ELEC-E7330 Laboratory Course in Internet Technologies

Site-to-Site VPN with AWS

Contents

Introduction

A VPN (virtual private network) enables us to connect to a remote network from outside securely. It encrypts all the traffics and the connecting device acts like it is in the same private network. It prevents sniffing or other similar type of attacks. There are several types of VPN connection for example, remote access VPN, site-to-site VPN and so on [1]. Each category has their own use-cases. In this lab we will be configuring a site-to-site VPN, which connects two remote sites, and they can share data privately. One use case of this type of VPN is to connect several branches of a big corporation that are in different regions. In our configuration, we will connect a VPC configured in AWS with a private network or other words on-prem network. As in recent times everything has been shifting on cloud, this is an important skill to be able to connect your device which may sit on your premises to the machines in the cloud. For example, it is possible that a company has some kind of legacy applications which are running on premise for a long time and now they want to develop the capability of those applications. So, they put a database on the cloud and now your application has some components in the cloud, and some are in your premise. To connect them securely, we can configure site-to-site VPN and those machines or services will share or access data over the VPN connection. VPC (virtual private cloud) is nothing but a private network in the cloud and it has all the components like physical network e.g., route tables, subnets and so on. To connect a VPC to our private network securely, we will configure a site-to-site VPN [2]. There are some mandatory components that needs to be present to have a working VPN connection and they are specific to AWS. Some of them are:

VPN connections: a VPN connection connect on-prem devices to AWS's virtual private network (VPC)

VPN tunnel: a tunnel passes encrypted data between devices. It creates a link between those networks. In AWS, each VPN connections has two tunnels, one is for regular data transmission and another one is for fall over.

Customer Gateway (CGW): at the end of each tunnel, two devices need to be sat. Those two devices are responsible for data encryption-decryption, secure data transfer and so on. The device which sits on the customer side is the customer gateway and on the other end, virtual private gateway. But in this case, customer gateway is also configured in AWS i.e., it is an AWS component. It provides all the information about the on-prem device configuration for VPN. For details see [2].

Virtual Private Gateway (VGW): this is the endpoint on AWS site and all the data coming in or out for a VPC will go through this gateway. A VGW can be attached to a single VPC.

Customer Gateway Device (CGD): this is the physical device which sits on the edge of your network and have all the configuration for site-to-site VPN. It can be a physical or software-based router and there is a long list of devices that can be used as customer gateway device. Please check [3] for more details.

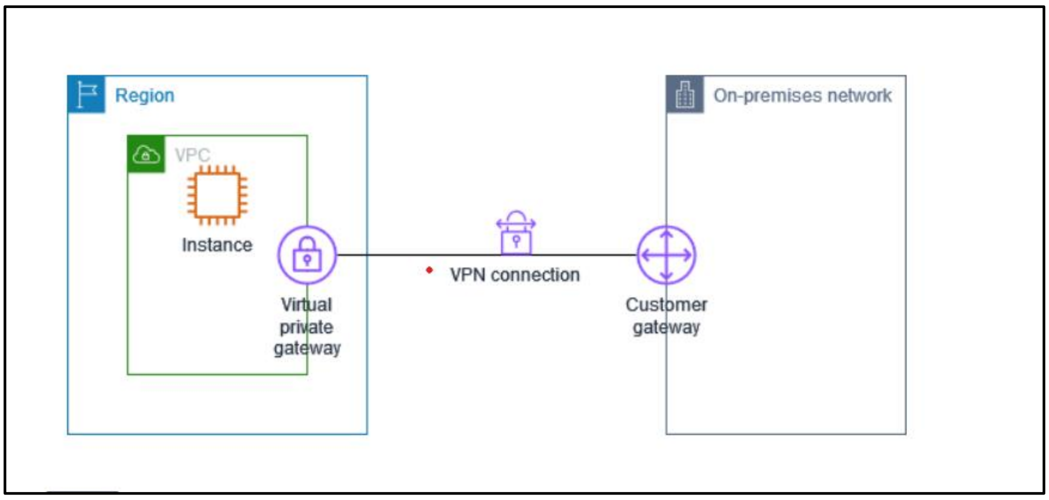


Figure 1: site-to-site VPN topology

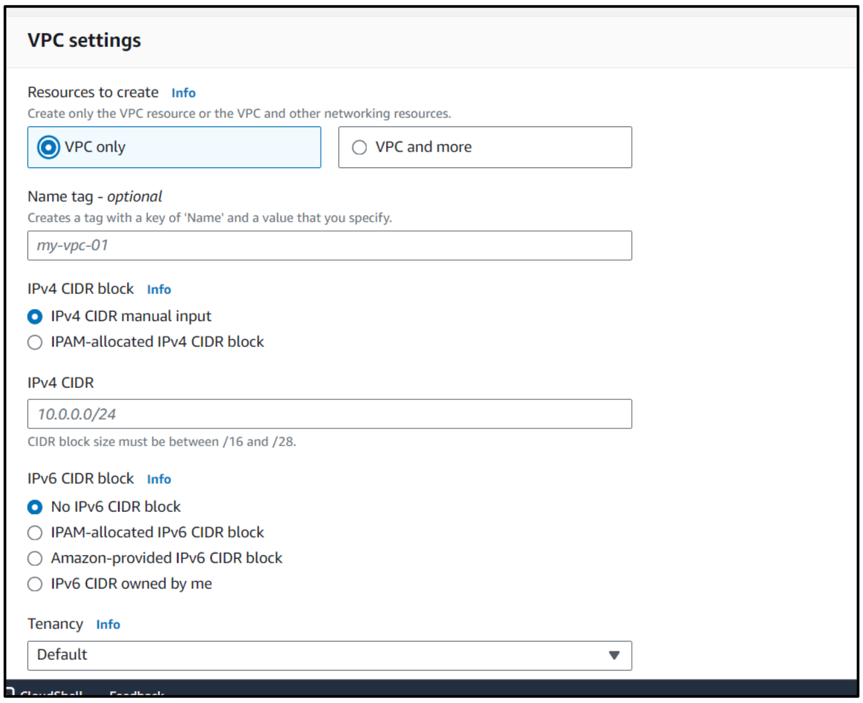
Part 1: Configuring VPN on AWS side.

A detailed instruction has been given on AWS site about how to configure necessary components for site-site-site VPN. To follow the step-by-step guide visit [4]. Also note that we will use static routing throughout this lab.

1.1 Create a new VPC:

At first, create a VPC on AWS. Chose 'VPC only' option when creating the VPC. From the AWS console go services -->Networking & content delivery --> VPC --> create vpc and select the following:

- Name tag: site-to-site vpc
- IPv4 CIDR block (manual input): 10.100.0.0/16
- Tenancy: default
- No IPv6 CIDR block.



VPC settings

Resources to create Info
Create only the VPC resource or the VPC and other networking resources.
[✓ VPC only] VPC and more

Name tag - *optional*
Creates a tag with a key of 'Name' and a value that you specify.
my-vpc-01

IPv4 CIDR block Info
[✓ IPv4 CIDR manual input]
IPAM-allocated IPv4 CIDR block

IPv4 CIDR
10.0.0.0/24
CIDR block size must be between /16 and /28.

IPv6 CIDR block Info
[✓ No IPv6 CIDR block]
IPAM-allocated IPv6 CIDR block
Amazon-provided IPv6 CIDR block
IPv6 CIDR owned by me

Tenancy Info
Default

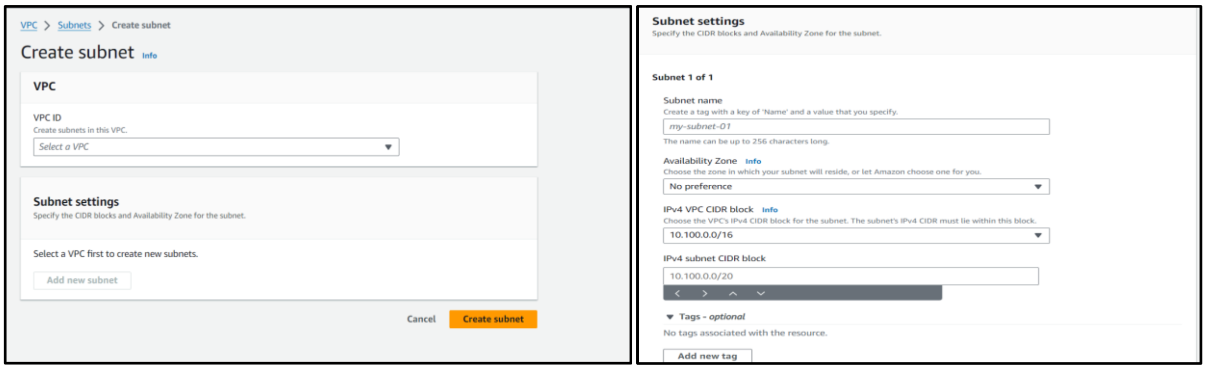
Figure 2: creating a new VPC.

For a step-by-step guide, follow [5].

1.2 Create a new subnet:

In this step, create a new subnet. From the VPC console page, go 'Subnets' → 'Create subnet' → select the ID of the VPC that has created in the previous step and then input the following:

- Subnet name: server-side-subnet
- Availability zone: no preference
- IPv4 VPC CIDR block: will be selected automatically.
- IPv4 subnet CIDR block: 10.100.100.0/24



Create subnet Info

VPC

VPC ID
Create subnets in this VPC.
Select a VPC

Subnet settings
Specify the CIDR blocks and Availability Zone for the subnet.

Select a VPC first to create new subnets.

Subnet settings
Specify the CIDR blocks and Availability Zone for the subnet.

Subnet 1 of 1

Subnet name
Create a tag with a key of 'Name' and a value that you specify.
my-subnet-01
The name can be up to 256 characters long.

Availability Zone Info
Choose the zone in which your subnet will reside, or let Amazon choose one for you.
No preference

IPv4 VPC CIDR block Info
Choose the VPC's IPv4 CIDR block for the subnet. The subnet's IPv4 CIDR must lie within this block.
10.100.0.0/16

IPv4 subnet CIDR block
10.100.0.0/20

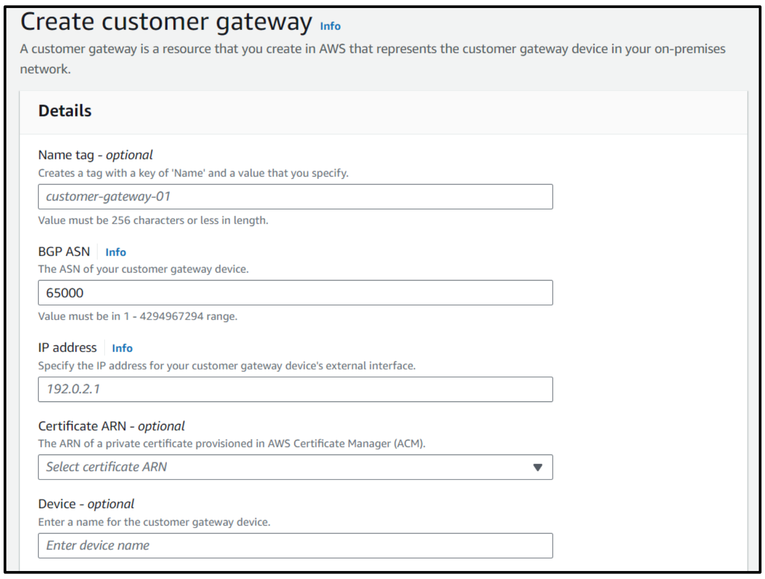
Tags - *optional*
No tags associated with the resource.

Figure 3: creating a new subnet.

1.3 Create a customer gateway (CGW):

On the VPC console page, select 'customer gateways' from the left menu → 'create customer gateway' and then input the following:

- Name tag: customerGW
- BGP ASN: leave the default value.
- IP address: put the public IP address of your on-prem router (e.g., 195.148.124.78)


Create customer gateway Info

A customer gateway is a resource that you create in AWS that represents the customer gateway device in your on-premises network.

Details

Name tag - *optional*
Creates a tag with a key of 'Name' and a value that you specify.
customer-gateway-01
Value must be 256 characters or less in length.

BGP ASN Info
The ASN of your customer gateway device.
65000
Value must be in 1 - 4294967294 range.

IP address Info
Specify the IP address for your customer gateway device's external interface.
192.0.2.1

Certificate ARN - *optional*
The ARN of a private certificate provisioned in AWS Certificate Manager (ACM).
Select certificate ARN

Device - *optional*
Enter a name for the customer gateway device.
Enter device name

Figure 4: Creating a customer gateway.

For more detail instruction, follow [4].

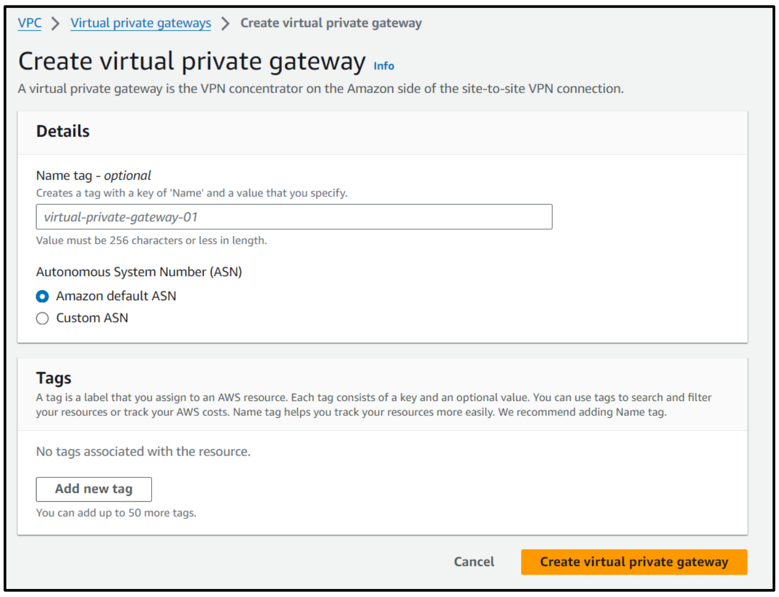
1.4 Create a virtual private gateway (VGW):

On the VPC page, select 'virtual private gateways' from the left panel →'create virtual private gateway' and set the following value:

- Name tag: virtualPrivateGateway
- Autonomous System Number (ASN): Amazon default ASN

After that, you need to attach this gateway to the VPC that you created before i.e., 'site-to-site vpc'. For this,

- Select the vgw that you created.
- Click 'Actions' and select 'attach to a vpc'
- Select the VPC ID for 'site-to-site vpc' and finally, 'attach to VPC'.


VPC > Virtual private gateways > Create virtual private gateway

Create virtual private gateway Info

A virtual private gateway is the VPN concentrator on the Amazon side of the site-to-site VPN connection.

Details

Name tag - *optional*

Creates a tag with a key of 'Name' and a value that you specify.

virtual-private-gateway-01

Value must be 256 characters or less in length.

Autonomous System Number (ASN)

[✓ Amazon default ASN] Custom ASN

Tags

A tag is a label that you assign to an AWS resource. Each tag consists of a key and an optional value. You can use tags to search and filter your resources or track your AWS costs. Name tag helps you track your resources more easily. We recommend adding Name tag.

No tags associated with the resource.

Add new tag

You can add up to 50 more tags.

Cancel   Create virtual private gateway

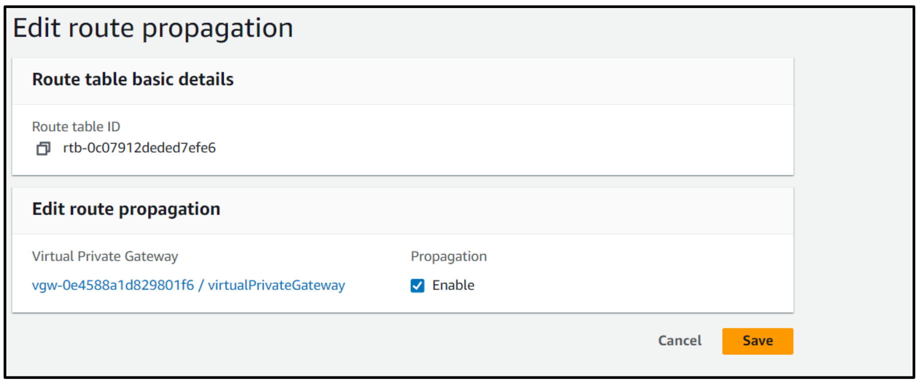
Figure 5: Creating a virtual private gateway (VGW)

For more details, visit [4].

1.5 Configure routing:

To route traffic between your on-prem and AWS network, you need to enable 'route propagation'. By doing this, routes will be automatically created in the routing table to access your on-prem network. To enable route propagation, follow the steps below:

- From the VPC console page, select 'route tables' from the left panel.
- Select the table ID that is associated with the VPC 'site-to-site vpc' i.e. the VPC that you created before
- From 'Actions' select 'Edit route propagation' and select 'Enable'
- And finally, 'save'.


Edit route propagation

Route table basic details

Route table ID
rtb-0c07912deded7efe6

Edit route propagation

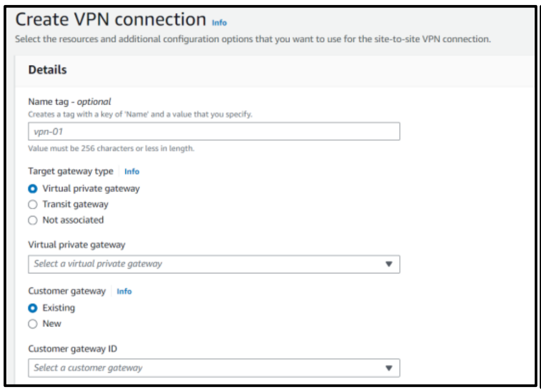
Virtual Private Gateway	Propagation
vgw-0e4588a1d829801f6 / virtualPrivateGateway	✓ Enable

Figure 6: Enabling route propagation

1.6 Create a VPN connection:

So far, we've configured customer and virtual private gateway, enabled route propagation and all the required components. Now we need to create the actual VPN connection for site-to-site VPN. Follow the steps below to create VPN connection:

- Select 'site-to-site VPN connections' from the left panel on the VPC console page, and 'Create VPN connection'
- Name tag: vpn-onprem, Target gateway type: virtual private gateway and select the VGW ID that you created.
- Customer gateway: existing and select the ID for the customer gateway that you created.
- Routing options: static, static IP prefixes: 10.38.0.0/24 or any other private subnet (on-prem) that you want to connect.
- For tunnel options, click 'Tunnel 1 options'
- 'Advanced options for tunnel 1': Edit tunnel 1 options.
- For 'DPD timeout action', select 'none'
- Do this for tunnel 2 also.


Create VPN connection Info

Select the resources and additional configuration options that you want to use for the site-to-site VPN connection.

Details

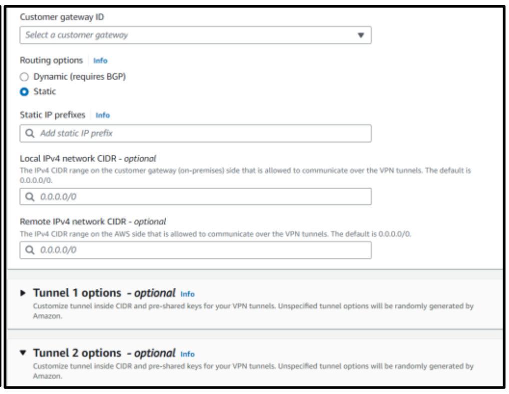
Name tag - *optional*
Creates a tag with a key of 'Name' and a value that you specify.
vpn-01
Value must be 256 characters or less in length.

Target gateway type Info
[✓ Virtual private gateway] Transit gateway Not associated

Virtual private gateway
Select a virtual private gateway

Customer gateway Info
[✓ Existing] New

Customer gateway ID
Select a customer gateway


Customer gateway ID
Select a customer gateway

Routing options Info
Dynamic (requires BGP) [✓ Static]

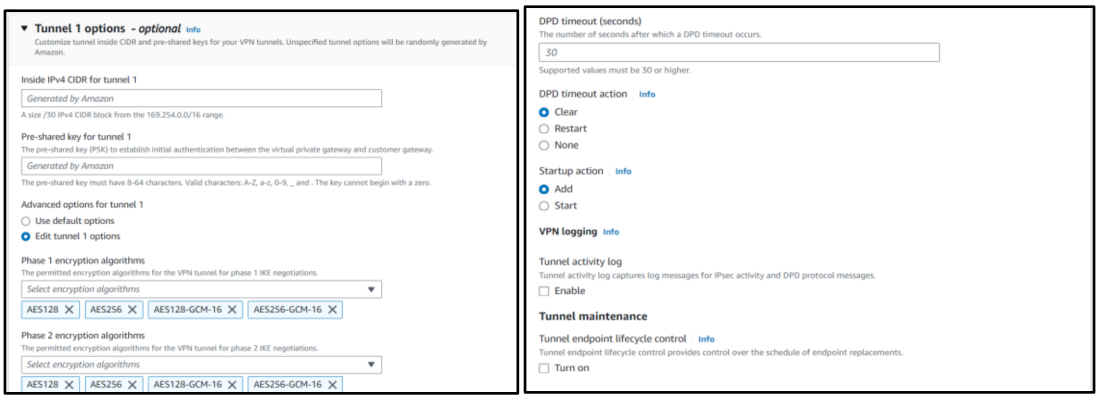
Static IP prefixes Info
Add static IP prefix

Local IPv4 network CIDR - *optional*
The IPv4 CIDR range on the customer gateway (on-premises) side that is allowed to communicate over the VPN tunnels. The default is 0.0.0.0/0.
0.0.0.0/0

Remote IPv4 network CIDR - *optional*
The IPv4 CIDR range on the AWS side that is allowed to communicate over the VPN tunnels. The default is 0.0.0.0/0.
0.0.0.0/0

Tunnel 1 options - *optional* Info
Customize tunnel inside CIDR and pre-shared keys for your VPN tunnels. Unspecified tunnel options will be randomly generated by Amazon.

Tunnel 2 options - *optional* Info
Customize tunnel inside CIDR and pre-shared keys for your VPN tunnels. Unspecified tunnel options will be randomly generated by Amazon.



▾ Tunnel 1 options - *optional* Info
Customize tunnel inside CIDR and pre-shared keys for your VPN tunnels. Unspecified tunnel options will be randomly generated by Amazon.

Inside IPv4 CIDR for tunnel 1
Generated by Amazon
A size /30 IPv4 CIDR block from the 169.254.0.0/16 range.

Pre-shared key for tunnel 1
The pre-shared key (PSK) to establish initial authentication between the virtual private gateway and customer gateway.
Generated by Amazon
The pre-shared key must have 8-64 characters. Valid characters: A-Z, a-z, 0-9, _ and . The key cannot begin with a zero.

Advanced options for tunnel 1
○ Use default options
● Edit tunnel 1 options

Phase 1 encryption algorithms
The permitted encryption algorithms for the VPN tunnel for phase 1 IKE negotiations.
Select encryption algorithms
AES128 ✕ AES256 ✕ AES128-GCM-16 ✕ AES256-GCM-16 ✕

Phase 2 encryption algorithms
The permitted encryption algorithms for the VPN tunnel for phase 2 IKE negotiations.
Select encryption algorithms
AES128 ✕ AES256 ✕ AES128-GCM-16 ✕ AES256-GCM-16 ✕

DPD timeout (seconds)
The number of seconds after which a DPD timeout occurs.
30
Supported values must be 30 or higher.

DPD timeout action Info
● Clear
○ Restart
○ None

Startup action Info
● Add
○ Start

VPN logging Info

Tunnel activity log
Tunnel activity log captures log messages for IPsec activity and DPD protocol messages.
☐ Enable

Tunnel maintenance

Tunnel endpoint lifecycle control Info
Tunnel endpoint lifecycle control provides control over the schedule of endpoint replacements.
☐ Turn on

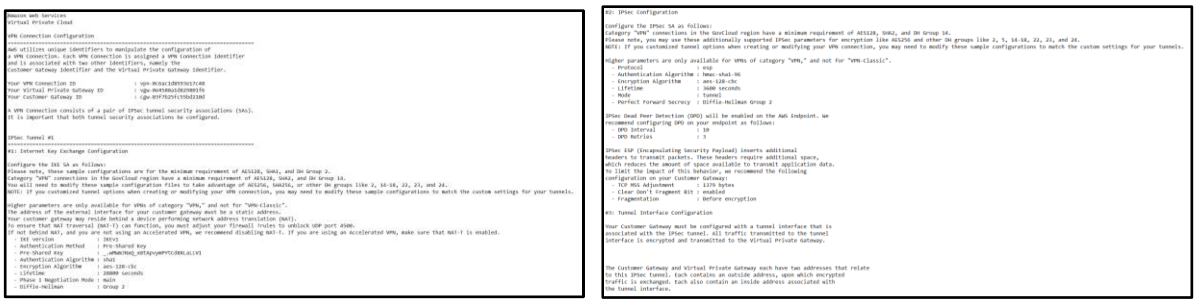
Figure 7: Creating a VPN connection.

For more details, visit [4].

1.7 Downloading the configuration for on-prem device:

Now, everything is ready on the AWS side, and we need to configure the on-prem device i.e., the customer gateway device to connect to the AWS's site-to-site VPN. To configure the customer device, we need all the configuration value that has been used for AWS. For example, encryption, hashing algorithm, IP address for the tunnels, and so on. The configuration file needs to be downloaded from the AWS site and then we need to use those configuration value to configure on-prem device. To do this,

- Select the vpn connections that you created in the previous step and select 'Download configuration.'
- For the vendor, select 'Generic' as we are not using any listed device and 'ikev1' for IKE version.
- A text file will be downloaded and thoroughly go through the configuration file. A sample configuration file will look like this:



```
Amazon Web Services
Virtual Private Cloud

VPN Connection Configuration
================================================================================
AWS utilizes unique identifiers to manipulate the configuration of
a VPN Connection. Each VPN Connection is assigned a VPN Connection Identifier
and is associated with two other identifiers, namely the
Customer Gateway Identifier and the Virtual Private Gateway Identifier.

Your VPN Connection ID                  : vpn-0c6ac1d8593e17c48
Your Virtual Private Gateway ID         : vgw-0e4588a1d829801f6
Your Customer Gateway ID                : cgw-03f7b25fc55bd210d

A VPN Connection consists of a pair of IPSec tunnel security associations (SAs).
It is important that both tunnel security associations be configured.


IPSec Tunnel #1
================================================================================
#1: Internet Key Exchange Configuration

Configure the IKE SA as follows:
Please note, these sample configurations are for the minimum requirement of AES128, SHA1, and DH Group 2.
Category "VPN" connections in the GovCloud region have a minimum requirement of AES128, SHA2, and DH Group 14.
You will need to modify these sample configuration files to take advantage of AES256, SHA256, or other DH groups like 2, 14-18, 22, 23, and 24.
NOTE: If you customized tunnel options when creating or modifying your VPN connection, you may need to modify these sample configurations to match the custom settings for your tunnels.

Higher parameters are only available for VPNs of category "VPN," and not for "VPN-Classic".
The address of the external interface for your customer gateway must be a static address.
Your customer gateway may reside behind a device performing network address translation (NAT).
To ensure that NAT traversal (NAT-T) can function, you must adjust your firewall !rules to unblock UDP port 4500.
If not behind NAT, and you are not using an Accelerated VPN, we recommend disabling NAT-T. If you are using an Accelerated VPN, make sure that NAT-T is enabled.
  - IKE version              : IKEv1
  - Authentication Method    : Pre-Shared Key
  - Pre-Shared Key           : _.wMWmJHbQ_X8tApvymPYtCd8XLaLLV1
  - Authentication Algorithm : sha1
  - Encryption Algorithm     : aes-128-cbc
  - Lifetime                 : 28800 seconds
  - Phase 1 Negotiation Mode : main
  - Diffie-Hellman           : Group 2
```

```
#2: IPSec Configuration

Configure the IPSec SA as follows:
Category "VPN" connections in the GovCloud region have a minimum requirement of AES128, SHA2, and DH Group 14.
Please note, you may use these additionally supported IPSec parameters for encryption like AES256 and other DH groups like 2, 5, 14-18, 22, 23, and 24.
NOTE: If you customized tunnel options when creating or modifying your VPN connection, you may need to modify these sample configurations to match the custom settings for your tunnels.

Higher parameters are only available for VPNs of category "VPN," and not for "VPN-Classic".
  - Protocol                 : esp
  - Authentication Algorithm : hmac-sha1-96
  - Encryption Algorithm     : aes-128-cbc
  - Lifetime                 : 3600 seconds
  - Mode                     : tunnel
  - Perfect Forward Secrecy  : Diffie-Hellman Group 2

IPSec Dead Peer Detection (DPD) will be enabled on the AWS Endpoint. We
recommend configuring DPD on your endpoint as follows:
  - DPD Interval             : 10
  - DPD Retries              : 3

IPSec ESP (Encapsulating Security Payload) inserts additional
headers to transmit packets. These headers require additional space,
which reduces the amount of space available to transmit application data.
To limit the impact of this behavior, we recommend the following
configuration on your Customer Gateway:
  - TCP MSS Adjustment       : 1379 bytes
  - Clear Don't Fragment Bit : enabled
  - Fragmentation            : Before encryption

#3: Tunnel Interface Configuration

Your Customer Gateway must be configured with a tunnel interface that is
associated with the IPSec tunnel. All traffic transmitted to the tunnel
interface is encrypted and transmitted to the Virtual Private Gateway.



The Customer Gateway and Virtual Private Gateway each have two addresses that relate
to this IPSec tunnel. Each contains an outside address, upon which encrypted
traffic is exchanged. Each also contain an inside address associated with
the tunnel interface.
```

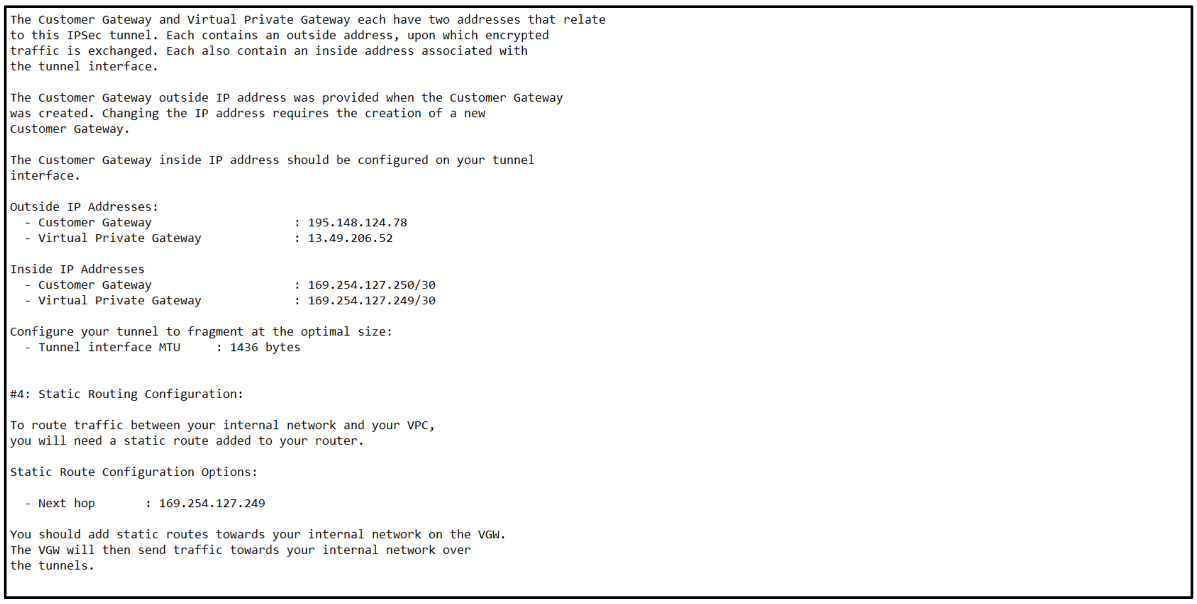

```
The Customer Gateway and Virtual Private Gateway each have two addresses that relate
to this IPSec tunnel. Each contains an outside address, upon which encrypted
traffic is exchanged. Each also contain an inside address associated with
the tunnel interface.

The Customer Gateway outside IP address was provided when the Customer Gateway
was created. Changing the IP address requires the creation of a new
Customer Gateway.

The Customer Gateway inside IP address should be configured on your tunnel
interface.

Outside IP Addresses:
  - Customer Gateway                    : 195.148.124.78
  - Virtual Private Gateway             : 13.49.206.52

Inside IP Addresses
  - Customer Gateway                    : 169.254.127.250/30
  - Virtual Private Gateway             : 169.254.127.249/30

Configure your tunnel to fragment at the optimal size:
  - Tunnel interface MTU     : 1436 bytes


#4: Static Routing Configuration:

To route traffic between your internal network and your VPC,
you will need a static route added to your router.

Static Route Configuration Options:

  - Next hop       : 169.254.127.249

You should add static routes towards your internal network on the VGW.
The VGW will then send traffic towards your internal network over
the tunnels.
```

Figure 8: Example configuration for customer gateway device.

For more details, visit [4].

1.8 Connecting VPC to the internet

To access the internet from the VPC, we need to connect an internet gateway to the VPC. Otherwise, any machine that is configured in this VPC, will not be able to access internet for further processing. For example, to test the VPN connectivity, we will configure a database on AWS and for this, we would need an internet connection. Let's create an internet gateway and connect it to our VPC 'site-to-site vpc'.

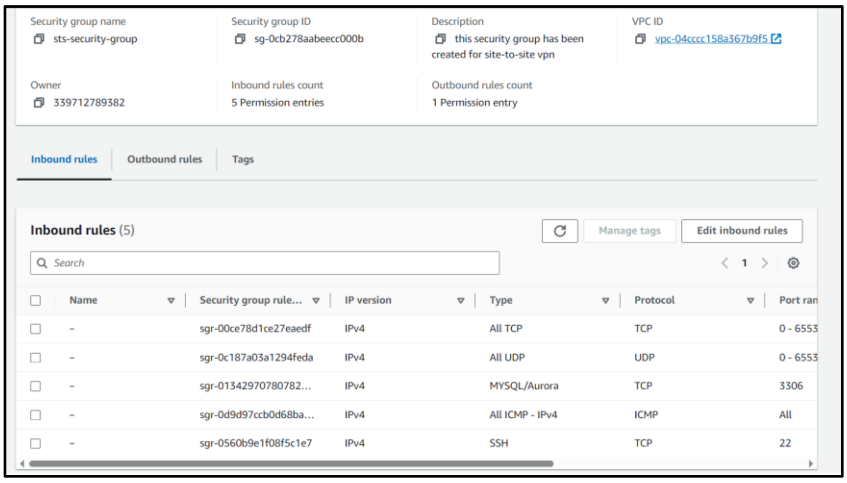
- From the 'services' go to the 'VPC' page
- Select 'Internet gateways' from the left panel and then 'create internet gateway'.
- Name tag: sts-gw and click 'create internet gateway'.
- In the next page, at the top right corner, select 'Attach to a VPC' and select the ID for 'site-to-site vpc'.
- Now our vpc has internet connection.

1.9 Configure and connect to an EC2 instance.

To test our site-to-site VPN connection, we would need a database running on AWS and a webserver running in the private network. To configure the database, we first need an EC2 instance, where we will configure mysql database later. Let's spin up an EC2 instance first by following the steps:

- Go 'services' → 'compute' → 'EC2' → 'Launch instance'.
- Name tag: database-1, select ubuntu image and instance type 't3.micro'.
- Generate a new ssh key pair with rsa algorithm and save the .pem file, which you will need later for connecting to the instance.
- In the network settings, click 'Edit' and select the 'site-to-site vpc' and the subnet will be selected automatically.

- Auto-assign public IP: enable.
- For the firewall, create a new security group and add inbound rule so that you can access the following service: ssh, icmp (for pinging), MYSQL (port 3306). For testing purpose, it is acceptable that you keep your instance wide open for numerous services and ports. But in practice, one should open only the required ports and services. Your security group rule should look something like this:



Security group name	Security group ID	Description	VPC ID
sts-security-group	sg-0cb278aabeecc000b	this security group has been created for site-to-site vpn	vpc-04cccc158a367b9f5
Owner	Inbound rules count	Outbound rules count	
339712789382	5 Permission entries	1 Permission entry	

Inbound rules (5)

Name	Security group rule...	IP version	Type	Protocol	Port ran
-	sgr-00ce78d1ce27eaedf	IPv4	All TCP	TCP	0 - 6553
-	sgr-0c187a03a1294feda	IPv4	All UDP	UDP	0 - 6553
-	sgr-01342970780782...	IPv4	MYSQL/Aurora	TCP	3306
-	sgr-0d9d97ccb0d68ba...	IPv4	All ICMP - IPv4	ICMP	All
-	sgr-0560b9e1f08f5c1e7	IPv4	SSH	TCP	22

Figure 9: Example of security group inbound rule.

- Now click ‘Launch instance’
- Select the instance that you configured and click ‘Connect’ from the upper right corner.
- Select ‘connect using EC2 instance connect’ and ‘Connect’.
- You can also connect to the instance from your terminal, and this is a convenient way. For this, note down the public IP of the instance and from your terminal, run ***‘sudo ssh -i <.pem file name> ubuntu@<public-ip>’***
- Now this instance is ready to install and set up mysql database, which is described in the subsequent section.

Part 2: Configuring VPN on Vyos (on-prem)

2.1 Configuration of Vyos

In this lab, we will Vyos software-based router as our customer gateway device and this router will sit behind a NAT device. Before configuring the site-to-site VPN on Vyos, we need to set up the Vyos on APU at the first place. You will be provided with an APU device and that APU device is pre-installed with VMware esxi. To setup a Vyos router on the APU, do the following steps:

- ✓ Download the iso file from the vyos site.
- ✓ Upload the iso file to the APU and spin up a virtual machine using that iso file.
- ✓ Create two port groups and two virtual switches, named them with public and private (e.g. ‘pub_switch/pub_port’) and attach them to physical NICs (e.g. VMNIC 1, VMNIC 2). The default vSwitch is attached to VMNIC 0 and it works as a management port. Use VMNIC 1 as public gateway and VMNIC 2 as private. Attach VMNIC 1 to the pub_switch as an uplink and VMNIC 2 to the priv_switch.

- ✓ Add two additional network adapters to the machine (which you created earlier. For example, pub_port and priv_port). The default adapter should be attached to the ‘VM Network’.
- ✓ When the machine is ready, login using the default credentials ‘vyos/vyos’ unlike other system, vyos will not be installed until you run ‘install image’ and follow the successive wizard. After the installation, reboot the system. For detailed instruction follow [6].
- ✓ Now, configure the interfaces eth1 and eth2. ‘eth1’ will be used for internet access (public) and set this interface from 10.38.0.x/24 and ‘eth2’ from 192.168.254.x/24.
- ✓ Use the following command to configure those interfaces:

  configure
  set interfaces ethernet ethx address 10.38.0.x/24
  set service ssh
  set protocols static route 0.0.0.0/0 next-hop 10.38.0.254 # this command set default gateway.
  commit
  save
- ✓ Now you can use ssh to connect to the ‘vyos’ machine using 10.38.0.x address.

2.2 Site-to-site configuration:

In step 1.7, we downloaded the necessary configuration for our gateway device i.e., vyos router from AWS. Now we need to configure the site-to-site VPN according to those suggested value. Especially, we need to configure these settings:

1. IKE SA (phase 1)
2. IPSEC SA (phase 2)
3. Tunnel interface

For a step-by-step instruction about configuring VPN on vyos, follow [7]. Please note that from the configuration file, it is recommended to configure two tunnels and two pair of IP address has been given for this purpose. But in our case, **we will configure a single tunnel**. Also note that, for each tunnel, we have two types of IP address:

i. Inside IP address: these are the private IPs that must be set up for the tunnel interface. One IP is for the customer side and the other one is for AWS site or virtual private gateway.
ii. Outside IP address: these are the public IPs. One is the external IP of the NAT device in your private network and the other one is provide by AWS.

- ▪ Login to the vyos router first and enter configuration mode by running ‘conf’ and run the following commands:

Authentication setting:

set vpn ipsec interface eth1

set vpn ipsec authentication psk vgw-0e4588a1d829801f6 id '10.38.0.149'

set vpn ipsec authentication psk vgw-0e4588a1d829801f6 id '13.49.206.52'

set vpn ipsec authentication psk vgw-0e4588a1d829801f6 secret '_.wMWmJNxQ_x8tApvymPYtCd8XLaLLV1'

[‘vgw-0e4588a1d829801f6’ = vpn peer name; you can give a descriptive name here.

’10.38.0.149’ = IP address of the vyos router’s eth1 interface

’13.49.206.52’ = IP address from AWS site, you need to take it from the configuration file.

'_.wMWmJNxQ_x8tApvymPYtCd8XLaLLV1' = this is a pre-shared secret, it can be found from the configuration file]

IKE setting:

```
set vpn ipsec ike-group IKE-GROUP key-exchange ikev1

set vpn ipsec ike-group IKE-GROUP lifetime 28800

set vpn ipsec ike-group IKE-GROUP proposal 1 dh-group 2

set vpn ipsec ike-group IKE-GROUP proposal 1 encryption aes128

set vpn ipsec ike-group IKE-GROUP proposal 1 hash sha1

set vpn ipsec ike-group IKE-GROUP mode 'main'

set vpn ipsec ike-group IKE-GROUP dead-peer-detection action restart

set vpn ipsec ike-group IKE-GROUP dead-peer-detection interval '10'

set vpn ipsec ike-group IKE-GROUP dead-peer-detection timeout 30
```

[‘IKE-GROUP’ = is the name of the IKE SA, you can use any descriptive name here.]

IPSEC setting:

```
set vpn ipsec esp-group ESP-GROUP lifetime 3600

set vpn ipsec esp-group ESP-GROUP proposal 1 encryption aes128

set vpn ipsec esp-group ESP-GROUP proposal 1 hash sha1

set vpn ipsec esp-group ESP-GROUP mode 'tunnel'

set vpn ipsec esp-group ESP-GROUP pfs 'enable'
```

[‘ESP-GROUP’ = is the name of ESP setting, any descriptive name can be used here.]

Tunnel 1 setting:

```
set interfaces vti vti1 address 169.254.127.250/30

set vpn ipsec site-to-site peer vgw-0e4588a1d829801f6 authentication mode pre-shared-secret

set vpn ipsec site-to-site peer vgw-0e4588a1d829801f6 authentication local-id '10.38.0.149'

set vpn ipsec site-to-site peer vgw-0e4588a1d829801f6 authentication remote-id
'13.49.206.52'
```

set vpn ipsec site-to-site peer vgw-0e4588a1d829801f6 connection-type initiate

set vpn ipsec site-to-site peer vgw-0e4588a1d829801f6 description ipsec

set vpn ipsec site-to-site peer vgw-0e4588a1d829801f6 local-address '10.38.0.149'

set vpn ipsec site-to-site peer vgw-0e4588a1d829801f6 remote-address '13.49.206.52'

set vpn ipsec site-to-site peer vgw-0e4588a1d829801f6 ike-group IKE-GROUP

set vpn ipsec site-to-site peer vgw-0e4588a1d829801f6 vti bind vti1

set vpn ipsec site-to-site peer vgw-0e4588a1d829801f6 vti esp-group ESP-GROUP

set protocols static route 10.100.100.0/24 next-hop 169.254.127.249

[169.254.127.250/30 = inside ip address for the customer gateway

169.254.127.249 = inside ip address of the virtual private gateway (AWS side)

'13.49.206.52' = public IP of the AWS side]

Don't forget to commit and save your configuration!!

commit # it will take a considerable amount of time.

save

exit

Checking the tunnel state:

After running all those commands, now it's the time to check the connection, whether it's working or not. For this, run the following commands and you must see the status as 'UP'.

show vpn ike sa

show vpn ipsec sa

If you see the connection is up, congratulations! Your set up is correct. You can also check if you can ping the other side of the tunnel, for example the private IP on the AWS side.

2.3 Setting up a Debian machine on APU

To setup the apache webserver, first configure a machine on APU. Follow the similar way as you created the vyos router on the APU.

. Especially consider the following:

- Choose ubuntu server 22.04 iso image.
- Username/password: 'lab/lab'
- In addition to the default network adapter, add another network adapter 'Pub_port' that you created before.

- During the installation, configure the network settings as static ip and put '10.38.0.0/24' as the subnet, '10.38.0.x' as the static ip address and choose '10.38.0.y' (IP address of the vyos router) as the default gateway.
- You can also specify some nameservers like 8.8.8.8, 8.8.4.4, etc.

At this point, you have a running Debian machine in the private network, and you also configured the VPN connection and it's up and running (let's assume). Now from this Debian machine, you should be able to ping all the tunnel interface, both customer and AWS side. You should also be able to ping the EC2 instance with private IP.

ping <tunnel-interface-at-customer-side>

ping <tunnel-interface-at-AWS -side>

ping <private-IP-of-EC2-instance>

If all the pinging is working fine, then all the setup is correct, and you have a working site-to-site vpn. For validating the connection between database and apache server, follow the next section.

Part 3: Validating the VPN connection.

To validate the VPN connection, we need an apache server running on the local network, and a mysql server running on AWS. The mysql server will be configured on an instance in AWS and it must be in the same VPC that is involved in the site-to-site peer connection. On the other hand, the apache server will be run on a Debian machine, which located in the private subnet that is allowed for the site-to-site connection i.e. in the 10.38.0.0/24 network.

3.1 Setting up mysql server [8]:

- Connect and login to the EC2 instance.
- Then run,
  sudo apt update
  sudo apt install mysql-server
  sudo mysql
  mysql>ALTER USER 'root'@'localhost' IDENTIFIED WITH mysql_native_password BY 'lab';
  mysql>exit
  sudo mysql_secure_installation *and follow the wizard.*
- Now login again as the root user but this time use the command,
  mysql -u root -p *and enter the password 'lab'*
  CREATE DATABASE db;
  CREATE USER 'lab'@'%' IDENTIFIED BY 'lab';
  GRANT ALL ON db.* TO 'lab'@'%';
  exit
- Login again by using the new user 'lab'
  mysql -u lab -p *and enter the password 'lab'*
  SHOW DATABASES;
  CREATE TABLE db.todo_list (
  item_id INT AUTO_INCREMENT,
  content VARCHAR(255),
  PRIMARY KEY(item_id)
  );

INSERT INTO db.todo_list (content) VALUES ("My first important item");
SELECT * FROM db.todo_list;
Exit

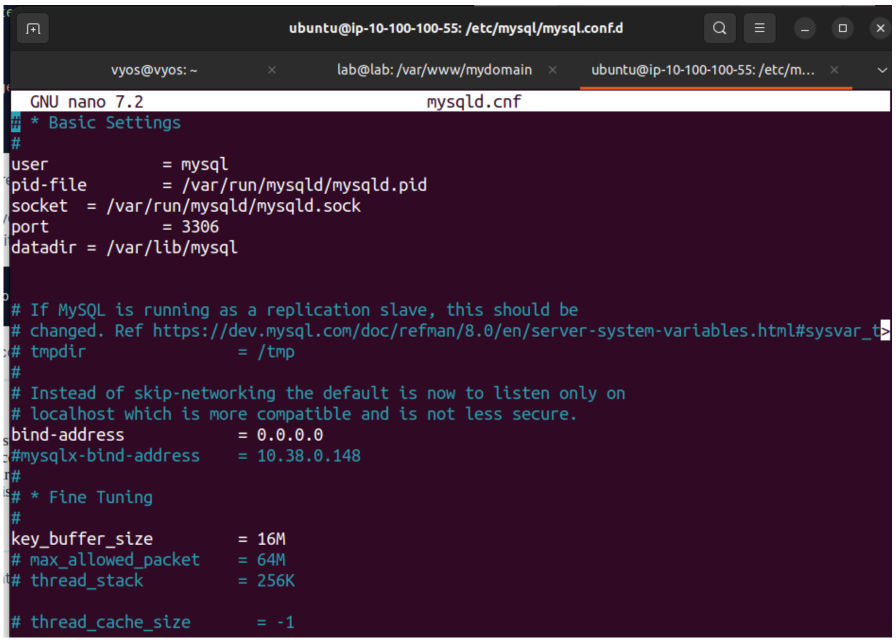
- Restart the mysql server and check the status.
  sudo systemctl restart mysql
  sudo systemctl status mysql

- With this setting, the mysql server will only listen to the localhost. We need to bind the ip address that this server should listen to, in our case, it would be 10.38.0.148. But it is safe to put 0.0.0.0 as the bind address [9]. We also need to enable port 3306. To apply all these settings, open the file mysqld.cnf by running,

  Sudo nano /etc/mysql/mysql.conf.d/mysqld.cnf

  And put the value as following:

  user = mysql
  pid-file = /var/run/mysqld/mysqld.pid
  socket = /var/run/mysqld/mysqld.sock
  port = 3306
  datadir = /var/lib/mysql
  bind-address = 0.0.0.0


ubuntu@ip-10-100-100-55: /etc/mysql/mysql.conf.d
vyos@vyos: ~
lab@lab: /var/www/mydomain
ubuntu@ip-10-100-100-55: /etc/m...

```
  GNU nano 7.2                                 mysqld.cnf
# * Basic Settings
#
user            = mysql
pid-file        = /var/run/mysqld/mysqld.pid
socket  = /var/run/mysqld/mysqld.sock
port            = 3306
datadir = /var/lib/mysql


# If MySQL is running as a replication slave, this should be
# changed. Ref https://dev.mysql.com/doc/refman/8.0/en/server-system-variables.html#sysvar_t>
# tmpdir                = /tmp
#
# Instead of skip-networking the default is now to listen only on
# localhost which is more compatible and is not less secure.
bind-address            = 0.0.0.0
#mysqlx-bind-address    = 10.38.0.148
#
# * Fine Tuning
#
key_buffer_size         = 16M
# max_allowed_packet    = 64M
# thread_stack          = 256K

# thread_cache_size       = -1
```

Figure 10: Configuration file for MYSQL server.

- To make sure that firewall will not create any problem for accessing the mysql server on port 3306, run the following ufw commands:

  sudo ufw allow 22,3306/tcp
  sudo ufw disable
  sudo ufw enable

sudo ufw status

To get more detailed instructions, follow [8] [9].

3.2 Setting up apache server with PHP:

We will run a simple webpage by using apache and php. At first, we will setup a running apache webserver and then will convert it to a dynamic server with php which will be connected to a mysql server. In this part, a detail instruction will be provided for setting up apache and php. The setup for mysql can be found in the previous section. To get a comprehensive guide about setting up apache, and php, visit [8]. Let's set up apache and php by following the steps below:

Installation:

- Login to the instance in your private network e.g. 10.38.0.148
- Then run the following commands to install apache and php.
  sudo apt update
  sudo apt install apache2
  sudo apt install php libapache2-mod-php php-mysql
  php -v
- Now we have apache and php installed in our system.
  sudo systemctl restart apache2
  sudo systemctl status apache2

Creating virtual host:

Now, to enable this server for our custom domain, we need to setup a virtual host. In a single server, if we want to set up several domains, we need to configure virtual host for each of them. Let's assume that our new domain is 'mydomain' and to enable this domain, create a virtual host by following the steps below:

- At first, create a new directory 'mydomain' at /var/www
  sudo mkdir /var/www/mydomain
- Create a configuration file for your new domain. To do this, at first copy the configuration file for the default file and then change the value for your new domain.

  cd /etc/apache2/sites-available
  cp 000-default.conf mydomain.conf

  Now edit the configuration file 'mydomain.conf' and put the value as:

  ServerName mydomain
  ServerAlias www.mydomain
  DocumentRoot /var/www/mydomain
  ErrorLog ${APACHE_LOG_DIR}/error.log
  CustomLog ${APACHE_LOG_DIR}/access.log combined

- Now enable your new site
  sudo a2ensite mydomain

- To disable the default website instead of your custom one, run the following command:

sudo a2dissite 000-default

- Test the configuration and restart apache2.
  sudo apache2ctl configtest
  sudo systemctl reload apache2

- At this point, our custom site 'mydomain' has no webpage to display. We can create a simple webpage and put the file in the document root.

  nano /var/www/mydomain/index.html

and include the following html code:

```
<html>
  <head>
    <title>your_domain website</title>
  </head>
  <body>
    <h1>Hello World!</h1>
    <p>This is the landing page of <strong>your_domain</strong>.</p>
  </body>
</html>
```

Now check the configuration of your custom site by visiting the webpage via lynx.

lynx http://localhost and it will display the above webpage.

N.B. If lynx is not installed in the system, install it first.

Configuring php:

Now as we have the apache server up and running, we can configure php so that apache can handle request for php. We need to replace the index.html file with index.php to server php request. But before that, we need to make sure that index.php get higher priority than index.html otherwise our site will be ended up displaying static index.html page. To do this, we need to change the following settings:

sudo nano /etc/apache2/mods-enabled/dir.conf
and put index.php at the beginning like this:

```
<IfModule mod_dir.c>

        DirectoryIndex index.php index.html index.cgi index.pl index.xhtml index.htm

</IfModule>
```

Restart the apache service.

We will be running a todo app using php. A database has already been created and it contains some data. By using php, we will render those value. To connect the php server to the database running on AWS, do the following:

Create a todo_list.php file.

nano /var/www/mydomain/todo_list.php and set the following value:

```
<?php

$host = "10.100.100.55"; # private ip address of the EC2 instance
$user = "lab";
$password = "lab";
$database = "db";
$table = "todo_list";

try {
  $db = new PDO("mysql:host=$host;port=3306;dbname=$database", $user, $password);
  echo "<h2>TODO</h2><ol>";
  foreach($db->query("SELECT content FROM $table") as $row) {
    echo "<li>" . $row['content'] . "</li>";
  }
  echo "</ol>";
} catch (PDOException $e) {
    print "Error!: " . $e->getMessage() . "<br/>";
    die();
}
```

3.3 Checking the connection:

Now check whether the php server can pull those value that already created on mysql.

lynx http://localhost/todo_list.php and you must see the items in your list.

It means that our php server has successfully connected to the database running on AWS. It also proves that our site-to-site vpn is working as we are accessing the database by using the private ip of the instance.

We can also check whether those two machines are using the VPN connection or not. To check this, run tcpdump on the vyos router’s vti interface and make the request once again from the private network.

tcpdump -I vti7 # run this on vyos

lynx http://localhost/todo_list.php # run this on apache server in the private network.

You must see some traffic between the apache server and mysql server and they are using the tunnel interface to transfer data.

References

[1] "What Are the Different Types of VPN?," paloalto, [Online]. Available: https://www.paloaltonetworks.com/cyberpedia/types-of-vpn. [Accessed 26 June 2024].

[2] "What is AWS Site-to-Site VPN?," AWS, [Online]. Available: https://docs.aws.amazon.com/vpn/latest/s2svpn/VPC_VPN.html. [Accessed 26 June 2024].

[3] "Your customer gateway device," AWS, [Online]. Available: https://docs.aws.amazon.com/vpn/latest/s2svpn/your-cgw.html. [Accessed 26 June 2024].

[4] "Getting started with AWS Site-to-Site VPN," AWS, [Online]. Available: https://docs.aws.amazon.com/vpn/latest/s2svpn/SetUpVPNConnections.html. [Accessed 26 June 2024].

[5] "Create a VPC," AWS, [Online]. Available: https://docs.aws.amazon.com/vpc/latest/userguide/create-vpc.html. [Accessed 26 June 2024].

[6] J. Nguyen, "Quick Guide: VyOS Installation and Setup," Vyos, 12 March 2024. [Online]. Available: https://support.vyos.io/support/solutions/articles/103000093618-quick-guide-vyos-installation-and-setup. [Accessed 26 June 2024].

[7] "Site-to-Site," Vyos, [Online]. Available: https://docs.vyos.io/en/latest/configuration/vpn/site2site_ipsec.html. [Accessed 28 June 2024].

[8] "How To Install Linux, Apache, MySQL, PHP (LAMP) Stack on Ubuntu," 27 February 2024. [Online]. Available: https://www.digitalocean.com/community/tutorials/how-to-install-lamp-stack-on-ubuntu#step-2-installing-mysql. [Accessed 27 June 2024].

[9] "How to Bind a MySQL Server to an IP Address?," 27 December 2021. [Online]. Available: https://www.alibabacloud.com/blog/how-to-bind-a-mysql-server-to-an-ip-address_598396. [Accessed 27 June 2024].

[10] "Site-to-Site," Vyos, [Online]. Available: https://docs.vyos.io/en/latest/configuration/vpn/site2site_ipsec.html. [Accessed 28 June 2024].

What Are the Different Types of VPN? - Palo Alto Networks

What is AWS Site-to-Site VPN? - AWS Site-to-Site VPN (amazon.com)

Your customer gateway device - AWS Site-to-Site VPN (amazon.com)

Getting started with AWS Site-to-Site VPN - AWS Site-to-Site VPN (amazon.com)

Releases · vyos/vyos-rolling-nightly-builds (github.com)

Quick Guide: VyOS Installation and Setup : VyOS Support Portal