled, multiple cores are used to parallelize the IPsec encryption and decryption processing. To enable this feature, include the **`flow parallelization encapsulation udp`** command in the IPsec profile configuration.

Note: The feature must be enabled on both sides of the tunnel. Other vendors do not support Flow Parallelization.

Note: This feature should be used with GRE over IPsec.

If the IPsec session is established without the feature enabled, complete the following tasks:

- Under the IPsec profile for the tunnel use the **`flow parallelization encapsulation udp`** command to enable the feature.
- Shutdown the tunnel on the tunnel interface.
- Bring the tunnel back up on the tunnel interface. After it is up, this enables the feature.

11.3 Using IPsec on CloudEOS and vEOS Router Instances

The CloudEOS and vEOS Router establishes and maintains GRE-over-IPsec and VTI IPsec tunnels for the secure or encrypted communications between peer CloudEOS and vEOS Router instances.

- Topology
- Configuring IPsec Tunnels on CloudEOS and vEOS Router Instances
- Examples of Running-configurations for GRE-over-IPsec Tunnels
- Examples of Running-configurations for VTI IPsec Tunnels

11.3.1 Topology

Use the vEOS Router to establish and maintain IPsec tunnels between peer vEOS Router instances in different topologies of varying complexity.

The diagram below represents a basic IPsec tunnel configuration in which vEOS Router instances are using an IPsec tunnel.

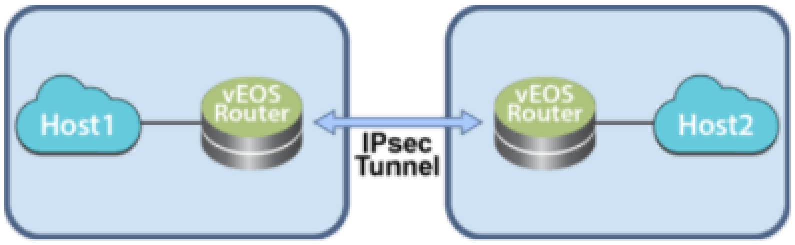


Figure 18: vEOS Router Instance Using a Basic IPsec Tunnel

The vEOS Router establishes and maintains IPsec tunnels for secure or encrypted communications between vEOS Router instances and third party devices peer router instances.

The basic process for establishing secure communications using IPsec involves the following tasks:

- Creating IKE Policy for establishing IKE with the peer.
- Specifying the encryption, integrity protocols for the Security Association (SA) Policy.
- Apply IKE and SA policies to a given profile.
- Apply the profile to a tunnel interface.

11.3.2 Configuring IPsec Tunnels on CloudEOS and vEOS Router Instances

Use this procedure to configure GRE-over-IPsec or VTI IPsec tunnels on peer CloudEOS and vEOS Router instances.

The procedure provides all of the steps required to set up either GRE-over-IPsec or VTI IPsec tunnels. Most of the steps are the same for both tunnel types (steps 1 through 6 are the same). Step 7 is the step to select the tunnel type.

Note: CloudEOS and vEOS Router by default uses IKE version 2 for all IPsec tunnels. To configure a tunnel that uses IKE version 1, explicitly configure the CloudEOS and vEOS Router to use IKE version 1.

Procedure

Complete the following steps to configure GRE-over-IPsec or VTI IPsec tunnels on CloudEOS and vEOS Router instances. This configuration will be the default IKE version 2 procedure.

1. Use this command to enter IP security mode.

```
switch(config)#ip security
```

2. To use IKE version 1, complete the following before completing the default IKE version the steps below.

```
switch(config)#ip security
switch(config-ipsec)#ike policy ike-peerRtr
switch(config-ipsec-ike)#version 1
```

3. Create an IKE Policy to be used to communicate with the peer to establish IKE. You have the option of configuring multiple IKE policies.

   The default IKE Policy values are:

   - **Encryption- *AES256***
   - **Integrity - *SHA256***
   - **DH group - *Group 14***
   - **IKE lifetime - *8 hours***

```
switch(config-ipsec)#ike policy ike-vrouter
switch(config-ipsec-ike)#encryption aes256
switch(config-ipsec-ike)#integrity sha256
switch(config-ipsec-ike)#dh-group 24
switch(config-ipsec-ike)#version 2
```

4. If the router is behind a NAT, configure the **local-id** with the local public IP address. The public IP corresponds to the underlying interface over which the IKE communications are done with the peer.

```
switch(config-ipsec-ike)#local-id <public ip address>
```

5. Create an IPsec Security Association policy to be used in the data path for encryption and integrity. Use the option of enabling Perfect Forward Secrecy by configuring a DH group to the SA.

   In this example, ***AES256*** is used for encryption, ***SHA 256*** is used for integrity, and Perfect Forward Secrecy is enabled (the **DH group** is ***14***).

```
switch(config-ipsec)#sa policy sa-vrouter
switch(config-ipsec-sa)#esp encryption aes256
switch(config-ipsec-sa)#esp integrity sha256
switch(config-ipsec-sa)#pfs dh-group 14
switch(config-ipsec-sa)#sa lifetime 2
switch(config-ipsec-sa)#exit
```

6. Bind or associate the IKE and SA policies together using an IPsec profile. Provide a shared-key, which must be common on both peers. The default profile assigns default values for all parameters that are not explicitly configured in the other profiles.

   In this example, **tunnel mode** is set to **transport**. The IKE Policy ***ike-peerRtr*** and SA Policy ***sa-peerRtr*** are applied to profile ***peer-Rtr***. Dead Peer Detection is enabled and configured to delete the connection when the peer is down for more than 50 seconds. The peer ***peer-Rtr*** is set to be the responder.

```
switch(config-ipsec)#profile default
switch(config-ipsec-profile)#ike-policy ikedefault
switch(config-ipsec-profile)#sa-policy sadefault
switch(config-ipsec-profile)#shared-key arista
switch(config-ipsec)#profile vrouter
switch(config-ipsec-profile)#ike-policy ike-vrouter
switch(config-ipsec-profile)#sa-policy sa-vrouter
switch(config-ipsec-profile)#dpd 10 50 clear
switch(config-ipsec-profile)#connection add
switch(config-ipsec-profile)#mode transport
```

7. Configure the WAN interface to be the underlying interface for the tunnel. You must specify an L3 address for the tunnel. If you do not, the vEOS Router cannot route packets using the tunnel.

```
switch(config)#interface Et1
switch(config-if-Et1)#no switchport
switch(config-if-Et1)#ip address 1.0.0.1/24
switch(config-if-Et1)#mtu 1500
```

8. Apply the IPsec profile to a new tunnel interface. You create the new tunnel interface as part of this step. You can choose to configure the tunnel as a GRE-over-IPsec tunnel, or a VTI IPsec tunnel.

   (GRE-over-IPsec): In this example, the new tunnel interface is *Tunnel0*. The new tunnel interface is configured to use IPsec, and the tunnel mode is set to GRE. The other end of the tunnel also needs to be configured as a GRE-over-IPsec tunnel.

```
switch(config)#interface tunnel0
switch(config-if-Tu0)#ip address 1.0.3.1/24
switch(config-if-Tu0)#tunnel mode gre
switch(config-if-Tu0)#mtu 1394
switch(config-if-Tu0)#tunnel source 1.0.0.1
switch(config-if-Tu0)#tunnel destination 1.0.0.2
switch(config-if-Tu0)#tunnel ipsec profile  vrouter
```

(VTI IPsec): To configure a VTI IPsec tunnel, you need to set the tunnel mode to *tunnel mode ipsec*. The other tunnel element settings are the same as the settings for GRE-over-IPsec.

```
switch(config)#interface tunnel0
switch(config-if-Tu0)#ip address 1.0.3.1/24
switch(config-if-Tu0)#tunnel mode ipsec
switch(config-if-Tu0)#mtu 1394
switch(config-if-Tu0)#tunnel source 1.0.0.1
switch(config-if-Tu0)#tunnel destination 1.0.0.2
switch(config-if-Tu0)#tunnel ipsec profile vrouter
```

Optional Steps

To move the tunnel interface to a different VRF, complete step 9. To achieve high throughput, complete step 10.

9. Create the GRE-over-IPsec tunnel interface in a VRF using the `vrf forwarding` command. If a VRF is needed, create one then create and configure the GRE tunnel interface. If tunnels in different VRFs need to share the IPsec connection, configure the same tunnel source, destination, IPsec profile, and a unique tunnel key for each tunnel.

Note: If tunnels in different VRFs need to share the IPsec connection, specify the same source, destination, and IPsec profile.

```
switch(config)#vrf definition red
switch(config-vrf-red)#rd 1:3
switch(config-vrf-red)#interface tunnel0
switch(config-if-Tu0)#tunnel key 100
switch(config-if-Tu0)#vrf forwarding red
switch(config-if-Tu0)#ip address 1.0.3.1/24
switch(config-if-Tu0)#mtu 1394
switch(config-if-Tu0)#tunnel source 1.0.0.1
switch(config-if-Tu0)#tunnel destination 1.0.0.2
switch(config-if-Tu0)#tunnel key 100
switch(config-if-Tu0)#tunnel ipsec profile vrouter
switch(config)#vrf definition blue
switch(config-vrf-blue)#rd 1:4
switch(config-vrf-blue)#interface tunnel1
switch(config-if-Tu1)#tunnel key 200
switch(config-if-Tu1)#vrf forwarding blue
switch(config-if-Tu1)#ip address 1.0.4.1/24
switch(config-if-Tu1)#tunnel mode gre
switch(config-if-Tu1)#mtu 1394
switch(config-if-Tu1)#tunnel source 1.0.0.1
switch(config-if-Tu1)#tunnel destination 1.0.0.2
switch(config-if-Tu1)#tunnel ipsec profile vrouter
```

10. Enable the IPsec flow parallelization feature to achieve high throughput over the IPsec tunnel. To enable the feature, include the flow parallelization encapsulation udp command in the IPsec profile configuration. Then, apply the IPsec profile configuration to the tunnel interface.

(IPsec profile configuration)

```
switch(config-ipsec)#profile vrouter
switch(config-ipsec-profile)#ike-policy ike-vrouter
switch(config-ipsec-profile)#sa-policy sa-vrouter
switch(config-ipsec-profile)#dpd 10 50 clear
switch(config-ipsec-profile)#connection start
switch(config-ipsec-profile)#mode transport
switch(config-ipsec-profile)#flow parallelization encapsulation udp
```

Example: (Applying IPsec profile to tunnel interface)

```
switch(config)#interface tunnel0
switch(config-if-Tu0)#tunnel ipsec profile vrouter
```

Note: Repeat step 9 on the other end of the tunnel. The IPsec flow parallelization feature must be enabled on both end of the tunnel.

11.3.3 Examples of Running-configurations for GRE-over-IPsec Tunnels

The following examples show the running configurations for two CloudEOS and vEOS Router instances (CloudEOS and vEOS1 and CloudEOS and vEOS2). The instances are the tunnel endpoints of a GRE-over-IPsec tunnel.

Running Configuration for CloudEOS and vEOS1

```
ip security
ike policy ikebranch1
integrity sha256
dh-group 15
!
sa policy sabranch1
sa lifetime 2
pfs dh-group 14
!
profile hq
mode tunnel
ike-policy ikebranch1
sa-policy sabranch1
connection add
shared-key keyAristaHq
dpd 10 50 clear
!
interface Tunnel1
mtu 1404
ip address 1.0.3.1/24
tunnel mode gre
tunnel source 1.0.0.1
tunnel destination 1.0.0.2
tunnel ipsec profile hq
!
interface Ethernet1
no switchport
ip address 1.0.0.1/24
!
```

Running Configuration for CloudEOS and vEOS2

```
ip security
ike policy ikebranch1
integrity sha256
dh-group 15
!
ike policy ikebranch2
dh-group 15
version 1
local-id 200.0.0.1
!
ike policy ikedefault
```

```
!
sa policy sabranch1
sa lifetime 2
pfs dh-group 14
!
profile hq
mode tunnel
ike-policy ikebranch1
sa-policy sabranch1
connection start
shared-key keyAristaHq
dpd 10 50 clear
!
interface Tunnel1
mtu 1404
ip address 1.0.3.2/24
tunnel mode gre
tunnel source 1.0.0.2
tunnel destination 1.0.0.1
tunnel ipsec profile hq
!
interface Ethernet2
no switchport
ip address 1.0.0.2/24
!
```

11.3.4 Examples of Running-configurations for VTI IPsec Tunnels

The following examples show the running configurations for two CloudEOS and vEOS Router instances (CloudEOS and vEOS1 and CloudEOS and vEOS2). The instances are the tunnel endpoints of a VTI IPsec tunnel.

Running Configuration for CloudEOS and vEOS1

```
ip security
ike policy ikebranch1
integrity sha256
dh-group 15
!
sa policy sabranch1
sa lifetime 2
pfs dh-group 14
!
profile hq
mode tunnel
ike-policy ikebranch1
sa-policy sabranch1
connection add
shared-key keyAristaHq
dpd 10 50 clear
!
interface Ethernet1
no switchport
ip address 1.0.0.1/24
!
interface Management1
ip address dhcp
!
interface Tunnel1
mtu 1404
ip address 1.0.3.1/24
tunnel mode ipsec
tunnel source 1.0.0.1
tunnel destination 1.0.0.2
tunnel ipsec profile hq
!
```

Running Configuration for CloudEOS and vEOS2

```
ip security
ike policy ikebranch1
integrity sha256
dh-group 15
!
ike policy ikebranch2
dh-group 15
version 1
local-id 200.0.0.1
!
ike policy ikedefault
!
sa policy sabranch1
sa lifetime 2
pfs dh-group 14
!
profile hq
mode tunnel
ike-policy ikebranch1
sa-policy sabranch1
connection start
shared-key keyAristaHq
dpd 10 50 clear
!
interface Ethernet2
no switchport
ip address 1.0.0.2/24
!
interface Management1 ip address dhcp
!
interface Tunnel1
mtu 1404
ip address 1.0.3.2/24
tunnel mode ipsec
tunnel source 1.0.0.2
tunnel destination 1.0.0.1
tunnel ipsec profile hq
!
```

11.4 Using IPsec on CloudEOS and vEOS and Third Party Devices

The CloudEOS and vEOS Router establishes and maintains IPsec tunnels for secure or encrypted communications between CloudEOS and vEOS Router instances and third party devices peer router instances.

The basic process for establishing secure communications using IPsec involves these tasks:

- Creating IKE Policy for establishing IKE with the peer.
- Specifying the encryption, integrity protocols for the Security Association (SA) Policy.
- Apply IKE and SA policies to a given profile.
- Apply the profile to a tunnel interface.

- Topology

- Interoperability Support
- CloudEOS and vEOS Router and Palo Alto Firewall VM
- CSR Router Show Commands
- IPsec Show Commands
- CloudEOS and vEOS Routers and CSR
- CloudEOS and vEOS Routers and AWS Specific Cloud Configuration

11.4.1 Topology

Use the vEOS Router to establish and maintain IPsec tunnels between vEOS Router instances and third party router instances in different topologies of varying complexity.

The following diagram represents a basic IPsec tunnel configuration in where a vEOS Router instance and a third party router instance is connected using an IPsec tunnel.

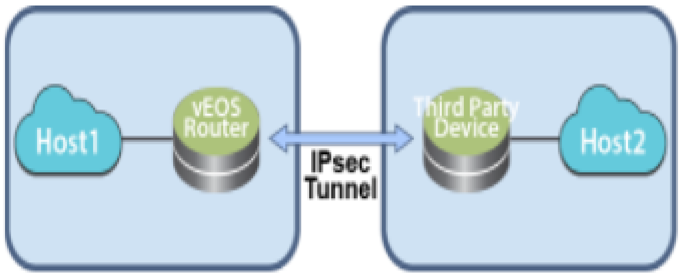


Figure 19: IPsec Interoperability

11.4.2 Interoperability Support

The CloudEOS and vEOS Router establishes and maintains IPsec tunnels for the secure or encrypted communications between CloudEOS and vEOS Router instances and third party device peer router instances.
Below lists the types of IPsec tunnels to set up between CloudEOS and vEOS Router instances and third party virtual router instances.

- **Palo Alto Firewall VM**
  - Set up these types of IPsec tunnels between CloudEOS and vEOS Router instances and Palo Alto firewall VM router instances.
    - VTI IPsec
- **CSR**
  - Set up these types of IPsec tunnels between CloudEOS and vEOS Router instances and CSR router instances.
    - GRE-over-IPsec
    - VTI IPsec
- **AWS VPN Specific Cloud**
  - Set up these types of IPsec tunnels between CloudEOS and vEOS Router instances and AWS VPN Specific Cloud router instances.
    - VTI IPsec
- **vSRX**
  - Set up these types of IPsec tunnels between CloudEOS and vEOS Router instances and vSRX router instances.
    - VTI IPsec

Note: See Supported Tunnel Types for descriptions of the GRE-over-IPsec and VTI IPsec tunnel types.

- CloudEOS and vEOS Routers and Palo Alto firewall AM
- CloudEOS and vEOS Routers and CSR
- CloudEOS and vEOS and AWS Specific Cloud

11.4.3 CloudEOS and vEOS Router and Palo Alto Firewall VM

The CloudEOS and vEOS Router establishes and maintains IPsec tunnels for secure or encrypted communications between CloudEOS and vEOS Router instances and third party device peer router instances.

- CloudEOS and vEOS Router Configuration
- Configuring VTI IPsec Tunnels

11.4.3.1 CloudEOS and vEOS Router Configuration

Use this procedure to configure GRE-over-IPsec tunnels on a CloudEOS and vEOS Router instance. Once the procedure is complete, configure the other tunnel end-point on the third party peer router.

Note: The CloudEOS and vEOS Router by default uses IKE version 2 for all IPsec tunnels. If you want to configure a GRE-over-IPsec tunnel that uses IKE version 1, explicitly configure the CloudEOS and vEOS Router to use IKE version 1.

Procedure

Complete the following steps to configure the CloudEOS and vEOS Router instance to share a GRE-over IPsec tunnel.

To use IKE version 1, complete the section below, then continue with the following steps. To use the default version IKE version 2, begin with Step 1 below.

```
switch(config)#ip security
switch(config-ipsec)#ike policy ike-peerRtr
switch(config-ipsec-ike)#version 1
```

1. Use this command to enter IP security mode.

   ```
   switch(config)#ip security
   ```

2. Create an IKE Policy used to communicate with the peer to establish IKE Phase 1. There is an option of configuring multiple IKE policies.

   The default IKE Policy values are:

   - **Encryption** - AES256
   - **Integrity** - SHA256
   - **DH group** - Group 14
   - **IKE lifetime** - 8 hours

   ```
   switch(config-ipsec)#ike policy ike-vrouter
   switch(config-ipsec-ike)#encryption aes256
   switch(config-ipsec-ike)#integrity sha256
   switch(config-ipsec-ike)#dh-group 24
   switch(config-ipsec-ike)#version 2
   switch(config-ipsec-ike)#exit
   switch(config-ipsec)#ike policy ike-default
   switch(config-ipsec-ike)#version 2
   switch(config-ipsec-ike)#exit
   ```

3. If the router is behind a NAT, configure the **local-id** with the local public IP address.

```
switch(config-ipsec-ike)#local-id <public ip address>
```

4. Create an IPsec Security Association policy used in the data path for encryption and integrity. The is an option of enabling Perfect Forward Secrecy by configuring a DH group to the SA.

   In this example, ***AES256*** is used for encryption, ***SHA 256*** is used for integrity, and Perfect Forward Secrecy is enabled (the DH group is ***14***).

```
switch(config-ipsec)#sa policy sa-vrouter
switch(config-ipsec-sa)#esp encryption aes256
switch(config-ipsec-sa)#esp integrity sha256
switch(config-ipsec-sa)#pfs dh-group 14
switch(config-ipsec-sa)#sa lifetime 2
switch(config-ipsec-sa)#exit
switch(config-ipsec)#sa policy sa-default
switch(config-ipsec-sa)#exit
```

5. Bind or associate the IKE and SA policies together using a IPsec profile. Provide a shared-key, which must be common on both peers. The default profile assigns default values for all parameters that are not explicitly configured in the other profiles.

   In this example, tunnel mode is set to **transport**. The IKE Policy *ike-peerRtr* and SA Policy ***sa-peerRtr*** are applied to profile ***peer-Rtr***. Dead Peer Detection is enabled and configured to delete the connection when the peer is down for more than 50 seconds. The peer (***peer-Rtr***) is set to be the responder.

```
switch(config-ipsec)#profile default
switch(config-ipsec-profile)#ike-policy ikedefault
switch(config-ipsec-profile)#sa-policy sadefault
switch(config-ipsec-profile)#shared-key arista

switch(config-ipsec)#profile peer-Rtr
switch(config-ipsec-profile)#ike-policy ike-peerRtr
switch(config-ipsec-profile)#sa-policy sa-peerRtr
switch(config-ipsec-profile)#dpd 10 50 clear
switch(config-ipsec-profile)#connection add
switch(config-ipsec-profile)#mode transport
```

6. Configure the WAN interface to be the underlying interface for the tunnel. Specify an L3 address for the tunnel. If the L3 address is not specified, the vEOS Router cannot route packets using the tunnel.

```
switch(config)#interface Et1
switch(config-if-Et1)#no switchport
switch(config-if-Et1)#ip address 1.0.0.1/24
switch(config-if-Et1)#mtu 1500
```

7. Apply the IPsec profile to a new tunnel interface. Create the new tunnel interface as part of this step.

   In this example, the new tunnel interface is ***Tunnel0***. The new tunnel interface is configured to use IPsec, and the tunnel mode is set to **GRE**. Configure the other end of the tunnel also as a GRE-over-IPsec tunnel.

```
switch(config)#interface tunnel0
switch(config-if-Tu0)#ip address 1.0.3.1/24
switch(config-if-Tu0)#tunnel mode gre
switch(config-if-Tu0)#mtu 1400
switch(config-if-Tu0)#tunnel source 1.0.0.1
switch(config-if-Tu0)#tunnel destination 1.0.0.2
switch(config-if-Tu0)#tunnel ipsec profile vrouter
```

8. Create the GRE-over-IPsec tunnel interface in a VRF using the **`vrf forwarding`** command. Create the VRF, if needed, then create and configure the GRE tunnel interface. Make sure to specify the tunnel key that is unique across all tunnels.

   Note: If tunnels in different VRFs need to share the IPsec connection, specify the same **source**, **destination**, and **ipsec profile**.

```
switch(config)#vrf definition red
switch(config-vrf-red)#rd 1:3
switch(config-vrf-red)#interface tunnel0
switch(config-if-Tu0)#ip address 1.0.3.1/24
switch(config-if-Tu0)#vrf forwarding red
switch(config-if-Tu0)#tunnel mode gre
switch(config-if-Tu0)#mtu 1400
switch(config-if-Tu0)#tunnel source 1.0.0.1
switch(config-if-Tu0)#tunnel destination 1.0.0.2
switch(config-if-Tu0)#tunnel key 100
switch(config-if-Tu0)#tunnel ipsec profile vrouter

switch(config)#vrf definition blue
switch(config-vrf-blue)#rd 1:4
switch(config-vrf-blue)#interface tunnel1
switch(config-if-Tu1)#ip address 1.0.4.1/24
switch(config-if-Tu1)#vrf forwarding blue
switch(config-if-Tu1)#tunnel mode gre
switch(config-if-Tu1)#mtu 1400
switch(config-if-Tu1)#tunnel source 1.0.0.1
switch(config-if-Tu1)#tunnel destination 1.0.0.2
switch(config-if-Tu1)#tunnel key 200
switch(config-if-Tu1)#tunnel ipsec profile vrouter
```

9. Configure the GRE-over-IPsec tunnel on the peer router.

11.4.3.2 Configuring VTI IPsec Tunnels

The CloudEOS and vEOS Router gives the ability to configure VTI IPsec tunnels between a CloudEOS and vEOS Router instance and a third party peer router instance (such as a Palo Alto firewall VM). First, complete the set up of the tunnel on the CloudEOS and vEOS Router instance, then set up the other end of the tunnel on the third party peer router instance.

- Palo Alto Firewall VM Configuration
- CloudEOS and vEOS and Palo Alto Firewall VM Pairing (VTI IPsec Tunnel)
- CloudEOS and vEOS Router Configuration

11.4.3.2.1 Palo Alto Firewall VM Configuration

Use this configuration when pairing a Palo Alto firewall VM instance and CloudEOS and vEOS Router instance as tunnel endpoints of an IPsec VTI IPsec tunnel.

Note: Refer to the Palo Alto firewall VM documentation for configuration details, including the different interfaces to use to complete the configuration and all the parameters and options.

Supported Tunnel Types

Set up IPsec VTI tunnels when using the Palo Alto firewall VM as a peer router instance with a CloudEOS and vEOS Router instance. IPsec GRE-over-IPsec tunnels using this combination of router instances as peers is not permitted.

Configuration Guidelines

The following are guidelines to follow when configuring the Palo Alto firewall VM.

- IP address settings.

Configure the first interface to be configured (typically named ***eth0***), as the management interface. Use the public IP address on this interface to open the GUI of the Palo Alto firewall VM.

- Management interface.

  Use this interface only for control plane traffic.
- Management profile.

  When configuring the profile, select all of the protocols allowed on the management interface.

Procedure

1. Create a new management profile. Select all of the protocols allowed on the management interface.
2. Create a new tunnel interface and specify the following parameters.
   - **Name:** (for example, ***tunnel 1***.)
   - **Virtual router:** (Select the existing virtual router.)
   - **Security Zone:** (Select the layer 3 internal zone, which is the zone from which the traffic originates.)
   - **IP address:** (Tunnel IP address.)
3. Add a new IKE Crypto profile and specify the IKE options.

   Note: Make sure the settings match the IKE settings on the other end of the tunnel (the CloudEOS and vEOS Router instance). This setting ensures that the IKE negotiation is successful.

   - **Name:** (can be any name.)
   - **Virtual router:** (Select the existing virtual router.)
   - **Security Zone:** (Select the layer 3 internal zone, which is the zone from which the traffic originates.)
   - **IP address:** (Tunnel IP address.)
4. Configure the IKE gateway.

   Note: Make sure the pre-shared key matches the key defined on the other end of the tunnel (the CloudEOS and vEOS Router instance).
5. Add a new IKE Crypto profile for the IKE options.

   Note: Make sure the settings match the IKE settings on the other end of the tunnel (the CloudEOS and vEOS Router instance). This setting ensures that the IKE negotiation of IPsec SAs is successful.
6. Create a new IPsec tunnel, and select the tunnel interface, IKE gateway, IKE crypto profile, and IKE crypto profile defined earlier in the procedure. Selecting these elements binds them to the new tunnel interface.

   Note: Enter the destination IP address of the tunnel interface of the CloudEOS and vEOS Router in the Destination IP option (one of the Tunnel Monitor settings on the Palo Alto firewall VM).
7. Create a new static route for the network that is behind the remote tunnel endpoint. This new static route ensures that the traffic flows through the tunnel to the other tunnel endpoint.
8. `Commit` (save) the configuration.

11.4.3.2.2 CloudEOS and vEOS and Palo Alto Firewall VM Pairing (VTI IPsec Tunnel)

The following example shows a VTI IPsec tunnel between a CloudEOS and vEOS Router instance and a third party Palo Alto firewall VM router instance.

Running Configuration for CloudEOS and vEOS1

```
ip security
  ike policy ikebranch1
```

```
    integrity sha256
    dh-group 15
  !
sa policy sabranch1
    sa lifetime 2
    pfs dh-group 14
  !
profile hq
    ike-policy ikebranch1
    sa-policy sabranch1
    connection add
    shared-key keyAristaHq
    dpd 10 50 clear
  !
interface Ethernet1
  no switchport
  ip address 1.0.0.1/24
!
interface Management1
  ip address dhcp
!

interface Tunnel1
  mtu 1404
  ip address 1.0.3.1/24
  tunnel mode ipsec
  tunnel source 1.0.0.1
  tunnel destination 1.0.0.2
  tunnel ipsec profile hq
!
```

Running Configuration on Palo Alto Firewall VM

```
"ike": {
               "crypto-profiles": {
                  "ike-crypto-profiles": [
                 {
                        "@name": "veos12-IKE-Phase1",
                        "hash": {
                           "member": "sha512"
                        },
                        "dh-group": {
                           "member": "group20"
                        },
                        "encryption": {
                           "member": "aes-256-cbc"
                        },
                        "lifetime": {
                           "hours": "8"
                        }
                     }
        ]

 "ipsec-crypto-profiles": [
               {
                        "@name": "veos12-IPSEC-Phase2",
                        "esp": {
                           "authentication": {
                              "member": "sha256"
                           },
                           "encryption": {
                              "member": "aes-256-cbc"
```

```
                            }
                         },
                         "lifetime": {
                            "hours": "2"
                         },
                         "dh-group": "group20"
                      }

"gateway": {
                  "entry": {
                     "@name": "veos12-IKE-Gateway",
                     "authentication": {
                        "pre-shared-key": {
                           "key": "-AQ==ocHnGzxJ4JVLomPyHuZNlg84S7I=BCiu0
HIvFeFOSQOx/gmhNQ=="
                        }
                     },
                     "protocol": {
                        "ikev1": {
                           "dpd": {
                              "enable": "yes",
                              "interval": "100",
                              "retry": "100"
                           },
                           "ike-crypto-profile": "veos12-IKE-Phase1"
                        },
                        "ikev2": {
                           "dpd": {
                              "enable": "yes"
                           },
                           "ike-crypto-profile": "veos12-IKE-Phase1"
                        },
                        "version": "ikev2-preferred"
                     }

 "tunnel": {
               "ipsec": {
                  "entry": {
                     "@name": "veos12-IPSEC-Tunnel",
                     "auto-key": {
                        "ike-gateway": {
                           "entry": {
                              "@name": "veos12-IKE-Gateway"
                           }
                        },
                        "ipsec-crypto-profile": "veos12-IPSEC-Phase2"
                     },
                     "tunnel-monitor": {
                        "enable": "yes",
                        "destination-ip": "1.0.3.1",
                        "tunnel-monitor-profile": "Test"
                     },
                     "tunnel-interface": "tunnel.1",
                     "disabled": "no"
                  }
               }
            }
         }
```

11.4.3.2.3 CloudEOS and vEOS Router Configuration

Use this procedure to configure VTI IPsec tunnels on an Arista router instance. Complete the procedure, then configure the other tunnel endpoint on the third party peer router.

Note: The CloudEOS and vEOS Router by default uses IKE version 2 for all IPsec tunnels. To configure a VTI IPsec tunnel that uses IKE version 1, explicitly configure the CloudEOS and vEOS Router instance to use IKE version 1.

Procedure

Complete the following steps to configure a CloudEOS and vEOS Router instance to share a VTI IPsec tunnel.

To use IKE version 1, complete the section below, then continue with the steps below. To use IKE version 2, which is the default version, start with Step 1 below.

```
switch(config)#ip security
switch(config-ipsec)#ike policy ike-peerRtr
switch(config-ipsec-ike)#version 1
```

1. Use this command to enter IP security mode.

   ```
   switch(config)#ip security
   ```

2. Create an IKE Policy to communicate with the peer to establish IKE Phase 1 options. There is the option of configuring multiple IKE policies.

   The default IKE Policy values are:

   - Encryption - AES256
   - Integrity - SHA256
   - DH group - Group 14
   - IKE lifetime - 8 hours

   ```
   switch(config)#ip security
   switch(config-ipsec)#ike policy ike-vrouter-PA
   switch(config-ipsec)#integrity sha512
   switch(config-ipsec)#encryption aes256
   switch(config-ipsec)#dh-group 20
   ```

3. If the router is behind a NAT, configure the local-id with the local public IP address.

   ```
   switch(config-ipsec-ike)#local-id <public ip  address>
   ```

4. Create an IPsec Security Association policy in the data path for encryption and integrity. There is the option of enabling Perfect Forward Secrecy by configuring a DH group to the SA.

   In this example, *AES256* is used for encryption, *SHA 256* is used for integrity, and Perfect Forward Secrecy is enabled (the DH group is *20*).

   ```
   switch(config-ipsec)#sa policy sa-vrouter-PA
   switch(config-ipsec)#esp encryption aes256
   switch(config-ipsec)#esp integrity sha256
   switch(config-ipsec)#sa lifetime 2
   switch(config-ipsec)#pfs dh-group 20
   ```

5. Bind or associate the IKE and SA policies together using an IPsec profile. Provide a shared-key, which must be common on both peers. The default profile assigns default values for all parameters that are not explicitly configured in the other profiles.

   In this example, the IKE Policy *ike-vrouter-PA* and SA Policy *sa-vrouter-PA* are applied to profile *vrouter-PA*. Dead Peer Detection is enabled and configured to delete the connection when the peer is down for more than 30 seconds.

   ```
   switch(config-ipsec)#profile vrouter-PA
   switch(config-ipsec-profile)#ike-policy ike-vrouter-PA
   switch(config-ipsec-profile)#sa-policy sa-vrouter-PA
   switch(config-ipsec-profile)#connection start
   ```

```
switch(config-ipsec-profile)#shared-key Arista1234
switch(config-ipsec-profile)#dpd 10 30 clear
```

6. Create a tunnel interface for the VTI tunnel. When tunnel mode is set to IPsec, configure a tunnel key on the vEOS Router instance to ensure that traffic can be forwarded through the tunnel.

```
switch(config)#interface Tunnel1
switch(config-if-Tu1)#mtu 1400
switch(config-if-Tu1)#ip address 1.0.3.1/24
switch(config-if-Tu1)#tunnel mode ipsec
switch(config-if-Tu1)#tunnel source 10.2.201.149
switch(config-if-Tu1)#tunnel destination 10.3.31.30
switch(config-if-Tu1)#tunnel ipsec profile vrouter-PA
```

Configure the VTI IPsec tunnel on the peer router (see Palo Alto Firewall VM Configuration).

11.4.4 CSR Router Show Commands

Describes the available CSR Router show commands and their example outputs.

Use the different show commands for CSR router instances to do the following:

- **View all Existing ISAKMP SAs**
- **View all Existing IPsec SAs**
- **View Crypto (Encryption) Session Details**
- **View IKEv2 SAs**
- **View IKEv2 SA Details**

View all Existing ISAKMP SAs

Use the **`show crypto isakmp sa`** command to view the ISAKMP SAs for all existing or current IPsec connections.

Example

```
switch#show crypto isakmp sa
IPv4 Crypto ISAKMP SA
dst          src          state          conn-id status
1.0.0.1      1.0.0.2      QM_IDLE        1331 ACTIVE
vrouter-ikev1-isakmp-profile

IPv6 Crypto ISAKMP SA
```

View all Existing IPsec SAs

Use the **`show crypto ipsec sa`** command to view the IPsec SAs for all existing or current IPsec connections.

Example

```
switch#show crypto ipsec sa

interface: Tunnel0
      Crypto map tag: Tunnel0-head-0, local addr  1.0.0.2

   protected vrf: (none)


   local ident (addr/mask/prot/port):
(1.0.0.2/255.255.255.255/47/0)
   remote ident (addr/mask/prot/port):
(1.0.0.1/255.255.255.255/47/0)
```

```
    current_peer 1.0.0.1 port 500
       PERMIT, flags={origin_is_acl,}
       #pkts encaps: 1, #pkts encrypt: 1, #pkts digest:  1f
       #pkts decaps: 1, #pkts decrypt: 1, #pkts verify:  1
       #pkts compressed: 0, #pkts decompressed: 0
       #pkts not compressed: 0, #pkts compr. failed:  0
       #pkts not decompressed: 0, #pkts decompress failed:  0
       #send errors 0, #recv errors 0

       local crypto endpt.: 1.0.0.2, remote crypto endpt.:
1.0.0.1
       plaintext mtu 1438, path mtu 1500, ip mtu 1500, ip mtu idb
GigabitEthernet2
       current outbound spi: 0xCB8FB740(3415193408)
       PFS (Y/N): N, DH group: none
       Dummy packet: Initializing
       inbound esp sas:
       spi: 0x36383677(909653623)
       transform: esp-aes esp-sha-hmac ,
       in use settings ={Tunnel, }
       conn id: 5287, flow_id: CSR:3287, sibling_flags
FFFFFFFF80004048, crypto map: Tunnel0-head-0
       sa timing: remaining key lifetime (k/sec):  (4607999/3598)
       IV size: 16 bytes
       replay detection support: Y
       Status: ACTIVE(ACTIVE)

       inbound ah sas:

       inbound pcp sas:

       outbound esp sas:
       spi: 0xCB8FB740(3415193408)
       transform: esp-aes esp-sha-hmac ,
       in use settings ={Tunnel, }
       conn id: 5288, flow_id: CSR:3288, sibling_flags
FFFFFFFF80004048, crypto map: Tunnel0-head-0
       sa timing: remaining key lifetime (k/sec):  (4607999/3598)
       IV size: 16 bytes
       replay detection support : Y
       Status: ACTIVE(ACTIVE)

outbound ah sas:

outbound pcp sas:
```

View Crypto (Encryption) Session Details

Use the **`show crypto session detail`** command to view details about the crypto session for all current IPsec connections.

Example

```
switch#show crypto session detail
Crypto session current status

Code: C - IKE Configuration mode, D - Dead Peer Detection
K - Keepalives, N - NAT-traversal, T - cTCP encapsulation
X - IKE Extended Authentication, F - IKE Fragmentation
R - IKE Auto Reconnect

Interface: Tunnel0
Profile: vrouter-ikev1-isakmp-profile
```

```
Uptime: 00:20:23
Session status: UP-ACTIVE
Peer: 1.0.0.1 port 500 fvrf: (none) ivrf: (none)
      Phase1_id: 1.0.0.1
      Desc: (none)
   Session ID: 0
   IKEv1 SA: local 1.0.0.2/500 remote 1.0.0.1/500 Active
      Capabilities:(none) connid:1332 lifetime:07:39:35
   IPSEC FLOW: permit 47 host 1.0.0.2 host 1.0.0.1
      Active SAs: 2, origin: crypto map
      Inbound: #pkts dec'ed 42 drop 0 life (KB/Sec)
4607997/2375
      Outbound: #pkts enc'ed 44 drop 0 life (KB/Sec)
4607995/2375
```

View IKEv2 SAs

Use the **`show crypto ikev2 sa`** command to view summary information about all IKE version 2 SAs in use by existing IPsec connections.

Example

```
switch#show crypto ikev2 sa
  IPv4 Crypto IKEv2  SA

Tunnel-id   Local               Remote              fvrf/ivrf    Status
1           3.3.3.3/500         3.3.3.1/500         none/none    READY

Encr: AES-CBC, keysize: 128, PRF: sha256, Hash: SHA96,
DH Grp:14, Auth sign: PSK, Auth verify: PSK
   Life/Active Time: 86400/5349 sec

IPv6 Crypto IKEv2  SA
```

View IKEv2 SA Details

Use the **`show crypto ikev2 sa detailed`** command to view details about all IKE version 2 SAs in use by existing IPsec connections.

Example

```
switch#show crypto ikev2 sa detailed
  IPv4 Crypto IKEv2 SA

Tunnel-id   Local           Remote          fvrf/ivrf    Status
1           3.3.3.3/500     3.3.3.1/500     none/none    READY

Encr: AES-CBC, keysize: 128, PRF: sha256, Hash: SHA96,
DH Grp:14, Auth sign: PSK, Auth verify: PSK
    Life/Active Time: 86400/5358 sec
    CE id: 1351, Session-id: 6
    Status Description: Negotiation done
    Local spi: 9FA0B7B1F7746E69     Remote spi:
4B1652D32691E8AF
    Local id: 3.3.3.3
    Remote id: 3.3.3.1
    Local req msg id:   4      Remote req msg id:   8
    Local next msg id:  4      Remote next msg id:  8
    Local req queued:   4      Remote req queued:   8
    Local window:   5          Remote window: 1
    DPD configured for 0 seconds, retry 0
    Fragmentation not configured.
    Extended Authentication not configured.
    NAT-T is not detected
    Cisco Trust Security SGT is disabled
    Initiator of SA : Yes

IPv6 Crypto IKEv2 SA
```

11.4.5 IPsec Show Commands

The CloudEOS and vEOS Router provides commands to view all current or established IPsec tunnels and to view all profiles currently in use by established tunnels.

The show commands are:

- **show ip security connection**
- **show ip security connection detail**

Examples

The example below shows the use of the show ip security connection command to view a summary of all current (established) IPsec tunnels.

```
switch#show ip security connection
Tunnel SourceDest Status Uptime
Tunnel01.0.0.1 1.0.0.2Established14 minutes

Input OutputReauth Time
589 bytes 608 bytes   8 hours
7 pkts36 pkts
```

The example below shows the use of the show ip security connection detail command to view the details for a specified IPsec tunnel.

```
switch#show ip security connection detail
source address 1.0.0.1, dest address 1.0.0.2
 Inbound SPI 0x672F6CC3:
request id 1, mode transport replay-window 32, seq 0x0
stats errors:
 replay-window 0, replay 0, integrity_failed 0
lifetime config:
 softlimit 18446744073709551615 bytes, hardlimit 18446744073709551615
 bytes
 softlimit 18446744073709551615 pkts, hardlimit 18446744073709551615 pkts
 expire add 0 secs, hard 0 secs
lifetime current:
 589 bytes, 7 pkts
 add time Wed Aug 17 17:50:28 2016, use time Wed Aug 17 17:50:31 2016
 Outbound SPI 0xc5f3c373:
request id 1, mode transport replay-window 32, seq 0x0
stats errors:
 replay-window 0, replay 0, integrity_failed 0
lifetime config:
 softlimit 18446744073709551615 bytes, hardlimit 18446744073709551615
 bytes
 softlimit 18446744073709551615 pkts, hardlimit 18446744073709551615 pkts
 expire add 0 secs, hard 0 secs
lifetime current:
 608 bytes, 7 pkts
 add time Wed Aug 17 17:50:28 2016, use time Wed Aug 17 17:50:31 2016
```

The example below shows the use of the show ip sec applied-profile command to view all profiles currently in use by established tunnels.

```
switch#show ip sec applied-profile
Profile Name Interface
Arista Tunnel0
```

11.4.6 CloudEOS and vEOS Routers and CSR

Use this configuration process to set up GRE-over-IPsec tunnels on CSR peer routers. Procedures are provided for configuration using IKE version 1, or IKE version 2. Make sure to use the correct procedure based on the selected version of IKE.

11.4.6.1 CSR Configuration

The configuration of VTI IPsec tunnels on CSR peer router instances is almost identical to the configuration of GRE-over-IPsec tunnels on CSR peer router instances. The only difference in the configurations is tunnel mode.

For VTI IPsec tunnels, tunnel mode must be set to **ipsec** instead of **gre** (for GRE-over-IPsec tunnels, tunnel mode must be set to **gre**.)

This example shows a basic VTI IPsec tunnel configuration for a CSR peer router instance.

Example

```
switch(config)#interface Tunnel0
switch(config-if)#ip address 1.0.3.1 255.255.255.0
switch(config-if)#tunnel source 10.3.31.30
switch(config-if)#tunnel destination 10.2.201.149
switch(config-if)#tunnel mode ipsec ipv4
switch(config-if)#tunnel protection ipsec profile vrouter-ikev1-ipsec-
profile
```

Note: Make sure you use the correct procedure based on the version of IKE you need to use.

11.4.6.2 Sharing IPsec Connections

On CSR, the user can configure multiple GRE tunnels to use the same IPsec connection.

The user needs to add an extra shared keyword after the profile name on every tunnel interface that is to be shared.

```
switch(config)#interface Tunnel0
switch(config-if)#tunnel protection ipsec profile vrouter-ikev2-ipsec-
profile shared
switch(config-if)#exit
```

11.4.6.3 IKEv1 Configuration

The CSR configuration to create a GRE over IPsec tunnel is similar the CloudEOS and vEOS Router setup using **`ikev1 version`**.

To ensure that the v EOS Router can establish a tunnel with CSR, it needs to set the ikev1 version as follows:

```
switch(config)#ip security
switch(config-ipsec)#ike policy ike-peerRtr
switch(config-ipsec-ike)#version 1
```

1. Enter the configuration terminal mode to configure IPsec.

```
switch#config terminal
```

2. Configure a pre-shared key for the vEOS Router and CSR to authenticate each other. Create a keyring to hold the keys.

```
switch(config)#crypto keyring vrouter-keyring
switch(conf-keyring)#pre-shared-key address 1.0.0.2 key arista
```

3. Create an ISAKMP policy. The policy's function is to communicate with the peer to establish IKE Phase 1. In the example below, a policy with AES256 is created with the following parameters: **SHA1**, **DH group** ***15***, **authentication pre-share**, and a **lifetime** of ***28800*** seconds.

```
switch(config)#crypto isakmp policy 1
switch(config-isakmp)#encr aes 256
switch(config-isakmp)#hash sha
switch(config-isakmp)#authentication pre-share
switch(config-isakmp)#group 15
switch(config-isakmp)#lifetime 28800
```

4. Create an ISAKMP profile associated with the vEOS Router to match its outside IP Address and the keyring that was created earlier to identify the pre-shared secret.

```
switch(config)#crypto isakmp profile vrouter-ikev1-isakmp-profile
switch(conf-isa-prof)#keyring vrouter-keyring
switch(conf-isa-prof)#match identity address 1.0.0.2 255.2-55.255.255
switch(conf-isa-prof)#local-address GigabitEthernet2
```

5. Create the IPsec transform-set configuration settings. The transform-set defines the encryption and hash algorithm for the child/IPsec SA. This example creates a transform-set with AES cipher for the ESP encryption and SHA1 for the authentication. The mode for the IPsec is set to **`transport`** mode.

```
switch(config)#crypto ipsec transform-set vrouter-tset esp-aes 256 esp-
sha-hmac
switch(cfg-crypto-trans)#mode transport
```

6. Create the IPsec profile which includes the **transform-set**, **SA idle time**, **lifetime**, and replay windows used to create the child SA.

```
switch(config)#crypto ipsec profile vrouter-ikev1-ipsec-profile
switch(ipsec-profile)#set security-association idle-time 3600
switch(ipsec-profile)#set security-association dummy seconds 3600
switch(ipsec-profile)#set transform-set vrouter-tset
switch(ipsec-profile)#set isakmp-profile vroute-ikev1-isakmp-profile
```

7. Configure the WAN interface as the underlying interface for the tunnel. To be able to route packets, the tunnel is given an L3 IP address.

```
switch(config)#interface GigabitEthernet2
switch(config-if)#ip address 1.0.0.2 255.255.255.0
switch(config-if)#mtu 9001
switch(config-if)#negotiation auto
```

8. Apply the IPsec profile to a tunnel interface. The example creates a tunnel interface (***Tunnel0***) and configures the tunnel interface to use IPsec.

```
switch(config-if)#exit
switch(config)#interface Tunnel0
switch(config-if)#ip address 1.0.3.1 255.255.255.0
switch(config-if)#tunnel source 1.0.0.2
switch(config-if)#tunnel destination 1.0.0.1
switch(config-if)#tunnel protection ipsec profile vrouter-ikev1-ipsec-
profile
switch(config-if)#exit
```

11.4.6.4 IKEv2 Configuration

The CSR configuration to create a GRE over IPsec tunnel is similar to the CloudEOS and vEOS Router setup using **`ikev2 version`**.

By default, the CloudEOS and vEOS Router is configured to run in IKEv2 version. Make sure the version is not set to 1 under the **`ike`** policy. The configuration steps for CSR IKEv2 are a bit different to that of IKEv1.

Complete the following steps to configure the CSR.

1. Enter the configuration terminal mode to configure IPsec.

```
switch#configure terminal
```

2. Create a pre-shared key for CSR and the CloudEOS and vEOS Router to authenticate each other. Create a keyring to hold the keys. Specify the peer CloudEOS and vEOS Router under which the keys and matching IP address of peer are configured.

```
switch(config)#crypto keyring vrouter-ikev2-keyring
switch(conf-keyring)#pre-shared-key address 1.0.0.2 key arista
```

3. Create an IKEv2 proposal to specify the **encryption**, **integrity**, and **group**. In the example, it specifies ***AES256***, ***SHA1***, and DH group ***14***.

```
switch(config)#crypto ikev2 proposal vrouter-ikev2-proposal
switch(config-ikev2-proposal)#encryption aes-cbc-256
switch(config-ikev2-proposal)#integrity sha1
switch(config-ikev2-proposal)#group 14
switch(config-ikev2-proposal)#exit
```

4. Create an IKEv2 policy and attach the proposal created in the previous step.

```
switch(config)#crypto ikev2 policy vrouter-ikev2-policy
switch(config-ikev2-policy)#match fvrf any
switch(config-ikev2-policy)#proposal vrouter-ikev2-proposal
switch(config-ikev2-policy)#exit
```

5. Create an IKEv2 profile and specify the match identity for the remote peer's **address**, **authentication pre-share**, and the keyring that was previously created.

```
switch(config)#crypto ikev2 profile vrouter-ikev2-profile
switch(config-ikev2-profile)#match fvrf any
switch(config-ikev2-profile)#match identity remote address 1.0.0.1
 255.255.255.255
switch(config-ikev2-profile)#authentication remote pre-share key arista
switch(config-ikev2-profile)#authentication local pre-share key arista
switch(config-ikev2-policy)#exit
```

6. Create the IPsec transform-set configuration settings. This step is similar to the step in IKEv1 configuration. The transform-set defines the encryption and hash algorithm for the child/IPsec SA. The example creates a transform-set with AES cipher for the ESP encryption and ***SHA1*** for the authentication. The mode for the IPsec is set to the **`transport`** mode.

```
switch(config)#crypto ipsec transform-set vrouter-tset esp-aes 256 esp-
sha-hmac
switch(cfg-crypto-trans)#mode transport
```

7. Create the IPsec profile similar to IKEv1. This profile includes the **transform-set**, **SA idle time**, **lifetime**, and replay windows that are used to create the child SA and specifies the IKEv2 profile to use.

```
switch(config)#crypto ipsec profile vrouter-ikev2-ipsec-profile
switch(ipsec-profile)#set security-association idle-time 3600
switch(ipsec-profile)#set security-association dummy seconds 3600
switch(ipsec-profile)#set transform-set vrouter-tset
switch(ipsec-profile)#set ikev2-profile vrouter-ikev2-profile
switch(ipsec-profile)#exit
```

8. Configure the interface to use as the underlying interface for the tunnel. To be able to route packets, the tunnel is given an L3 IP address.

```
switch(config)#interface GigabitEthernet2
switch(config-if)#ip address 1.0.0.1 255.255.255.0
switch(config-if)#negotiation auto
```

9. Apply the IPsec profile to a tunnel interface. The example creates a tunnel interface (***Tunnel0***) and configures the tunnel interface to use IPsec.

```
switch(config-if)#exit
switch(config)#interface Tunnel0
switch(config-if)#ip address 1.0.3.1 255.255.255.0
switch(config-if)#tunnel path-mtu-discovery
switch(config-if)#tunnel source 1.0.0.1
switch(config-if)#tunnel destination 1.0.0.2
switch(config-if)#tunnel protection ipsec profile vrouter-ikev2-ipsec-
profile
switch(config-if)#exit
```

11.4.6.5 CloudEOS and vEOS Router (GRE-over-IPsec Tunnel)

The IPsec tunnels represented in these examples include GRE-over-IPsec tunnels on CloudEOS and vEOS Router instances.

Running Configuration for CloudEOS and vEOS

```
ip security
ike policy ikebranch1 encryption aes256 dh-group 15
!
sa policy sabranch1 sa lifetime 2
pfs dh-group 14
!
profile hq
ike-policy ikebranch1 sa-policy sabranch1 connection add
shared-key keyAristaHq dpd 10 50 clear
!
interface Tunnel1
ip address 1.0.3.1/24 tunnel mode gre tunnel source 1.0.0.1
tunnel destination 1.0.0.2 tunnel ipsec profile hq
interface Ethernet1 no switchport
ip address 1.0.0.1/24
```

11.4.6.6 CloudEOS and vEOS Router (VTI IPsec Tunnel)

The IPsec tunnels represented in these examples include VTI IPsec tunnels between CloudEOS and vEOS Router instances and third party CSR router instances.

Running Configuration for CloudEOS and vEOS

```
ip security
  ike policy ikebranch1
    encryption aes256
    dh-group 15
  !
  sa policy sabranch1
    sa lifetime 2
    pfs dh-group 14
  !
  profile hq
    ike-policy ikebranch1
    sa-policy sabranch1
    connection add
    shared-key keyAristaHq
    dpd 10 50 clear
  !
interface Tunnel1
  ip address 1.0.3.1/24
  tunnel mode ipsec
  tunnel source 1.0.0.1
  tunnel destination 1.0.0.2
  tunnel key 100
  tunnel ipsec profile hq
interface Ethernet1
  no switchport
  ip address 1.0.0.1/24
```

11.4.6.7 CSR Commands

The CSR router has show commands for several IPsec tunnel elements on CSR router instances.

11.4.6.8 CSR Router Show Commands

Describes the available CSR Router show commands and their example outputs.

Use the different show commands for CSR router instances to do the following:

- **View all Existing ISAKMP SAs**
- **View all Existing IPsec SAs**
- **View Crypto (Encryption) Session Details**
- **View IKEv2 SAs**
- **View IKEv2 SA Details**

View all Existing ISAKMP SAs

Use the **`show crypto isakmp sa`** command to view the ISAKMP SAs for all existing or current IPsec connections.

Example

```
switch#show crypto isakmp sa
IPv4 Crypto ISAKMP SA
dst          src         state         conn-id status
1.0.0.1      1.0.0.2     QM_IDLE       1331 ACTIVE
vrouter-ikev1-isakmp-profile

IPv6 Crypto ISAKMP SA
```

View all Existing IPsec SAs

Use the **`show crypto ipsec sa`** command to view the IPsec SAs for all existing or current IPsec connections.

Example

```
switch#show crypto ipsec sa

interface: Tunnel0
      Crypto map tag: Tunnel0-head-0, local addr  1.0.0.2

   protected vrf: (none)


   local ident (addr/mask/prot/port):
(1.0.0.2/255.255.255.255/47/0)
   remote ident (addr/mask/prot/port):
(1.0.0.1/255.255.255.255/47/0)
   current_peer 1.0.0.1 port 500
      PERMIT, flags={origin_is_acl,}
      #pkts encaps: 1, #pkts encrypt: 1, #pkts digest:  1f
      #pkts decaps: 1, #pkts decrypt: 1, #pkts verify:  1
      #pkts compressed: 0, #pkts decompressed: 0
      #pkts not compressed: 0, #pkts compr. failed:  0
      #pkts not decompressed: 0, #pkts decompress failed:  0
      #send errors 0, #recv errors 0

      local crypto endpt.: 1.0.0.2, remote crypto endpt.:
1.0.0.1
      plaintext mtu 1438, path mtu 1500, ip mtu 1500, ip mtu idb
GigabitEthernet2
      current outbound spi: 0xCB8FB740(3415193408)
      PFS (Y/N): N, DH group: none
      Dummy packet: Initializing
      inbound esp sas:
      spi: 0x36383677(909653623)
      transform: esp-aes esp-sha-hmac ,
      in use settings ={Tunnel, }
      conn id: 5287, flow_id: CSR:3287, sibling_flags
FFFFFFFF80004048, crypto map: Tunnel0-head-0
      sa timing: remaining key lifetime (k/sec):  (4607999/3598)
      IV size: 16 bytes
      replay detection support: Y
      Status: ACTIVE(ACTIVE)

      inbound ah sas:

      inbound pcp sas:

      outbound esp sas:
      spi: 0xCB8FB740(3415193408)
      transform: esp-aes esp-sha-hmac ,
      in use settings ={Tunnel, }
      conn id: 5288, flow_id: CSR:3288, sibling_flags
FFFFFFFF80004048, crypto map: Tunnel0-head-0
      sa timing: remaining key lifetime (k/sec):  (4607999/3598)
      IV size: 16 bytes
      replay detection support : Y
      Status: ACTIVE(ACTIVE)

outbound ah sas:
```

```
outbound pcp sas:
```

View Crypto (Encryption) Session Details

Use the **`show crypto session detail`** command to view details about the crypto session for all current IPsec connections.

Example

```
switch#show crypto session detail
Crypto session current status

Code: C - IKE Configuration mode, D - Dead Peer Detection
K - Keepalives, N - NAT-traversal, T - cTCP encapsulation
X - IKE Extended Authentication, F - IKE Fragmentation
R - IKE Auto Reconnect

Interface: Tunnel0
Profile: vrouter-ikev1-isakmp-profile
Uptime: 00:20:23
Session status: UP-ACTIVE
Peer: 1.0.0.1 port 500 fvrf: (none) ivrf: (none)
      Phase1_id: 1.0.0.1
      Desc: (none)
   Session ID: 0
   IKEv1 SA: local 1.0.0.2/500 remote 1.0.0.1/500 Active
      Capabilities:(none) connid:1332 lifetime:07:39:35
   IPSEC FLOW: permit 47 host 1.0.0.2 host 1.0.0.1
      Active SAs: 2, origin: crypto map
      Inbound: #pkts dec'ed 42 drop 0 life (KB/Sec)
4607997/2375
      Outbound: #pkts enc'ed 44 drop 0 life (KB/Sec)
4607995/2375
```

View IKEv2 SAs

Use the **`show crypto ikev2 sa`** command to view summary information about all IKE version 2 SAs in use by existing IPsec connections.

Example

```
switch#show crypto ikev2 sa
  IPv4 Crypto IKEv2  SA

Tunnel-id   Local               Remote              fvrf/ivrf    Status
1           3.3.3.3/500         3.3.3.1/500         none/none    READY

Encr: AES-CBC, keysize: 128, PRF: sha256, Hash: SHA96,
DH Grp:14, Auth sign: PSK, Auth verify: PSK
   Life/Active Time: 86400/5349 sec

IPv6 Crypto IKEv2  SA
```

View IKEv2 SA Details

Use the **`show crypto ikev2 sa detailed`** command to view details about all IKE version 2 SAs in use by existing IPsec connections.

Example

```
switch#show crypto ikev2 sa detailed
```

```
  IPv4 Crypto IKEv2 SA

Tunnel-id   Local           Remote          fvrf/ivrf    Status
1           3.3.3.3/500     3.3.3.1/500     none/none    READY

Encr: AES-CBC, keysize: 128, PRF: sha256, Hash: SHA96,
DH Grp:14, Auth sign: PSK, Auth verify: PSK
    Life/Active Time: 86400/5358 sec
    CE id: 1351, Session-id: 6
    Status Description: Negotiation done
    Local spi: 9FA0B7B1F7746E69     Remote spi:
4B1652D32691E8AF
    Local id: 3.3.3.3
    Remote id: 3.3.3.1
    Local req msg id:   4      Remote req msg id:   8
    Local next msg id:  4      Remote next msg id:  8
    Local req queued:   4      Remote req queued:   8
    Local window:   5          Remote window: 1
    DPD configured for 0 seconds, retry 0
    Fragmentation not configured.
    Extended Authentication not configured.
    NAT-T is not detected
    Cisco Trust Security SGT is disabled
    Initiator of SA : Yes

IPv6 Crypto IKEv2 SA
```

11.4.7 CloudEOS and vEOS Routers and AWS Specific Cloud Configuration

Describes the configuration steps for an AWS specific cloud on a CloudEOS and vEOS Router instance.

11.4.7.1 IPsec Between the CloudEOS and vEOS Router and AWS Specific Cloud Configuration

Describes the steps and the running configuration for setting up an IPsec connection between the CloudEOS and vEOS Router and the AWS Specific Cloud. The AWS Specific Cloud only supports IKE1 and not IKE2.

The following configurations are for the minimum requirement of AES128, SHA1, and DH Group 2. These can be modified to take advantage of AES256, SHA256, or other DH groups such as 5, 14-17, and 24.

11.4.7.2 Running-configuration of the CloudEOS and vEOS Router and AWS Specific Cloud

The sample configuration below sets up the running configuration of the CloudEOS and vEOS Router and AWS Specific Cloud. In the configuration, the **local-id** is the external IP of the router when it is behind a NAT device, and the tunnel destination is the external IP of the AWS Specific Cloud.

```
ip security
   ike policy AWS-IKE1
      integrity sha1
      version 1
      local-id 52.165.228.195
   !
   ike policy ikedefault
      encryption aes256
   !
   sa policy AWS-SA1
      esp encryption aes128
      esp integrity sha1
      pfs dh-group 14
   !
   profile AWS-profile
      ike-policy AWS-IKE1
      sa-policy AWS-SA1
      connection start
      sharded-key LwYbARmDJmpFGAOrAbPGk2uQiWwvbmfU
   !
   profile default
      ike-policy
      sa-policy AWS-SA1
      shared-key arista
   !
interface Tunnel1
   ip address 169.254.11.162/30
   tunnel mode ipsec
   tunnel source 10.2.0.4
   tunnel destination 52.53.75.160
   tunnel ipsec profile AWS-profile
```

11.4.7.3 AWS Specific Cloud Configuration

1. **Internet Key Exchange Configuration**

The address of the external interface for a customer gateway must be a static address. the customer gateway can reside behind a device performing network Address Translation (NAT) To ensure that NAT Transversal (NAT-T) can function, add, and update the firewall rules, allow UDP port 4500. Disable NAT-T if the customer gateway is not behind a NAT gateway.

- **Authentication Method:** Pre-Shared Key
- **Pre-Shared Key:** LwYbARmDJmpFGOrAbPGk2uQiWwvbmfU
- **Authentication Algorithm:** sha1
- **Encryption Algorithm:** aes-128-cbc
- **Lifetime:** 28800 seconds
- **Phase 1 Negotiation Method:** main
- **Perfect Forward Secrecy:** Diffie-Hellman Group 2

11.4.7.4 AWS Specific Cloud Configuration Modifications

1. **Internet Key Exchange SA Configuration**

   The address of the external interface for the customer gateway must be a static address. The customer gateway can reside behind a device performing Network Address Translation (NAT). To make sure that NAT traversal (NAT-T) functions correctly, add or update the firewall rule to allow UDP port 4500. Disable NAT-T if the customer gateway is not behind a NAT gateway.

   Use the following sample configuration files to set up an Internet key exchange SA configuration.

   - **Authentication Method:** Pre-shared Key
   - **Pre-shard Key:** LwYbARmDJmpFGAOrAbPGk2uQiWwvbmfU
   - **Authentication Algorithm:** sha1
   - **Encryption Algorithm:** aes-128-cbc
   - **Lifetime**: 28800 seconds
   - **Phase 1 Negotiation Mode:** main
   - **Perfect Forward Secrecy:** Diffie-Hellman Group 2
2. **IPsec Configuration**

   Use the following sample configuration files to configure the IPsec. Modification of the sample configuration files may be need to take advantage of additionally supported IPsec parameters for encryption, such as AES256 and other DH groups like 2, 5, 14-18, 22, 23, and 24.

- **Protocol:** esp
- **Authentication Algorithm:** hmac-sha-96
- **Encryption Algorithm:** aes-128-cbc
- **Lifetime:** 3600 seconds
- **Mode:** tunnel
- **Perfect Forward Secrecy:** Diffie-Hellman Group2

The IPsec Dead Peer Detection (DPD) is enabled on the AWS Specific Cloud endpoint. Configure the DPD on your endpoint as follows:

- **DPD interval:** 10
- **DPD Retries:** 3

The IPsec Encapsulating Security Payload (ESP) inserts additional headers to transmit the packets. These headers require additional space, which reduces the amount of space available to transmit application data. The following configuration is recommended on the customer gateway to limit the impact of this behavior:

- **TCP MSS Adjustment:** 1379 bytes
- **Clear Don't fragment Bit:** enabled
- **Fragmentation:** Before encryption

3. **Tunnel Interface Configuration**

   Configure the customer gateway with a tunnel interface that associates with the IPsec tunnel. All traffic transmitted to the tunnel interface is encrypted and transmitted to the virtual private gateway.

   The customer gate and the virtual private gateway each have two addresses that relate to this IPsec tunnel. Each one contains an outside address, where the encrypted traffic is exchanged. Both gateways also contain an inside address associated with the tunnel interface. The customer gateway outside IP address is provided upon creation of the customer gateway. To change the IP address of the customer gateway, create a new customer gateway. The customer gateway inside IP address must be configured on the interface tunnel.

   Outside IP Addresses:

   - **Customer Gateway:** 52.165.228.195
   - **Virtual Private Gateway:** 52.53.75.160

     The customer gateway IP address is the IP address of the firewall that the CloudEOS and vEOS instance in the DC with NAT behind.

     The virtual private gateway IP address is the external IP address of the AWS Specific Cloud.

   Inside IP Addresses

   - **Customer Gateway:** 169.254.11.162/30
   - **Virtual Private Gateway:** 169.254.11.161/30

   The virtual private gateway IP address is the tunnel IP address of the AWS Specific Cloud.

4. **Static Routing Configuration**

   The router traffic between the internal network and the VPC an AWS Specific Cloud, add a static router to the CloudEOS and vEOS Router.

   Next Hop: 169.254.11.162

   Any subnet that requires a route to DC must have a route pointing to the AWS Specific Cloud tunnel IP address.

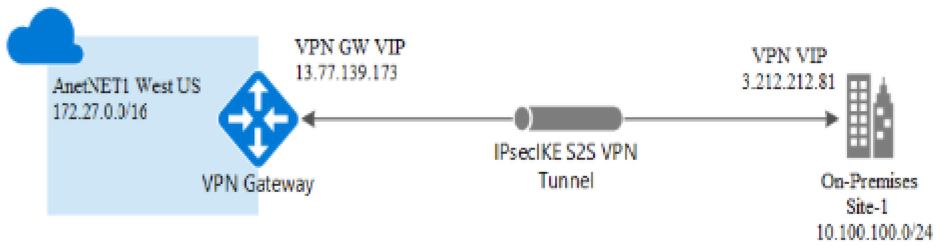
   For traffic destined to the Internet Network, add static routes on the VGW.

11.5 CloudEOS IPsec Connectivity to Azure Virtual Network Gateway

This document describes how to establish IPsec connection between CloudEOS router and Azure Virtual Network Gateway. This document also documents how to establish a BGP connection over the IPsec tunnel.

11.5.1 Creating an IPsec Azure Virtual Network Gateway

The following topology is for IPsec Azure Virtual Network Gateway.



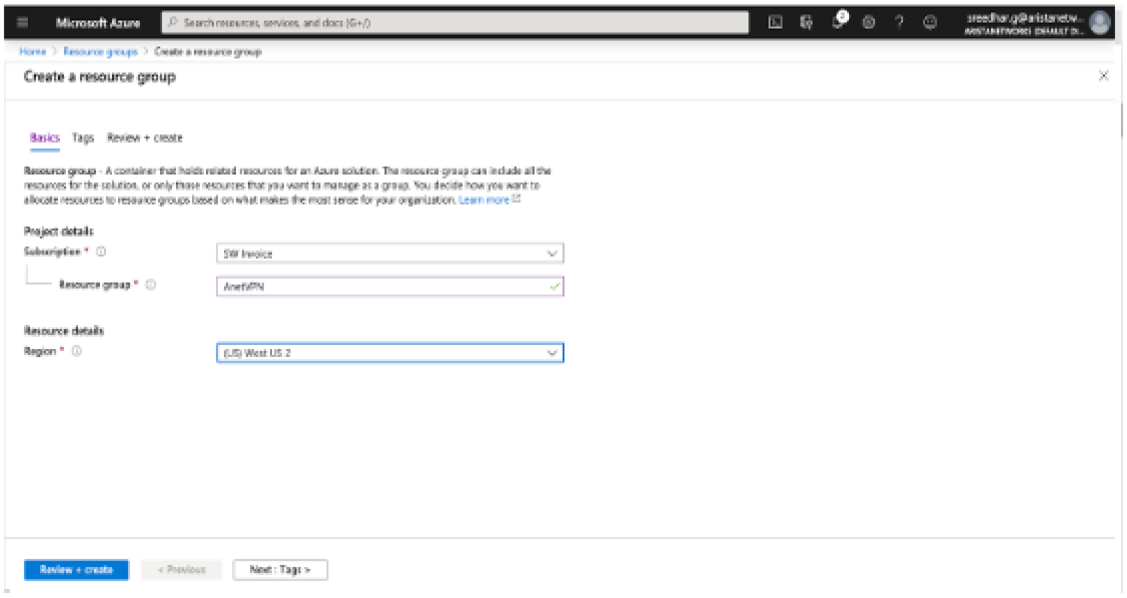
The following steps are to create an IPsec Azure Virtual Network Gateway.

1. Create a Resource Group.
2. Create the Virtual Network.
3. Create Virtual Network Gateway.
4. Configure Local Network Gateway.
5. Create Site-to-site Connections.

For more information on creating an IPsec Azure Virtual Network Gateway, refer to:https://docs.microsoft.com/en-us/azure/vpn-gateway/vpn-gateway-howto-site-to-site-resource-manager-portal

11.5.1.1 Creating a Resource Group

- Create a new resource group if not already created, under this group all other resources such as Virtual Network Gateway, Virtual Networks and other resources are created. For example, here a resource group **AnetVPN** is created.


Microsoft Azure
Search resources, services, and docs (G+/)
Home > Resource groups > Create a resource group
Basics
Tags
Review + create
Review + create
< Previous
Next : Tags >

Create a resource group

Resource group - A container that holds related resources for an Azure solution. The resource group can include all the resources for the solution, or only those resources that you want to manage as a group. You decide how you want to allocate resources to resource groups based on what makes the most sense for your organization. Learn more

Project details

Subscription *: SW Invoice

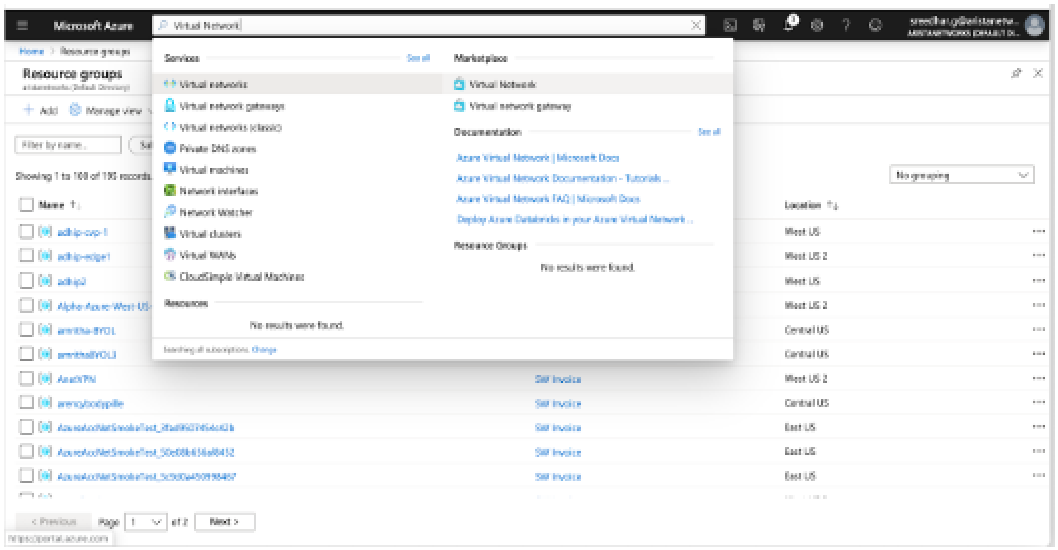
Resource group *: AnetVPN

Resource details

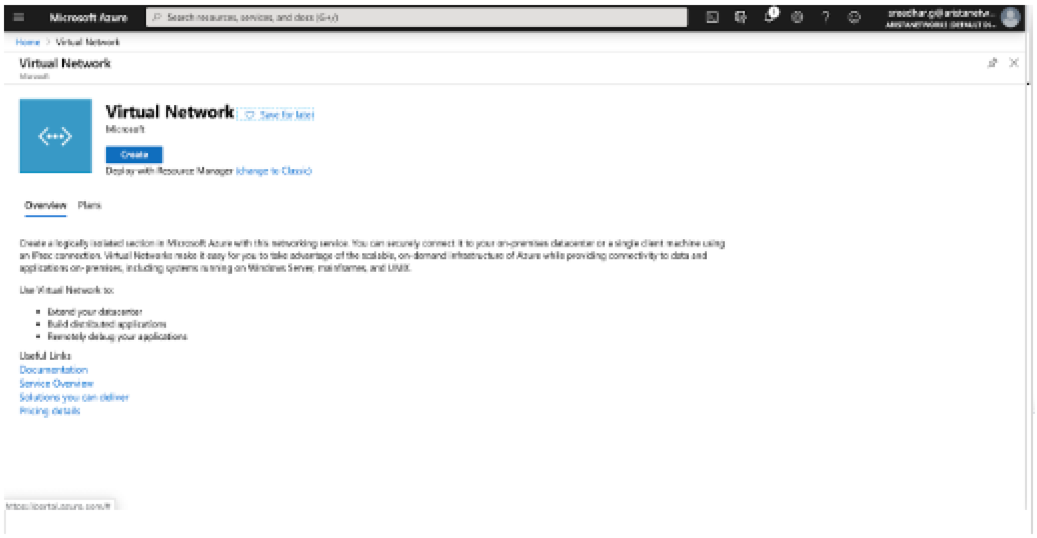
Region *: (US) West US 2

11.5.1.2 Creating a Virtual Network

1. A Virtual Network is created in the Azure Cloud, which is reached through the Azure Virtual Network Gateway. For example, a virtual network **AnetNet1**, with an IP address space 172.27.0.0/16 is created. A subnet AnetSubnet1(172.27.1.0/24) is also created, which is used as the subnet for Virtual Network Gateway.



2. Click on the **Create** button.



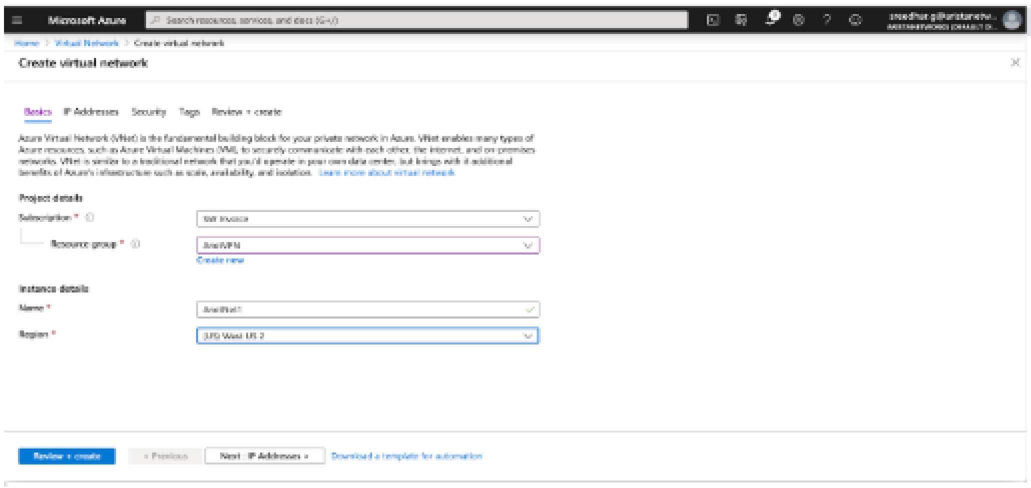
Create a logically isolated section in Microsoft Azure with this networking service. You can securely connect it to your on-premises datacenter or a single client machine using an IPsec connection. Virtual Networks make it easy for you to take advantage of the scalable, on-demand infrastructure of Azure while providing connectivity to data and applications on-premises, including systems running on Windows Server, mainframes, and UNIX.

Use Virtual Network to:

- Extend your datacenter
- Build distributed applications
- Remotely debug your applications

Useful Links
Documentation
Service Overview
Solutions you can deliver
Pricing details

3. Fill-in the mandatory fields in the Project Details section.



Create virtual network

Basics | IP Addresses | Security | Tags | Review + create

Azure Virtual Network (VNet) is the fundamental building block for your private network in Azure. VNet enables many types of Azure resources, such as Azure Virtual Machines (VM), to securely communicate with each other, the internet, and on-premises networks. VNet is similar to a traditional network that you'd operate in your own data center, but brings with it additional benefits of Azure's infrastructure such as scale, availability, and isolation. Learn more about virtual network

Project details

Subscription *: SW Invoice

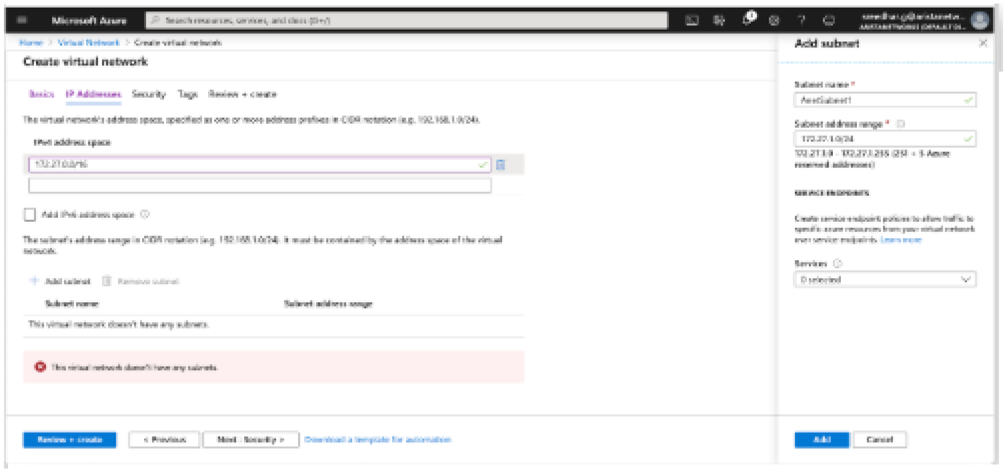
Resource group *: AristVPN

Instance details

Name *: AristNet1

Region *: (US) West US 2

4. Fill-in the IP address section with the IP address and Subnets.



Create virtual network

The virtual network's address space, specified as one or more address prefixes in CIDR notation (e.g. 192.168.1.0/24).

IPv4 address space: 172.27.0.0/16

The subnet's address range in CIDR notation (e.g. 192.168.1.0/24). It must be contained by the address space of the virtual network.

This virtual network doesn't have any subnets.

Add subnet

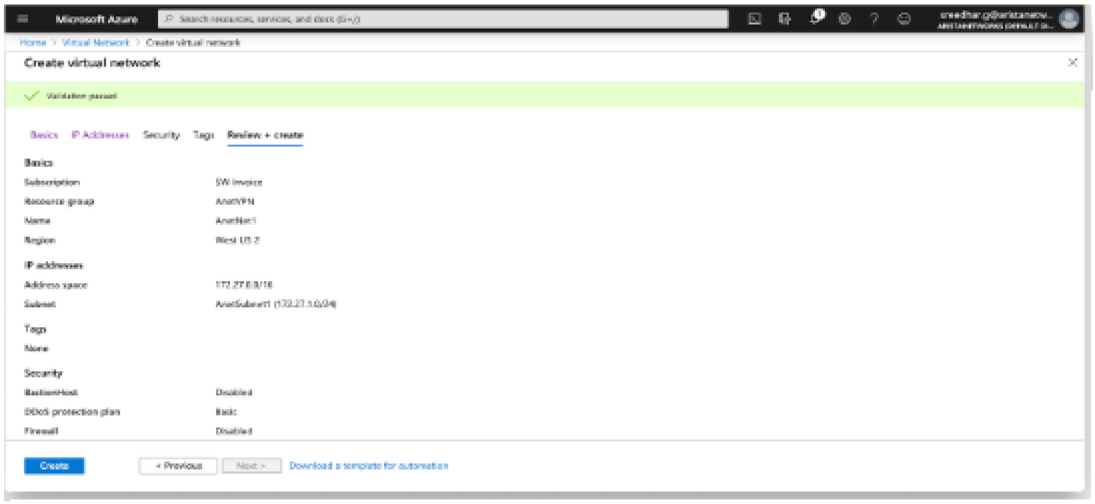
Subnet name *: AristSubnet1

Subnet address range *: 172.27.1.0/24

172.27.1.0 - 172.27.1.255 (251 + 5 Azure reserved addresses)

Services: 0 selected

5. Click on the **Review+create** tab to validate the deployment.

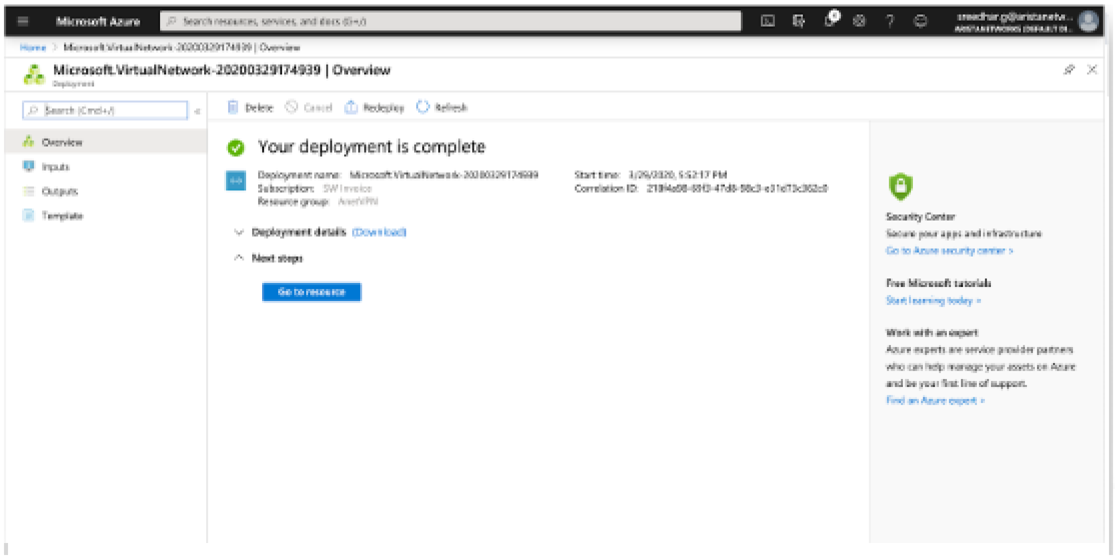


Create virtual network

Validation passed

Basics	
Subscription	SW Invoice
Resource group	AristVPN
Name	AristNet1
Region	West US 2
IP addresses	
Address space	172.27.0.0/16
Subnet	AristSubnet1 (172.27.1.0/24)
Tags	
None	
Security	
BastionHost	Disabled
DDoS protection plan	Basic
Firewall	Disabled

6. Finally if the deployment passes the validation, you see this screen.



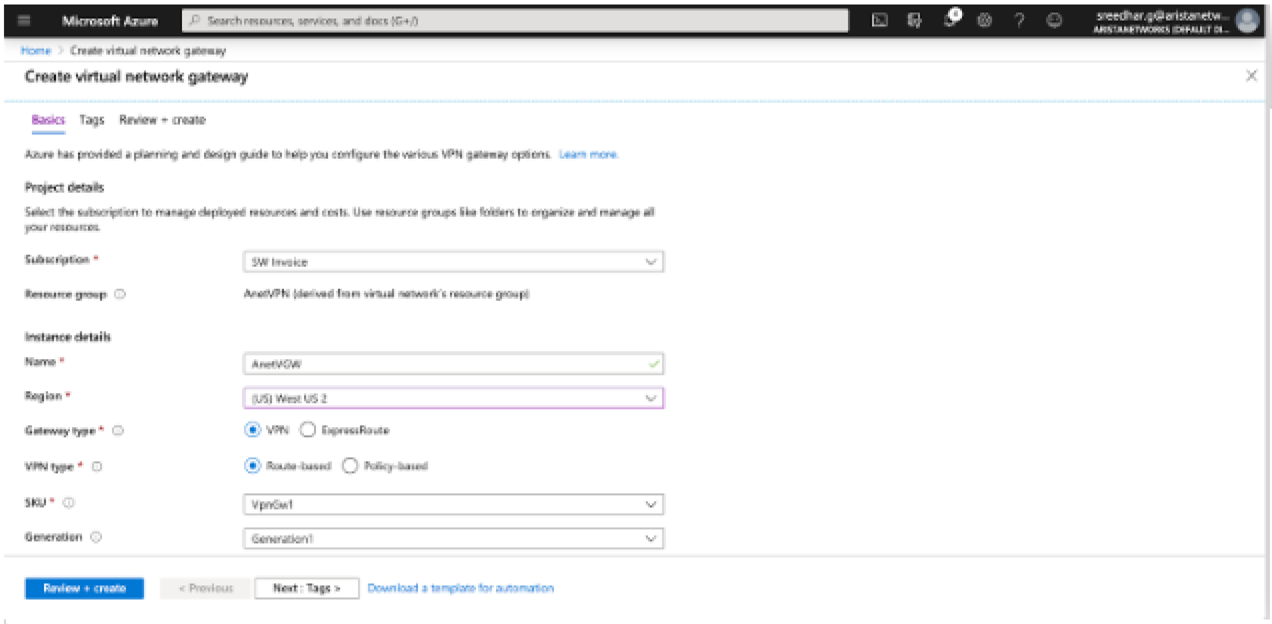
Microsoft.VirtualNetwork-20200329174939 | Overview

Your deployment is complete

Deployment name: Microsoft.VirtualNetwork-20200329174939
Subscription: SW Invoice
Resource group: AnetVPN
Start time: 3/29/2020, 5:52:17 PM
Correlation ID: 2184a08-8813-47d8-98c3-e31e73c362c0

11.5.1.3 Creating an Virtual Network Gateway

1. After creating the virtual network, a virtual network gateway(AnetVGW) is created. The Virtual Network Gateway must have a public IP address. By default BGP is disabled on the Virtual Network Gateway. In this example below the BGP enabled to demonstrate the BGP session over the IPsec connection.



Create virtual network gateway

Azure has provided a planning and design guide to help you configure the various VPN gateway options. Learn more.

Project details

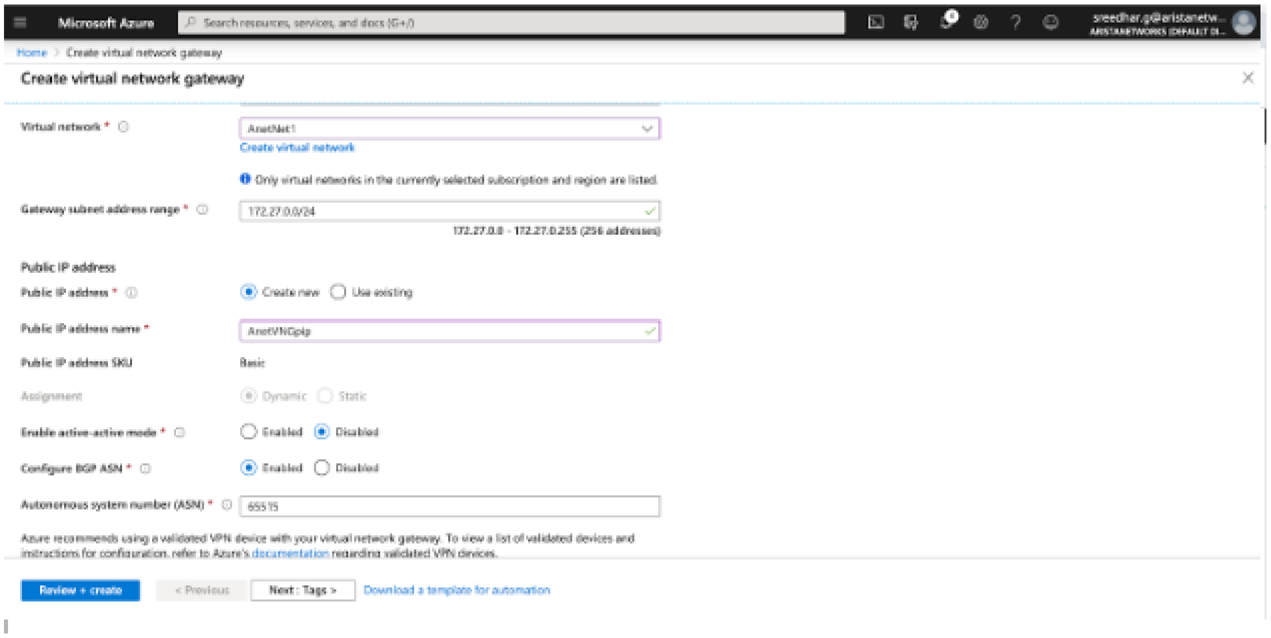
Select the subscription to manage deployed resources and costs. Use resource groups like folders to organize and manage all your resources.

Subscription *	SW Invoice
Resource group	AnetVPN (derived from virtual network's resource group)

Instance details

Name *	AnetVGW
Region *	(US) West US 2
Gateway type *	[✓ VPN] ExpressRoute
VPN type *	[✓ Route-based] Policy-based
SKU *	VpnGw1
Generation	Generation1

2. Provide the public IP address name.



Create virtual network gateway

Virtual network * : AnetNet1

Create virtual network

Only virtual networks in the currently selected subscription and region are listed.

Gateway subnet address range * : 172.27.0.0/24

172.27.0.0 - 172.27.0.255 (256 addresses)

Public IP address

Public IP address * : [✓ Create new] Use existing

Public IP address name * : AnetVNGpip

Public IP address SKU : Basic

Assignment : [✓ Dynamic] Static

Enable active-active mode * : Enabled [✓ Disabled]

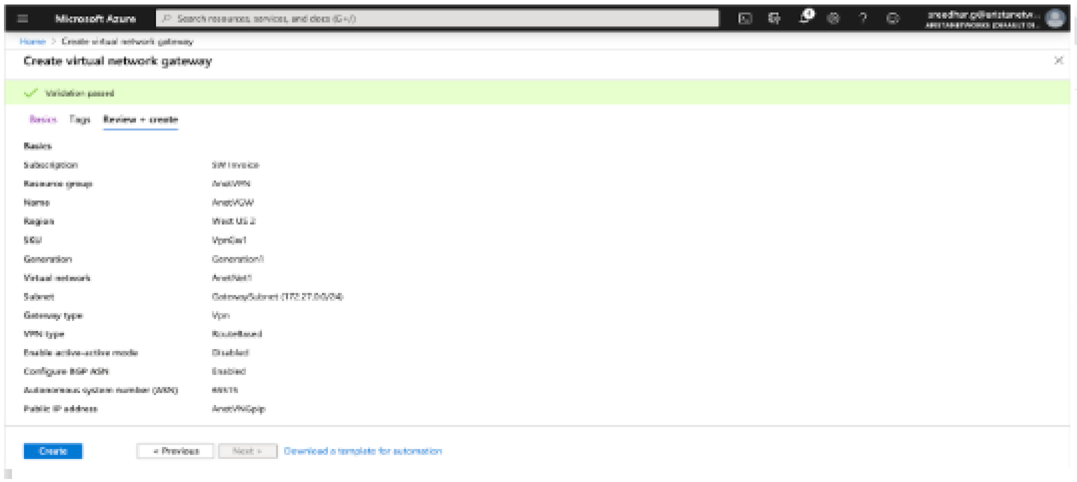
Configure BGP ASN * : [✓ Enabled] Disabled

Autonomous system number (ASN) * : 65515

Azure recommends using a validated VPN device with your virtual network gateway. To view a list of validated devices and instructions for configuration, refer to Azure's documentation regarding validated VPN devices.

Microsoft Azure
Search resources, services, and docs (G+/)
Home > Create virtual network gateway
Review + create
< Previous
Next : Tags >
Download a template for automation

3. Click on the **Review+create** tab to proceed with the deployment.



Create virtual network gateway

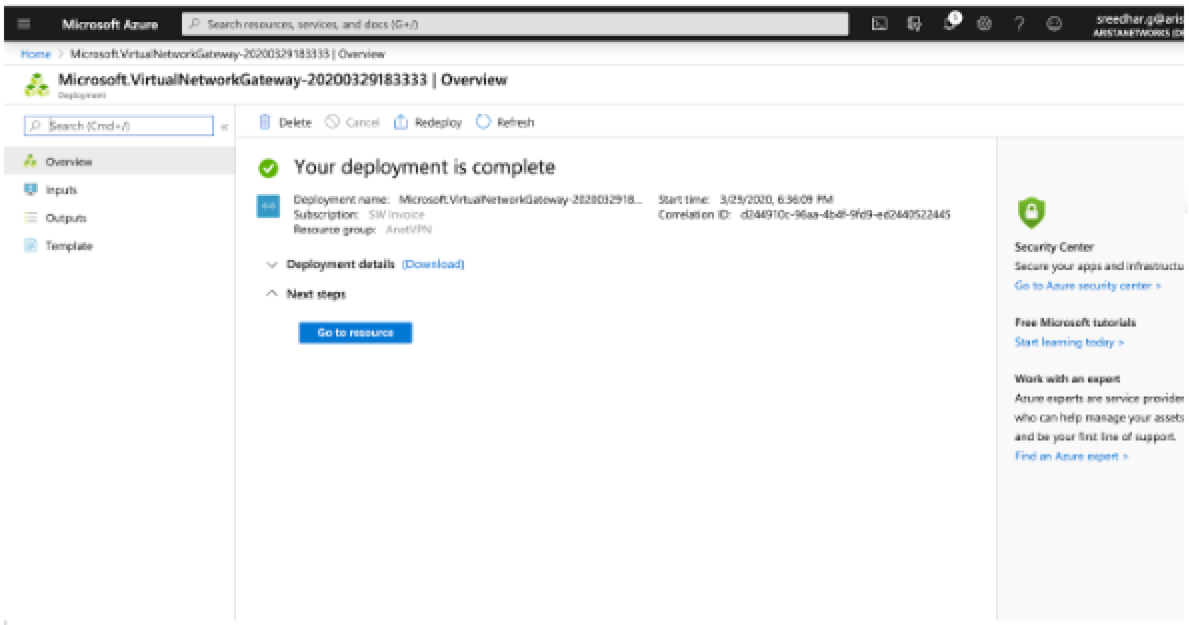
Validation passed

Basics

Subscription	SW Invoice
Resource group	AnetVPN
Name	AnetVGW
Region	West US 2
SKU	VpnGw1
Generation	Generation1
Virtual network	AnetNet1
Subnet	GatewaySubnet (172.27.0.0/24)
Gateway type	Vpn
VPN type	RouteBased
Enable active-active mode	Disabled
Configure BGP ASN	Enabled
Autonomous system number (ASN)	65515
Public IP address	AnetVNGpip

Microsoft Azure
Search resources, services, and docs (G+/)
Home > Create virtual network gateway
Basics
Tags
Review + create
Create
< Previous
Next >
Download a template for automation

4. Finally you see this page on successful deployment.

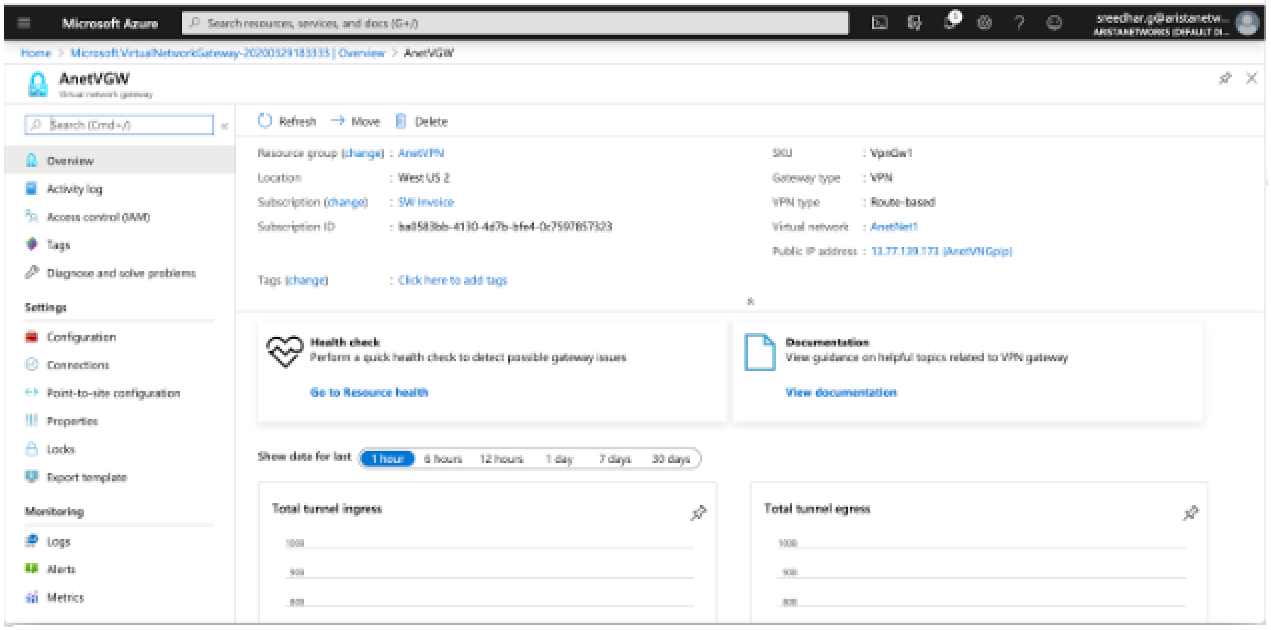


Your deployment is complete

Deployment name: Microsoft.VirtualNetworkGateway-2020032918...
Subscription: SW Invoice
Resource group: AnetVPN
Start time: 3/29/2020, 6:36:09 PM
Correlation ID: d244910c-96aa-4b4f-9fd9-ed2440522445

Microsoft Azure
Home > Microsoft.VirtualNetworkGateway-20200329183333 | Overview
Microsoft.VirtualNetworkGateway-20200329183333 | Overview
Delete
Cancel
Redeploy
Refresh
Overview
Inputs
Outputs
Template
Deployment details (Download)
Next steps
Go to resource
Security Center
Secure your apps and infrastructu
Go to Azure security center >
Free Microsoft tutorials
Start learning today >
Work with an expert
Azure experts are service provider who can help manage your assets and be your first line of support.
Find an Azure expert >

5. This page provides you information about the resources and other information related to the deployment.

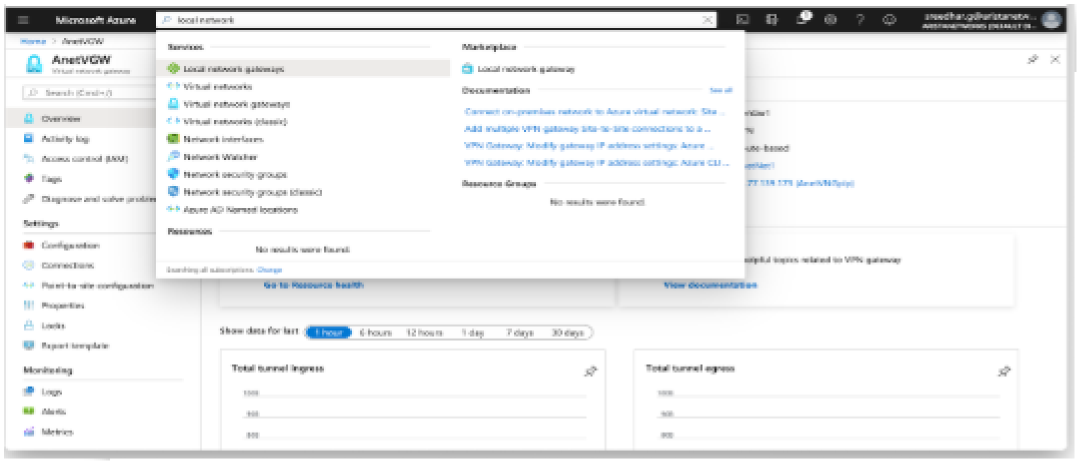


AnetVGW
Virtual network gateway

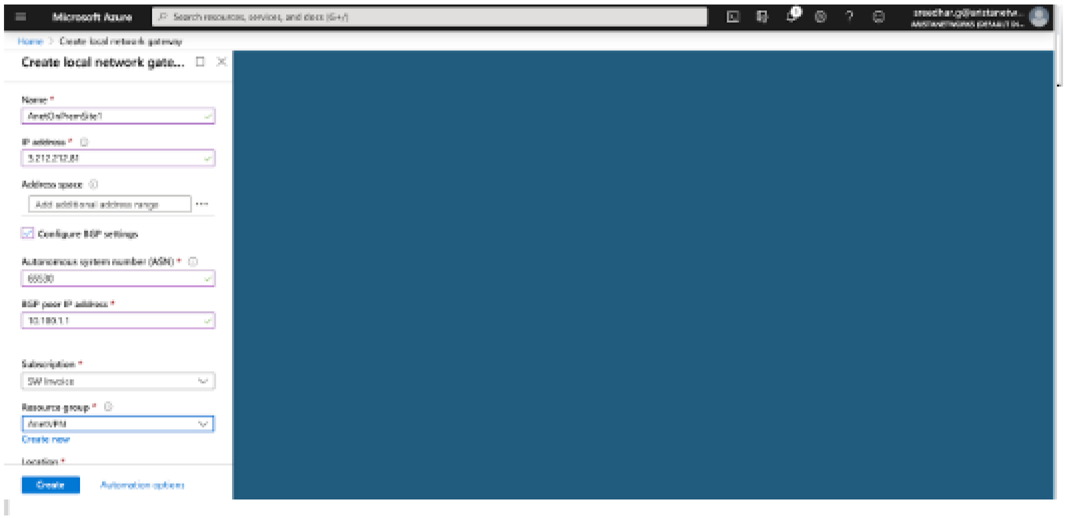
Resource group (change)	: AnetVPN	SKU	: VpnGw1
Location	: West US 2	Gateway type	: VPN
Subscription (change)	: SW Invoice	VPN type	: Route-based
Subscription ID	: ba0583bb-4130-4d7b-bfe4-0c7597857323	Virtual network	: AnetNet1
		Public IP address	: 13.77.139.173 (AnetVNGpip)
Tags (change)	: Click here to add tags		

11.5.1.4 Configuring the Local Network Gateway

1. At a customer site an on-prem router (referred as Local Network Gateway) is connected to the Azure Virtual Network Gateway. The public IP address of the On-prem router along with the BGP peering address and ASN is configured in the Local Network Gateway.



2.



Create local network gate...

Name *
AnetOnPremSite1

IP address *
3.212.212.81

Address space
Add additional address range

☑ Configure BGP settings

Autonomous system number (ASN) *
65530

BGP peer IP address *
10.180.1.1

Subscription *
SW Invoice

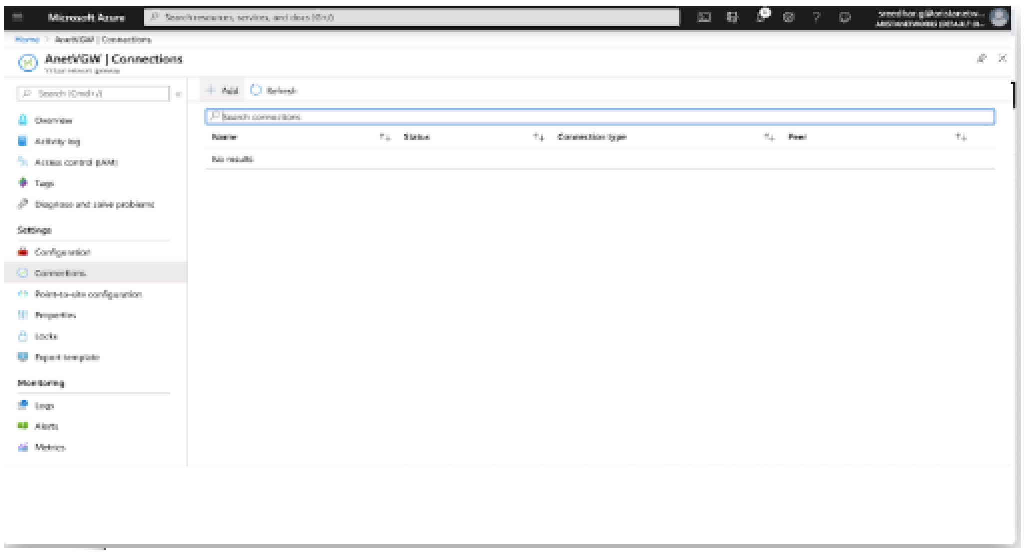
Resource group *
AnetVPN

Location *

11.5.1.5 Creating Site-to-site Connections

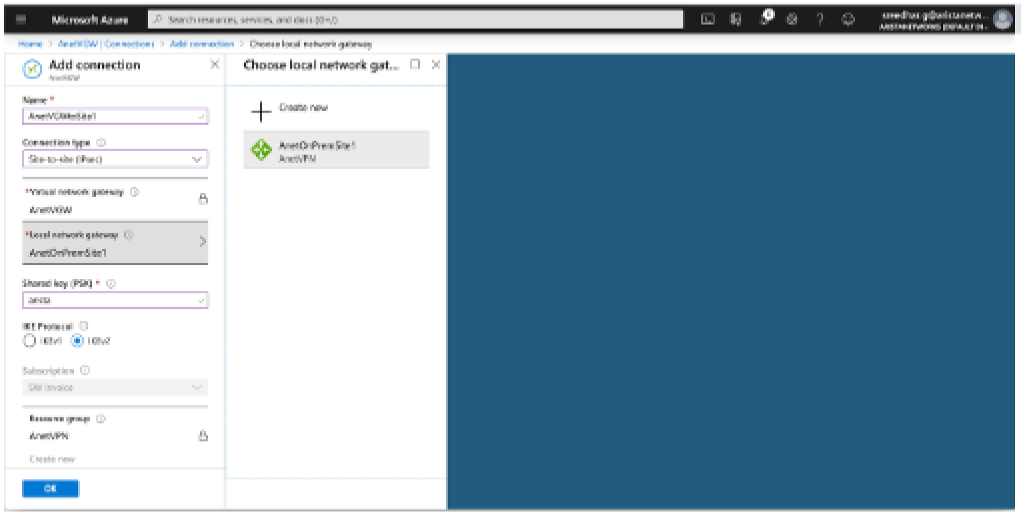
A site-to-site connection is configured to connect a Virtual Network Gateway to the Local Network Gateway. In addition to this IKE version and shared-key used for IKE authentication is configured. The rest of the cryptographic parameters cannot be configured from the Azure portal, but can be configured using Power shell. The complete list of Azure crypto suites is found here:https://docs.microsoft.com/en-us/azure/vpn-gateway/vpn-gateway-ipsecikepolicy-rm-powershell#params

1.


AnetVGW | Connections

Name | Status | Connection type | Peer

No results

2. 
Add connection

Name: AnetVGWtoSite1

Connection type: Site-to-site (IPsec)

Virtual network gateway: AnetVGW

Local network gateway: AnetOnPremSite1

Shared key (PSK): arista

IKE Protocol: IKEv1 [✓ IKEv2]

Resource group: AnetVPN

Choose local network gateway: AnetOnPremSite1 (AnetVPN)

11.5.2 Configuring CloudEOS IPsec

This section describes the CloudEOS configuration instance. The following are the default cryptographic parameters used in Azure Virtual Network Gateway configuration.

```
IKE - Ikev2/AES256/SHA256/DH-Group2
IPsec - ESP/AES256/SHA256
```

11.5.2.1 Configure the IKE Policy

```
CloudEOS(config-ipsec-ike)#ip security
CloudEOS(config-ipsec)#ike policy ikeAzure
CloudEOS(config-ipsec-ike)#encryption aes256
CloudEOS(config-ipsec-ike)#integrity sha256
CloudEOS(config-ipsec-ike)#version 2
CloudEOS(config-ipsec-ike)#dh-group 2
CloudEOS(config-ipsec-ike)#ex
CloudEOS(config-ipsec)#
```

11.5.2.2 Configure the SA Policy

```
CloudEOS(config-ipsec)#sa policy saAzure
CloudEOS(config-ipsec-sa)#esp encryption aes256
CloudEOS(config-ipsec-sa)#esp integrity sha256
CloudEOS(config-ipsec-sa)#ex
CloudEOS(config-ipsec)#
```

11.5.2.3 Configure the Profile

```
CloudEOS(config-ipsec)#profile profAzure
CloudEOS(config-ipsec-profile)#ike-policy ikeAzure
CloudEOS(config-ipsec-profile)#sa-policy saAzure
CloudEOS(config-ipsec-profile)#connection start
CloudEOS(config-ipsec-profile)#shared-key arista
CloudEOS(config-ipsec-profile)#ex
CloudEOS(config-ipsec)#
```

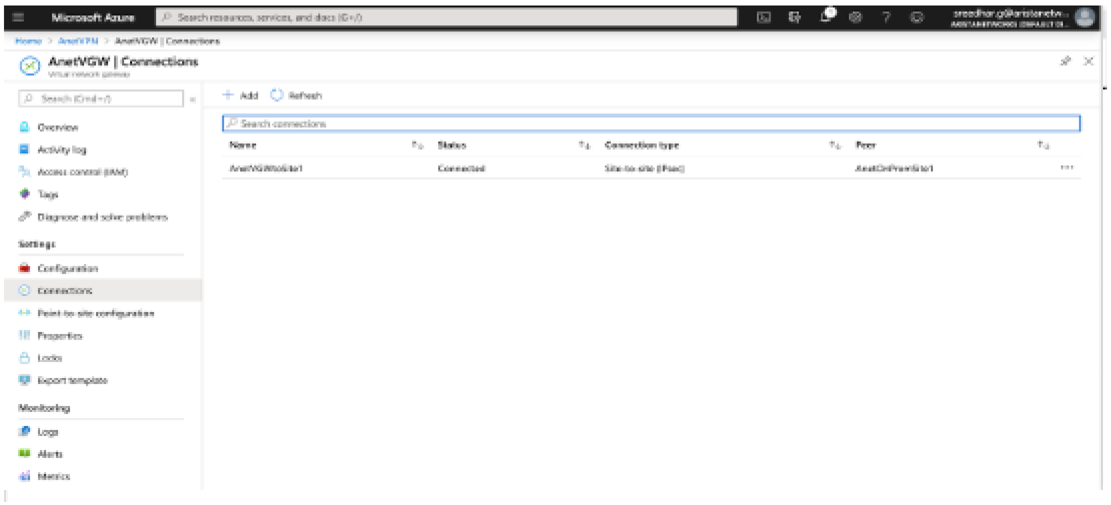
11.5.2.4 Configuring the IPsec Tunnel (VTI) Interface

```
CloudEOS(config)#interface Tunnel 1
CloudEOS(config-if-Tu1)#ip address 10.100.1.1/24
CloudEOS(config-if-Tu1)#tunnel mode ipsec
CloudEOS(config-if-Tu1)#tunnel source 3.212.212.81
CloudEOS(config-if-Tu1)#tunnel destination 13.77.139.173
CloudEOS(config-if-Tu1)#tunnel ipsec profile profAzure
! IPSec adds an overhead of up to 82 bytes. Example: A GRE tunnel with an
 MTU=1476 should be changed to 1394 when using IPSec.
CloudEOS(config-if-Tu1)#ex
CloudEOS(config)#show
```

11.5.2.5 Verifying the IPsec Connection

```
CloudEOS(config)#show ip security  connection
Tunnel      Source             Dest                   Status
 Uptime      Input            Output           Rekey Time
Tunnel1    3.212.212.81    13.77.139.173          Established       1
 second    0 bytes          0 bytes          44 minutes

           0 pkts           0 pkts
```



Name	Status	Connection type	Peer
AnetVGWtoSite1	Connected	Site-to-site (IPsec)	AnetOnPremSite1

11.5.2.6 On Prem CloudEOS behind a NAT Device

If the on-prem CloudEOS instance is behind a NAT device, configure the public IP address in local-ID under the IKE policy configuration as shown in the example below.

```
CloudEOS#ip security
   ike policy ikeAzure
      encryption aes256
      dh-group 2
      local-id 3.212.212.81
```

11.5.3 BGP over IPsec

In the BGP configuration in section Creating Virtual Network Gateway, the BGP configuration is added for AnetOnPremSite1 with ASN as 65530 and BGP peer IP address as 10.100.1.1. In this scenario, BGP address and the IP address on the tunnel interface are same, but, this is not a configuration limitations both the IP addresses can be different.

```
CloudEOS(config)#router bgp 65530
CloudEOS(config-router-bgp)#neighbor 172.27.0.254 remote-as 65515
CloudEOS(config-router-bgp)#neighbor 172.27.0.254 update-source Tunnel1
CloudEOS(config-router-bgp)#neighbor 172.27.0.254 ebgp-multihop 4
CloudEOS(config-router-bgp)#address-family ipv4
CloudEOS(config-router-bgp-af)#neighbor 172.27.0.254 activate
CloudEOS(config-router-bgp-af)#network 10.100.100.0/24
CloudEOS(config-router-bgp-af)#ex
CloudEOS(config-router-bgp)#ex
CloudEOS(config)#
```

11.5.3.1 BGP Routes Advertised to Neighbor

```
CloudEOS(config)#show ip bgp  neighbors 172.27.0.254 advertised-routes
BGP routing table information for VRF default
Router identifier 198.18.0.65, local AS number 65530
Route status codes: s - suppressed, * - valid, > - active, # - not
 installed, E - ECMP head, e - ECMP
                    S - Stale, c - Contributing to ECMP, b - backup, L -
 labeled-unicast, q - Queued for advertisement
                    % - Pending BGP convergence
Origin codes: i - IGP, e - EGP, ? - incomplete
RPKI Origin Validation codes: V - valid, I - invalid, U - unknown
```

```
AS Path Attributes: Or-ID - Originator ID, C-LST - Cluster List, LL
 Nexthop - Link Local Nexthop

          Network                Next Hop              Metric  LocPref
 Weight  Path
 * >      10.100.100.0/24        10.100.1.1            -       -       -
      65530 i
```

11.5.3.2 BGP Routes Received from the Neighbor

```
CloudEOS(config)#show ip bgp  neighbors 172.27.0.254  received-routes
BGP routing table information for VRF default
Router identifier 198.18.0.65, local AS number 65530
Route status codes: s - suppressed, * - valid, > - active, # - not
 installed, E - ECMP head, e - ECMP
                    S - Stale, c - Contributing to ECMP, b - backup, L -
 labeled-unicast
                    % - Pending BGP convergence
Origin codes: i - IGP, e - EGP, ? - incomplete
RPKI Origin Validation codes: V - valid, I - invalid, U - unknown
AS Path Attributes: Or-ID - Originator ID, C-LST - Cluster List, LL
 Nexthop - Link Local Nexthop

          Network                Next Hop              Metric  LocPref
 Weight  Path
 * >      172.27.0.0/16          172.27.0.254          -       -       -
      65515 i
CloudEOS(config)#
```

11.5.3.3 Verifying the BGP Connection

```
CloudEOS(config)#show ip bgp summ
BGP summary information for VRF default
Router identifier 198.18.0.65, local AS number 65530
Neighbor Status Codes: m - Under maintenance
  Neighbor          V  AS          MsgRcvd   MsgSent  InQ OutQ  Up/Down
 State   PfxRcd PfxAcc
  172.27.0.254     4  65515      194           214          0     0
   00:00:06  Estab   1          1
CloudEOS(config)#
```

Chapter 12

Subinterface Support for vEOS

Subinterfaces are logical L3 interfaces commonly used in the L2/L3 boundary device. They enable the division of a single ethernet interface into multiple logical L3 interfaces, based on the incoming 802.1q tag (VLAN-ID).

For a subinterface to be operational on an Ethernet or port channel interface, the parent interface must be configured as a routed port and be administratively up, and a VLAN must be configured on the subinterface. If the parent interface goes down, all subinterfaces automatically go down as well, but will come back up with the same configuration once the parent interface is up.

Note, that a port channel should not contain Ethernet interfaces with subinterfaces configured on them, and that subinterfaces cannot be members of a port channel.

Subinterfaces are named by adding a period followed by a unique subinterface number to the name of the parent interface. Note that the subinterface number has no relation to the ID of the VLAN corresponding to the subinterface.

Configuring Routing Features on a Subinterface

Once a subinterface is created, the following features can be configured on it:

- IP Routing
- VRF
- IP ACL
- Policy Map
- Basic QOS
- COS rewrite
- VRRP

Creating a Subinterface

To create a subinterface on an Ethernet or port channel interface:

Step 1: Bring up the parent interface and ensure that it is configured as a routed port.

```
switch(config)#interface Ethernet1/1
switch(config-if-Et1/1)#no switchport
switch(config-if-Et1/1)#no shutdown
```

Step 2: Configure a VLAN on the subinterface. The encapsulation dot1q vlan command is also used for VLAN translation, but in this context it associates a VLAN with the subinterface.

```
switch(config-if-Et1/1)#interface Ethernet1/1.1
switch(config-if-Et1/1.1)#encapsulation dot1q vlan 100
```

Step 3: Configure an IP address on the subinterface (optional) and ensure that it is up.

```
switch(config-if-Et1/1)#ip address 10.0.0.1/24
switch(config-if-Et1/1)#no shutdown
switch(config-if-Et1/1)#
```

Chapter 13

Port Mirroring with Greenspan

The Mirroring features allows the mirroring of source port packets in Rx, Tx and both directions to a local destination port. Mirroring with Greenspan feature allows mirroring of source packets in Rx direction on to GRE tunnel.

Configuring Mirroring with Greenspan and Port

- These commands configure a mirroring session with destination for GRE tunnel interface and regular port.

Syntax

```
// Source port
switch(config)#monitor session <name> source <interface> [ rx | tx |
 both ] [ ip access-group <access-list-name>

// GREENSPAN destination
switch(config)#monitor session <name> destination tunnel mode gre source
 <ipAddress> destination <ipAddress> ttl <ttlValue> dscp <dscpValue>

// Port destination
switch(config)#monitor session <name> destination <interface>
```

Example

```
switch(config)#monitor session r1 source Ethernet2 rx ip access-group a3
switch(config)#monitor session r1 destination tunnel mode gre source
 75.75.75.75 destination 100.1.0.2
```

Displaying Mirroring with Greenspan and Port Information

These two commands display the monitor and platform related mirroring information.

- show monitor session
- show platform sfe mirroring

Example

- The **show monitor session** command displays the session information.

```
switch#show monitor session

Session r1
------------------------

Programmed in HW: Yes

Source Ports:

  Rx Only:     Et2(IPv4 ACL: a3)

Destination Ports:
```

```
            status          source            dest            TTL   DSCP
       proto        VRF fwd-drop
    Gre1 :  active          75.75.75.75       100.1.0.2 128     0
 0x88be     default           no
      next hop interfaces: Et5
```

- The **show platform sfe mirroring** command displays the platform specific SFE mirroring information.

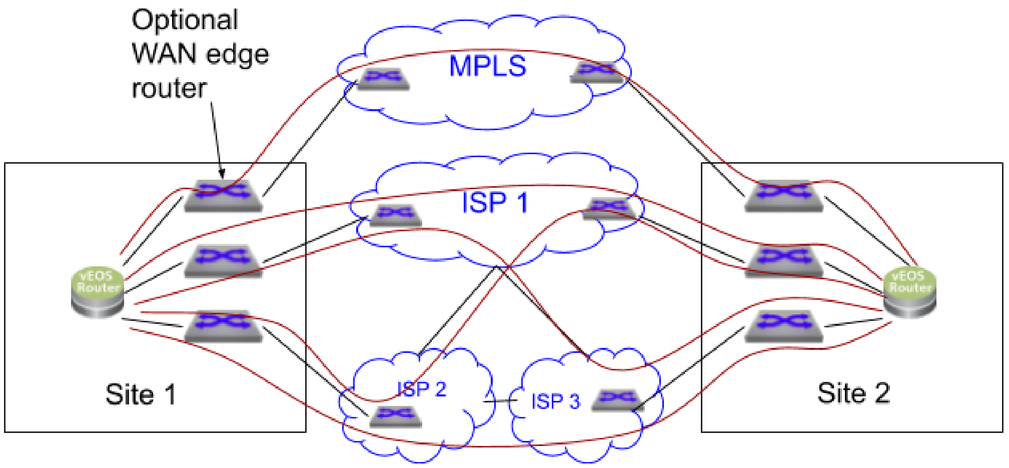
```
switch#show platform sfe mirroring
Session: r1, mirrorGroup: 3, globalId: 0
 srcIntf: Ethernet2, direction: Rx
 tgtIntf: Gre1
  encapType: Gre
 Sfe module: Mir_r1_Rx_et2
  Sfe srcIntf: Ethernet2, direction: Rx, gate: 0
  Sfe tgtIntf: Ethernet5, gate: 1
  Copy Success: 47523752, Fail: 0
  Forward Success: 53109025, Fail: 0
```

Chapter 14

Dynamic Path Selection

The CloudEOS and vEOS supports the Dynamic Path Selection that selects the path for the traffic to optimize application performance in the enterprise deployments.

The enterprise network sites like Data centers, Branches, Public Cloud (AWS VPC, Azure VNet, and others) are connected through multiple SPs (MPLS, Internet, LTE). Enterprises deploy edge routers to connect these sites over the SP WAN networks and in some cases building GRE or IPsec tunnels between sites. For high availability reasons, there are at least two WAN networks or paths available between sites.

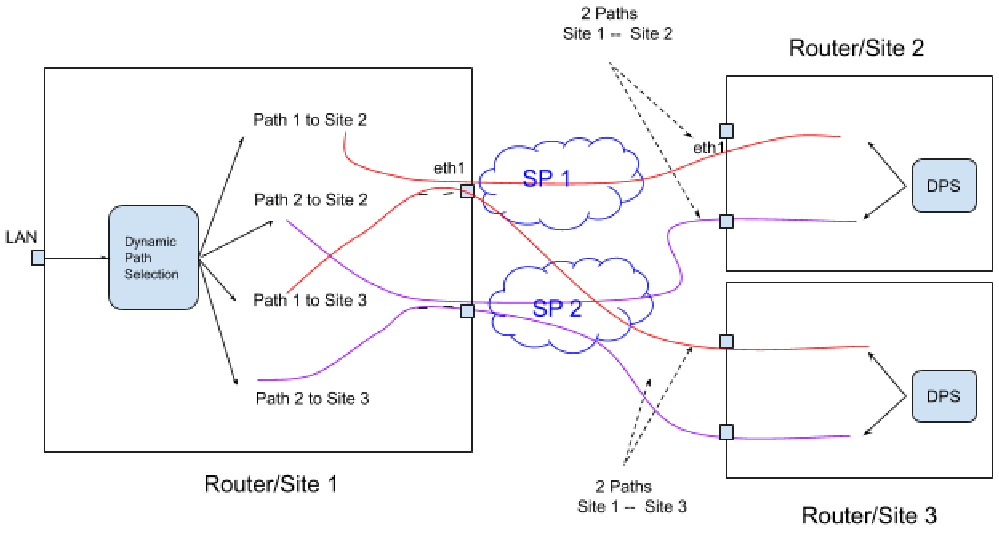


In the above example there are 5 paths, 1 MPLS path and are four paths through ISPs: ISP1, ISP1-ISP3, ISP2-ISP3, ISP2-ISP1. Different ISP have different costs, bandwidth, WAN characteristics, SLAs, and so on. This is ideal for users wanting to use various SPs in a cost effective manner without sacrificing application performance. The traditional enterprises use MPLS VPNs which provides a very good WAN characteristics such as (latency, etc), but, at a very high costs. Internet has been gaining adoption as an alternative WAN to MPLS that offers much higher bandwidth at lower costs. Also, MPLS VPNs are not available in all geographies. While ISPs are more readily available and at a lower cost, however, maintaining application performance for traffic across sites is a big problem because ISPs don't offer a good SLAs. The traditional routing solutions do not address the requirements to optimize routing across WAN SP networks.

This chapter includes the following sections:

14.1 Overview

This section describes the functional overview of the Dynamic Path Selection feature. The below figure shows three routers in different sites interconnected through two SPs. In this example, Site 1 is a hub site and is connected to both Site 2 and Site 3. There are two paths between site1 to site 2 and two paths from site 1 to site 3.



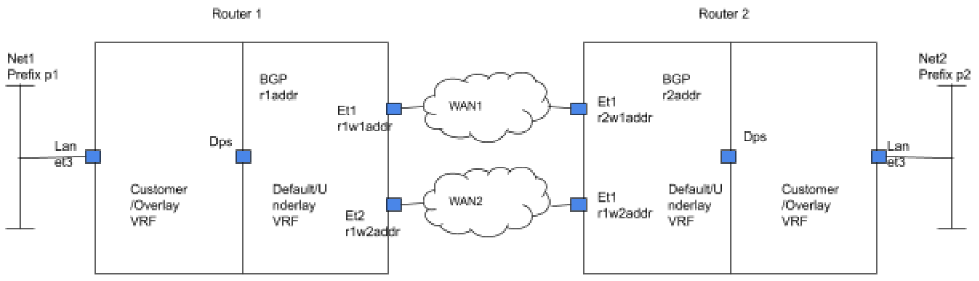
14.1.1 Path Definition

A “path” represents a pair of interfaces, a source interface and a destination interface through which traffic can flow from site to site. For example, eth1/router1 -- eth1/router2 is a path. Note, that there could be many paths through the same egress interface. The “path” does not refer to the actual network path the packet takes through the SP network. There could be multiple network paths in SP network from customer’s edge router to another edge router. Also, the network paths could change. A path is unidirectional and path characteristics is tracked in each direction.

14.1.2 Dynamic Load Balancing

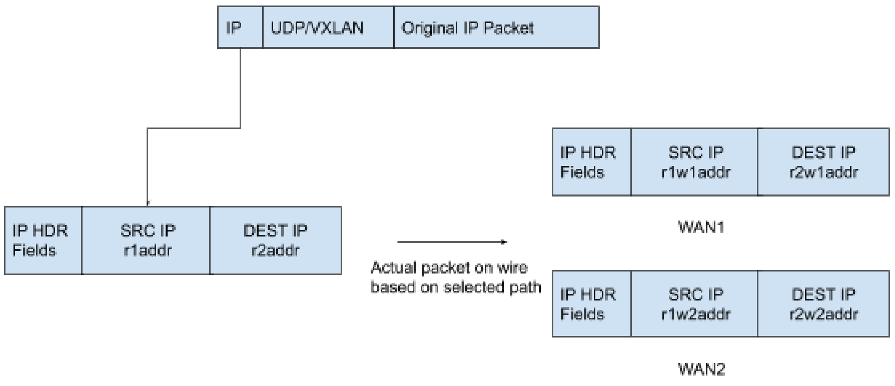
Selects the best path (destination IP and egress interface) to a destination for a given application. The algorithm has to select the best paths based on user specified priorities or constraints, and dynamically load balance flows across selected paths.

14.1.3 WAN Overlay using VXLAN



Note, that the routers are connected to two SPs in the above diagram. All customer prefixes are on the overlay network and if the VTEP IP r1addr and r2addr addresses are accessible through SP networks then the VXLAN overlay would work similar to the datacenter network. However the VTEP IP address is an internal IP address and is not routable over SP networks. While it is possible to make the VTEP IP address routable over MPS network (unlike ISP), since we want to dynamically load balance across SP networks we will not advertise the VTEP IP address over MPLS.

However, the WAN interfaces have SP routable IP address. For example, r1w1 IP address is routable on WAN1 and r1w2 IP address is routable on WAN2. The forwarding engine will replace the VTEP address on the packet based on the path selected before sending it to SP network.



Therefore for router 1:

- The router VTEP IP V1 is the nexthop for all the customer prefixes and the customer prefixes p1, and others are advertised using EVPN type 5 address family.
- VTEP IP V1 is reached through the two publically routable WAN IP addresses r1w1 address and r1w2 address.

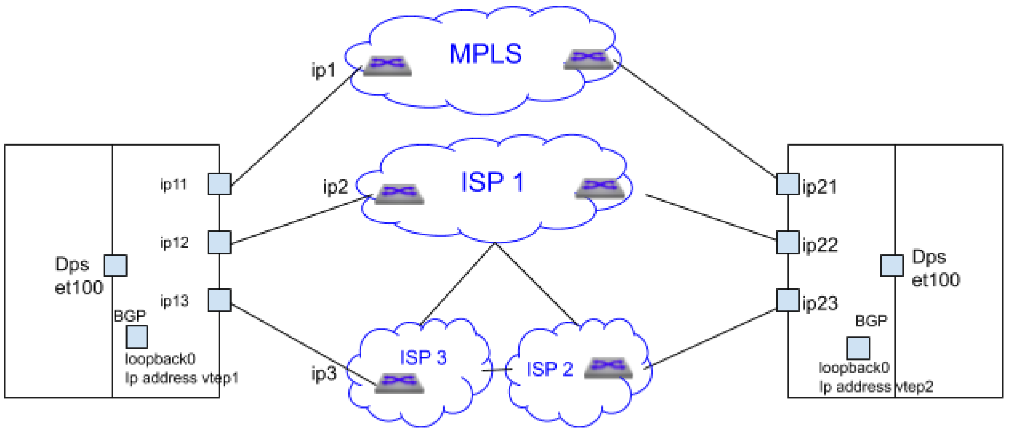
For this, the router needs to know SP routable IP addresses through which it can reach each.

14.1.4 DPS / Et100 Interface

This interface is similar to the VXLAN interface. All the inter-site WAN traffic flows through this interface. If any policies that are applied to the packet before encapsulation is applied to this interface. Currently the DPS interface is represented by et100 interface that is created by default.

Note: The et100 interface supports **TCP MSS Ceiling** for all DPS encapsulated packets. For more information on **TCP MSS Ceiling**, refer Section 28.9 TCP MSS Ceiling in the EOS user manual.

14.1.5 Peer VTEP Reachability



In the above figure there are five paths between the two sites:

- MPLS - Ip11, ip21
- Internet Ip12 - ip22
- Internet Ip12 - ip23
- Internet Ip13 - ip22

- Internet Ip13 - ip23

Currently peer VTEP reachability needs to be configured statically, but, in future this is exchanged through BGP. BGP runs on the same loopback interface used as VXLAN source VREP interface in underlay.

The router tracks if the configured paths are available using routing updates, interface state and so on, and programs the available paths for forwarding.

14.1.6 Control Plane Traffic

The BGP traffic that is going between sites will all go through DPS interface and leverage path selection feature to ensure that the BGP traffic leverages all the path selection features. Different path selection policies can be setup for different control plane traffic types as for end applications.

14.1.7 Load Balancing Algorithm

Algorithm selects the path that meets all the criteria for an application. If there are multiple paths that meet then it load balances across the available paths. If none of the paths meet the criteria is it picks the one with the lowest loss rate.

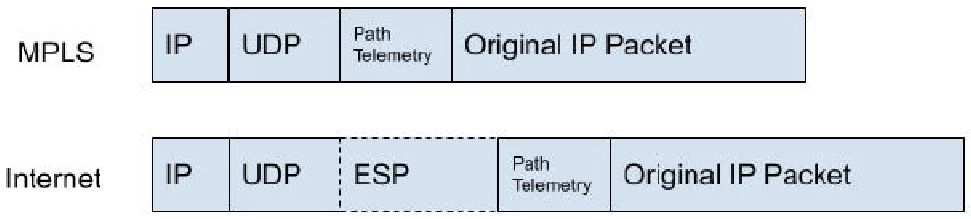
The selected path for a given flow is then stored in flow cache. The chosen path is not reevaluated for constraints. Packets from that flow will take the same path even if the path characteristics no longer meet the user specified criteria.

Events that trigger the re-selection of path for a flow are as shown below:

- When path is no longer active.
- When the flow is remapped to a different application.
- When user has changed the constraints or priority such that the path is no longer valid for this flow.

14.1.8 Path Telemetry

Path Telemetry feature provides the ability to determine WAN path state and measure its characteristics including latency (one way delay), jitter, packet loss rate and throughput.



The outer IP header uses the WAN IP addresses on local and peer WAN interfaces of the path. IP header is followed by a UDP header where the destination port is set to be 4793 by default or to be the port number configured by user in CLI. When IPsec is enabled, destination port is set to be 4500. A path telemetry header is inserted in between of UDP/ESP header and the inner IP packet for path characteristics measurement purpose.

Path State Determination

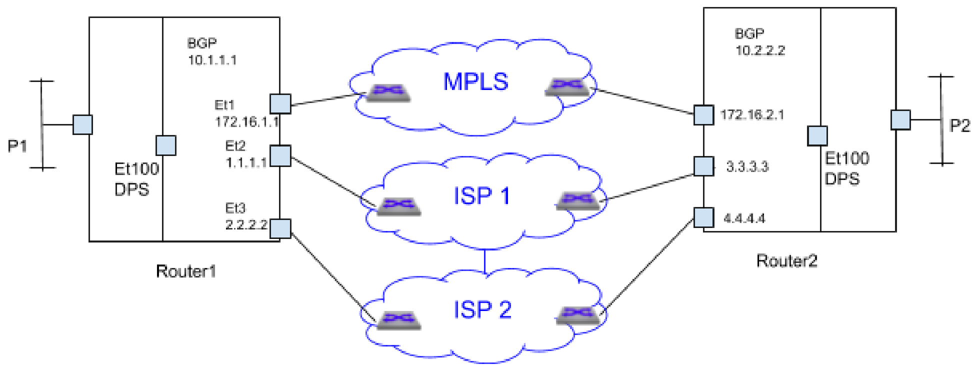
Path telemetry uses keepalive and feedback packets to determine path state. It sends out keepalive periodically (once per second) and if it receives peer's feedback packet, the path is considered as active and its characteristics is measured. Accordingly, if feedback packet is not received within a certain period of time (for 5 keepalive we sent), the path is considered as inactive and is not used for path selection.

14.2 Configuration

This section describes the commands to configure and verify the Dynamic Path Selection feature.

14.2.1 Defining Paths

A “path” represents a pair of interfaces (or their IP addresses), a source interface and a destination interface through which traffic can flow from site to site.



For example, in the above figure there are two paths from Router1 to Router2

1. MPLS path - 172.16.1.1 -- 172.16.2.1
2. 4 Internet paths

- 1.1.1.1 -- 3.3.3.3
- 1.1.1.1 -- 4.4.4.4
- 2.2.2.2 -- 3.3.3.3
- 2.2.2.2 -- 4.4.4.4

However, some of the paths are crossing ISPs, for example, 1.1.1.1 -- 4.4.4.4 is going from router1 through ISP1, ISP2 to router2. In some customer scenarios ISP2 could be an LTE SP and could be purely as a backup in case ISP1 fails. In this case the paths 1.1.1.1 -- 4.4.4.4 and 2.2.2.2 -- 3.3.3.3 should not be used.

Path-group similar to nexthop-group is used to group the paths in order to

- Restrict paths - define which paths are valid among the available paths like the LTE backup SP discussed before
- Apply specific policies to path group. Eg apply encryption for all Internet paths

Path group commands are configured under “router path-selection” as shown below. The commands are explained in the subsections.

```
router path-selection
path-group <group-name>
local interface <intf-name>
## more local interface commands
## that belong to the same path-group, eg Internet
peer static router-ip <ip-address>
ipv4 address <ip-addr1>
## more IP addresses through which the router can be reached
```

The router-IP is the same as the VTEP-IP. local is used to configure the local WAN IP address or interface part of the path-group. Peer is used to configure the remote VTEP reachability statically.

Each combination of peer and local IP address is a potential path. If routing resolves the remote IP through a local interface then that local-remote IP pair becomes a real path that is used for forwarding.

In the topology in the above figure two groups are defined.

1. mpls-group
2. Internet-group

Further if paths need to be restricted through the Internet, the Internet groups can be divided into more groups. For example, the customer can define ISP1 and ISP2-ISP3 as separate groups create 2 Internet paths instead of 4.

14.2.1.1 Creating Path-Groups under Path-Selection

Syntax

router path-selection path-group <name>

name: name of the path group

Example

```
switch(config)#router path-selection
switch(config-dynamic-path-selection)#
switch(config-dynamic-path-selection)#path-group mpls
```

14.2.1.2 Specifying Local Interfaces under Path-Group Sub-Mode

Syntax

path-group <name> local interface <intf-name>

local interface: is used to configure the local WAN interface part of the path-group. The IP addresses assigned to the WAN interface is used as WAN IP. Multiple interfaces can be specified. For example, if there are two ISP connections.

Example

In the above deployment: ether1 is part of MPLS path-group.

```
switch(config-dynamic-path-selection)#path-group mpls
switch(config-path-group-mpls)#local interface ether1

Ethernet 2 and 3 are part of Internet path-group
switch(config-dynamic-path-selection)#path-group internet
switch(config-path-group-internet)#local interface ether2
switch(config-path-group-internet)#local interface ether3
```

14.2.1.3 Specifying Remote VTEPs and their Reachability Statically

Syntax

path-group <name> peer static router-ip <ip-address> ipv4 address <ip-addr1> ipv4 address <ip-addr2> ## more IP addresses through which the router can be reached

peer static is used to configure the remote VTEP reachability statically via routable IP addresses over the SP network. The **router-IP** is the VTEP IP address. In the case of Internet, the routable IP address is a public IP address. In the case of MPLS it is Enterprise specific private IP address that the MPLS provider knows how to reach. Typically customer edge routers (CEs) are configured to exchange subnets by running eBGP to the SP's PE router.

Example

In the above deployment for the MPLS path group Router2's router IP 10.2.2.2 is reachable via Router2's MPLS IP address 172.16.2.1

```
switch(config-dynamic-path-selection)#path-group mpls
switch(config-path-group-mpls)#peer static router-ip 10.2.2.2
switch(config-peer-router-ip-10.2.2.2-mpls)#ipv4 address 172.16.2.1

For the Internet path group Router2's router IP 10.2.2.2 is reachable via
 two IP addresses only via ISP1 3.3.3.3 and another through ISP2 4.4.4.4

switch(config-dynamic-path-selection)#path-group internet
switch(config-path-group-internet)#peer static router-ip 10.2.2.2
switch(config-peer-router-ip-10.2.2.2-internet)#ipv4 address 3.3.3.3
switch(config-peer-router-ip-10.2.2.2-internet)#ipv4 address 4.4.4.4
```

It is important to note that once local and remote IP addresses are specified for a path-group then all combinations of local and remote IP address is a potential path for load balancing.

Example

Consider the following configuration that corresponds to the topology in the above figure :

```
switch(config)#router path-selection
switch(config-dynamic-path-selection)#path-group mpls
switch(config-path-group-mpls)#local interface et1
switch(config-path-group-mpls)#peer static router-ip 10.2.2.2
switch(config-peer-router-ip-10.2.2.2-mpls)#ipv4 address 172.16.2.1
switch(config-peer-router-ip-10.2.2.2-mpls)#path-group internet
switch(config-path-group-internet)#local interface et2
switch(config-path-group-internet)#local interface et3
switch(config-path-group-internet)#peer static router-ip 10.2.2.2
switch(config-peer-router-ip-10.2.2.2-internet)#ipv4 address 3.3.3.3
switch(config-peer-router-ip-10.2.2.2-internet)#ipv4 address 4.4.4.4

The paths defined are
MPLS path - 172.16.1.1 -- 172.16.2.1
4 Internet paths
1.1.1.1 -- 3.3.3.3
1.1.1.1 -- 4.4.4.4
2.2.2.2 -- 3.3.3.3
2.2.2.2 -- 4.4.4.4

However if ISP2 is a LTE and the customer does not want paths to cross
 over from ISP1 to LTE then the configuration should be

switch(config)#router path-selection
switch(config-dynamic-path-selection)#path-group mpls
switch(config-path-group-mpls)#local interface et1
switch(config-path-group-mpls)#peer static router-ip 10.2.2.2
switch(config-peer-router-ip-10.2.2.2-mpls)#ipv4 address 172.16.2.1
switch(config-peer-router-ip-10.2.2.2-mpls)#path-group internet
switch(config-path-group-internet)#local interface et2
switch(config-path-group-internet)#peer static router-ip 10.2.2.2
switch(config-peer-router-ip-10.2.2.2-internet)#ipv4 address 3.3.3.3
switch(config-peer-router-ip-10.2.2.2-internet)#path-group lte
switch(config-path-group-lte)#local interface et3
switch(config-path-group-lte)#peer static router-ip 10.2.2.2
switch(config-peer-router-ip-10.2.2.2-lte)#ipv4 address 4.4.4.4

In the above case the paths are
MPLS path - 172.16.1.1 -- 172.16.2.1
Internet path 1.1.1.1 -- 3.3.3.3
```

```
LTE path 2.2.2.2 -- 4.4.4.4.
```

14.2.2 Underlay DPS Configuration

For DPS paths and EVPN routes to be exchanged we need to configure VXLAN with a private IP address of a loopback interface and DPS interface should be configured as L3 interface. Please note that the configuration for DPS interface has to be split up and configured under two interfaces VXLAN1 and et100. In future they are replaced with one single DPS interface.

14.2.2.1 DPS Interface Configuration

For the DPS interface add any private IP address to make it an Layer 3 interface. However, the assigned IP address is not used for routing.

Syntax

interface Ethernet100 no switchport ip address 11.0.0.1/24

Example

```
switch(config)#interface ethernet 100
switch(config-if-Et100)#no switchport
switch(config-if-Et100)#ip address 11.0.0.1/24
```

14.2.2.2 VXLAN Configuration

In the example below 1.1.1.1 is a private IP which is configured in loopback 0 interface is used as VXLAN source interface.

Example

```
switch(config)#interface loopback 0
switch(config-if-Lo0)#ip address 1.1.1.1/32
switch(config-if-Lo0)#interface vxlan1
switch(config-if-Vx1)#vxlan source-interface loopback 0
switch(config-if-Vx1)#vxlan udp-port 4789
switch(config-if-Vx1)#vxlan vrf vrf1 vni 100
```

BGP runs on the same loopback IP as VXLAN source interface IP. In the above example BGP runs on ips 1.1.1.1, 2.2.2.2, and 3.3.3.3 on each peer.

For underlay routing add the remote peer routes via DPS interface and statically add an ARP entry for remote peer. In future versions of EOS the underlay routing also be handled by BGP.

Example

```
switch(config)#ip route 2.2.2.2/32 ethernet 100
switch(config)#ip route 3.3.3.3/32 ethernet 100
switch(config)#arp 2.2.2.2 00:00:33:02:00:00 arpa
switch(config)#arp 3.3.3.3 00:00:33:03:00:00 arpa
The above configuration makes the peers reachable via DPS.
```

14.2.3 Applying Policies for Path Groups

The policies for the path groups are applied on all the paths in the group. The following policy is supported:

14.2.3.1 Encrypting Path-Group

Applying IPsec to the group will enable encryption on all the paths in the group as per the applied IPsec profile. This policy is used to encrypt all Internet paths. This configuration simplifies IPsec configuration as the customer does not have to specify what traffic to encrypt.

Syntax

path-group <name> ipsec profile <ipsec-profile-name> Applying IPsec profile will cause all the paths in the path group to be encrypted based on the algorithms and authentication mechanisms as per the profile.

14.2.4 Configuring Load Balancing Profile

Load balancing policy is configured under **router path-selection** as shown.

Syntax

router path-selection load-balance policy <name> latency <milliseconds> jitter <milliseconds> loss-rate <0.00-100.00 percentage> path-group <group-name> [priority <number>] path-group <group-name> The commands are explained in the following subsections.

14.2.4.1 Specifying Path Groups to the Load Balancer

Syntax

router path-selection load-balance policy <name> path-group <group-name> path-group <group-name>

When multiple path-groups are specified flows are load balanced across all the paths in the specified path-groups.

Example

For example, configuring load balancing for best effort traffic across 1 MPLS path and 4 Internet paths.

```
switch(config)#router path-selection
switch(config-dynamic-path-selection)#load-balance policy best-effort
switch(config-load-balance-policy-best-effort)#path-group mpls
switch(config-load-balance-policy-best-effort)#path-group internet
```

14.2.4.2 Specifying Constraints for Path Selection

Syntax

router path-selection load-balance policy <name> latency <milliseconds> jitter <milliseconds> loss-rate <0.00-100.00 percentage>

Latency, jitter and loss-rate constraints can be specified for path selection. There can be more than one path that meets the constraints in which case the flows are load balanced across all the selected paths. All constraints need to be met. If none of the paths meet the constraints, then the path with the lowest loss rate is chosen as the best path.

Example

For example, configuring load balancing for voice traffic with preference for paths with latency less than 50ms, loss at 1%.

```
switch(config-path-selection)#load-balance policy voice
switch(config-load-balance-policy-voice)#path-group mpls
switch(config-load-balance-policy-voice)#path-group internet
switch(config-load-balance-policy-voice)#latency 50
switch(config-load-balance-policy-voice)#loss-rate 1
```

In this case, the traffic is load balanced across all the paths that meet the constraints. If none matches then the traffic is sent to the best path.

14.2.4.3 Specifying Preference to a Path-Group

Syntax

router path-selection load-balance policy <name> path-group <group-name> [priority <number>] path-group <group-name>

Preference can be specified for path-groups. Flows are load balanced based on path group priority. The lower the number the higher the priority is given to the path group. If not specified, default policy is 1 (highest). If multiple path groups in the same load-balance profile have same priority traffic will be load balanced among them. If no paths in a path-group are available then paths from the next lower priority is considered. Paths may not be available because of the following reasons:

1. Interface is down
2. Route is not resolved
3. Path keepalives have failed
4. Specified constraints for the load balancing policy is not met

Example

For example, configuring load balancing for voice traffic with MPLS path preference and Internet as backup.

```
switch(config-dynamic-path-selection)#load-balance policy voice
switch(config-load-balance-policy-voice)#path-group mpls
switch(config-load-balance-policy-voice)#path-group internet
```

When MPLS path is down then all the existing flows are forwarded through Internet paths. When MPLS path is up again, all the new flows are forwarded through MPLS paths.

14.2.5 Classification - Application Profiles

The existing commands in EOS are as shown below.

Syntax

application traffic recognition application ipv4 http-8080 { protocol <proto> [destination-port { <port_num> | <port-range> }] } protocol tcp destination-port 8080 protocol tcp destination-port 8000 application ipv4 app2-service protocol tcp destination-port 8001-8080

Applications is specified either with custom signatures specified using the **application** configuration as shown above or can be imported from a DPI engine. Application configuration might have to be extended to address the path-selection use case.

Syntax

Applications can be grouped and other attributes like the traffic class can be specified using **application-profile** as below.

application traffic recognition application-profile <app-xyz> application <app-name-1> application <app-name-2>

Example

Traffic-class is used for QoS in the datapath for path selection, queuing, rate limiting, and for other QoS configuration. This example is for “platinum” application profile for all critical traffic like voice.

```
switch(config)#application traffic recognition
switch(config-app-recognition)#application-profile gold
switch(config-app-profile-gold)#application voice
switch(config-app-profile-gold)#traffic-policies

“bronze” profile for best effort
switch(config-app-recognition)#application-profile bronze
switch(config-app-profile-bronze)#application best-effort
switch(config-app-profile-bronze)#traffic-policies
```

14.2.5.1 Path Selection Policy

The load balancing policy can be specified based on the application.

Syntax

router path-selection policy <dps-policy-name> <rule key> application-profile <profile-name> load-balance <load balance policy name> <rule key> application-profile <profile-name> load-balance <load balance policy name>

Sequence numbers are required since a flow can potentially match multiple application profiles. Also, we have “set load-balance” as a sub-mode so we can add other actions for “match application-profile”.

Example

```
switch(config)#router path-selection
switch(config-dynamic-path-selection)#policy dynamic
switch(config-policy-dynamic)#10 application-profile voice
switch(config-policy-rule-key-10-dynamic)#load-balance voice
switch(config-policy-rule-key-10-dynamic)#20 application-profile best
switch(config-policy-rule-key-20-dynamic)#load-balance best
```

14.2.5.2 Applying Path Selection Policy

All traffic going from site to site will go through VTI interfaces and is VXLAN encapsulated. Different classification and path selection policies are specified for each VRF. For example, the test VRF can have simple application classification and load balancing policy.

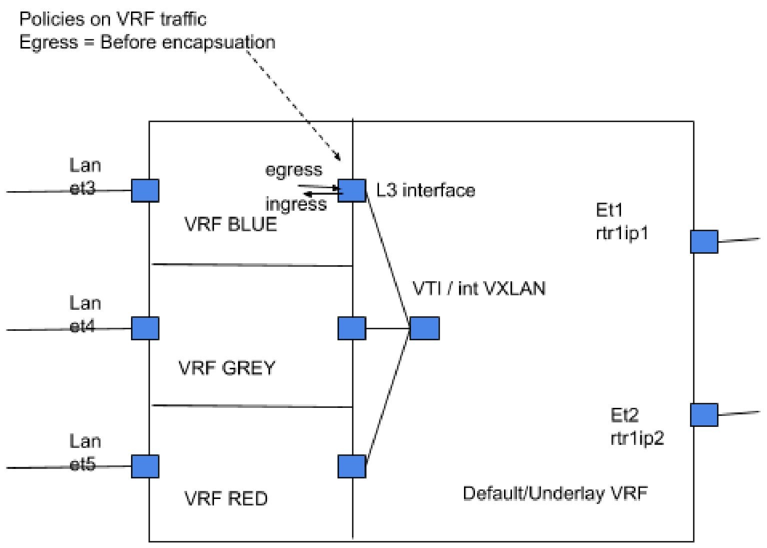
Syntax

router path-selection vrf <vrf-name> path-selection-policy <policy-name>

VRF **“all”** can be specified to apply policy on all VRFs. In case both **“all”** and **per VRF** policy is specified, only the per VRF policy is applied.

The policy (classification and load balancing) needs to be applied to the datapath once it is determined that traffic is going from site to site. This is done to avoid the classification overhead for LAN to LAN traffic. When policy is applied on a VRF it is actually applied on the egress direction on the hidden SVI interface for the VTI (VXLAN tunnel interface). If there is no VTI configured then this policy is ignored.

When policy is applied on a VRF it is actually applied on the egress direction on the hidden SVI interface for the VTI (VXLAN tunnel interface) as shown below. If there is no VTI configured then this policy is ignored.



Example

```
switch(config)#router path-selection
switch(config-dynamic-path-selection)#vrf red
switch(config-vrf-red)#path-selection-policy production
switch(config-vrf-red)#
```

14.2.6 Path Telemetry UDP Port

By default, the path telemetry protocol uses 4793 as the destination UDP port number for encapsulation purpose. The below command is used to configure the UDP port for DPS.

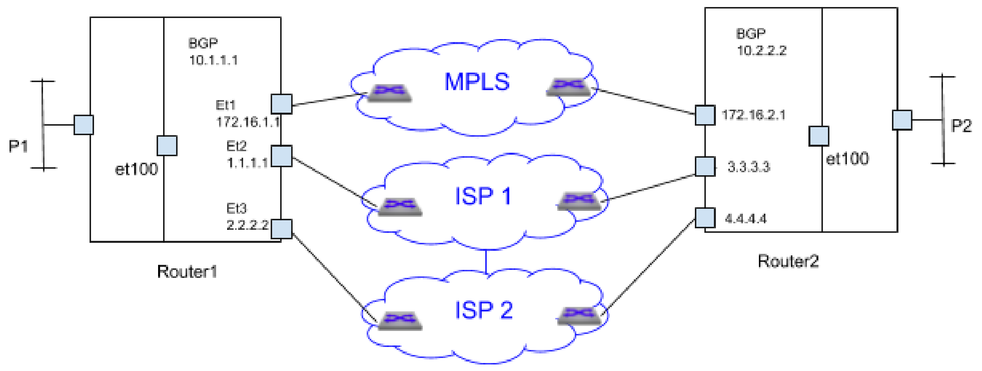
Syntax

router path-selection encapsulation path-telemetry udp port <number>

Example

```
switch(config)#router path-selection
switch(config-dynamic-path-selection)#encapsulation path-telemetry udp
 port 4794
```

14.2.7 Complete Path Selection Configuration Example



Note: That applications like Voice, Skype-Voice, SCP, FTP in the example below is defined under “application traffic recognition” but is not shown below.

Example 1

```
switch#application traffic recognition
switch(config-app-recognition)#application-profile platinum
switch(config-app-profile-platinum)#application voice
switch(config-app-profile-platinum)#application skype-voice
switch(config-app-profile-platinum)#application-profile bronze
switch(config-app-profile-bronze)#application scp
switch(config-app-profile-bronze)#application ftp
switch(config-app-profile-bronze)#router path-selection
switch(config-dynamic-path-selection)#path-group mpls
switch(config-path-group-mpls)#local interface et1
switch(config-path-group-mpls)#peer static router-ip 10.2.2.2
switch(config-peer-router-ip-10.2.2.2-mpls)#ipv4 address 172.16.2.1
switch(config-peer-router-ip-10.2.2.2-mpls)#path-group internet
switch(config-path-group-internet)#local interface et2
switch(config-path-group-internet)#local interface et3
switch(config-path-group-internet)#peer static router-ip 10.2.2.2
switch(config-peer-router-ip-10.2.2.2-internet)#ipv4 address 3.3.3.3
switch(config-peer-router-ip-10.2.2.2-internet)#ipv4 address 4.4.4.4
switch(config-dynamic-path-selection)#load-balance policy voice
switch(config-load-balance-policy-voice)#latency 50
switch(config-load-balance-policy-voice)#path-group mpls
switch(config-load-balance-policy-voice)#path-group internet priority 2
switch(config-load-balance-policy-voice)#load-balance policy best-effort
switch(config-load-balance-policy-best-effort)#path-group mpls
switch(config-load-balance-policy-best-effort)#path-group internet
switch(config-load-balance-policy-best-effort)#load-balance policy
 default
switch(config-load-balance-policy-default)#path-group internet
switch(config-load-balance-policy-default)#policy dynamic
switch(config-policy-dynamic)#10 application-profile platinum
switch(config-policy-rule-key-10-dynamic)#load-balance voice
switch(config-policy-rule-key-10-dynamic)#20 application-profile bronze
switch(config-policy-rule-key-20-dynamic)#load-balance best-effort
switch(config-dynamic-path-selection)#policy dynamic
switch(config-policy-dynamic)#interface ethernet 100
switch(config-if-Et100)#no switchport
switch(config-if-Et100)#ip address 11.0.0.1/24
switch(config-if-Et100)#interface loopback 0
switch(config-if-Lo0)#ip address 10.1.1.1/32
switch(config-if-Lo0)#interface vxlan 1
switch(config-if-Vx1)#vxlan source-interface loopback 0
switch(config-if-Vx1)#vxlan udp-port 4789
switch(config-if-Vx1)#vxlan vrf vrf1 vni 100
switch(config-if-Vx1)#ip route 10.2.2.2/32 ethernet 100
switch(config)#arp 10.2.2.2 00:00:33:02:00:00 arpa
switch(config)#
```

Example 2

```
Site-1
switch(config)#router path-selection
switch(config-dynamic-path-selection)#path-group 1
switch(config-path-group-1)#local interface ethernet 5
!
switch(config-path-group-1)#peer static router-ip 22.22.22.22
switch(config-peer-router-ip-22.22.22.22-1)#ipv4 address
8.0.1.5
!
switch(config-peer-router-ip-22.22.22.22-1)#load-balance
policy policy-1
switch(config-load-balance-policy-policy-1)#path-group 1
```

```
!
switch(config-load-balance-policy-policy-1)#policy policy-1
switch(config-policy-policy-1)#default-match
switch(config-policy-default-rule-policy-1)#load-balance
policy-1
!
switch(config-policy-default-rule-policy-1)#vrf default
switch(config-vrf-default)#path-selection-policy policy-1
!
switch(config-dynamic-path-selection)#vrf et1
switch(config-vrf-et1)#path-selection-policy policy-1
!
switch(config-vrf-et1)#vrf instance et1
switch(config-vrf-et1)#interface ethernet 1
switch(config-if-Et1)#description LAN-interface
switch(config-if-Et1)#no switchport
switch(config-if-Et1)#ip address 4.0.1.5/24
!
switch(config)#vrf instance et1
switch(config-vrf-et1)#interface ethernet 1
switch(config-if-Et1)#description LAN-interface
switch(config-if-Et1)#no switchport
switch(config-if-Et1)#ip address 4.0.1.5/24
!
switch(config-if-Et1)#interface ethernet 5
switch(config-if-Et5)#description WAN-Interface
switch(config-if-Et5)#no switchport
switch(config-if-Et5)#ip address 5.0.1.5/24
!
switch(config-if-Et5)#interface ethernet 100
switch(config-if-Et100)#no switchport
switch(config-if-Et100)#ip address 10.0.0.2/24
!
switch(config-if-Et100)#interface loopback 1
switch(config-if-Lo1)#ip address 11.11.11.11/32
!
switch(config-if-Lo1)#interface vxlan 1
switch(config-if-Vx1)#vxlan source-interface loopback 1
switch(config-if-Vx1)#vxlan udp-port 4789
switch(config-if-Vx1)#vxlan vrf et1 vni 5
!
switch(config-if-Vx1)#ip route 22.22.22.22/32 ethernet 100
!
switch(config)#arp 22.22.22.22 22:22:22:22:22:22 arpa
!
switch(config)#ip routing
switch(config)#ip routing vrf et1
!
switch(config)#router bgp 32
switch(config-router-bgp)#neighbor 5.0.1.1 remote-as 501
switch(config-router-bgp)#neighbor 5.0.1.1 maximum-routes
12000
switch(config-router-bgp)#neighbor 22.22.22.22 remote-as 43
switch(config-router-bgp)#neighbor 22.22.22.22 update-source
 loopback 1
switch(config-router-bgp)#neighbor 22.22.22.22 ebgp-multihop
switch(config-router-bgp)#neighbor 22.22.22.22 send-community
 extended
switch(config-router-bgp)#neighbor 22.22.22.22 maximum-routes
 12000
switch(config-router-bgp)#redistribute static
!
switch(config-router-bgp)#address-family evpn
switch(config-router-bgp-af)#neighbor 22.22.22.22 activate
```

```
!
switch(config-router-bgp-af)#exit
switch(config-router-bgp)#address-family ipv4
switch(config-router-bgp-af)#no neighbor 22.22.22.22 activate
switch(config-router-bgp-af)#exit
!
switch(config)#router bgp 32
switch(config-router-bgp)#vrf et1
switch(config-router-bgp-vrf-et1)#rd 4.0.1.5:0
switch(config-router-bgp-vrf-et1)#route-target import evpn
9.0.1.5:0
switch(config-router-bgp-vrf-et1)#route-target export evpn
4.0.1.5:0
switch(config-router-bgp-vrf-et1)#router-id 4.0.1.5
switch(config-router-bgp-vrf-et1)#network 4.0.1.0/24
switch(config-router-bgp-vrf-et1)#network 50.0.0.0/24
switch(config-router-bgp-vrf-et1)#exit

switch(config-router-bgp)#exit
switch(config)#
----------------------------------------------------------------------
-----------
Site-2
switch(config)#router path-selection
switch(config-dynamic-path-selection)#path-group 1
switch(config-path-group-1)#local interface ethernet 1
!
switch(config-path-group-1)#peer static router-ip 11.11.11.11
switch(config-peer-router-ip-11.11.11.11-1)#ipv4 address
5.0.1.5
!
switch(config-peer-router-ip-11.11.11.11-1)#load-balance
policy policy-1
switch(config-load-balance-policy-policy-1)#path-group 1
!
switch(config-load-balance-policy-policy-1)#policy policy-1
switch(config-policy-policy-1)#default-match
switch(config-policy-default-rule-policy-1)#load-balance
policy-1
!
switch(config-policy-default-rule-policy-1)#vrf default
switch(config-vrf-default)#path-selection-policy policy-1
!
switch(config-dynamic-path-selection)#vrf et5
switch(config-vrf-et5)#path-selection-policy policy-1
!
switch(config-vrf-et5)#vrf instance et5
switch(config-vrf-et5)#interface ethernet 1
switch(config-if-Et1)#description WAN-Interface
switch(config-if-Et1)#no switchport
switch(config-if-Et1)#ip address 8.0.1.5/24
!
switch(config)#vrf instance et5
switch(config-vrf-et5)#interface ethernet 5
switch(config-if-Et5)#description LAN-interface
switch(config-if-Et5)#no switchport
switch(config-if-Et5)#ip address 9.0.1.5/24
!
switch(config-if-Et5)#interface ethernet 100
switch(config-if-Et100)#no switchport
switch(config-if-Et100)#ip address 10.0.0.1/24
!
switch(config-if-Et100)#interface loopback 1
switch(config-if-Lo1)#ip address 22.22.22.22/32
!
switch(config-if-Lo1)#interface vxlan 1
switch(config-if-Vx1)#vxlan source-interface loopback 1
switch(config-if-Vx1)#vxlan udp-port 4789
switch(config-if-Vx1)#vxlan vrf et5 vni 5
!
switch(config-if-Vx1)#ip route 11.11.11.11/32 ethernet 100
!
switch(config)#arp 11.11.11.11 11:11:11:11:11:11 arpa
!
switch(config)#ip routing
switch(config)#ip routing vrf et5
!
switch(config)#router bgp 43
switch(config-router-bgp)#maximum-paths 16
switch(config-router-bgp)#neighbor 8.0.1.1 remote-as 701
switch(config-router-bgp)#neighbor 8.0.1.1 maximum-routes
12000
switch(config-router-bgp)#neighbor 11.11.11.11 remote-as 32
switch(config-router-bgp)#neighbor 11.11.11.11 update-source
 loopback 1
switch(config-router-bgp)#neighbor 11.11.11.11 ebgp-multihop
switch(config-router-bgp)#neighbor 11.11.11.11 send-community
 extended
switch(config-router-bgp)#neighbor 11.11.11.11 maximum-routes
 12000
!
switch(config-router-bgp)#address-family evpn
switch(config-router-bgp-af)#neighbor 11.11.11.11 activate
switch(config-router-bgp-af)#exit
!
switch(config-router-bgp)#address-family ipv4
switch(config-router-bgp-af)#no neighbor 11.11.11.11 activate
switch(config-router-bgp-af)#exit
!
switch(config)#router bgp 40
switch(config-router-bgp)#vrf et5
switch(config-router-bgp-vrf-et5)#rd 9.0.1.5:0
switch(config-router-bgp-vrf-et5)#route-target import evpn
4.0.1.5:0
switch(config-router-bgp-vrf-et5)#route-target export evpn
9.0.1.5:0
switch(config-router-bgp-vrf-et5)#router-id 9.0.1.5
switch(config-router-bgp-vrf-et5)#network 9.0.1.0/24
switch(config-router-bgp-vrf-et5)#network 51.0.0.0/24
switch(config-router-bgp-vrf-et5)#exit
switch(config-router-bgp)#exit
switch(config)#
```

14.3 DPS Display Commands

The following **show commands** are used to verify the various information of the Dynamic Path Selection application.

14.3.1 Path Telemetry Show Commands

These two show commands provide path telemetry status:

`show monitor telemetry path characteristics` **[detail][*destination DSTIP*][*path-name NAME*][*peer PEERIP*] [*source SRCIP*] [*traffic-class TC*]**

`show monitor telemetry path counters` **[detail][** ***destination DSTIP*** **][** ***path-name NAME*** **][** ***peer PEERIP*** **] [** ***source SRCIP*** **][** ***traffic-class TC*** **]**

Example

- The **`show monitor telemetry path characteristics`** command displays the path state, latency, jitter, and other information.

```
switch#show monitor telemetry path characteristics
PathName   TrafficClass  TxState  Latency(ms)  Jitter(ms)
 Throughput(Mbps)  LossRate(%)
path1      0             active   3.520         1.122          10.00
           0.01
path2      0             active   35.220        2.330          10.00
           1.01

switch#show monitor telemetry path characteristics detail
Peer: 10.1.10.5
  PathName: path1
  Source: 156.142.20.23, Destination: 156.142.40.21
    Traffic Class: 0
      TxState: active
      Latency: 3.520 ms
      Jitter:  1.122 ms
      Throughput: 10.00 Mbps
      LossRate: 0.01 %
  PathName: path2
  Source: 156.142.20.24, Destination: 156.142.40.22
    Traffic Class: 0
      TxState: active
      Latency: 35.220 ms
      Jitter:  2.330  ms
      Throughput: 1000 Mbps
      LossRate: 1.01 %
```

- The **`show monitor telemetry path counters`** displays the input output bytes and packets and flow information.

```
switch#show monitor telemetry path counters
PathName   TrafficClass  InBytes  InPkts  InPktsDrop  OutBytes  OutPkts
  OutPktsDrop
path1      0             4553300  1022    0           5341333   752
  0
path2      0             4553300  1022    0           5341333   752
  0

kvs17-b10#show monitor telemetry path counters detail
Peer: 10.1.10.5
  PathName: path1
  Source: 156.142.20.23, Destination: 156.142.40.21
    Traffic Class: 0
      InBytes: 4553300
      InPkts: 1022
      InPktsDrop: 0
      OutBytes: 5341333
      OutPkts: 752
      OutPktsDrop: 0
```

Both path characteristics and path counters show results can be filtered by path name, destination IP, source IP, remote IP and traffic class. And both of them have detail version output and brief version output, default version is brief version as shown.

14.3.2 IPsec Show Commands

The following IPsec show commands filter IPsec connections based on path name and remote IP address. The IPsec show results are filtered using the following options like Tunnel, Detail, Path, and VRF.

Examples

- The **`show ip security connection path`** command displays all path based IP security connections.

```
switch#show ip security connection path
Name    Source   Dest   Status       Uptime       Input      Output
 Rekey Time
Path1   ip1      ip3    Established  22 minutes   0 bytes    0 bytes
 34 minutes
                                        0 pkts     0 pkts
Path2   ip2      ip3    Established  22 minutes   0 bytes    0 bytes
 34 minutes
                                        0 pkts     0 pkts
Path2   ip5      ip6    Established  22 minutes   0 bytes    0 bytes
 34 minutes
                                        0 pkts     0 pkts
```

- The **`show ip security connection path name`** command displays IPsec path connections based on the path name.

```
switch#show ip security connection path name path1
Name    Source   Dest   Status       Uptime       Input      Output
 Rekey Time
Path1   ip1      ip3    Established  22 minutes   0 bytes    0 bytes
 34 minutes
                                       0 pkts     0 pkts
```

- The**`show ip security connection path peer`** command displays the IPsec path connections based on the remote router IP.

```
switch#show ip security connection path peer ip3
Name    Source   Dest   Status       Uptime       Input      Output
 Rekey Time
Path1   ip1      ip3    Established  22 minutes   0 bytes    0 bytes
 34 minutes
                                        0 pkts    0 pkts
Path2   ip2      ip3    Established  22 minutes   0 bytes    0 bytes
 34 minutes
                                        0 pkts    0 pkts
```

14.3.3 Load balance and Application Counters

These counters display the statistics of load balancing based on application profile, overlay VRF and remote node IP:

`show path-selection load-balance counter` [detail] [*application-profile APPNAME*] [*peer PEERIP*] [*vrf VRFNAME*]

`show path-selection application counters`[*application-profile APPNAME*] [*peer PEERIP*] [*vrf VRFNAME*]

Examples

- The **`show path-selection load-balance counter`** command displays for every (application profile, overlay VRF and remote IP), per path group flow count and the throughput of path group.

```
switch#show path-selection load-balance counters
AppProfile                Vrf        Peer             PathGroup
 Path       Flows      Throughput(Mbps)
app1                      vrf1       11.0.1.1         transit0
 path2      0          0.00
app2                      vrf1       11.0.1.1         transit1
 path1      0          0.00
default_app               default    11.0.1.1         transit0
 path2      0          0.00
                                                      transit1
 path1      0          0.00
```

- The **`show path-selection load-balance counters detail`** command displays for every (application profile, overlay VRF and remote IP), per path group flow count, out bytes, out packets and the throughput of path group.

```
switch#show path-selection load-balance counters detail

AppProfile                Vrf        Peer             PathGroup
    Path       Flows      Throughput(Mbps)   OutBytes
 OutPkts
app1                      vrf1       11.0.1.1         transit0
 path2      0          0.00              0                      0

app2                      vrf1       11.0.1.1         transit1
 path1      0          0.00              0                      0

default_app               default    11.0.1.1         transit0
 path2      0          0.00              1052                   17

                                                      transit1
 path1      0          0.00              1321                   17
```

- The **`show path-selection application counters`** command displays the application profile, overlay VRF and remote IP out bytes, out packets and throughput.

```
switch#show path-selection application counters
 AppProfile   VRF   Peer      Throughput OutBytes   OutPackets

  Silver      Red   10.0.0.1    15        3000           15
```

Output of both **`show path-selection load-balance counters`** and **`show path-selection application counters`** can be filtered by application-profile name, peer IP address and vrf name.

14.4 Clear Commands

The following commands clears the DPS related counters:

Syntax

Clear load balancing and application counters:

clear path-selection counters **Clear path telemetry counters:**

clear monitor telemetry path counters

14.5 Troubleshooting

In order for DPS to work, the following needs to be working.

1. Verify the paths are in the “Estab” or “Estab IPSec” state using “show path-selection paths” command. If the path is not in established state.
    - ARP Pending - Make sure the next-hop to the path destination IP is available.
    - Route Pending - Make sure a route to the path destination IP is available through the local interface for the path.
    - IPSec Pending - Check IPSec connection with “show ip security connection” or other IPSec related commands between the path’s local interface and the path’s destination.
2. If the paths are in Estab state, verify the paths are active and available using “show monitor telemetry path characteristics”
    - If a path is inactive, make sure IP connectivity is working between the path’s source IP/interface and destination IP. Ping the path destination with the path source IP could be one of the ways to verify this. And also, to check the configuration and make sure that the paths are configured symmetrically on both sites.
    - Check and make sure there are DPS communications between the source and destination IPs using TCP dump on et100.
3. Paths are active but ping between loopbacks of the two sites is not working. Loopbacks should be reachable through overlay.
    - Check your interface VXLAN1 configuration.
    - Check and make sure you have applied a policy with default match to your “vrf default” configuration in DPS.
4. Site-to-Site loopback IPs are reachable but data traffic is not going through.
    - Check your EVPN configuration. Make sure the remote routes are in your VRF route table of your sites.
    - Make sure your DPS configuration has proper policy, application profile, default match and load-balance profile

ECMP

Equal-cost multi-path routing (**ECMP**) is a routing plan of action where next-hop packet forwarding to a single destination takes place over multiple "best paths" which tie for top place in routing metric calculations. Use multi-path routing in conjunction with most routing protocols, because it is a per-hop decision limited to a single router.

15.1 Adding ECMP

Describes how to add ECMP to new or existing VMs.

1. Have the following line in a device's running-configuration.

   If an instance is created with an older, pre-vEOS 4.20.5 image, add the command line in the example below.

   If an instance is created with vEOS 4.20.5 or later image, there is no need for additional configuration changes because the command line appears in the configuration by default.

```
agent KernelFib environment KERNELFIB_PROGRAM_ALL_ECMP='true'
```

2. Reload the device or restart the KernelFib agent via **`agent KernelFib terminate`**.

   This step is needed only if the instance was created with an older, pre-vEOS 4.20.5 image.
3. To enable ECMP in a routing protocol, issue the **`maximum-paths <#>`** command inside the routing protocol used.

```
veos#configure terminal
veos(config)#router bgp 65112
veos(config-router-bgp)#maximum-paths 16
```

4. When ECMP starts, and there are multiple routes, display output may be similar to the following example.

```
veos#show ip route 10.4.3.0

VRF: default
Codes: C - connected, S - static, K - kernel,
       O - OSPF, IA - OSPF inter area, E1 - OSPF external type 1,
       E2 - OSPF external type 2, N1 - OSPF NSSA external type 1,
       N2 - OSPF NSSA external type2, B I - iBGP, B E - eBGP,
       R - RIP, I L1 - IS-IS level 1, I L2 - IS-IS level 2,
       O3 - OSPFv3, A B - BGP Aggregate, A O - OSPF Summary,
       NG - Nexthop Group Static Route, V - VXLAN Control Service,
       DH - DHCP client installed default route, M - Martian

 S      10.4.3.0/24 [1/0] via 190.19.11.2, Tunnel1
                          via 190.19.11.102, Tunnel3
```

5. To determine the route that the interface traffic takes to specific addresses, issue the **`bash ip route get <address>`** command to determine which link the traffic uses. In the following example, traffic to ***10.4.3.5*** takes Tunnel1, while traffic to ***10.4.3.6*** takes Tunnel3.

```
veos#bash ip route get 10.4.3.5
10.4.3.5 via 190.19.11.2 dev tun1 src 190.19.11.1
    cache
veos#bash ip route get 10.4.3.6
10.4.3.6 via 190.19.11.102 dev tun3 src 190.19.11.101
    cache
```

For additional information regarding ECMP, refer to the current release notes.