

Configure CSR1000v HA Version 3 on AWS, Azure and GCP

Contents

[Introduction](#)

[Prerequisites](#)

[Requirements](#)

[Components Used](#)

[Background Information](#)

[Topology](#)

[Network Diagram](#)

[Configure CSR1000v Routers](#)

[Cloud Independent Configuration](#)

[AWS Specific Configuration](#)

[Azure Specific Configuration](#)

[GCP Specific Configuration](#)

[Verify](#)

[Troubleshoot](#)

[Related Information](#)

Introduction

This document describes the steps to configure CSR1000v routers for High Availability Version 3 (HAV3) on Amazon Web Services (AWS), Microsoft Azure and Google Cloud Platform (GCP).

Prerequisites

Requirements

Cisco recommends that you have knowledge of these topics:

- AWS, Azure or GCP clouds.
- CSR1000v routers.
- Cisco IOS®-XE.

This article assumes the underlying network configuration has already been completed and focuses on the HAV3 configuration.

Full configuration details are found in the [Cisco CSR 1000v and Cisco ISRV Software Configuration Guide](#).

Components Used

The information in this document is based on these software and hardware versions:

- An AWS, Azure or GCP account.
- 2 CSR1000v routers.
- A minimum of Cisco IOS®-XE Polaris 16.11.1s

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

Background Information

Cisco recommends that you have knowledge of different HA versions available:

- HAv1: HA configuration is performed as IOS commands and relies on BFD as the mechanism to detect failure.
- HAv2/HA3: The implementation has been moved into the guestshell container as python scripts. BFD is optional and custom scripts can be written to detect failure and trigger failover. Azure HAv2 configuration is largely similar to HAv3 with minor differences in pip install packages and IOS redundancy configuration.
- HAv3: The implementation of HA has been largely moved out of the Cisco IOS®-XE code and runs in the guestshell container.

HA3 is available from Cisco IOS®-XE Polaris 16.11.1s and adds several new features:

- **Cloud Agnostic:** This version of high availability is functional on CSR 1000v routers on any cloud service provider. While there are some differences in the cloud terminology and parameters, the set of functions and scripts used to configure, control, and show the high availability features are common across the different cloud service providers. High Availability Version 3 (HA3) is supported in CSR 1000v routers on AWS, Azure, and GCP. Support for the GCP provider has been added in 16.11.1. Check with Cisco for current support of high availability in the individual provider's clouds.
- **Active/active operation:** You can configure both Cisco CSR 1000v routers to be active simultaneously, which allows for load sharing. In this mode of operation, each route in a route table has one of the two routers serve as the primary router and the other router as the secondary router. To enable load sharing, take all the routes and split them between the two Cisco CSR 1000v routers. Note that this functionality is new for AWS-based clouds.
- **Reversion to Primary CSR After Fault Recovery:** You can designate a Cisco CSR 1000v as the primary router for a given route. While this Cisco CSR 1000v is up, it is the next hop for the route. If this Cisco CSR 1000v fails, the peer Cisco CSR 1000v takes over as the next hop for the route, maintaining network connectivity. When the original router recovers from the failure, it reclaims ownership of the route and is the next hop router. This functionality is also new for the AWS-based clouds.
- **User-supplied Scripts:** The guestshell is a container in which you can deploy your own scripts. HA3 exposes a programming interface to user-supplied scripts. This implies you can now write scripts that can trigger both failover and reversion events. You can also develop your own algorithms and triggers to control which Cisco CSR 1000v provides the forwarding services for a given route. This functionality is new for AWS-based clouds.
- **New Configuration and Deployment Mechanism:** The implementation of HA has been moved out of the Cisco IOS®-XE code. High availability code now runs in the guestshell container. For further information on guestshell, see the "Guest Shell" section in the

Programmability Configuration Guide. In HA v3, the configuration of redundancy nodes is performed in the guestshell that uses a set of Python scripts. This feature has now been introduced for AWS-based clouds.

Note: Resources deployed in AWS, Azure, or GCP from the steps in this document can incur a cost.

Topology

Before configuration starts, it is important to understand the topology and design completely. This helps to troubleshoot any potential issues later on.

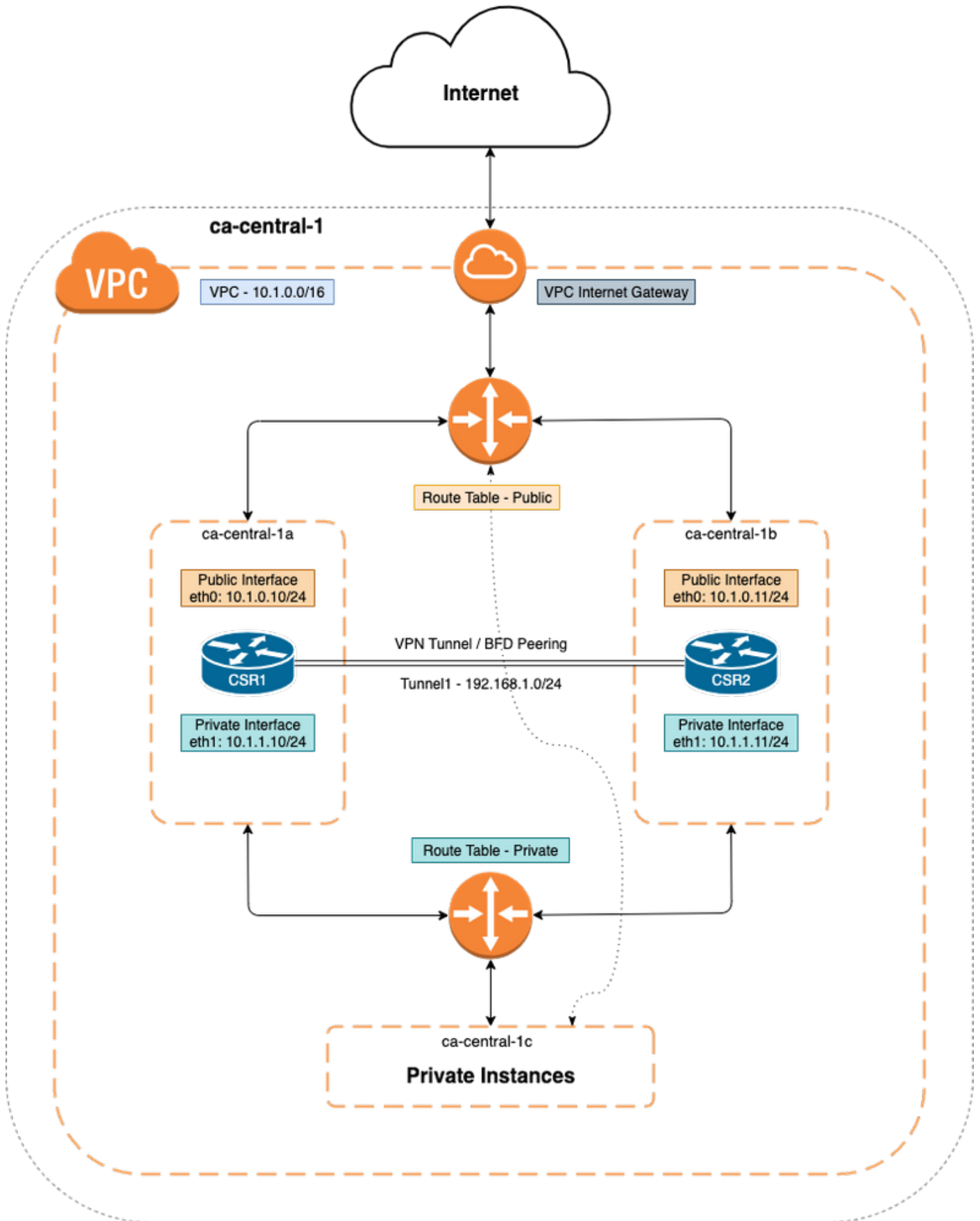
Although the network topology diagram is based on AWS, the underlying network deployment between clouds are relatively similar. The network topology is also independent of the HA version used, whether it be HA v1, HA v2 or HA v3.

For this topology example, HA redundancy is configured with these settings in AWS:

- 1x - Region
- 1x - VPC
- 3x - Availability Zones
- 4x - Network Interfaces/Subnets (2x Public Facing/2x Private Facing)
- 2x - Route Tables (Public & Private)
- 2x - CSR1000v routers (Cisco IOS®-XE 17.01.01)

There are 2x CSR1000v routers in an HA pair, in two different availability zones. The third zone is a private instance, which simulates a device in a private datacenter. Generally, all normal traffic must flow through the private (or inside) route table.

Network Diagram



Network Diagram

Configure CSR1000v Routers

Cloud Independent Configuration

Step 1. Configure IOX app-hosting and guestshell, this provides provide ip reachability into guestshell. This step can be automatically configured by default when CSR1000v is deployed.

```
vrf definition GS ! iox app-hosting appid guestshell app-vnic gateway1 virtualportgroup 0 guest-interface 0 guest-ipaddress
192.168.35.102 netmask 255.255.255.0 app-default-gateway 192.168.35.101 guest-interface 0 name-server0 8.8.8.8 ! interface
VirtualPortGroup0 vrf forwarding GS ip address 192.168.35.101 255.255.255.0 ip nat inside ! interface GigabitEthernet1 ip nat
outside ! ip access-list standard GS_NAT_ACL permit 192.168.35.0 0.0.0.255 ! ip nat inside source list GS_NAT_ACL interface
GigabitEthernet1 vrf GS overload !! The static route points to the G1 ip address's gateway ip route vrf GS 0.0.0.0 0.0.0.0
GigabitEthernet1 10.1.0.1 global
```

Step 2. Enable and login to guestshell.

```
Device#guestshell enable
Interface will be selected if configured in app-hosting
Please wait for completion
guestshell installed successfully
Current state is: DEPLOYED
guestshell activated successfully
Current state is: ACTIVATED
guestshell started successfully
Current state is: RUNNING
Guestshell enabled successfully

Device#guestshell
[guestshell@guestshell ~]$
```

Note: For more information about guestshell see - [Programmability Configuration Guide](#)

Step 3. Confirm guestshell is able to communicate to Internet.

```
[guestshell@guestshell ~]$ ping 8.8.8.8
PING 8.8.8.8 (8.8.8.8) 56(84) bytes of data.
64 bytes from 8.8.8.8: icmp_seq=1 ttl=109 time=1.74 ms
64 bytes from 8.8.8.8: icmp_seq=2 ttl=109 time=2.19 ms
64 bytes from 8.8.8.8: icmp_seq=3 ttl=109 time=2.49 ms
64 bytes from 8.8.8.8: icmp_seq=4 ttl=109 time=1.41 ms
64 bytes from 8.8.8.8: icmp_seq=5 ttl=109 time=3.04 ms
```

Step 4. (Optional) Enable Bi-directional Forwarding Detection (BFD) and a routing protocol as Enhanced Interior Gateway Routing Protocol (EIGRP) or Border Gateway Protocol (BGP) to the tunnel for peer failure detection. Configure either a VxLAN or IPsec tunnel between the Cisco CSR 1000v routers.

- IPsec tunnel between the Cisco CSR 1000v routers.

```
crypto isakmp policy 1 encr aes 256 authentication pre-share crypto isakmp key cisco address <peer_ip_address> crypto ipsec
transform-set uni-perf esp-aes 256 esp-sha-hmac mode tunnel crypto ipsec profile vti-1 set security-association lifetime kilobytes
disable set security-association lifetime seconds 86400 set transform-set uni-perf set pfs group2 interface Tunnel1 ip address
192.168.1.1 255.255.255.0 bfd interval 500 min_rx 500 multiplier 3 tunnel source GigabitEthernet1 tunnel destination
<peer_public_interface_ip_address> redundancy cloud-ha bfd peer <peer_router_ip_address> Example - #CSR1 ! interface
Tunnel1 ip address 192.168.1.1 255.255.255.0 bfd interval 500 min_rx 500 multiplier 3 tunnel source GigabitEthernet1 tunnel
destination 10.1.0.11 ! redundancy cloud-ha bfd peer 192.168.1.2 #CSR2 ! interface Tunnel1 ip address 192.168.1.2
255.255.255.0 bfd interval 500 min_rx 500 multiplier 3 tunnel source GigabitEthernet1 tunnel destination 10.1.0.10 ! redundancy
cloud-ha bfd peer 192.168.1.1
```

- VxLAN tunnel between the Cisco CSR 1000v routers.

```
Example: interface Tunnel100 ip address 192.168.1.1 255.255.255.0 bfd interval 500 min_rx 500 multiplier 3 tunnel source
GigabitEthernet1 tunnel mode vxlan-gpe ipv4 tunnel destination <peer_public_interface_ip_address> tunnel vxlan vni 10000
redundancy cloud-ha bfd peer <peer_router_ip_address>
```

Step 4.1. (Optional) Configure EIGRP over Tunnel Interfaces.

```
router eigrp 1 bfd interface Tunnel1 network 192.168.1.0 0.0.0.255
```

- Custom scripts can be used to trigger failover, example:

```
event manager applet Interface_GigabitEthernet2 event syslog pattern "Interface GigabitEthernet2, changed state to
administratively down" action 1 cli command "enable" action 2 cli command "guestshell run node_event.py -i 10 -e peerFail" exit
exit
```

AWS Specific Configuration

- AWS HA Parameters

Parameter	Switch	Description
Node Index	-i	Index that is used to uniquely identify this node. Valid values: 1-1023.
Region Name	-rg	Name of the region that contains the route table. For example, us-west-2.
Route Table Name	-t	Name of the route table to be updated. The name of the route table must begin with the substring rtb-. For example, rtb-001333c29ef2aec5f
Route	-r	If a route is unspecified, then the redundancy node is considered to apply to all routes in the routing table. The CSR cannot change routes which are of type local or gateway.
Next Hop Interface	-n	Name of the interface to which packets should be forwarded in order to reach the destination route. The name of the interface must begin with the substring eni-. For example, eni-07160c7e740ac8ef4.
Mode	-m	Indicates whether this router is the primary or secondary router for servicing this route. Valid values are primary or secondary. This is an optional parameter. The default value is secondary.

Step 1. Configure authentication with IAM.

In order to CSR1000v router to update a routing table in the AWS network, router needs to be authenticated. In AWS, you must create a policy that permits the CSR 1000v router to access the route table. An IAM role is then created that use this policy and applied to the EC2 resource.

After the CSR 1000v EC2 instances are created, the IAM role created needs to be attached to each router.

The policy used in the new IAM role is:

```
{ "Version": "2012-10-17", "Statement": [ { "Sid": "VisualEditor0", "Effect": "Allow", "Action": [ "logs:CreateLogStream",
"cloudwatch:", "s3:", "ec2:AssociateRouteTable", "ec2:CreateRoute", "ec2:CreateRouteTable", "ec2>DeleteRoute",
"ec2>DeleteRouteTable", "ec2:DescribeRouteTables", "ec2:DescribeVpcs", "ec2:ReplaceRoute", "ec2:DescribeRegions",
"ec2:DescribeNetworkInterfaces", "ec2:DisassociateRouteTable", "ec2:ReplaceRouteTableAssociation", "logs:CreateLogGroup",
"logs:PutLogEvents" ], "Resource": "*" } ] }
```

Note: Refer to [IAM role with a Policy and associate it to the VPC](#) for detailed steps.

Step 2. Install the HA python package.

```
[guestshell@guestshell ~]$ pip install csr_aws_ha --user
[guestshell@guestshell ~]$ source ~/.bashrc
```

Step 3. Configure HA parameters on the primary router.

```
[guestshell@guestshell ~]$ create_node.py -i 10 -t rtb-01c5b0633a3422575 -rg ca-central-1 -n eni-0bc1912748614df2a -r 0.0.0.0/0 -m primary
```

Step 4. Configure HA parameters on the secondary router.

```
[guestshell@guestshell ~]$ create_node.py -i 10 -t rtb-01c5b0633a3422575 -rg ca-central-1 -n eni-0e351ab1b8f416728 -r 0.0.0.0/0 -m secondary
```

- Node format is:

```
create_node.py -i n -t rtb-private-route-table-id -rg region-id -n eni-CSR-id -r route(x.x.x.x/x) -m <primary|secondary>
```

Azure Specific Configuration

- Azure HA Parameters

The following table specifies the redundancy parameters that are specific to Microsoft Azure:

Parameter Switch	Switch	Description
Node Index	-i	The index that is used to uniquely identify this node. Valid values: 1-255.
Cloud Provider	-p	Specifies the type of Azure cloud: azure, azusgov, or azchina.
Subscription ID	-s	The Azure subscription id.
Resource Group Name	-g	The name of the route table to be updated.
Route Table Name	-t	The name of the route table to be updated.
Route	-r	IP address of the route to be updated in CIDR format. Can be IPv4 or IPv6 address. If a route is unspecified, then the redundancy node is considered to apply to all routes in the routing table of type "virtual appliance".
Next Hop Address	-n	The IP address of the next hop router. Use the IP address that is assigned to this CSR 1000v on the subnet which utilizes this route table. Can be an IPv4 or IPv6 address.
Mode	-m	Indicates whether this router is the primary or secondary router for servicing this route. Default value is secondary.

Note: The outside facing interface must be configured on GigabitEthernet1. This is the interface used to reach Azure API's. HA can not function properly otherwise. Within guestshell, ensure curl command can fetch metadata from Azure.

```
[guestshell@guestshell ~]$ curl -H "Metadata:true" http://169.254.169.254/metadata/instance?api-version=2020-06-01
```

Step 1. Authentication for CSR1000v API Calls must be enabled with either Azure Active Directory (AAD) or Managed Service Identity (MSI). Refer to [Configure Authentication for CSR1000v API Calls](#) for detailed steps. Without this step, the CSR1000v router can not be authorized to update the route table.

AAD parameters

Parameter Name	Switch	Description
Cloud Provider	-p	Specifies which Azure cloud is in use {azure azusgov azchina}
Tenant ID	-d	Identifies the AAD instance.
Application ID	-a	Identifies the application in AAD.
Application Key	-k	Access key that is created for the application. Key should be specified in unencoded URL format.

Step 2. Install the HA python package.

```
[guestshell@guestshell ~]$ pip install csr_azure_ha --user
[guestshell@guestshell ~]$ source ~/.bashrc
```

Step 3. Configure HA parameters on the primary router (MSI or AAD can be used for this step).

- With MSI authentication.

```
[guestshell@guestshell ~]$ create_node -i 10 -p azure -s xxxxxxxx-xxxx-xxxx-xxxx-xxxxxxxxxxxxxxxx -g ResourceGroup -t Private-RouteTable -r 0.0.0.0/0 -n 10.1.0.10 -m primary
```

- With AAD authentication (Additional -a, -d, -k flags required).

```
[guestshell@guestshell ~]$ create_node -i 10 -p azure -s xxxxxxxx-xxxx-xxxx-xxxx-xxxxxxxxxxxxxxxx -g ResourceGroup -t Private-RouteTable -r 0.0.0.0/0 -n 10.1.0.10 -m primary -a 1e0f69c3-b6aa-46cf-b5f9-xxxxxxxx -d ae49849c-2622-4d45-b95e-xxxxxxxx -k bDEN1k8batJqpeqjAuUvaUCZn5Md6rWEi=
```

Step 4. Configure HA parameters on the secondary router.

- With MSI authentication

```
[guestshell@guestshell ~]$ create_node -i 10 -p azure -s xxxxxxxx-xxxx-xxxx-xxxx-xxxxxxxxxxxxxxxx -g ResourceGroup -t Private-RouteTable -r 0.0.0.0/0 -n 10.1.0.11 -m secondary
```

- With AAD authentication (Additional -a, -d, -k flags required)

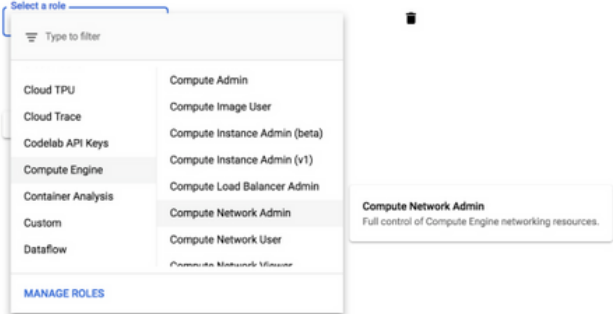
```
[guestshell@guestshell ~]$ create_node -i 10 -p azure -s xxxxxxxx-xxxx-xxxx-xxxx-xxxxxxxxxxxxxxxx --g ResourceGroup -t Private-RouteTable -r 0.0.0.0/0 -n 10.0.0.11 -m secondary -a 1e0f69c3-b6aa-46cf-b5f9-xxxxxxxx -d ae49849c-2622-4d45-b95e-xxxxxxxx -k bDEN1k8batJqpeqjAuUvaUCZn5Md6rWEi=
```

GCP Specific Configuration

- GCP HA Parameters

Parameter	Is this parameter required?	Switch	Description
Node Index	Yes	-i	The index that is used to uniquely identify this node. Valid values: 1–255.
Cloud Provider	Yes	-p	Specify gcp for this parameter.
Project	Yes	-g	Specify the Google Project ID.
routeName	Yes	-a	The route name for which this CSR is next hop. For example from Fig. 2, if we are configuring node on CSR 1, this would be route-vpc2-csr1.
peerRouteName	Yes	-b	The route name for which the BFD peer CSR is next hop. For example from Fig. 2, if we are configuring node on CSR 1, this would be route-vpc2-csr2.
Route	yes	-r	The IP address of the route to be updated in CIDR format. Can be IPv4 or IPv6 address. If a route is unspecified, then the redundancy node is considered to apply to all routes in the routing table of type virtual appliance. Note: Currently Google cloud does not have IPv6 support in VPC.
Next hop address	Yes	-n	The IP address of the next hop router. Use the IP address that is assigned to this CSR 1000v on the subnet which utilizes this route table. The value can be an IPv4 or IPv6 address. Note: Currently Google cloud does not have IPv6 support in VPC.
hopPriority	Yes	-o	The route priority for the route for which the current CSR is the next hop.
VPC	Yes	-v	The VPC network name where the route with the current CSR as the next hop exists.

Note: Ensure the service account associated with the CSR 1000v routers at least have a Compute Network Admin permission.

Command or Action	Purpose
Ensure that the service account associated with the CSR 1000v routers at least have a Compute Network Admin permission.	<p>Create service account</p> <p>1 Service account details — 2 Grant this service account access to project (optional) — 3 Grant users access to this service account (optional)</p> <p>Service account permissions (optional)</p> <p>Grant this service account access to project-avvays so that it has permission to complete specific actions on the resources in your project. Learn more</p>  <p>You can also provide the required permissions in a credentials file with name 'credentials.json' and place it under the /home/guestshell directory. The credentials file overrides the permissions supplied through the service account associated with the CSR 1000v instance.</p>

369497

Step 1. Install the HA python package.

```
[guestshell@guestshell ~]$ pip install csr_gcp_ha --user
[guestshell@guestshell ~]$ source ~/.bashrc
```

Step 2. Configure HA parameters on the primary router.

```
[guestshell@guestshell ~]$ create_node -i 1 -g -r dest_network -o 200 -n nexthop_ip_addr -a route-vpc2-csr1 -b route-vpc2-csr2 -p gcp -v vpc_name
```

Step 3. Configure HA parameters on the secondary router.

```
[guestshell@guestshell ~]$ create_node -i 1 -g -r dest_network -o 200 -n nexthop_ip_addr -a route-vpc2-csr2 -b route-vpc2-csr1 -p gcp -v vpc_name
```

Verify

Use this section to confirm that your configuration works properly.

Step 1. Trigger a failover with the node_event.py peerFail flag.

```
[guestshell@guestshell ~]$ node_event.py -i 10 -e peerFail 200: Node_event processed successfully
```

Step 2. Navigate to the Private Route Table of your Cloud Provider, verify the route has updated the next-hop to the new IP address.

Troubleshoot

There is currently no specific troubleshooting information available for this configuration.

Related Information

- Detailed HAV3 configuration steps are found at [Cisco CSR 1000v and Cisco ISRV Software Configuration Guide](#)
- Azure HAV2 configuration is largely similar to HAV3 with minor differences in pip install packages and IOS redundancy configuration. Documentation is found at [CSR1000v HA Version 2 Configuration Guide on Microsoft Azure](#)
- Azure HAV1 configuration with CLI is found at [CSR1000v HA Redundancy Deployment Guide on Microsoft Azure with AzureCLI 2.0](#)
- AWS HAV1 configuration is found at [CSR1000v HA Redundancy Deployment Guide on Amazon AWS](#)
- [Technical Support & Documentation - Cisco Systems](#)