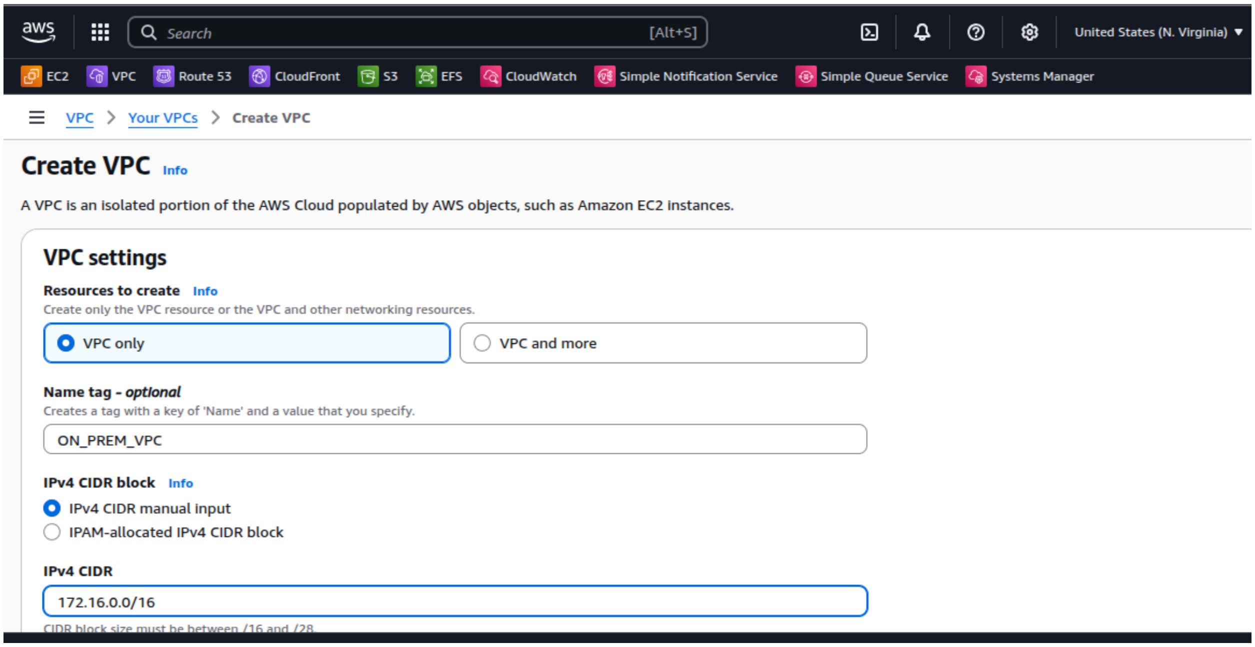
AWS Site-to-Site VPN Setup – Stepwise Explanation

Project Overview

A Site-to-Site VPN allows secure connectivity between your **on-premises network** and an **AWS VPC** over the internet using IPsec tunnels. This setup enables resources in your VPC to communicate with your on-premises network securely.

STEP 1

Create VPC in north virginia region which will act as on-prem infrastructure



Create VPC Info

A VPC is an isolated portion of the AWS Cloud populated by AWS objects, such as Amazon EC2 instances.

VPC settings

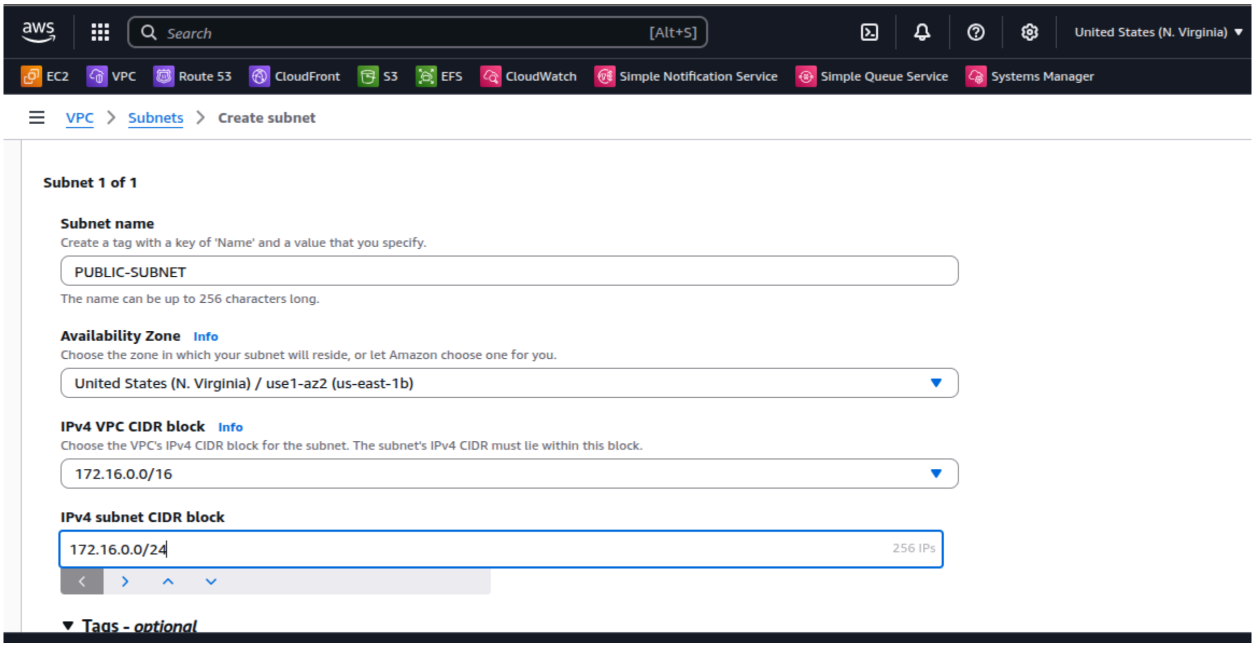
Resources to create Info
Create only the VPC resource or the VPC and other networking resources.
● VPC only ○ VPC and more

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

IPv4 CIDR block Info
● IPv4 CIDR manual input
○ IPAM-allocated IPv4 CIDR block

IPv4 CIDR
172.16.0.0/16
CIDR block size must be between /16 and /28.

STEP 2
Create Public Subnet from where connection will be established between Virginia and mumbai region


aws
Search [Alt+S]
United States (N. Virginia)
EC2 VPC Route 53 CloudFront S3 EFS CloudWatch Simple Notification Service Simple Queue Service Systems Manager
VPC > Subnets > Create subnet

Subnet 1 of 1

Subnet name
Create a tag with a key of 'Name' and a value that you specify.
PUBLIC-SUBNET
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.
United States (N. Virginia) / use1-az2 (us-east-1b)

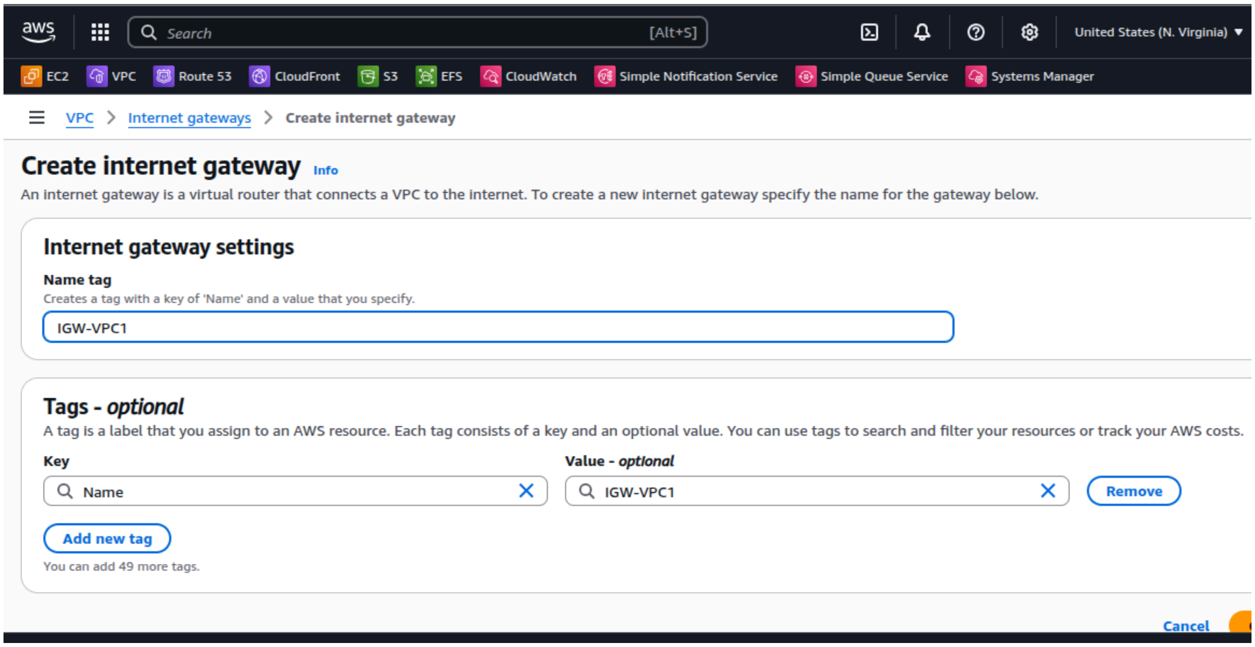
IPv4 VPC CIDR block Info
Choose the VPC's IPv4 CIDR block for the subnet. The subnet's IPv4 CIDR must lie within this block.
172.16.0.0/16

IPv4 subnet CIDR block
172.16.0.0/24 256 IPs

▼ Tags - *optional*

STEP 3

Create Internet gateway so that internet can be attached in that vpc and public subnet created


aws
Search [Alt+S]
United States (N. Virginia)
EC2 VPC Route 53 CloudFront S3 EFS CloudWatch Simple Notification Service Simple Queue Service Systems Manager
VPC > Internet gateways > Create internet gateway

Create internet gateway Info

An internet gateway is a virtual router that connects a VPC to the internet. To create a new internet gateway specify the name for the gateway below.

Internet gateway settings

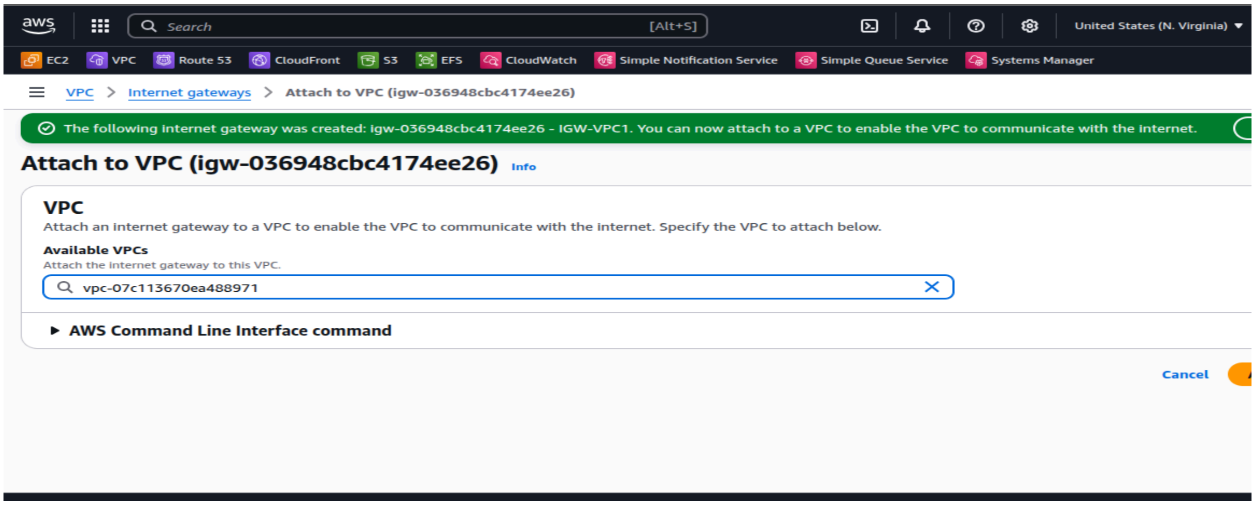
Name tag
Creates a tag with a key of 'Name' and a value that you specify.
IGW-VPC1

Tags - *optional*
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.

Key	Value - *optional*
Name	IGW-VPC1

You can add 49 more tags.

STEP 4
ATTACH INTERNET GATEWAY TO THE VPC



VPC > Internet gateways > Attach to VPC (igw-036948cbc4174ee26)

The following internet gateway was created: igw-036948cbc4174ee26 - IGW-VPC1. You can now attach to a VPC to enable the VPC to communicate with the internet.

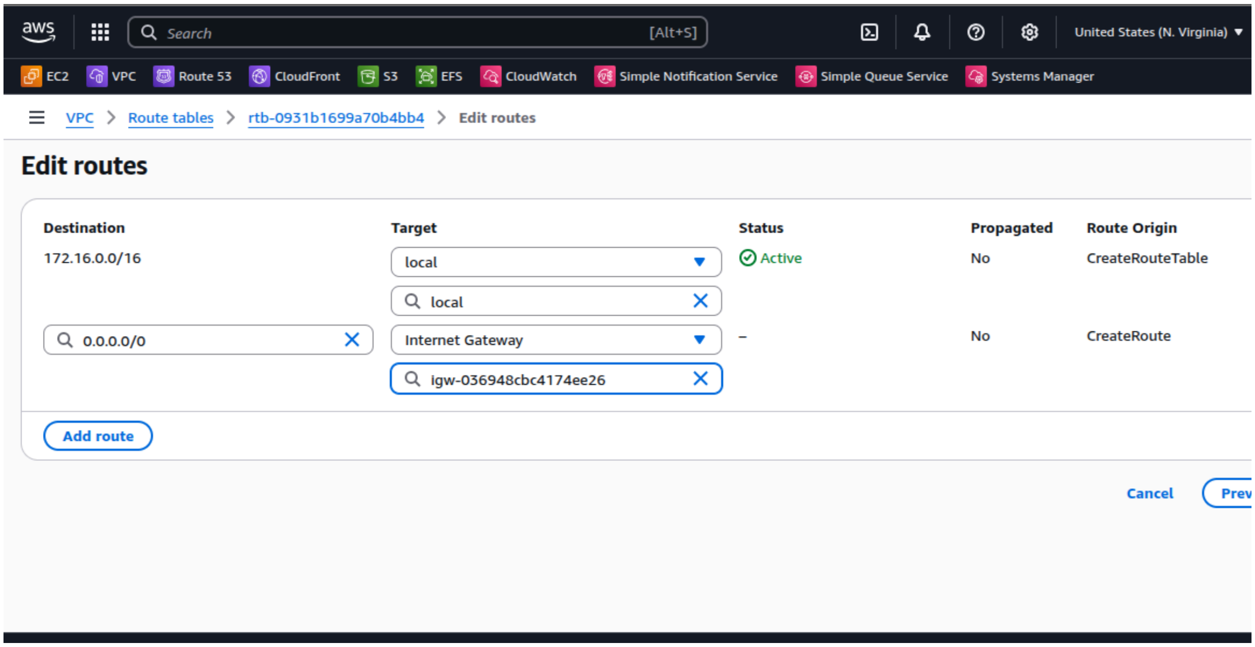
Attach to VPC (igw-036948cbc4174ee26) Info

VPC
Attach an internet gateway to a VPC to enable the VPC to communicate with the internet. Specify the VPC to attach below.

Available VPCs
Attach the internet gateway to this VPC.
vpc-07c113670ea488971

▶ AWS Command Line Interface command

STEP 5
Now edit the route table of public subnet and add internet route as 0.0.0.0/0 through internet gateway

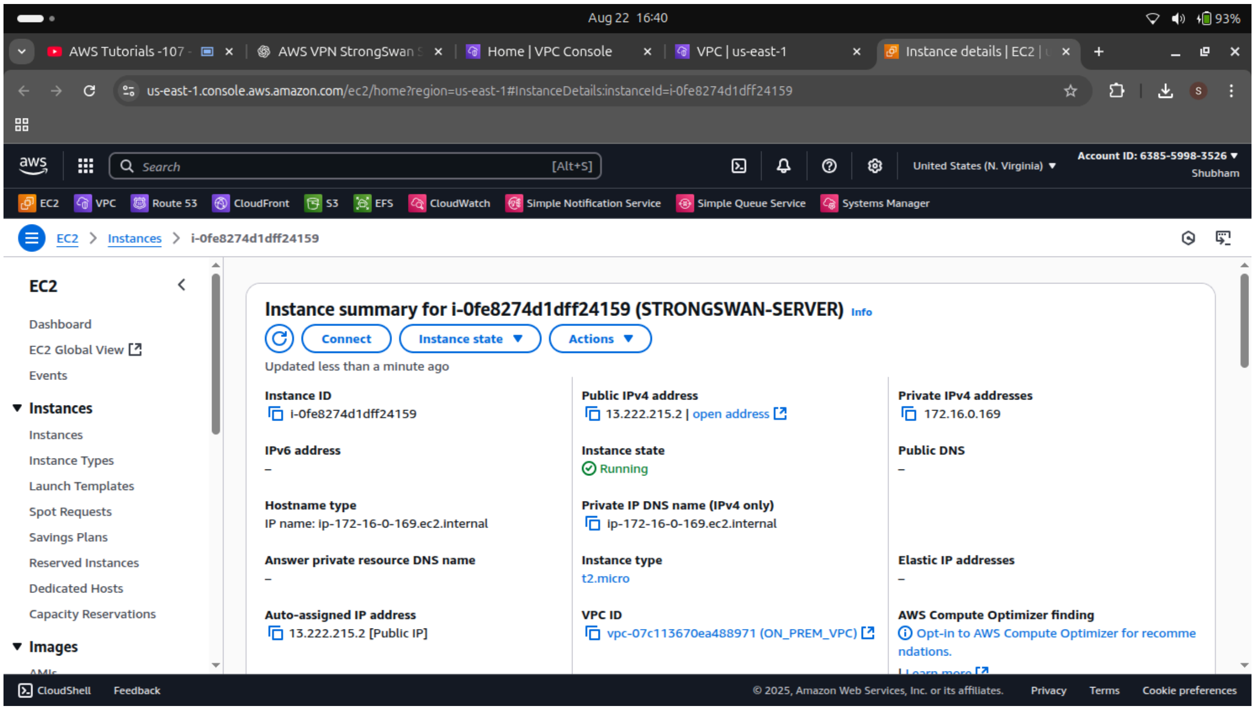


VPC > Route tables > rtb-0931b1699a70b4bb4 > Edit routes

Edit routes

Destination	Target	Status	Propagated	Route Origin
172.16.0.0/16	local local	✓ Active	No	CreateRouteTable
0.0.0.0/0	Internet Gateway igw-036948cbc4174ee26	–	No	CreateRoute

STEP 6
Now we will launch ec2 instance in on-prem virginia region and set it up as vpn server using strong swan ipsec protocol



us-east-1.console.aws.amazon.com/ec2/home?region=us-east-1#InstanceDetails:instanceId=i-0fe8274d1dff24159

EC2 > Instances > i-0fe8274d1dff24159

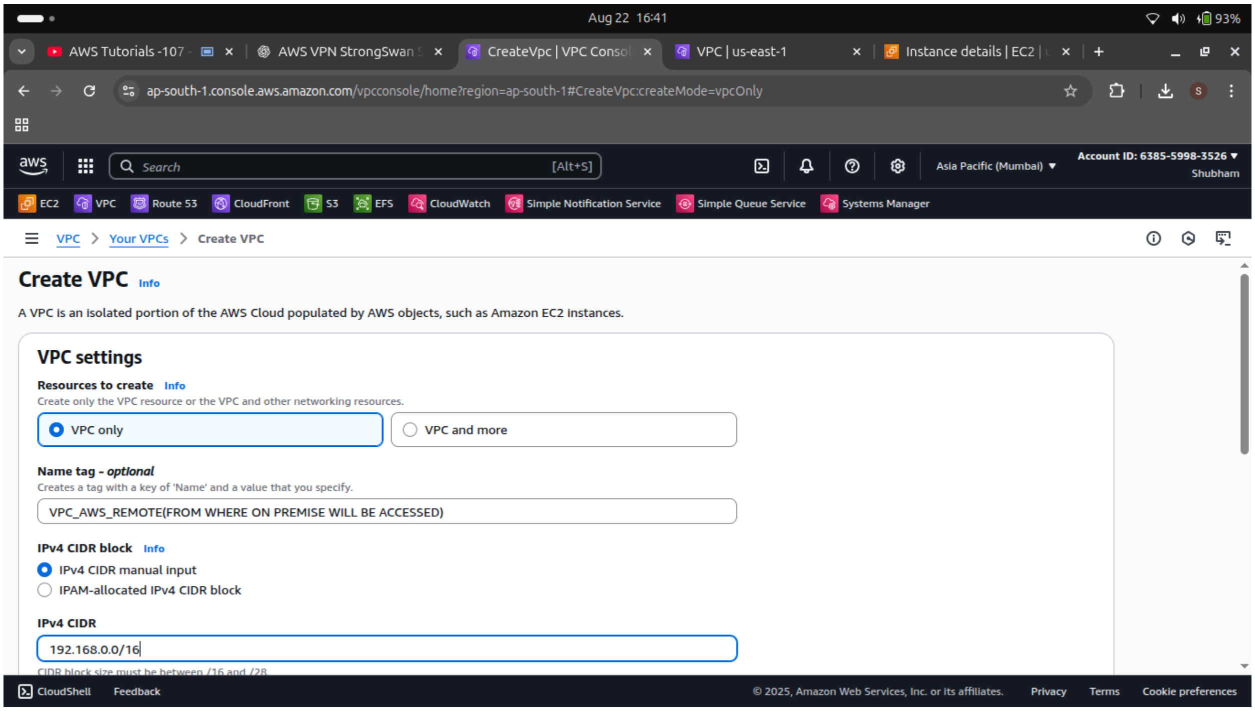
Instance summary for i-0fe8274d1dff24159 (STRONGSWAN-SERVER) Info

Updated less than a minute ago

Instance ID	Public IPv4 address	Private IPv4 addresses
i-0fe8274d1dff24159	13.222.215.2 | open address	172.16.0.169
IPv6 address	**Instance state**	**Public DNS**
–	Running	–
Hostname type	**Private IP DNS name (IPv4 only)**	
IP name: ip-172-16-0-169.ec2.internal	ip-172-16-0-169.ec2.internal	
Answer private resource DNS name	**Instance type**	**Elastic IP addresses**
–	t2.micro	–
Auto-assigned IP address	**VPC ID**	**AWS Compute Optimizer finding**
13.222.215.2 [Public IP]	vpc-07c113670ea488971 (ON_PREM_VPC)	Opt-in to AWS Compute Optimizer for recommendations.

STEP 7

Now we will also create our remote cloud region vpc in mumbai from where we will establish connection



ap-south-1.console.aws.amazon.com/vpcconsole/home?region=ap-south-1#CreateVpc:createMode=vpcOnly

VPC > Your VPCs > Create VPC

Create VPC Info

A VPC is an isolated portion of the AWS Cloud populated by AWS objects, such as Amazon EC2 instances.

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.

VPC_AWS_REMOTE(FROM WHERE ON PREMISE WILL BE ACCESSED)

IPv4 CIDR block Info

● IPv4 CIDR manual input

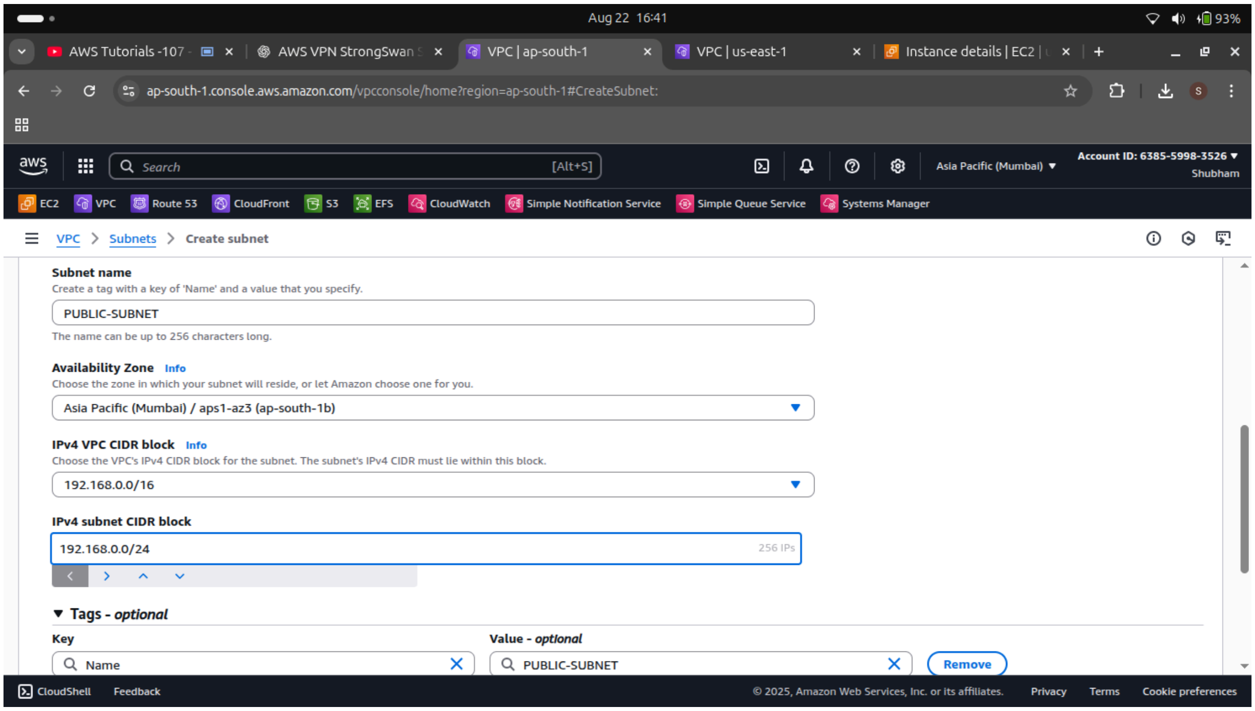
○ IPAM-allocated IPv4 CIDR block

IPv4 CIDR

192.168.0.0/16

CIDR block size must be between /16 and /28.

STEP 8
Now will create public subnet here as well



Subnet name
Create a tag with a key of 'Name' and a value that you specify.
PUBLIC-SUBNET
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.
Asia Pacific (Mumbai) / aps1-az3 (ap-south-1b)

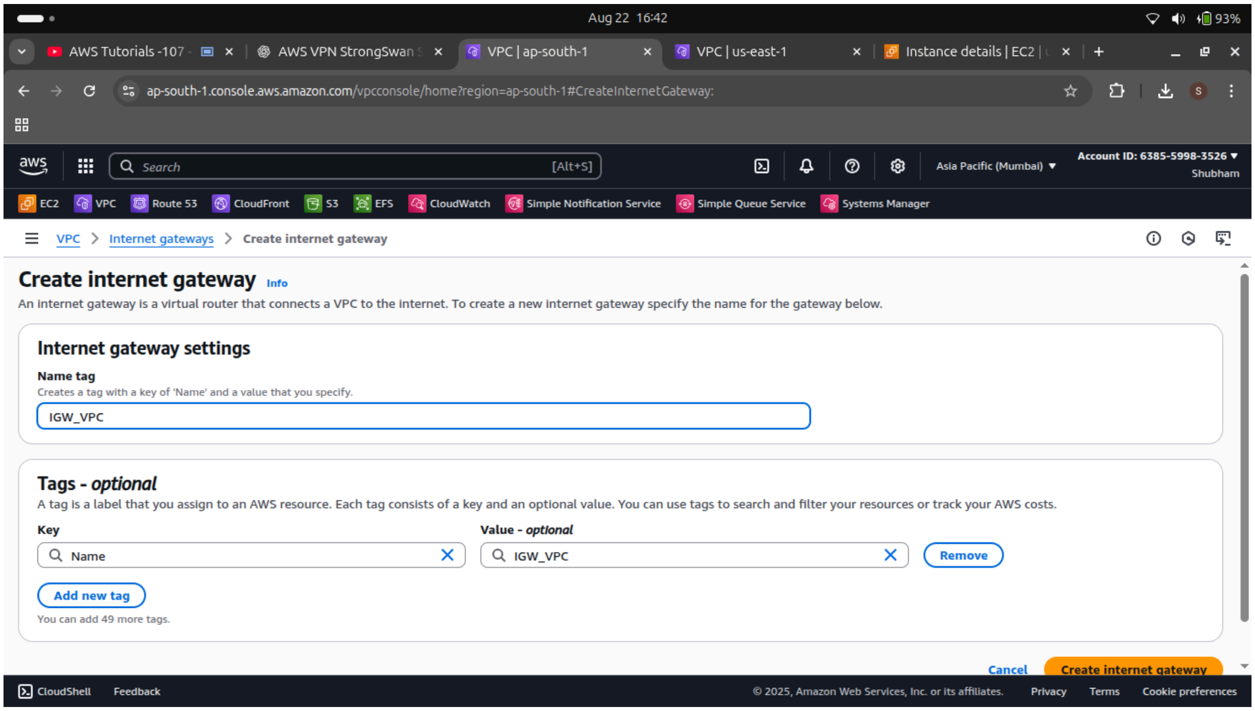
IPv4 VPC CIDR block Info
Choose the VPC's IPv4 CIDR block for the subnet. The subnet's IPv4 CIDR must lie within this block.
192.168.0.0/16

IPv4 subnet CIDR block
192.168.0.0/24 256 IPs

Tags - *optional*

Key	Value - *optional*
Name	PUBLIC-SUBNET

STEP 9
Similarly we will create internet gateway here as well and attach it to mumbai remote vpc



Create internet gateway Info
An internet gateway is a virtual router that connects a VPC to the internet. To create a new internet gateway specify the name for the gateway below.

Internet gateway settings
Name tag
Creates a tag with a key of 'Name' and a value that you specify.
IGW_VPC

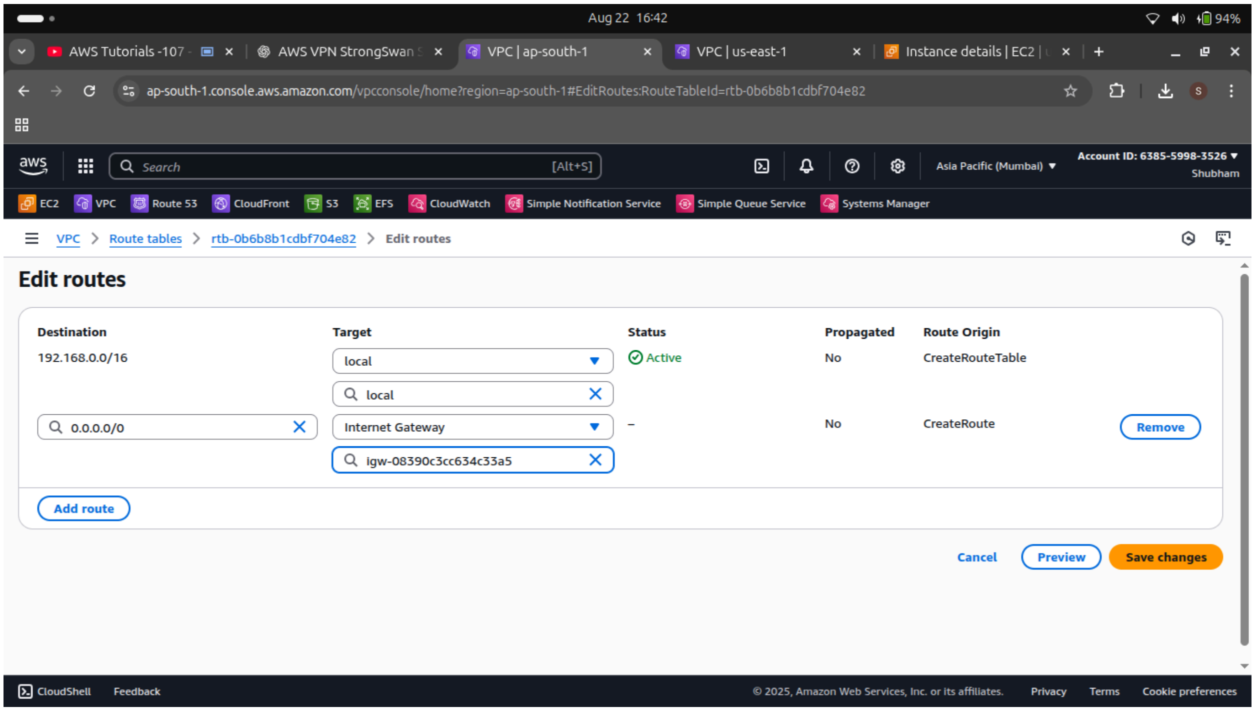
Tags - *optional*
A tag is a label that you assign to an AWS resource. Each tag consists of a key and an optional value. You can use tags to search and filter your resources or track your AWS costs.

Key	Value - *optional*
Name	IGW_VPC

You can add 49 more tags.

STEP 10

Again edit the route table here to allow internet traffic 0.0.0.0/0 to go through internet gateway

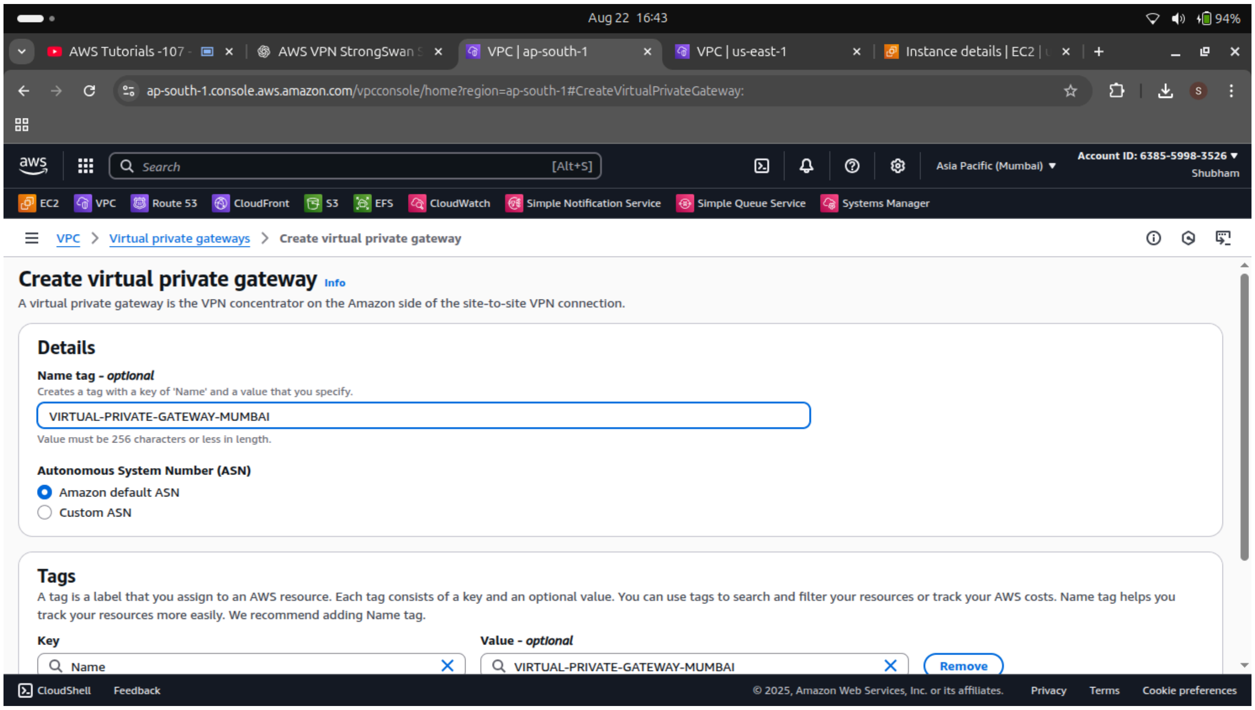


Edit routes

Destination	Target	Status	Propagated	Route Origin	
192.168.0.0/16	local local	Active	No	CreateRouteTable	
0.0.0.0/0	Internet Gateway igw-08390c3cc634c33a5	-	No	CreateRoute	Remove

STEP 11

Now we will create Virtua private gateway in mumbai remote region



Create virtual private gateway

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-MUMBAI

Value must be 256 characters or less in length.

Autonomous System Number (ASN)

● Amazon default ASN

○ Custom ASN

Tags

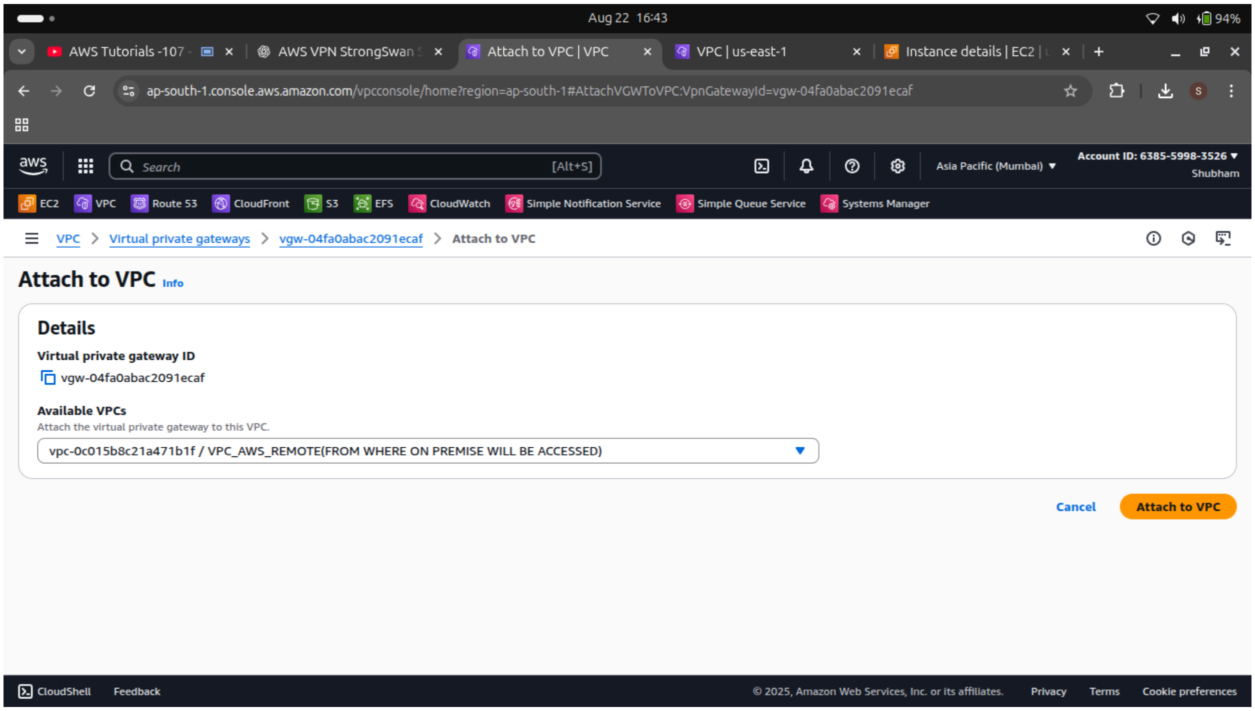
A tag is a label that you assign to an AWS resource. Each tag consists of a key and an optional value. You can use tags to search and filter your resources or track your AWS costs. Name tag helps you track your resources more easily. We recommend adding Name tag.

Key	Value - *optional*
Name	VIRTUAL-PRIVATE-GATEWAY-MUMBAI

Aug 22 16:43
ap-south-1.console.aws.amazon.com/vpcconsole/home?region=ap-south-1#CreateVirtualPrivateGateway:
VPC > Virtual private gateways > Create virtual private gateway
Asia Pacific (Mumbai)
Account ID: 6385-5998-3526
Remove

STEP 12

Now will attach this gateway to our mumbai remote vpc



Attach to VPC

Details

Virtual private gateway ID

vgw-04fa0abac2091ecaf

Available VPCs

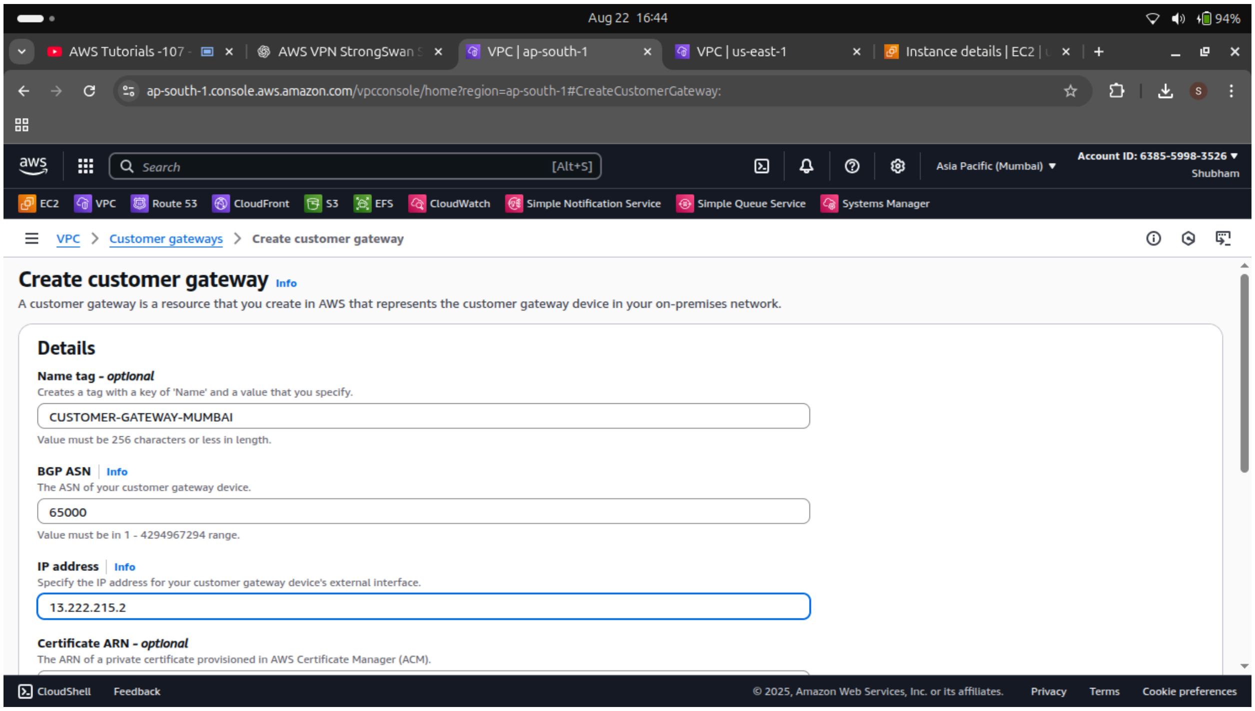
Attach the virtual private gateway to this VPC.

vpc-0c015b8c21a471b1f / VPC_AWS_REMOTE(FROM WHERE ON PREMISE WILL BE ACCESSED)

Aug 22 16:43
ap-south-1.console.aws.amazon.com/vpcconsole/home?region=ap-south-1#AttachVGWToVPC:VpnGatewayId=vgw-04fa0abac2091ecaf
VPC > Virtual private gateways > vgw-04fa0abac2091ecaf > Attach to VPC
Cancel
Attach to VPC

STEP 13

Now we will create customer gateway in mumbai region and paste down the elastic ip of virginia vpn server already launched previously



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-MUMBAI
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.

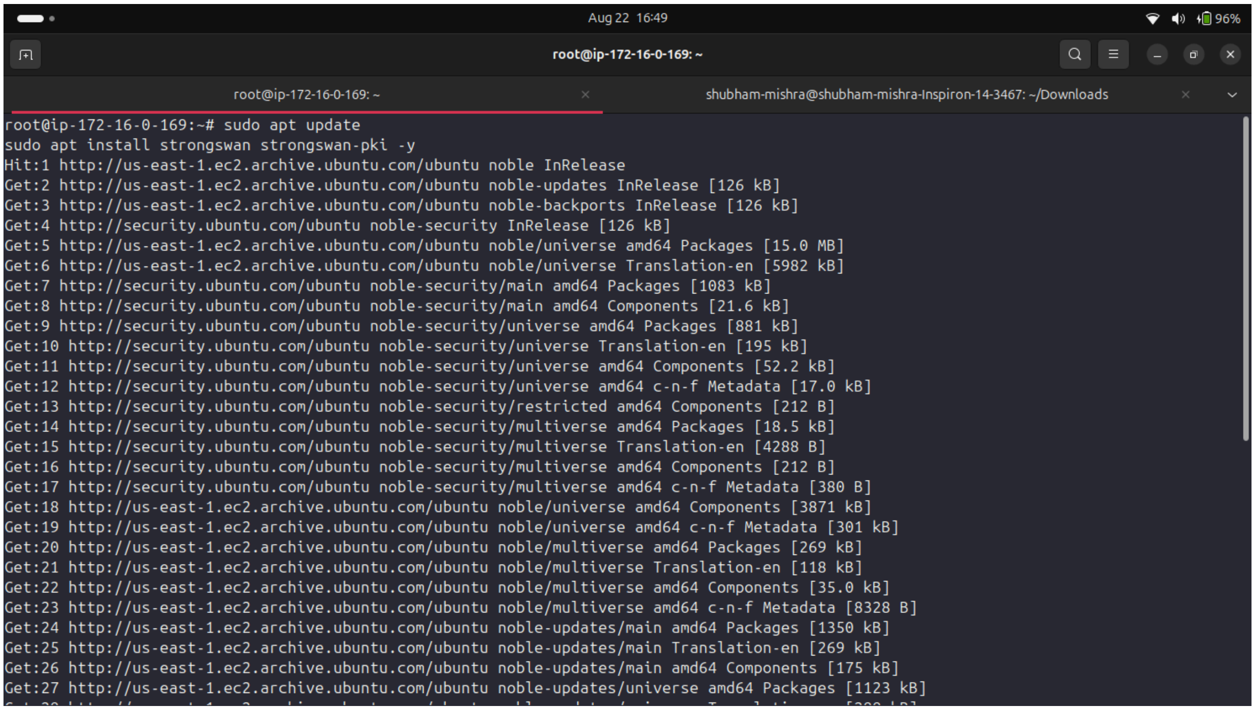
IP address | Info
Specify the IP address for your customer gateway device's external interface.
13.222.215.2

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

STEP 14

Now ssh into vpn server already launched into virginia region and run following commands

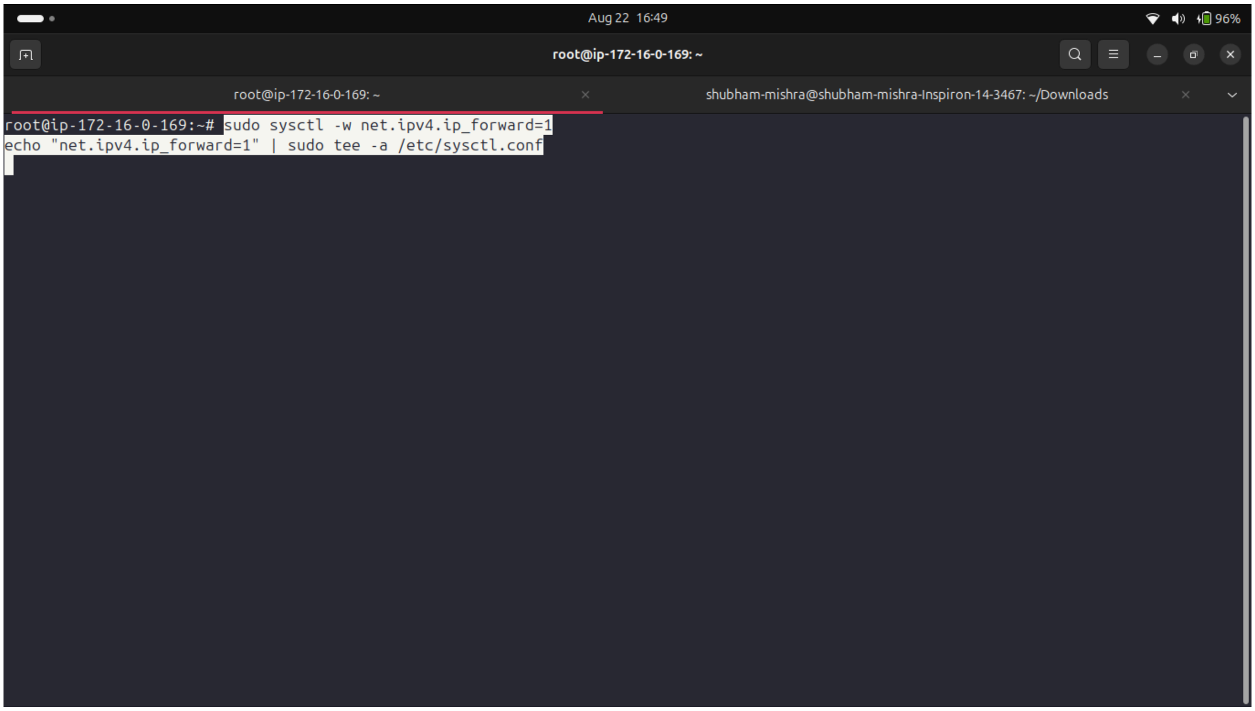
a) sudo apt update -y
b) sudo apt install strongswan-pki -y



```
root@ip-172-16-0-169:~# sudo apt update
sudo apt install strongswan strongswan-pki -y
Hit:1 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble InRelease
Get:2 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble-updates InRelease [126 kB]
Get:3 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble-backports InRelease [126 kB]
Get:4 http://security.ubuntu.com/ubuntu noble-security InRelease [126 kB]
Get:5 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble/universe amd64 Packages [15.0 MB]
Get:6 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble/universe Translation-en [5982 kB]
Get:7 http://security.ubuntu.com/ubuntu noble-security/main amd64 Packages [1083 kB]
Get:8 http://security.ubuntu.com/ubuntu noble-security/main amd64 Components [21.6 kB]
Get:9 http://security.ubuntu.com/ubuntu noble-security/universe amd64 Packages [881 kB]
Get:10 http://security.ubuntu.com/ubuntu noble-security/universe Translation-en [195 kB]
Get:11 http://security.ubuntu.com/ubuntu noble-security/universe amd64 Components [52.2 kB]
Get:12 http://security.ubuntu.com/ubuntu noble-security/universe amd64 c-n-f Metadata [17.0 kB]
Get:13 http://security.ubuntu.com/ubuntu noble-security/restricted amd64 Components [212 B]
Get:14 http://security.ubuntu.com/ubuntu noble-security/multiverse amd64 Packages [18.5 kB]
Get:15 http://security.ubuntu.com/ubuntu noble-security/multiverse Translation-en [4288 B]
Get:16 http://security.ubuntu.com/ubuntu noble-security/multiverse amd64 Components [212 B]
Get:17 http://security.ubuntu.com/ubuntu noble-security/multiverse amd64 c-n-f Metadata [380 B]
Get:18 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble/universe amd64 Components [3871 kB]
Get:19 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble/universe amd64 c-n-f Metadata [301 kB]
Get:20 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble/multiverse amd64 Packages [269 kB]
Get:21 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble/multiverse Translation-en [118 kB]
Get:22 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble/multiverse amd64 Components [35.0 kB]
Get:23 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble/multiverse amd64 c-n-f Metadata [8328 B]
Get:24 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble-updates/main amd64 Packages [1350 kB]
Get:25 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble-updates/main Translation-en [269 kB]
Get:26 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble-updates/main amd64 Components [175 kB]
Get:27 http://us-east-1.ec2.archive.ubuntu.com/ubuntu noble-updates/universe amd64 Packages [1123 kB]
```

STEP 15

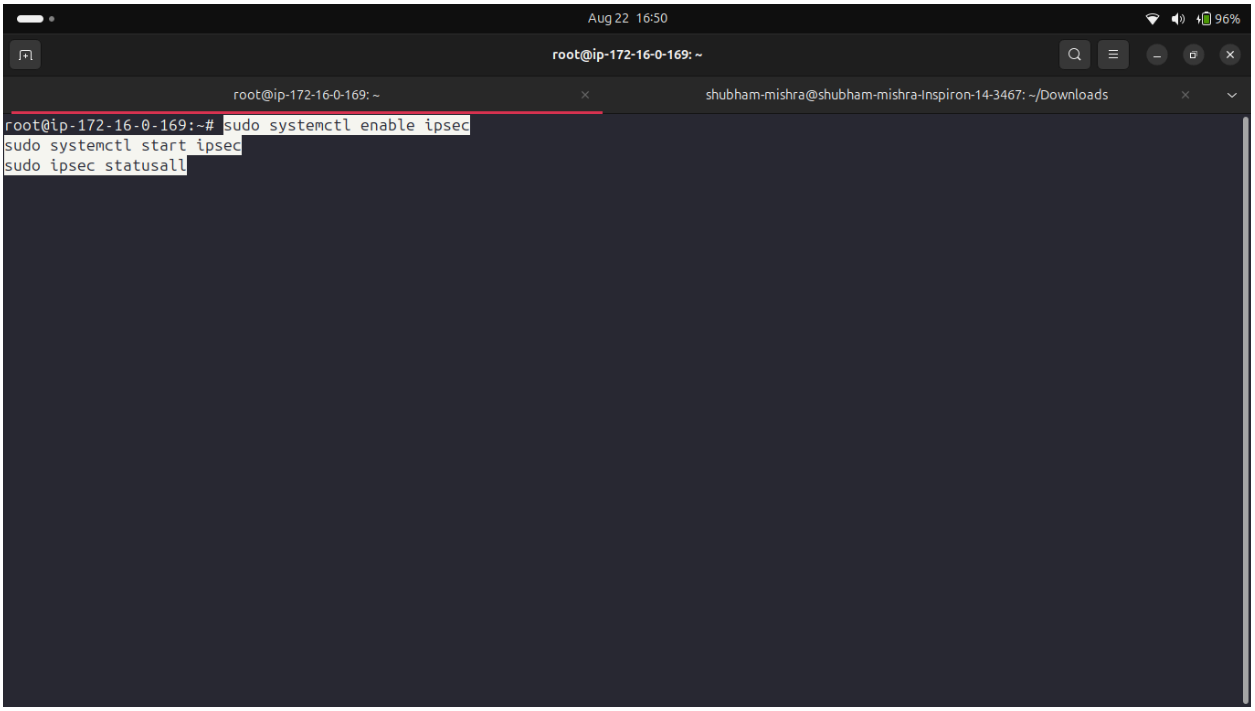
Now run following commands to setup strongswan and enable following settings



```
root@ip-172-16-0-169:~# sudo sysctl -w net.ipv4.ip_forward=1
echo "net.ipv4.ip_forward=1" | sudo tee -a /etc/sysctl.conf
```

STEP 16

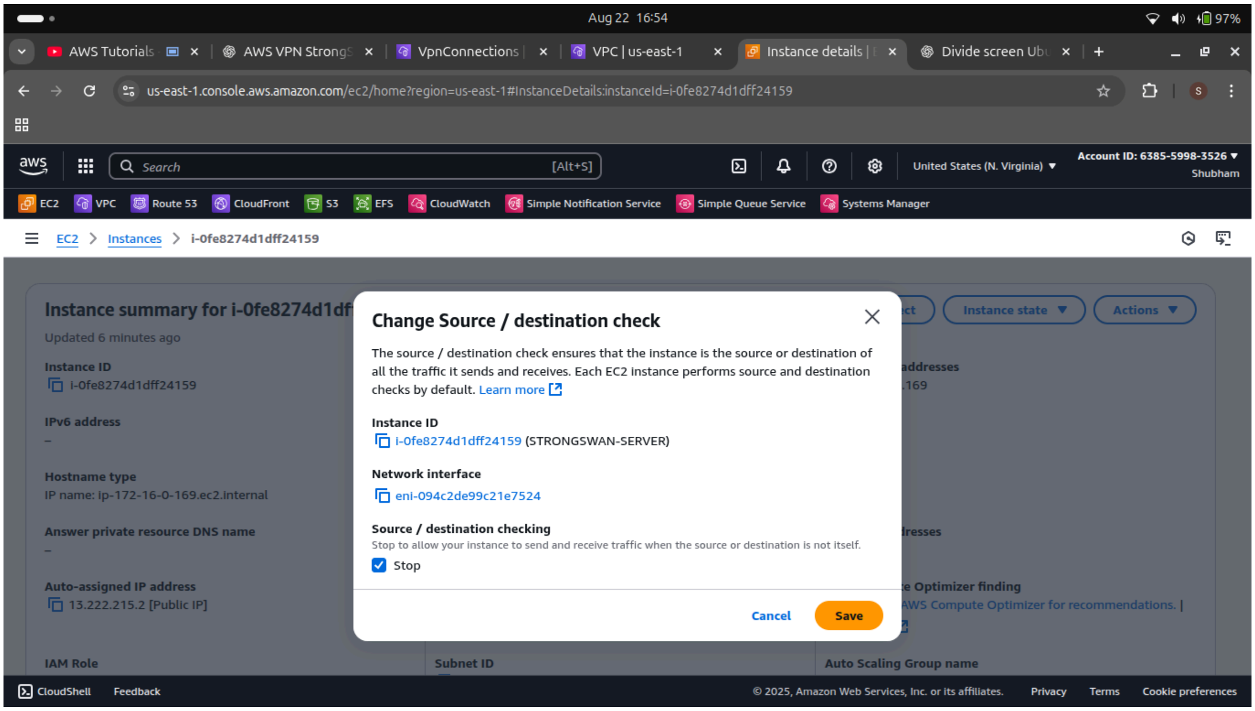
Run other commands show below to start ipsec ie strongswan



```
root@ip-172-16-0-169:~# sudo systemctl enable ipsec
sudo systemctl start ipsec
sudo ipsec statusall
```

STEP 17

Also make sure source destination check is stopped on vpn server in virginia region



Change Source / destination check

The source / destination check ensures that the instance is the source or destination of all the traffic it sends and receives. Each EC2 instance performs source and destination checks by default. Learn more

Instance ID
i-0fe8274d1dff24159 (STRONGSWAN-SERVER)

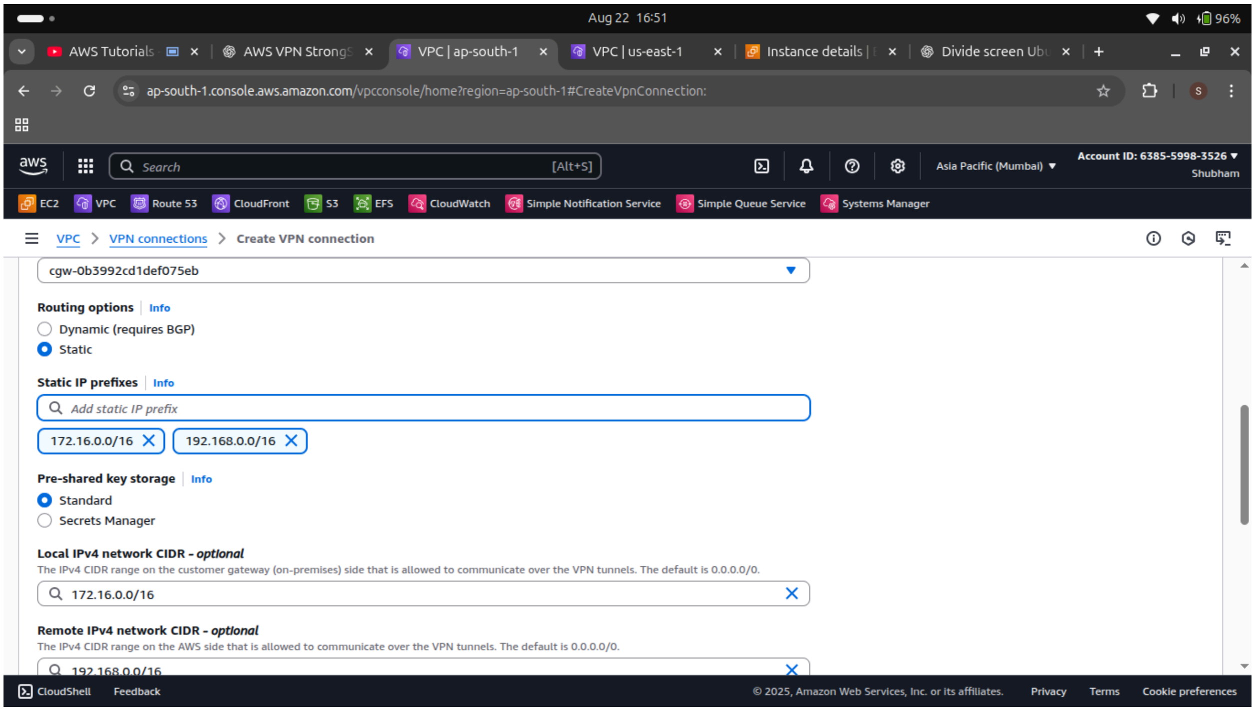
Network interface
eni-094c2de99c21e7524

Source / destination checking
Stop to allow your instance to send and receive traffic when the source or destination is not itself.

☑ Stop

Cancel Save

STEP 18
NOw we will create site to site vpn connection in mumbai region
Mention the cidr vpc ranges of mumbai and virginia region select routing as static
Also here local ipv4 refers to virginia region(on prem) cidr range and remote ipv4 refers to mumbai(remote) cidr range which we have to add.



VPC > VPN connections > Create VPN connection

cgw-0b3992cd1def075eb

Routing options | Info
○ Dynamic (requires BGP)
● Static

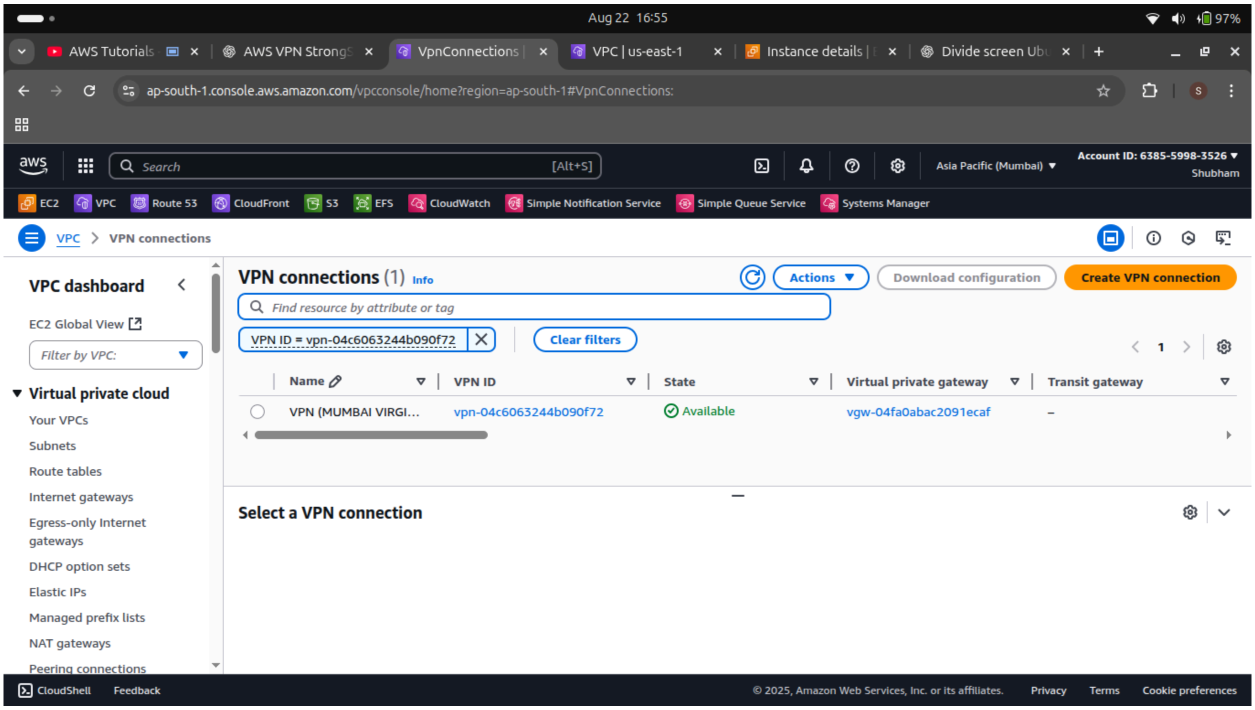
Static IP prefixes | Info
Add static IP prefix
172.16.0.0/16 ✕ 192.168.0.0/16 ✕

Pre-shared key storage | Info
● Standard
○ Secrets Manager

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.
172.16.0.0/16

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.
192.168.0.0/16

STEP 19
Wait for few minutes you can see vpn connection is in available state



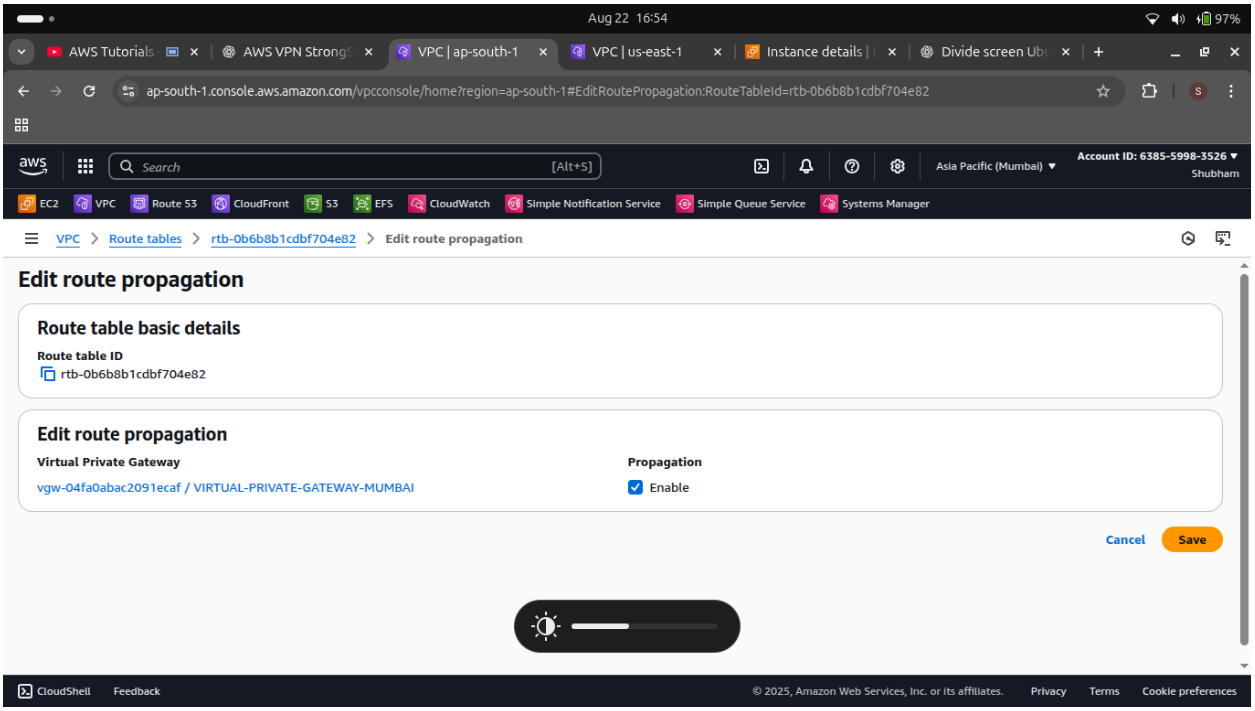
VPN connections (1) Info

VPN ID = vpn-04c6063244b090f72

	Name	VPN ID	State	Virtual private gateway	Transit gateway
○	VPN (MUMBAI VIRGI...	vpn-04c6063244b090f72	Available	vgw-04fa0abac2091ecaf	–

Select a VPN connection

STEP 20
Also make sure route propagation is enabled in mumbai vpc (remote) region public route table so that route can be propogated


Aug 22 16:54
97%
AWS Tutorials
AWS VPN StrongS
VPC | ap-south-1
VPC | us-east-1
Instance details |
Divide screen Ubu
ap-south-1.console.aws.amazon.com/vpcconsole/home?region=ap-south-1#EditRoutePropagation:RouteTableId=rtb-0b6b8b1cdbf704e82
Search [Alt+S]
Asia Pacific (Mumbai)
Account ID: 6385-5998-3526
Shubham
EC2 VPC Route 53 CloudFront S3 EFS CloudWatch Simple Notification Service Simple Queue Service Systems Manager
VPC > Route tables > rtb-0b6b8b1cdbf704e82 > Edit route propagation
Cancel
Save
CloudShell
Feedback
© 2025, Amazon Web Services, Inc. or its affiliates.
Privacy
Terms
Cookie preferences

Edit route propagation

Route table basic details

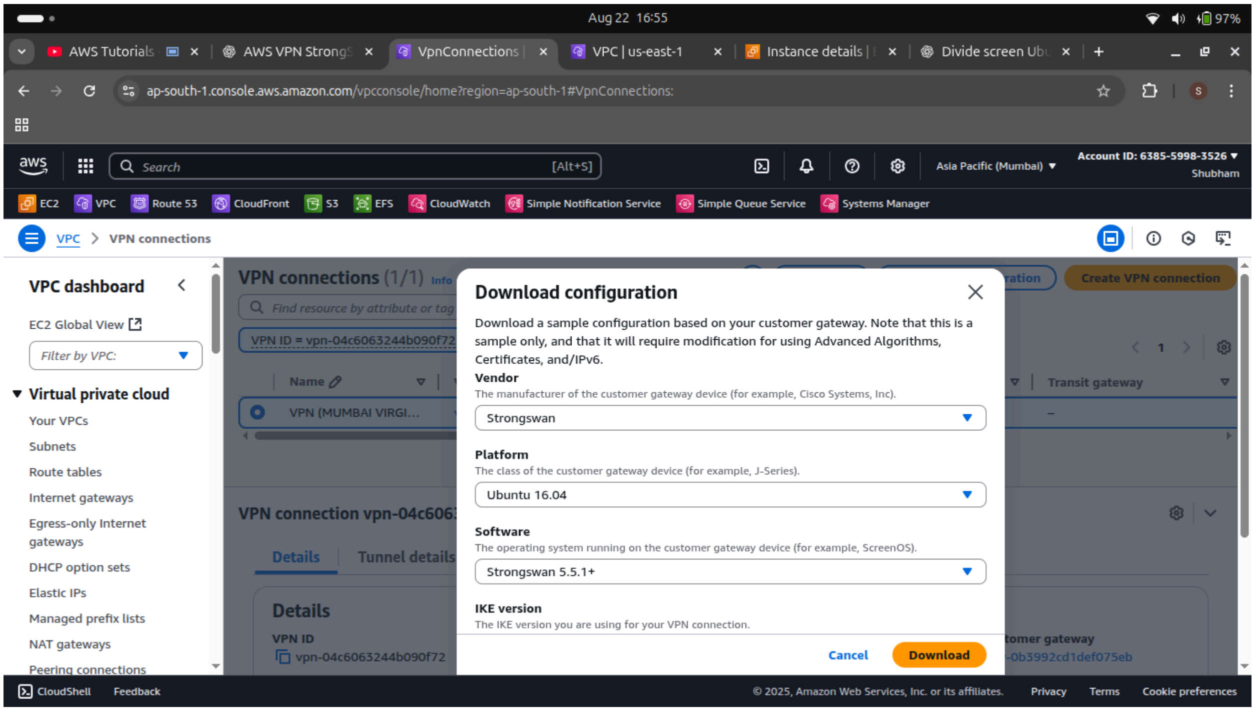
Route table ID
rtb-0b6b8b1cdbf704e82

Edit route propagation

Virtual Private Gateway	Propagation
vgw-04fa0abac2091ecaf / VIRTUAL-PRIVATE-GATEWAY-MUMBAI	☑ Enable

STEP 21

Now as our vpn connection is available we need to download configuration file for strongswan setup and make the necessary changes on vpn strongswan server in virginia region



Download configuration

Download a sample configuration based on your customer gateway. Note that this is a sample only, and that it will require modification for using Advanced Algorithms, Certificates, and/IPv6.

Vendor
The manufacturer of the customer gateway device (for example, Cisco Systems, Inc).
Strongswan

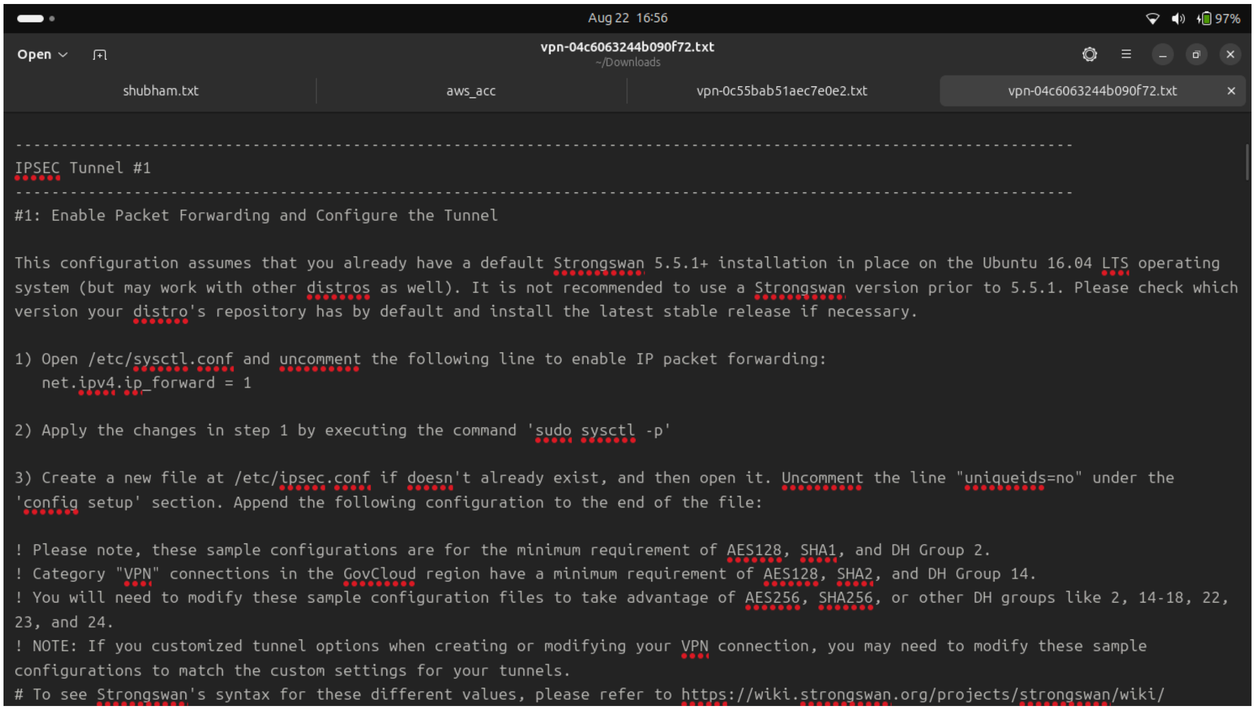
Platform
The class of the customer gateway device (for example, J-Series).
Ubuntu 16.04

Software
The operating system running on the customer gateway device (for example, ScreenOS).
Strongswan 5.5.1+

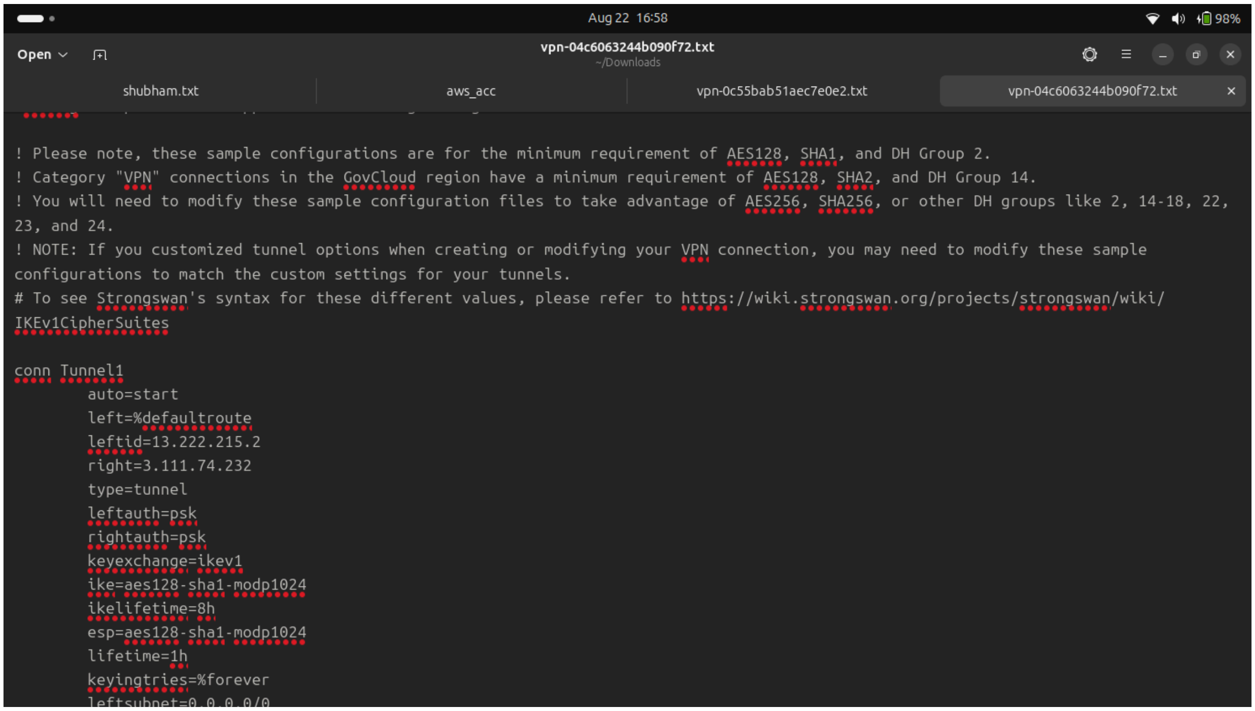
IKE version
The IKE version you are using for your VPN connection.

STEP 22

Follow the steps after downloading configuration file and make necessary changes in vpn server in north virginia region

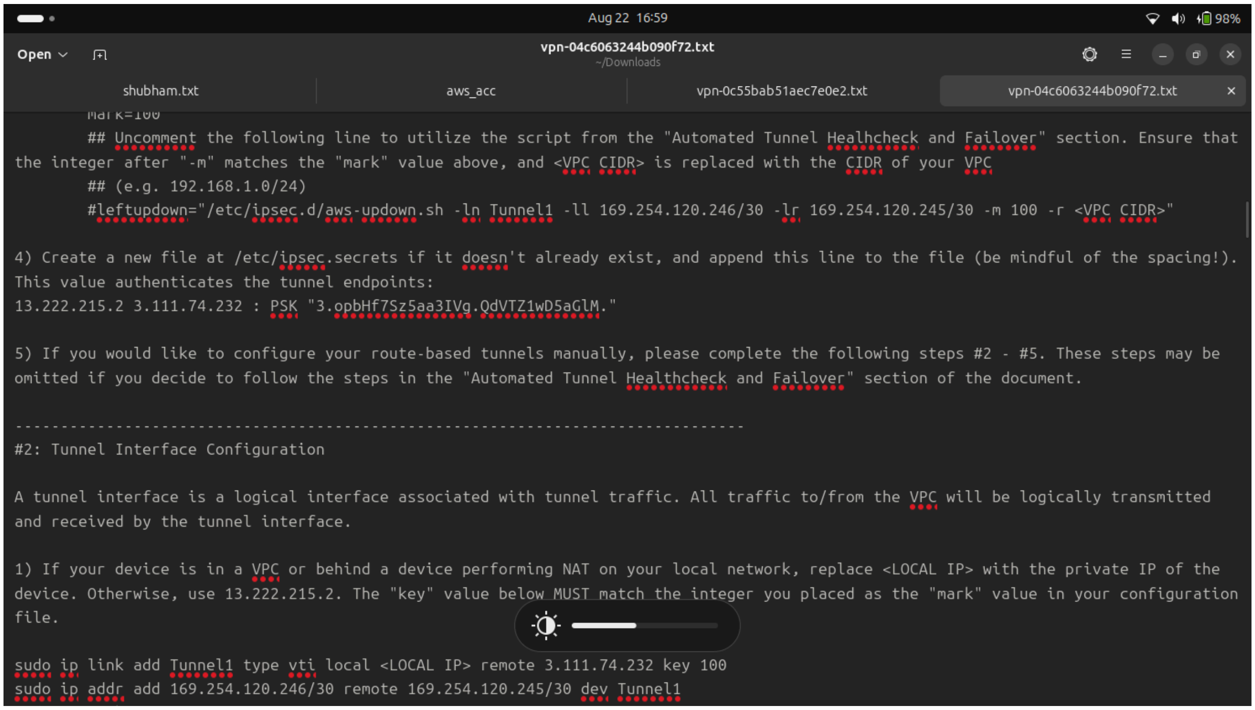

Aug 22 16:56
97%
Open
vpn-04c6063244b090f72.txt
~/Downloads
shubham.txt
aws_acc
vpn-0c55bab51aec7e0e2.txt
vpn-04c6063244b090f72.txt

```
----------------------------------------------------------------------------------------------------------------------
IPSEC Tunnel #1
----------------------------------------------------------------------------------------------------------------------
#1: Enable Packet Forwarding and Configure the Tunnel

This configuration assumes that you already have a default Strongswan 5.5.1+ installation in place on the Ubuntu 16.04 LTS operating
system (but may work with other distros as well). It is not recommended to use a Strongswan version prior to 5.5.1. Please check which
version your distro's repository has by default and install the latest stable release if necessary.

1) Open /etc/sysctl.conf and uncomment the following line to enable IP packet forwarding:
   net.ipv4.ip_forward = 1

2) Apply the changes in step 1 by executing the command 'sudo sysctl -p'

3) Create a new file at /etc/ipsec.conf if doesn't already exist, and then open it. Uncomment the line "uniqueids=no" under the
'config setup' section. Append the following configuration to the end of the file:

! 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.
# To see Strongswan's syntax for these different values, please refer to https://wiki.strongswan.org/projects/strongswan/wiki/
```


Aug 22 16:58
98%
Open
vpn-04c6063244b090f72.txt
~/Downloads
shubham.txt
aws_acc
vpn-0c55bab51aec7e0e2.txt
vpn-04c6063244b090f72.txt

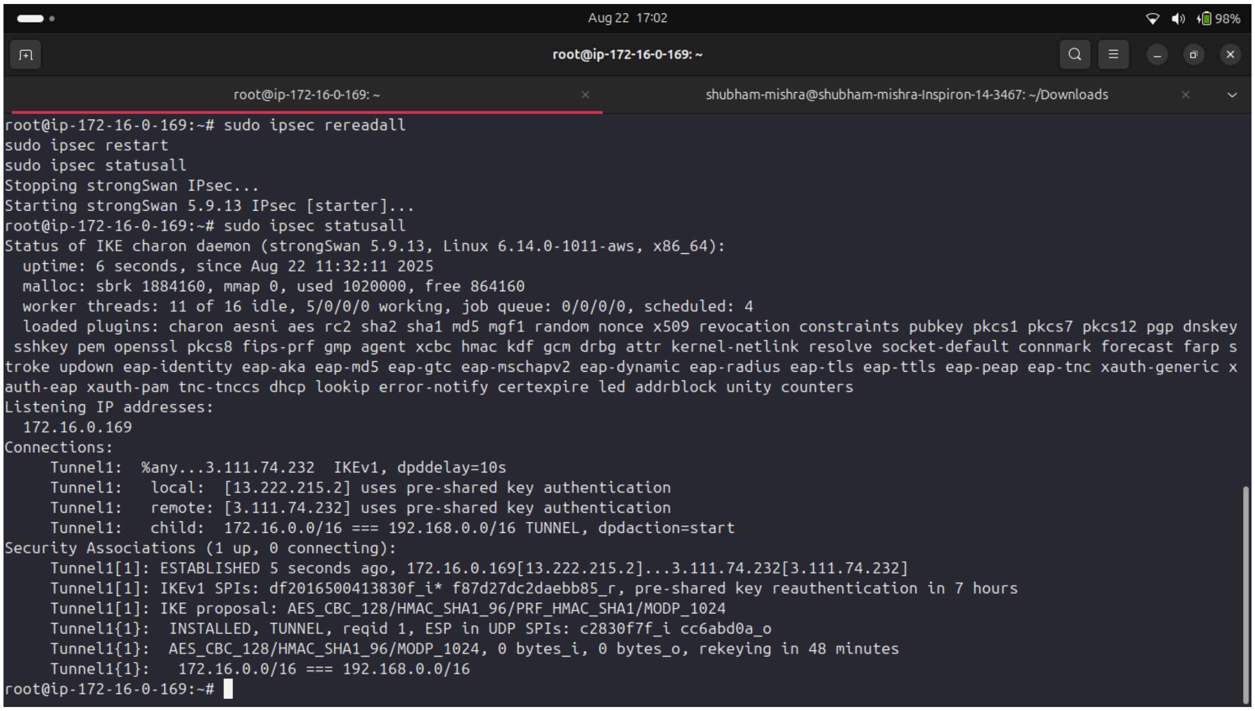
```
! Please note, these sample configurations are for the minimum requirement of AES128, SHA1, and DH Group 2.
! Category "VPN" connections in the GovCloud region have a minimum requirement of AES128, SHA2, and DH Group 14.
! You will need to modify these sample configuration files to take advantage 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.
# To see Strongswan's syntax for these different values, please refer to https://wiki.strongswan.org/projects/strongswan/wiki/
```

```
IKEv1CipherSuites

conn Tunnel1
        auto=start
        left=%defaultroute
        leftid=13.222.215.2
        right=3.111.74.232
        type=tunnel
        leftauth=psk
        rightauth=psk
        keyexchange=ikev1
        ike=aes128-sha1-modp1024
        ikelifetime=8h
        esp=aes128-sha1-modp1024
        lifetime=1h
        keyingtries=%forever
        leftsubnet=0.0.0.0/0
```

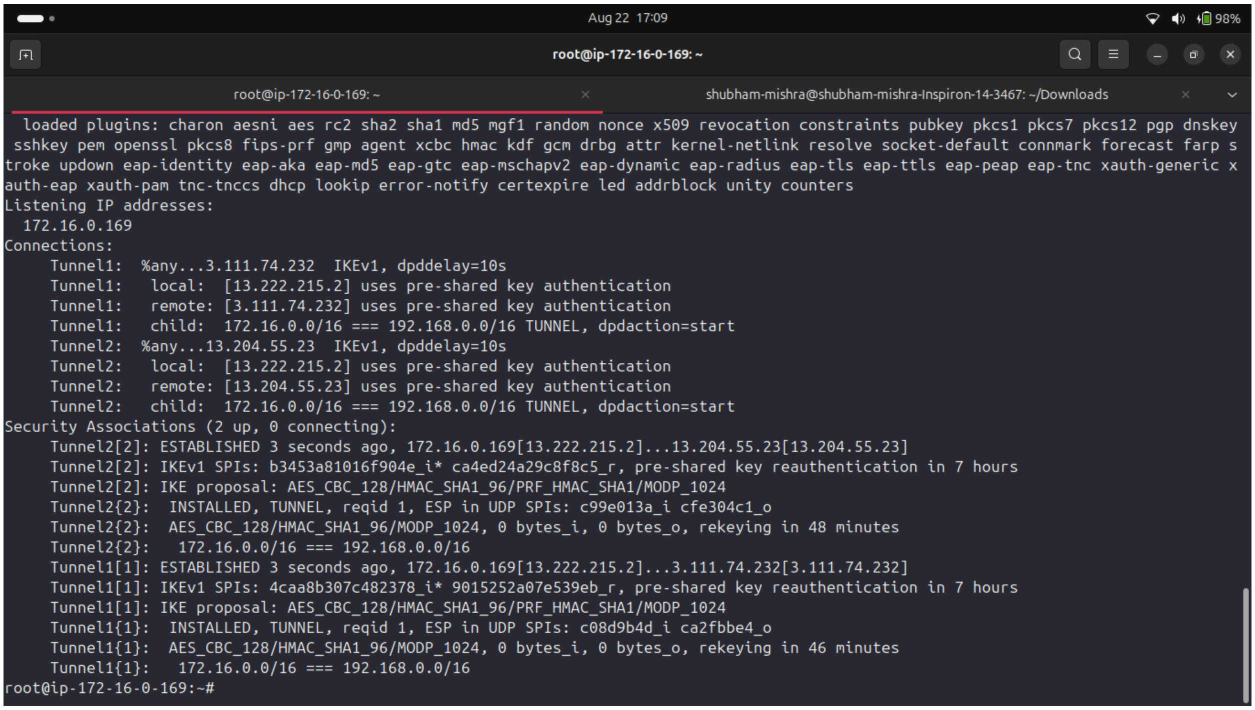

Aug 22 16:59
98%
Open
vpn-04c6063244b090f72.txt
~/Downloads
shubham.txt
aws_acc
vpn-0c55bab51aec7e0e2.txt
vpn-04c6063244b090f72.txt

```
        mark=100
        ## Uncomment the following line to utilize the script from the "Automated Tunnel Healhcheck and Failover" section. Ensure that
the integer after "-m" matches the "mark" value above, and <VPC CIDR> is replaced with the CIDR of your VPC
        ## (e.g. 192.168.1.0/24)
        #leftupdown="/etc/ipsec.d/aws-updown.sh -ln Tunnel1 -ll 169.254.120.246/30 -lr 169.254.120.245/30 -m 100 -r <VPC CIDR>"

4) Create a new file at /etc/ipsec.secrets if it doesn't already exist, and append this line to the file (be mindful of the spacing!).
This value authenticates the tunnel endpoints:
13.222.215.2 3.111.74.232 : PSK "3.opbHf7Sz5aa3IVg.QdVTZ1wD5aGlM."

5) If you would like to configure your route-based tunnels manually, please complete the following steps #2 - #5. These steps may be
omitted if you decide to follow the steps in the "Automated Tunnel Healthcheck and Failover" section of the document.

--------------------------------------------------------------------------------
#2: Tunnel Interface Configuration

A tunnel interface is a logical interface associated with tunnel traffic. All traffic to/from the VPC will be logically transmitted
and received by the tunnel interface.

1) If your device is in a VPC or behind a device performing NAT on your local network, replace <LOCAL IP> with the private IP of the
device. Otherwise, use 13.222.215.2. The "key" value below MUST match the integer you placed as the "mark" value in your configuration
file.

sudo ip link add Tunnel1 type vti local <LOCAL IP> remote 3.111.74.232 key 100
sudo ip addr add 169.254.120.246/30 remote 169.254.120.245/30 dev Tunnel1
```

STEP 23
AWS provides two vpn tunnel for high availability and fault tolerance make necessary changes for both tunnels and restart the vpn server services as show below



```
root@ip-172-16-0-169:~# sudo ipsec rereadall
sudo ipsec restart
sudo ipsec statusall
Stopping strongSwan IPsec...
Starting strongSwan 5.9.13 IPsec [starter]...
root@ip-172-16-0-169:~# sudo ipsec statusall
Status of IKE charon daemon (strongSwan 5.9.13, Linux 6.14.0-1011-aws, x86_64):
  uptime: 6 seconds, since Aug 22 11:32:11 2025
  malloc: sbrk 1884160, mmap 0, used 1020000, free 864160
  worker threads: 11 of 16 idle, 5/0/0/0 working, job queue: 0/0/0/0, scheduled: 4
  loaded plugins: charon aesni aes rc2 sha2 sha1 md5 mgf1 random nonce x509 revocation constraints pubkey pkcs1 pkcs7 pkcs12 pgp dnskey
 sshkey pem openssl pkcs8 fips-prf gmp agent xcbc hmac kdf gcm drbg attr kernel-netlink resolve socket-default connmark forecast farp s
troke updown eap-identity eap-aka eap-md5 eap-gtc eap-mschapv2 eap-dynamic eap-radius eap-tls eap-ttls eap-peap eap-tnc xauth-generic x
auth-eap xauth-pam tnc-tnccs dhcp lookip error-notify certexpire led addrblock unity counters
Listening IP addresses:
  172.16.0.169
Connections:
     Tunnel1:  %any...3.111.74.232  IKEv1, dpddelay=10s
     Tunnel1:   local:  [13.222.215.2] uses pre-shared key authentication
     Tunnel1:   remote: [3.111.74.232] uses pre-shared key authentication
     Tunnel1:   child:  172.16.0.0/16 === 192.168.0.0/16 TUNNEL, dpdaction=start
Security Associations (1 up, 0 connecting):
     Tunnel1[1]: ESTABLISHED 5 seconds ago, 172.16.0.169[13.222.215.2]...3.111.74.232[3.111.74.232]
     Tunnel1[1]: IKEv1 SPIs: df2016500413830f_i* f87d27dc2daebb85_r, pre-shared key reauthentication in 7 hours
     Tunnel1[1]: IKE proposal: AES_CBC_128/HMAC_SHA1_96/PRF_HMAC_SHA1/MODP_1024
     Tunnel1{1}:  INSTALLED, TUNNEL, reqid 1, ESP in UDP SPIs: c2830f7f_i cc6abd0a_o
     Tunnel1{1}:  AES_CBC_128/HMAC_SHA1_96/MODP_1024, 0 bytes_i, 0 bytes_o, rekeying in 48 minutes
     Tunnel1{1}:   172.16.0.0/16 === 192.168.0.0/16
root@ip-172-16-0-169:~#
```

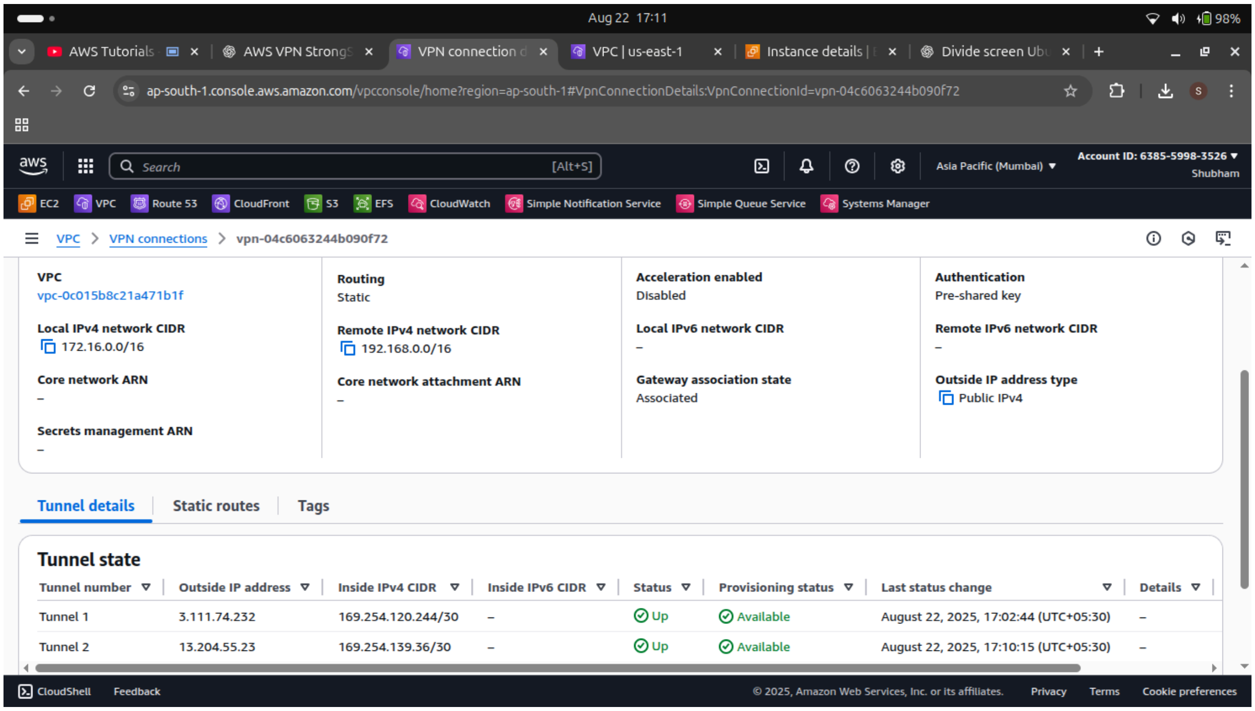


```
  loaded plugins: charon aesni aes rc2 sha2 sha1 md5 mgf1 random nonce x509 revocation constraints pubkey pkcs1 pkcs7 pkcs12 pgp dnskey
 sshkey pem openssl pkcs8 fips-prf gmp agent xcbc hmac kdf gcm drbg attr kernel-netlink resolve socket-default connmark forecast farp s
troke updown eap-identity eap-aka eap-md5 eap-gtc eap-mschapv2 eap-dynamic eap-radius eap-tls eap-ttls eap-peap eap-tnc xauth-generic x
auth-eap xauth-pam tnc-tnccs dhcp lookip error-notify certexpire led addrblock unity counters
Listening IP addresses:
  172.16.0.169
Connections:
     Tunnel1:  %any...3.111.74.232  IKEv1, dpddelay=10s
     Tunnel1:   local:  [13.222.215.2] uses pre-shared key authentication
     Tunnel1:   remote: [3.111.74.232] uses pre-shared key authentication
     Tunnel1:   child:  172.16.0.0/16 === 192.168.0.0/16 TUNNEL, dpdaction=start
     Tunnel2:  %any...13.204.55.23  IKEv1, dpddelay=10s
     Tunnel2:   local:  [13.222.215.2] uses pre-shared key authentication
     Tunnel2:   remote: [13.204.55.23] uses pre-shared key authentication
     Tunnel2:   child:  172.16.0.0/16 === 192.168.0.0/16 TUNNEL, dpdaction=start
Security Associations (2 up, 0 connecting):
     Tunnel2[2]: ESTABLISHED 3 seconds ago, 172.16.0.169[13.222.215.2]...13.204.55.23[13.204.55.23]
     Tunnel2[2]: IKEv1 SPIs: b3453a81016f904e_i* ca4ed24a29c8f8c5_r, pre-shared key reauthentication in 7 hours
     Tunnel2[2]: IKE proposal: AES_CBC_128/HMAC_SHA1_96/PRF_HMAC_SHA1/MODP_1024
     Tunnel2{2}:  INSTALLED, TUNNEL, reqid 1, ESP in UDP SPIs: c99e013a_i cfe304c1_o
     Tunnel2{2}:  AES_CBC_128/HMAC_SHA1_96/MODP_1024, 0 bytes_i, 0 bytes_o, rekeying in 48 minutes
     Tunnel2{2}:   172.16.0.0/16 === 192.168.0.0/16
     Tunnel1[1]: ESTABLISHED 3 seconds ago, 172.16.0.169[13.222.215.2]...3.111.74.232[3.111.74.232]
     Tunnel1[1]: IKEv1 SPIs: 4caa8b307c482378_i* 9015252a07e539eb_r, pre-shared key reauthentication in 7 hours
     Tunnel1[1]: IKE proposal: AES_CBC_128/HMAC_SHA1_96/PRF_HMAC_SHA1/MODP_1024
     Tunnel1{1}:  INSTALLED, TUNNEL, reqid 1, ESP in UDP SPIs: c08d9b4d_i ca2fbbe4_o
     Tunnel1{1}:  AES_CBC_128/HMAC_SHA1_96/MODP_1024, 0 bytes_i, 0 bytes_o, rekeying in 46 minutes
     Tunnel1{1}:   172.16.0.0/16 === 192.168.0.0/16
root@ip-172-16-0-169:~#
```

Here it comes that both vpn tunnels are up

STEP 24

Also in the console you can see both vpn tunnels are up



VPC
vpc-0c015b8c21a471b1f

Local IPv4 network CIDR
172.16.0.0/16

Core network ARN
–

Secrets management ARN
–

Routing
Static

Remote IPv4 network CIDR
192.168.0.0/16

Core network attachment ARN
–

Acceleration enabled
Disabled

Local IPv6 network CIDR
–

Gateway association state
Associated

Authentication
Pre-shared key

Remote IPv6 network CIDR
–

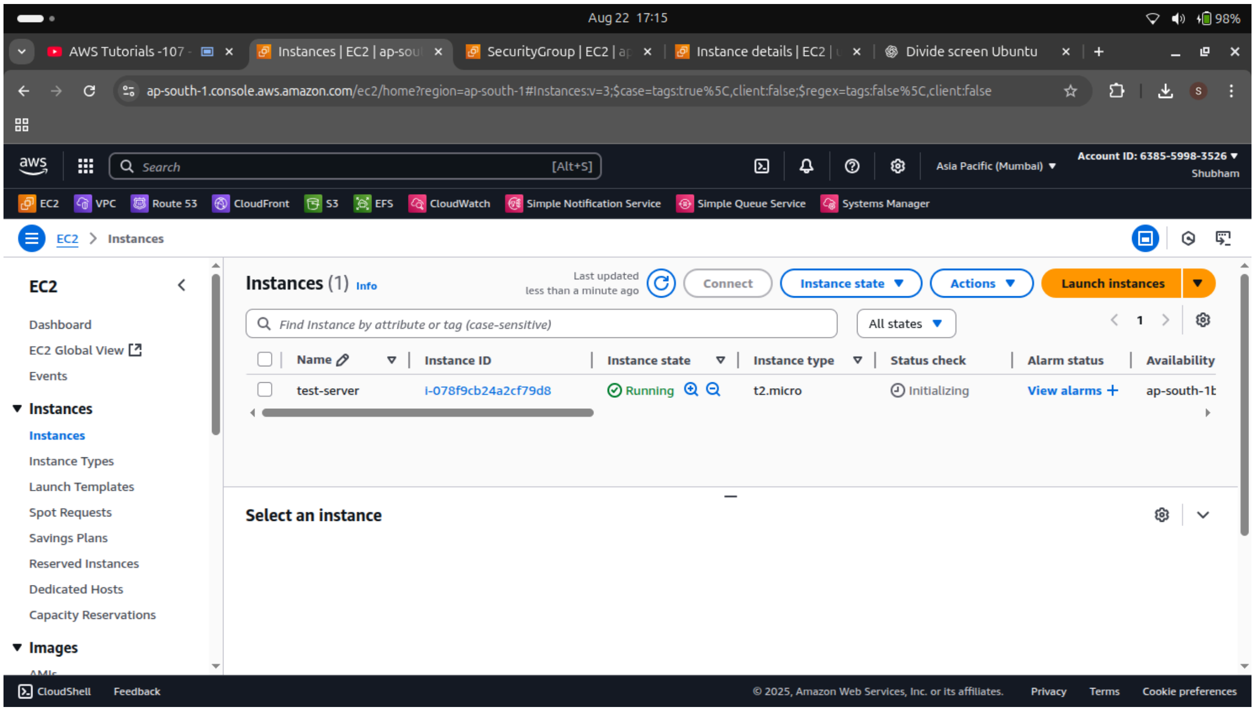
Outside IP address type
Public IPv4

Tunnel state

Tunnel number	Outside IP address	Inside IPv4 CIDR	Inside IPv6 CIDR	Status	Provisioning status	Last status change	Details
Tunnel 1	3.111.74.232	169.254.120.244/30	–	✓ Up	✓ Available	August 22, 2025, 17:02:44 (UTC+05:30)	–
Tunnel 2	13.204.55.23	169.254.139.36/30	–	✓ Up	✓ Available	August 22, 2025, 17:10:15 (UTC+05:30)	–

STEP 25

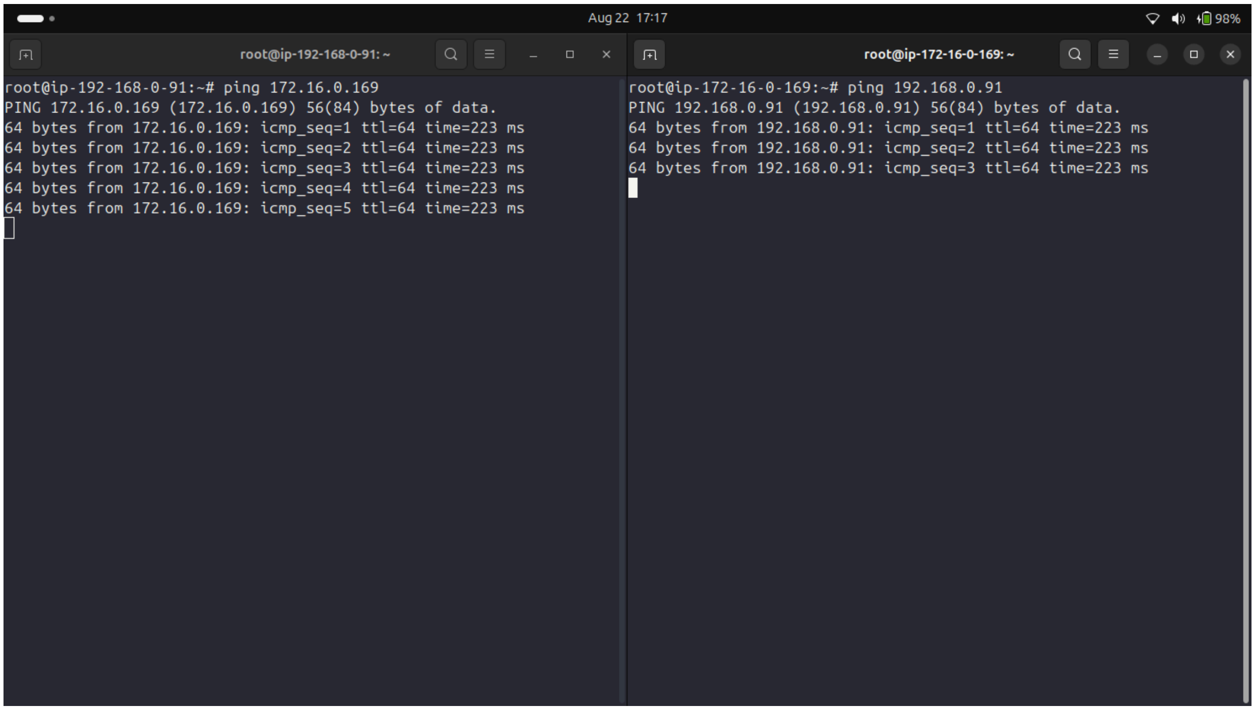
Now you can launch servers in mumbai remote region and check the tunnel connectivity



Name	Instance ID	Instance state	Instance type	Status check	Alarm status	Availability
test-server	i-078f9cb24a2cf79d8	Running	t2.micro	Initializing	View alarms +	ap-south-1b

STEP 26

You can see here servers in mumbai remote area public subnet where we setup the connectivity can communicate over private ip to the vpn server in virgina region



```
root@ip-192-168-0-91:~# ping 172.16.0.169
PING 172.16.0.169 (172.16.0.169) 56(84) bytes of data.
64 bytes from 172.16.0.169: icmp_seq=1 ttl=64 time=223 ms
64 bytes from 172.16.0.169: icmp_seq=2 ttl=64 time=223 ms
64 bytes from 172.16.0.169: icmp_seq=3 ttl=64 time=223 ms
64 bytes from 172.16.0.169: icmp_seq=4 ttl=64 time=223 ms
64 bytes from 172.16.0.169: icmp_seq=5 ttl=64 time=223 ms
```

```
root@ip-172-16-0-169:~# ping 192.168.0.91
PING 192.168.0.91 (192.168.0.91) 56(84) bytes of data.
64 bytes from 192.168.0.91: icmp_seq=1 ttl=64 time=223 ms
64 bytes from 192.168.0.91: icmp_seq=2 ttl=64 time=223 ms
64 bytes from 192.168.0.91: icmp_seq=3 ttl=64 time=223 ms
```

This confirms that vpn tunnel has been established and secure communication has been established

✅ Key Points to Mention

1. Purpose / Objective

- Demonstrated a **secure, encrypted connection** between an on-premises network and AWS VPC across regions.
- Enabled communication between resources in **Virginia (on-premises)** and **Mumbai AWS cloud environment**.
- Used for hybrid cloud architecture or disaster recovery setups.

---

2. Architecture Overview

- On-premises data center located in **Virginia** simulated using a Customer Gateway.
- AWS Virtual Private Cloud (VPC) deployed in the **Mumbai region**.

- Site-to-Site VPN tunnel established over the internet with **IPSec encryption**.
- Route propagation enabled between VPC subnets and on-premises networks.

---

3. Components Used

- **Virtual Private Gateway (VGW)** attached to Mumbai VPC.
- **Customer Gateway (CGW)** configured with on-premises public IP (Virginia).
- **VPN Connection** with two tunnels for redundancy.
- **Route Tables** updated to allow traffic between both ends.
- **Security Groups and NACLs** configured for controlled access.

---

4. Routing Details

- Static or dynamic routing using **BGP** (if configured).
- Custom routes added to ensure traffic flows between AWS and on-premises.
- Verified route propagation from VGW to Mumbai VPC and vice versa.

---

5. Encryption & Security

- Used **IPSec tunnels** with encryption algorithms like AES-256.
- Authentication via pre-shared keys.
- Controlled access via security groups allowing only required ports (e.g., ICMP, SSH).

---

6. Testing & Verification

- Used **ping tests** to check connectivity between instances in AWS and on-premises network.

- Verified routing table entries and VPN tunnel status (UP/DOWN).

---

7. Challenges Faced

- Configuring correct route propagation.

- Ensuring proper security group rules and network ACL settings.

- Handling failover with two tunnels.

---

8. Benefits / Use Cases

- Secure communication without exposing workloads over the public internet.

- Multi-region access for distributed teams or backup environments.

- Hybrid cloud architecture enabling scalable applications.
