Multi-Cloud VPN and Multi-Zone Subnetworks — Network Setup for Multi-Cloud Database Deployments

A tutorial for setting up a multi-cloud VPN

Christoph Bussler Follow
Aug 10 · 13 min read

Introduction

This blog is a tutorial on how to set up a VPN between Google Cloud and AWS, create multi-zone subnetworks in each cloud and test any-to-any connectivity. For part of the setup this blog follows the community tutorial Google Cloud HA VPN interoperability guide for AWS and cites verbatim from it. This blog goes beyond the content of the community tutorial and includes setting up subnetworks as well as perform any-to-any testing of the network setup.

Please note upfront that this is a step-by-step manual setup all the way in case you want to understand and to execute all the steps required yourself. If you are looking for an automatic setup, this blog is not for you. A post setting up a VPN using only using user interfaces is here, for example.

Setup phases

This blog divides up the setup into the following phases:

- **Project setup**. Create a project in Google Cloud and create a project in AWS. This is not further explained here — the instructions assume you have one project in each cloud setup and that you have the permissions to set up VPNs, subnets, firewall rules, compute engines, and additional resources as needed.

- **VPN configuration**. The initial setup phase is setting up a VPN between Google Cloud and AWS. This is a step-by-step set of instructions.
- **Subnet configuration**. The next phase is setting up subnets as well as firewalls and security settings.
- **Testing**. The final phase is testing of the setup by creating VMs in Google Compute Engine as well as EC2 for any-to-any ping testing.

After a successful testing phase the network setup is ready as network setup for multi-cloud use cases and workloads.

Notation

On the Google Cloud side `glcoud` commands are shown. Variable values that you have to select are denoted as `[…]`. They are introduced as needed at the first time of use. You could replace those variables consistently by a global text replace in order to prepare all commands consistently.

On the AWS side user interface instructions are provided and are mostly taken verbatim from Google Cloud HA VPN interoperability guide for AWS with sometimes small modifications (for clarification) that are explicitly marked as `[[...]]`.

It is recommended that you note down all chosen values of variables as well as names/values as typed into the various user interfaces to keep a record of the details. Here is one example of this approach: Configuration parameters and values. Another approach is creating a spreadsheet where you record all settings and variable values.

VPN configuration

In the Google Cloud project

- Create a network (GCP: Google Cloud Platform):

```
gcloud compute networks create [GCP_NETWORK_NAME] \
--subnet-mode custom \
--bgp-routing-mode global
```

- As a best practice, delete the default network in the project. This deletion is not required, but prevents accidental configuration of the incorrect network. You might have to delete several firewall rules first if the Cloud Shell indicates that:

```
gcloud compute networks delete default

// in case you need to delete firewall rules:
gcloud compute firewall-rules delete <firewall-rule-name>
```

- Add two subnets:

```
gcloud compute networks subnets create [GCP_SUBNET_1] \
--network [GCP_NETWORK_NAME] \
--region [GCP_REGION_SUBNET_1] \
--range [GCP_RANGE_SUBNET_1]

gcloud compute networks subnets create [GCP_SUBNET_2] \
--network [GCP_NETWORK_NAME] \
--region [GCP_REGION_SUBNET_2] \
--range [GCP_RANGE_SUBNET_2]
```

- Add a VPN gateway:

```
gcloud compute vpn-gateways create [GCP_VPN_GATEWAY] \
--network [GCP_NETWORK_NAME] \
--region [GCP_REGION_SUBNET_1]
```

- Add a router:

```
gcloud compute routers create [GCP_ROUTER] \
--region [GCP_REGION_SUBNET_1] \
--network [GCP_NETWORK_NAME] \
--asn [GCP_ASN]
```

In the AWS project

Note: Additions or modifications of mine in cited text are indicated by `[[...]]` so that any deviation from Google Cloud HA VPN interoperability guide for AWS is explicitly marked.

- Create a VPC with default tenancy
- Create an AWS [[virtual private]] gateway and attach it to the VPC:

```
1. In the AWS dashboard, under Virtual Private Network, select
Virtual Private Gateways.

2. Click Create Virtual Private Gateway

3. Enter a Name for the gateway.

4. Select Custom ASN and give the gateway an AWS ASN that doesn't
conflict with the [GCP_ASN].

5. Click Create Virtual Private Gateway. Click Close.

6. Attach the gateway to the AWS VPC by selecting the gateway you
just created, clicking Actions, Attach to VPC.

7. Pull down the menu and select a VPC for this gateway and click
Yes, Attach.

8. Note the ID of the gateway for the steps that follow.
```

- Create a site-to-site connection and customer gateway:

```
1. In the AWS dashboard, under Virtual Private Network select Site-
to-site VPN connections.

2. Click Create VPN connection.

3. Enter a Name for the connection.

4. Pull down the Virtual Private Gateway menu and select the ID of
the virtual private gateway you just created.

5. Under Customer gateway select New.
```

```
6. For IP address, use the public IP address that was automatically
generated for Interface: 0 [[label of the interface in the GCP
console]] of the HA VPN gateway you created earlier [[in GCP]].

7. For BGP ASN, use the value for [GCP_ASN].

8. Under Routing options make sure that dynamic is selected.

9. For Tunnel 1 and Tunnel 2, specify the Inside IP CIDR for your AWS
VPC network, which is a BGP IP address from a /30 CIDR in the
169.254.0.0/16 range. For example, 169.254.1.4/30. This address must
not overlap with the BGP IP address for the GCP side. When you
specify the pre-shared key, you must use the same pre-shared key for
the tunnel from the HA VPN gateway side. If you specify a key, AWS
doesn't autogenerate it.

10. Click Create VPN connection.
```

- Create the second site-to-site connection (the cited text states also to create a second AWS Gateway, but that is impossible. So I added `[[not:...]]` to indicate that this should not be done):

```
Repeat the above steps to create a second [[not: AWS Gateway,]]
site-to-site connection, and customer gateway, but use the IP address
generated for HA VPN gateway Interface: 1 [[(label of the interface
in the GCP console)]] instead. Use the same [GCP_ASN].
```

- Download the AWS configuration settings for each site-to-site connection:

```
1. As of this writing, you can download the configuration settings
from your AWS virtual private gateway into a text file that you can
reference when configuring HA VPN. You do this after you've
configured the HA VPN gateway and Cloud Router.

2. On the AWS VPC Dashboard screen, in the left navigation bar,
select Site-to-Site VPN Connections.

3. Check the checkbox for the first VPN connection to download.

4. At the top of the screen, click the middle button that reads
Download Configuration. The Download Configuration button provides
only one configuration file, the one for the selected connection.
```

```
5. In the pop-up dialog box, choose vendor: generic. There is no need
to change the rest of the fields, Click the Download button.

6. Repeat the above steps, but choose the second VPN connection to
download the file.
```

In the Google Cloud project

Create an external VPN gateway resource (also called Peer VPN Gateway) as follows.

When you create a GCP external VPN gateway resource for an AWS virtual private gateway, you must create it with four interfaces, as shown in the AWS topology diagram [[in here: Google Cloud HA VPN interoperability guide for AWS]].

Note: For successful configuration, you must use the public IP addresses for the AWS interfaces as referenced in the two AWS configuration files you downloaded earlier. You must also match each AWS public IP address exactly with a specific HA VPN interface. The instructions below describe this task in detail.

Use the following command to create the External VPN gateway resource. Replace the options as noted below:

```
[[The instructions in 1. ... 4. are for the command following 4.:]]

1. For [AWS_GW_IP_0], in the configuration file you downloaded for
AWS Connection 0, under IPSec tunnel #1, #3 Tunnel Interface
Configuration, use the IP address under Outside IP address, Virtual
private gateway.

2. For [AWS_GW_IP_1], in the configuration file you downloaded for
AWS Connection 0, under IPSec tunnel #2, #3 Tunnel Interface
Configuration, use the IP address under Outside IP address, Virtual
private gateway.

3. For [AWS_GW_IP_2], in the configuration file you downloaded for
AWS Connection 1, under IPSec tunnel #1, #3 Tunnel Interface
Configuration, use the IP address under Outside IP address, Virtual
private gateway.

4. For [AWS_GW_IP_3], in the configuration file you downloaded for
AWS Connection 1, under IPSec tunnel #2, #3 Tunnel Interface
Configuration, use the IP address under Outside IP address, Virtual
private gateway.
```

```
gcloud compute external-vpn-gateways create [GCP_PEER_GW_NAME] \
--interfaces \
0=[AWS_GW_IP_0],1=[AWS_GW_IP_1],2=[AWS_GW_IP_2],3=[AWS_GW_IP_3]
```

- Create VPN tunnels on the HA VPN gateway

Create four VPN tunnels, two for each interface, on the HA VPN gateway created previously.

In the following commands to create each tunnel, replace the options as noted in the configuration below:

```
[[The instructions in 1. ... 9. are for the commands following 9.:]]

1. Replace [GCP_TUNNEL_NAME_IF0] with the name of the tunnel to
[[e.g.]] tunnel-a-to-aws-connection-0-ip0.

2. Replace [GCP_TUNNEL_NAME_IF1] with the name of the tunnel to
[[e.g.]] tunnel-a-to-aws-connection-0-ip1.

3. Replace [GCP_TUNNEL_NAME_IF2] with the name of the tunnel to
[[e.g.]] tunnel-a-to-aws-connection-1-ip0.

4. Replace [GCP_TUNNEL_NAME_IF3] with the name of the tunnel to
[[e.g.]] tunnel-a-to-aws-connection-1-ip1.

5. Replace [GCP_PEER_GW_NAME] with a name of the external peer
gateway created earlier.

6. Replace [IKE_VERS] with 2. Although the AWS virtual private
gateway supports IKEv1 or IKEv2, using IKEv2 is recommended. All four
tunnels created in this example use IKEv2.

7. Replace [AWS_SHARED_SECRET_0] through [AWS_SHARED_SECRET_3] with
the shared secret, which must be the same as the shared secret used
for the partner tunnel you create on your AWS virtual gateway. See
Generating a strong pre-shared key for recommendations. You can also
find the shared secrets in the AWS configuration files that you
downloaded earlier.

8. Replace [INT_NUM_0] with the number 0 for the first interface on
the HA VPN gateway you created earlier.

9. Replace [INT_NUM_1] with the number 1 for the second interface on
the HA VPN gateway you created earlier.
```

- Create the tunnel to AWS Connection 0, IP address 0:

```
gcloud compute vpn-tunnels create [GCP_TUNNEL_NAME_IF0] \
--peer-external-gateway [GCP_PEER_GW_NAME] \
--peer-external-gateway-interface 0 \
--region [GCP_REGION_SUBNET_1] \
--ike-version [IKE_VERS] \
--shared-secret [AWS_SHARED_SECRET_0] \
--router [GCP_ROUTER] \
--vpn-gateway [GCP_VPN_GATEWAY] \
--interface [INT_NUM_0]
```

- Create the tunnel to AWS Connection 0, IP address 1

```
gcloud compute vpn-tunnels create [GCP_TUNNEL_NAME_IF1] \
--peer-external-gateway [GCP_PEER_GW_NAME] \
--peer-external-gateway-interface 1 \
--region [GCP_REGION_SUBNET_1] \
--ike-version [IKE_VERS] \
--shared-secret [AWS_SHARED_SECRET_1] \
--router [GCP_ROUTER] \
--vpn-gateway [GCP_VPN_GATEWAY] \
--interface [INT_NUM_0]
```

- Create the tunnel to AWS Connection 1, IP address 0

```
gcloud compute vpn-tunnels create [GCP_TUNNEL_NAME_IF2] \
--peer-external-gateway [GCP_PEER_GW_NAME] \
--peer-external-gateway-interface 2 \
--region [GCP_REGION_SUBNET_1] \
--ike-version [IKE_VERS] \
--shared-secret [AWS_SHARED_SECRET_2] \
--router [GCP_ROUTER] \
--vpn-gateway [GCP_VPN_GATEWAY] \
--interface [INT_NUM_1]
```

- Create the tunnel to AWS Connection 1, IP address 1

```
gcloud compute vpn-tunnels create [GCP_TUNNEL_NAME_IF3] \
--peer-external-gateway [GCP_PEER_GW_NAME] \
```

```
--peer-external-gateway-interface 3 \
--region [GCP_REGION_SUBNET_1] \
--ike-version [IKE_VERS] \
--shared-secret [AWS_SHARED_SECRET_3] \
--router [GCP_ROUTER] \
--vpn-gateway [GCP_VPN_GATEWAY] \
--interface [INT_NUM_1]
```

- Assign BGP IP addresses

Follow the instructions below to assign BGP IP addresses to Cloud Router interfaces and to BGP peer interfaces.

For each VPN tunnel, get the BGP IP addresses and ASNs for both AWS and GCP from the AWS configuration files you downloaded earlier.

Replace the options for the GCP side as noted below:

```
[[The instructions in 1. ... 4. are for the commands following the
next two blocks of instructions:]]

1. For [GCP_BGP_IP_0], in the configuration file you downloaded for
AWS Connection 0, under IPSec tunnel #1, #3 Tunnel Interface
Configuration, use the IP address under Inside IP address, Customer
gateway.

2. For [GCP_BGP_IP_1], in the configuration file you downloaded for
AWS Connection 0, under IPSec tunnel #2, #3 Tunnel Interface
Configuration, use the IP address under Inside IP address, Customer
gateway.

3. For [GCP_BGP_IP_2], in the configuration file you downloaded for
AWS Connection 1, under IPSec tunnel #1, #3 Tunnel Interface
Configuration, use the IP address under Inside IP address, Customer
gateway.

4. For [GCP_BGP_IP_3], in the configuration file you downloaded for
AWS Connection 1, under IPSec tunnel #2, #3 Tunnel Interface
Configuration, use the IP address under Inside IP address, Customer
gateway.
```

- Replace the options for the AWS side as noted below:

```
1. For [AWS_BGP_IP_0], in the configuration file you downloaded for
AWS Connection 0, under IPSec tunnel #1, #3 Tunnel Interface
Configuration, use the IP address under Inside IP address, Virtual
private gateway.

2. For [AWS_BGP_IP_1], in the configuration file you downloaded for
AWS Connection 0, under IPSec tunnel #2, #3 Tunnel Interface
Configuration, use the IP address under Inside IP address, Virtual
private gateway.

3. For [AWS_BGP_IP_2], in the configuration file you downloaded for
AWS Connection 1, under IPSec tunnel #1, #3 Tunnel Interface
Configuration, use the IP address under Inside IP address, Virtual
private gateway.

4. For [AWS_BGP_IP_3], in the configuration file you downloaded for
AWS Connection 1, under IPSec tunnel #2, #3 Tunnel Interface
Configuration, use the IP address under Inside IP address, Virtual
private gateway.
```

- [[For]] [AWS_PEER_ASN] Use the following ASNs under BGP subsection #4 (currently the ASN is the same in all four cases):

```
1. For [GCP_TUNNEL_NAME_IF0], in the AWS configuration file for AWS
Connection 0, use the Virtual Private Gateway ASN under IPSec Tunnel
#1.

2. For [GCP_TUNNEL_NAME_IF1], in the AWS configuration file for AWS
Connection 0, use the Virtual Private Gateway ASN under IPSec Tunnel
#2.

3. For [GCP_TUNNEL_NAME_IF2], in the AWS configuration file for AWS
Connection 1, use the Virtual Private Gateway ASN under IPSec Tunnel
#1.

4. For [GCP_TUNNEL_NAME_IF3], in the AWS configuration file for AWS
Connection 1, use the Virtual Private Gateway ASN under IPSec Tunnel
#2.

gcloud compute routers add-interface [GCP_ROUTER] \
--interface-name [GCP_ROUTER_INTERFACE_NAME_0] \
--vpn-tunnel [GCP_TUNNEL_NAME_IF0] \
--ip-address [GCP_BGP_IP_0] \
--mask-length 30 \
--region [GCP_REGION_SUBNET_1]

gcloud compute routers add-bgp-peer [GCP_ROUTER] \
--peer-name [GCP_BGP_PEER_NAME_0] \
```

```
--peer-asn [AWS_PEER_ASN] \
--interface [GCP_ROUTER_INTERFACE_NAME_0] \
--peer-ip-address [AWS_BGP_IP_0] \
--region [GCP_REGION_SUBNET_1]

gcloud compute routers add-interface [GCP_ROUTER] \
--interface-name [GCP_ROUTER_INTERFACE_NAME_1] \
--vpn-tunnel [GCP_TUNNEL_NAME_IF1] \
--ip-address [GCP_BGP_IP_1] \
--mask-length 30 \
--region [GCP_REGION_SUBNET_1]

gcloud compute routers add-bgp-peer [GCP_ROUTER] \
--peer-name [GCP_BGP_PEER_NAME_1] \
--peer-asn [AWS_PEER_ASN] \
--interface [GCP_ROUTER_INTERFACE_NAME_1] \
--peer-ip-address [AWS_BGP_IP_1] \
--region [GCP_REGION_SUBNET_1]

gcloud compute routers add-interface [GCP_ROUTER] \
--interface-name [GCP_ROUTER_INTERFACE_NAME_2] \
--vpn-tunnel [GCP_TUNNEL_NAME_IF2] \
--ip-address [GCP_BGP_IP_2] \
--mask-length 30 \
--region [GCP_REGION_SUBNET_1]

gcloud compute routers add-bgp-peer [GCP_ROUTER] \
--peer-name [GCP_BGP_PEER_NAME_2] \
--peer-asn [AWS_PEER_ASN] \
--interface [GCP_ROUTER_INTERFACE_NAME_2] \
--peer-ip-address [AWS_BGP_IP_2] \
--region [GCP_REGION_SUBNET_1]

gcloud compute routers add-interface [GCP_ROUTER] \
--interface-name [GCP_ROUTER_INTERFACE_NAME_3] \
--vpn-tunnel [GCP_TUNNEL_NAME_IF3] \
--ip-address [GCP_BGP_IP_3] \
--mask-length 30 \
--region [GCP_REGION_SUBNET_1]

gcloud compute routers add-bgp-peer [GCP_ROUTER] \
--peer-name [GCP_BGP_PEER_NAME_3] \
--peer-asn [AWS_PEER_ASN] \
--interface [GCP_ROUTER_INTERFACE_NAME_3] \
--peer-ip-address [AWS_BGP_IP_3] \
--region [GCP_REGION_SUBNET_1]
```

- Check the status:

```
gcloud compute routers get-status [GCP_ROUTER] \
--region [GCP_REGION_SUBNET_1] \
--format='flattened(result.bgpPeerStatus[].name,
result.bgpPeerStatus[].ipAddress,
result.bgpPeerStatus[].peerIpAddress)'
```

At this point we stop following Google Cloud HA VPN interoperability guide for AWS as that tutorial does not set up subnetworks, VMs or firewall rules for multi-zone subnetworks and any-to-any ping testing across those.

In the AWS project

- In the AWS console under VIRTUAL PRIVATE CLOUD in the section **Route Tables:**

```
1. Select the route table for the VPC you created

2. If it is not named yet, give it a name for ease of identification

3. In the lower part, click the tab called Route Propagation

4. Select Edit route propagation

5. Check the checkbox under Propagate

6. Click Save
```

Subnet configuration

In the AWS project

In AWS subnets are zonal resources, whereas in Google Cloud subnets are regional resources. For an HA deployment, in Google Cloud one subnet is required as all zones in that region are covered. In AWS, each zone in a region needs its own subnet for resources in that zone.

- Create a subnet for each zone in the region of the AWS project (if you want coverage of all zones). In section **Subnets**:

```
1. Click on Create subnet
```

```
2. Fill in the required information
```

In order to test connectivity:

```
1. Create VMs, one in each subnet. In order to ssh into an instance,
it needs a public IP.

2. When setting up VMs, create a new security group as it will have
to be modified specifically for the VPN setup and it allows you to
prevent open access from anywhere.

3. Modify the security group related to the EC2 instances in order to
allow ingress SSH traffic from your laptop IP
```

In addition, an Internet Gateway is needed:
https://docs.aws.amazon.com/vpc/latest/userguide/VPC_Internet_Gateway.html.

- Create an internet gateway. In section **Internet Gateways**:

```
1. Click Create internet gateway

2. Attached the internet gateway to the VPC. Select the internet
gateway and click Attach to VPC in the Actions drop down menu

3. Select the VPC and attach it to the internet gateway
```

- In **Route Tables**:

```
1. Select the route table of the VPC

2. In the tab Subnet Assocations associate all subnetworks to the
routing table of the VPC

3. In the tab Routes add the route from 0.0.0.0/0 to the internet
gateway
```

To allow the AWS EC2 instances to ping each other, the AWS security group requires ingress rules so that ingress from all subnets is allowed. Once you added the ingress rules for the subnets, you can ping each of the instances from each instance using their private IP addresses. For the ping ICMP ingress is sufficient.

To allow the GCP VM instances to ping the AWS EC2 instances, on the AWS security group for the EC2 instances, two ingress rules are to be added, one for each Google Cloud subnetwork.

In total there should be several ingress rules in the security group:

- One each for every AWS subnet
- One each for every GCP subnet
- One for your laptop to access the AWS EC2 VMs

At this point you can ssh into the EC2 instances you created and ping each from the other.

In the Google Cloud project

Two subnets in two different regions were created above. This setup allows resources in two regions to interact with each other in order to e.g. implement a disaster recovery strategy.

- The following firewall rules have to be implemented in order to allow resource communication:

```
gcloud compute firewall-rules create default-allow-ssh \
--network [GCP_NETWORK_NAME] \
--allow tcp:22 \
--source-ranges 0.0.0.0/0

gcloud compute firewall-rules create default-allow-icmp \
--network [GCP_NETWORK_NAME] \
--allow icmp \
--source-ranges 0.0.0.0/0
```

One way to test connectivity right away is to create a VM in each of the two regions, and have them ping each other. If that works, the firewall rules are set up correctly.

Testing

At this point you have ideally configured:

- one VM in each of the GCP subnetworks
- one VM in each of the AWS subnetworks

From a testing perspective each VM should be able to ping itself and each of the other VMs, in Google Cloud as well as AWS (aka, complete mesh reachability).

Summary

The instructions above

- setup a VPN between Google Cloud and AWS
- created a subnet in each of two regions in Google Cloud
- created one or more subnets in different AWS regions based on your choice
- tested the connectivity with VMs in Google Cloud and AWS.

At this point you have a multi-cloud, multi-region and multi-zone setup. You can deploy additional resources into the subnetworks. If you need additional network protocols between the resources, do not forget to setup firewall rules accordingly.

Acknowledgements

I'd like to thank Anibal Santiago (Anibal Santiago) for the thorough review and many comments to improve the accuracy of this content.

Disclaimer

Christoph Bussler is a Solutions Architect at Google, Inc. (Google Cloud). The opinions stated here are my own, not those of Google, Inc.

Thanks to Anibal Santiago.

Networking Cloud Google Cloud Platform AWS