Unix for Administration

Setup an IKEv2 VPN Server with StrongSwan on Ubuntu

By

Aravind Kumar

Professor:

Salem Osman

Table of contents

Introduction

- A virtual private network, or VPN, allows us to securely encrypt traffic as it travels through untrusted networks, such as those at the coffee shop, a conference, or an airport.
- IKEv2, or Internet Key Exchange v2, is a protocol that allows for direct IPSec tunneling between the server and client.
- In IKEv2 VPN implementations, IPSec provides encryption for the network traffic. IKEv2 is natively supported on some platforms (OS X 10.11+, iOS 9.1+, and Windows 10) with no additional applications necessary, and it handles client hiccups quite smoothly.

In this tutorial, we set up an IKEv2 VPN server using StrongSwan on an Ubuntu server and connect to it from Windows, macOS, Ubuntu, iOS, and Android clients.

Prerequisites

- We will need a sudo non-root user and a firewall.

Step 1 – Installing StrongSwan

- First, we will install StrongSwan, an open-source IPSec daemon which we will configure as our VPN server.
- We will also install the public key infrastructure component so that we can create a certificate authority to provide credentials for our infrastructure.

  Command – sudo apt install strongswan strongswan-pki

```
aravind@aravind-ubuntu:/$ sudo apt install strongswan strongswan-pki
Reading package lists... Done
Building dependency tree
Reading state information... Done
The following packages were automatically installed and are no longer required:
  galera-4 libdbd-mysql-perl libdbi-perl libmariadb3 libreadline5 libterm-readkey-perl linux-headers-5.4.0-26 linux-headers-5.4.0-26-generic
  linux-image-5.4.0-26-generic linux-modules-5.4.0-26-generic linux-modules-extra-5.4.0-26-generic mariadb-common socat
Use 'sudo apt autoremove' to remove them.
The following additional packages will be installed:
  libcharon-extauth-plugins libstrongswan libstrongswan-standard-plugins strongswan-charon strongswan-libcharon strongswan-starter
Suggested packages:
  libstrongswan-extra-plugins libcharon-extra-plugins
The following NEW packages will be installed:
  libcharon-extauth-plugins libstrongswan libstrongswan-standard-plugins strongswan strongswan-charon strongswan-libcharon strongswan-pki
  strongswan-starter
0 upgraded, 8 newly installed, 0 to remove and 1 not upgraded.
Need to get 931 kB of archives.
After this operation, 4,454 kB of additional disk space will be used.
Do you want to continue? [Y/n] y
Get:1 http://us.archive.ubuntu.com/ubuntu focal/main amd64 libstrongswan amd64 5.8.2-1ubuntu3 [356 kB]
Get:2 http://us.archive.ubuntu.com/ubuntu focal/main amd64 strongswan-libcharon amd64 5.8.2-1ubuntu3 [241 kB]
Get:3 http://us.archive.ubuntu.com/ubuntu focal/main amd64 strongswan-charon amd64 5.8.2-1ubuntu3 [22.2 kB]
Get:4 http://us.archive.ubuntu.com/ubuntu focal/main amd64 strongswan-starter amd64 5.8.2-1ubuntu3 [148 kB]
Get:5 http://us.archive.ubuntu.com/ubuntu focal/main amd64 libcharon-extauth-plugins amd64 5.8.2-1ubuntu3 [23.0 kB]
Get:6 http://us.archive.ubuntu.com/ubuntu focal/main amd64 libstrongswan-standard-plugins amd64 5.8.2-1ubuntu3 [67.3 kB]
Get:7 http://us.archive.ubuntu.com/ubuntu focal/main amd64 strongswan all 5.8.2-1ubuntu3 [18.2 kB]
Get:8 http://us.archive.ubuntu.com/ubuntu focal/universe amd64 strongswan-pki amd64 5.8.2-1ubuntu3 [55.1 kB]
Fetched 931 kB in 2s (386 kB/s)
Preconfiguring packages ...
Selecting previously unselected package libstrongswan.
```

- Now we have installed the strongswan, let's move on to creating our certificates.

Step 2 – Creating a certificate authority

- An IKEv2 server requires a certificate to identify itself to clients. To help us create the certificate required, the **strongswan-pki** package comes with a utility to generate a certificate authority and server certificates.
- To begin, let's create a few directories to store all the assets we will be working on. The directory structure matches some of the directories in **/etc/ipsec.d**, where we will eventually move all of the items we create.
- We will lock down the permissions so that our private files can't be seen by other users

  Command

  1. mkdir -p ~/pki/{cacerts,certs,private}
  2. chmod 700 ~/pki

```
aravind@aravind-ubuntu:/$ mkdir -p ~/pki/{cacerts,certs,private}
aravind@aravind-ubuntu:/$ chmod 700 ~/pki
aravind@aravind-ubuntu:/$
```

- Now that we have a directory structure to store everything, we can generate a root key. This will be a 4096-bit RSA key that will be used to sign our root certificate authority.

  Command - ipsec pki --gen --type rsa --size 4096 --outform pem > ~/pki/private/ca-key.pem

```
aravind@aravind-ubuntu:/$ ipsec pki --gen --type rsa --size 4096 --outform pem > ~/pki/private/ca-key.pem
aravind@aravind-ubuntu:/$
```

- Now that we have a key, we can move on to creating our root certificate authority, using the key to sign the root certificate.

  Command - ipsec pki --self --ca --lifetime 3650 --in ~/pki/private/ca-key.pem \ --type rsa --dn "CN=VPN root CA" --outform pem > ~/pki/cacerts/ca-cert.pem

```
aravind@aravind-ubuntu:/$ ipsec pki --self --ca --lifetime 3650 --in ~/pki/private/ca-key.pem \
> --type rsa --dn "CN=VPN root CA" --outform pem > ~/pki/cacerts/ca-cert.pem
aravind@aravind-ubuntu:/$
```

- Now that we've got our root certificate authority up and running, we can create a certificate that the VPN server will use.

Step 3 – Generating a certificate for the VPN Server

- We will now create a certificate and key for the VPN server. This certificate will allow the client to verify the server's authenticity using the CA certificate we just generated.
- First, create a private key for the VPN server

  Command - ipsec pki --gen --type rsa --size 4096 --outform pem > ~/pki/private/server-key.pem

```
aravind@aravind-ubuntu:/$ ipsec pki --gen --type rsa --size 4096 --outform pem > ~/pki/private/server-key.pem
aravind@aravind-ubuntu:/$
```

- Now, create and sign the VPN server certificate with the certificate authority's key we created in the previous step.
- Execute the following command, but change the Common Name (CN) and the Subject Alternate Name (SAN) field to your VPN server's DNS name or IP address:

  Command

  ipsec pki --pub --in ~/pki/private/server-key.pem --type rsa \ | ipsec pki --issue --lifetime 1825 \
  > --cacert ~/pki/cacerts/ca-cert.pem \
  > --cakey ~/pki/private/ca-key.pem \
  > --dn "CN=192.168.244.134" --san "192.168.244.134" \

> --flag serverAuth --flag ikeIntermediate --outform pem \
> ~/pki/certs/server-cert.pem

```
aravind@aravind-ubuntu:/$ ipsec pki --pub --in ~/pki/private/server-key.pem --type rsa \ | ipsec pki --issue --lifetime 1825 \
> --cacert ~/pki/cacerts/ca-cert.pem \
> --cakey ~/pki/private/ca-key.pem \
> --dn "CN=192.168.244.134" --san "192.168.244.134" \
> --flag serverAuth --flag ikeIntermediate --outform pem \
> ~/pki/certs/server-cert.pem
-----BEGIN CERTIFICATE-----
MIIFBTCCAu2gAwIBAgIIK+1er0I+90UwDQYJKoZIhvcNAQEMBQAwFjEUMBIGA1UE
AxMLVlBOIHJvb3QgQ0EwHhcNMjAwNTMwMjEwOTA2WhcNMjUwNTI5MjEwOTA2WjAa
MRgwFgYDVQQDEw8xOTIuMTY4LjI0NC4xMzQwggIiMA0GCSqGSIb3DQEBAQUAA4IC
DwAwggIKAoICAQDkxfSDDX82BcPzBw2mHxZqdVVg2TaQvs2qNzjnwojWTc0kWJi2
bxPLda8UywUSNOTM3+Y1o3LC08wdwprIqbI4G4rGt6CX8e6eI5IP8xZRh/VNuJtb
GB+jdQOMpJ+XgNc3M1RhIuZX14hDZn9PnFoFdKewcvU5DdCXDpC9nw0ivkPJ8uCY
oO87hNY2kIqqkBqKkknOL8ZUvK822+mQ8HVYRMoz0N5rGsxXzJta/du8SLV1x0V3
Az2ZmPUyb1l3DK21ydyyoxYoJldnSW19jJoiQG4VOM+JFAnNrwGStFqF8wk2+FDj
u1hkNGJRJGgo3P+C8o4wXoGOXQq8yn1HhMFdyXRJMNvgXcGLUoWJNtJke3SEkwdn
c4w3ObfwI8MKOZIKmEycPZa+5d5Ts4JmgGE1yUkaqrwn6nV9BO4ux8SKfs78NaeG
IlDCFBRIieYRkRst8To1NQNNmq15ulxJb8W8UimVnLRyqnjcd6PdwYpxH9X6+JFq
btw8/XKqp0fXVMJmglo6bkTJ1XCH6cLo3EG7NtsLoILdH8MXP6ycXMMGlZgmqV+Z
ZWgtorn1utnkMas31YLBMizdotGhaPYaC494tSWIYQXm7OpTuItwmkdDQ1ALyrAw
nTOWigQxA4SMij7CroC32WLR+utWU/Xvds85qiLs7fZCfZPHjuRa1lTgmwIDAQAB
o1MwUTAfBgNVHSMEGDAWgBSJUIHXcPSB/BAaTOS26P9WAsxvfzAPBgNVHREECDAG
hwTAqPSGMB0GA1UdJQQWMBQGCCsGAQUFBwMBBggrBgEFBQgCAjANBgkqhkiG9w0B
AQwFAAOCAgEAFFmHP7oPlnPoosERhls6+L03qd83u5BlRgf8gcKWG4YpuJGVeg2y
14ZVyaSACeDzsY64GCZACza9w/WLZngDF4/+6KyW474j8ObsxUUclr5VsCCO0bkK
LzVkcdwdvEM7Ze8sbLzUGYhitM+DsOZhkjAukdlMERYxdXoxCJQC266qCMzFl0ks
D5d1PFoipsmqAveiDU9IJb9VbMvIT1PM95hMh553qBcqqN8Befui7uuu4P0Z9f1m
```

```
bxUOLwBecQnAIrJMZKvqYqjcTCCh8vao8Sbccg1J+EZ4f6fs1t4E1ihpF1nXUPBN
aqWj3212yDiY2NR3IbtjklaSh1gJ9z540VSul+ZVsr8+9t1ceFOKcxHu8fdt33m5
Kq+gGiiREpuAyR+PxzBqb+ll6swAWOezYA6P7WUWhd5Ca8aQxbbwmLn1E/bL/Vsg
VuNgVxuaukDYS3ODxAAItJ7ejsCnFNHudANjShf0WwqDGUJHZ86IUZdXo6550nA8
OHf9hIedQIuyaF5feQ1UfbJI1W0av+psgzOPDMtJM63amzcpmL0xw2YEJdJUWoSF
N+FNFGJQ2135Z87hkll5qvObhQjUM8oHueKmXVToVaRFj3SM+uZH7vZAwbP4fug2
iVly+qmGrv8wUA7Nsm3KsF4p3FcJC4RlFQ4/SsHKYB1BAz39KbiJq1Q=
-----END CERTIFICATE-----
aravind@aravind-ubuntu:/$
```

- Now that we've generated all of the TLS/SSL files StrongSwan needs, we can move the files into place in the /etc/ipsec.d directory

  Command - sudo cp -r ~/pki/* /etc/ipsec.d/

```
aravind@aravind-ubuntu:/$ sudo cp -r ~/pki/* /etc/ipsec.d/
[sudo] password for aravind:
aravind@aravind-ubuntu:/$
```

- In this step, we've created a certificate pair that would be used to secure communications between the client and the server.
- We've also signed the certificates with the CA key, so the client will be able to verify the authenticity of the VPN server using the CA certificate.
- Now that have all of the certificates ready, we'll move on to configuring the software.

Step 4 – Configuring StrongSwan

- StrongSwan has a default configuration file with some examples, but we will have to do most of the configuration ourselves.
- Let's back up the file for reference before starting from scratch

  Command - sudo mv /etc/ipsec.conf{,.original}

```
aravind@aravind-ubuntu:/$ sudo mv /etc/ipsec.conf{,.original}
aravind@aravind-ubuntu:/$
```

- Now we will create and open a new blank configuration file

  Command - sudo nano /etc/ipsec.conf

```
aravind@aravind-ubuntu:/$ sudo nano /etc/ipsec.conf
aravind@aravind-ubuntu:/$
```

- Inside the /etc/ipsec.conf file enter the code specified below

  Code

  #First, we'll tell StrongSwan to log daemon statuses for debugging and allow duplicate connections.

```
config setup
   charondebug="ike 1, knl 1, cfg 0"
   uniqueids=no
```

  #We'll also tell StrongSwan to create IKEv2 VPN Tunnels and to automatically load this configuration section when it starts up.

```
conn ikev2-vpn
   auto=add
   compress=no
   type=tunnel
   keyexchange=ikev2
   fragmentation=yes
   forceencaps=yes
```

  #We'll also configure dead-peer detection to clear any "dangling" connections in case the client unexpectedly disconnects.

```
   dpdaction=clear
   dpddelay=300s
```

rekey=no

#Then, we'll configure the server (left) side IPSec parameters

left=%any

leftid=192.168.244.134

leftcert=server-cert.pem

leftsendcert=always

leftsubnet=0.0.0.0/0

#Next, we can configure the client (right) side IPSec parameters, like the private IP address ranges and DNS servers to use

right=%any

rightid=%any

rightauth=eap-mschapv2

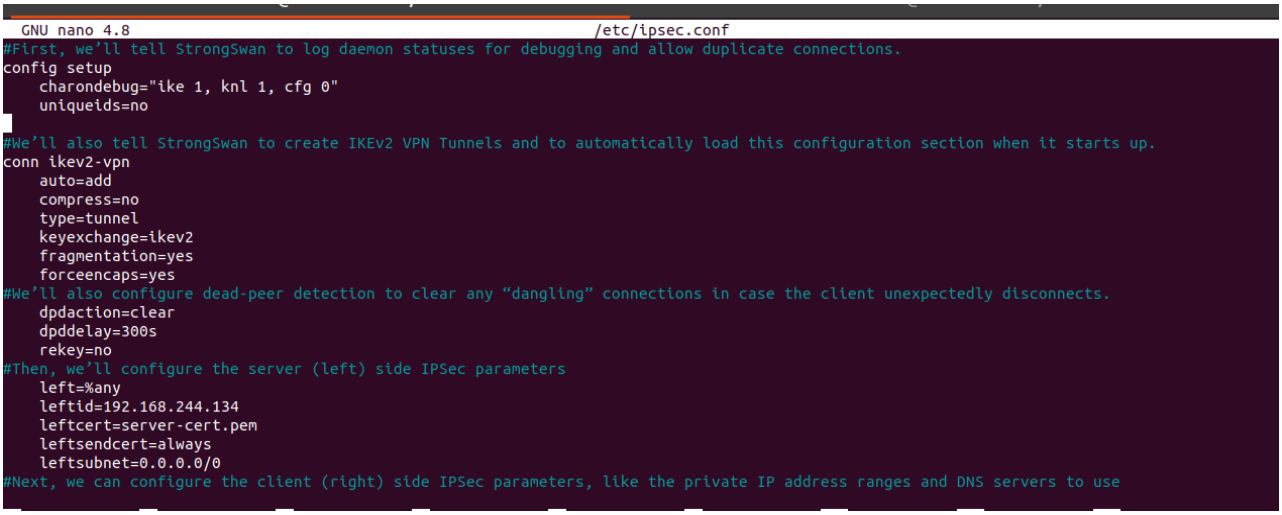
rightsourceip=10.10.10.0/24

rightdns=8.8.8.8,8.8.4.4

rightsendcert=never

#Finally, we'll tell StrongSwan to ask the client for user credentials when they connect

eap_identity=%identity



- In the configure the server (left) side IPSec parameters in the leftid parameter enter your domain name or IP address

```
#Next, we can configure the client (right) side IPSec parameters, like the private IP address ranges and DNS servers to use
    right=%any
    rightid=%any
    rightauth=eap-mschapv2
    rightsourceip=10.10.10.0/24
    rightdns=8.8.8.8,8.8.4.4
    rightsendcert=never
#Finally, we'll tell StrongSwan to ask the client for user credentials when they connect
    eap_identity=%identity
```

- Save and close the file.
- Now that we've configured the VPN parameters, let's move on to creating an account so our users can connect to the server.

Step 5 – Configuration VPN Authentication

- Our VPN server is now configured to accept client connections, but we don't have any credentials configured yet.
- We'll need to configure a couple things in a special configuration file called **ipsec.secrets:**
    - We need to tell StrongSwan where to find the private key for our server certificate, so the server will be able to authenticate to clients.
    - We also need to set up a list of users that will be allowed to connect to the VPN.
- Now open the secrets file for editing

  Command - sudo nano /etc/ipsec.secrets

```
aravind@aravind-ubuntu:/$ sudo nano /etc/ipsec.secrets
[sudo] password for aravind:
aravind@aravind-ubuntu:/$
```

- Now inside the file paste the below codes

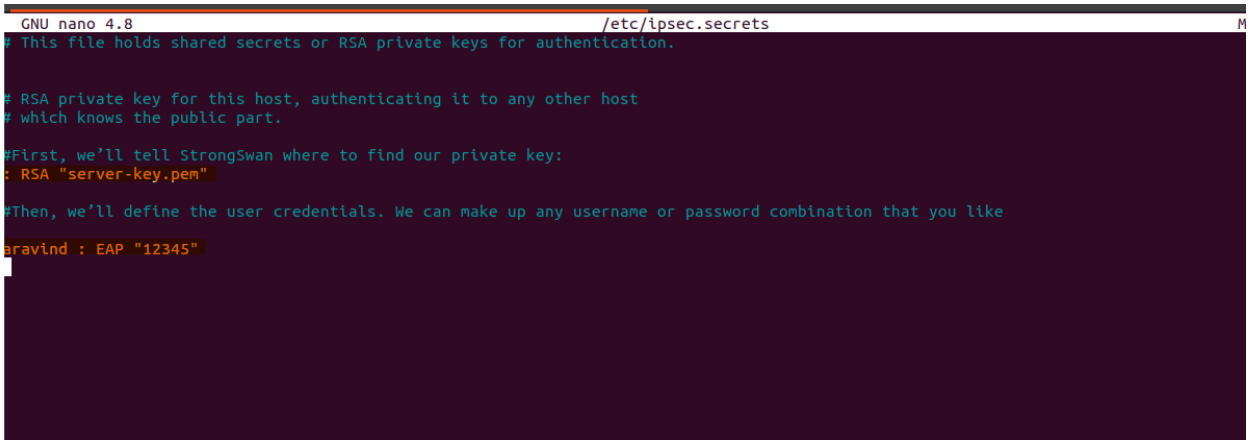
  Command

  : RSA "server-key.pem"

  your_username : EAP "your_password"

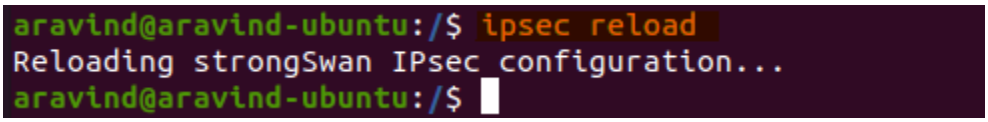
  Note

  We can make up any username or password combination that you like



- Save and close the file.
- Now that we've finished working with the VPN parameters, we'll reload the VPN service so that our configuration is applied

  Command – ipsec reload



Step 6 – Configuring the Firewall & Kernel IP Forwarding

- With the StrongSwan configuration complete, we need to configure the firewall to forward and allow VPN traffic through.
- Check whether OpenSSH is enabled in the UFW configuration (**Port 22**)

  Command – sudo ufw status

```
aravind@aravind-ubuntu:/$ sudo ufw status
[sudo] password for aravind:
Status: active

To                         Action      From
--                         ------      ----
80                         ALLOW       Anywhere
443                        ALLOW       Anywhere
22/tcp                     ALLOW       Anywhere
80 (v6)                    ALLOW       Anywhere (v6)
443 (v6)                   ALLOW       Anywhere (v6)
22/tcp (v6)                ALLOW       Anywhere (v6)

aravind@aravind-ubuntu:/$
```

- Now, we add a rule to allow UDP traffic to the standard IPSec ports, 500 and 4500

  Command - sudo ufw allow 500,4500/udp

```
aravind@aravind-ubuntu:/$ sudo ufw allow 500,4500/udp
Rule added
Rule added (v6)
aravind@aravind-ubuntu:/$
```

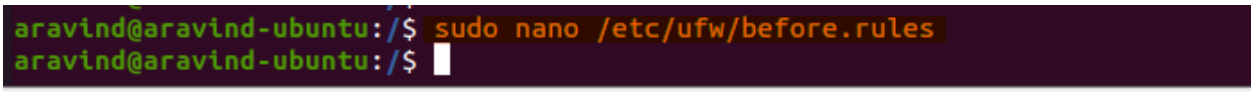
- Next, we will open up one of UFW's configuration files to add a few low-level policies for routing and forwarding IPSec packets.
- Before we do, we need to find which network interface on our server is used for internet access.
- We can find that by querying for the interface associated with the default route.
- The public interface should follow the word "dev"

  Command - ip route | grep default

```
aravind@aravind-ubuntu:/$ ip route | grep default
default via 192.168.244.2 dev ens33 proto dhcp metric 100
aravind@aravind-ubuntu:/$
```

- When we have our public network interface, open the **/etc/ufw/before.rules** file in the text editor

  Command - sudo nano /etc/ufw/before.rules



- Near the top of the file (before the *filter line), add the following configuration block

  Code

  *nat

  -A POSTROUTING -s 10.10.10.0/24 -o ens33 -m policy --pol ipsec --dir out -j ACCEPT

  -A POSTROUTING -s 10.10.10.0/24 -o ens33 -j MASQUERADE

  COMMIT

  *mangle

  -A FORWARD --match policy --pol ipsec --dir in -s 10.10.10.0/24 -o ens33 -p tcp -m tcp --tcp-flags SYN,RST SYN -m tcpmss --mss 1361:1536 -j TCPMSS --set-mss 1360

  COMMIT

- The ***nat lines** create rules so that the firewall can correctly route and manipulate traffic between the VPN clients and the internet.
- The ***mangle line** adjusts the maximum packet segment size to prevent potential issues with certain VPN clients.
- Next, after the ***filter** and chain definition lines, add one more block of configuration

  Code

  -A ufw-before-forward --match policy --pol ipsec --dir in --proto esp -s 10.10.10.0/24 -j ACCEPT
  -A ufw-before-forward --match policy --pol ipsec --dir out --proto esp -d 10.10.10.0/24 -j ACCEPT

- These lines tell the firewall to forward **ESP (Encapsulating Security Payload)** traffic so the VPN clients will be able to connect. **ESP** provides additional security for our VPN packets as they’re traversing untrusted networks.

```
# Don't delete these required lines, otherwise there will be errors

*nat
-A POSTROUTING -s 10.10.10.0/24 -o ens33 -m policy --pol ipsec --dir out -j ACCEPT
-A POSTROUTING -s 10.10.10.0/24 -o ens33 -j MASQUERADE
COMMIT

*mangle
-A FORWARD --match policy --pol ipsec --dir in -s 10.10.10.0/24 -o ens33 -p tcp -m tcp --tcp-flags SYN,RST SYN -m tcpmss --mss 1361:1536 -j T>
COMMIT

*filter
:ufw-before-input - [0:0]
:ufw-before-output - [0:0]
:ufw-before-forward - [0:0]
:ufw-not-local - [0:0]
# End required lines

#tell the firewall to forward ESP (Encapsulating Security Payload) traffic so the VPN clients will be able to connect
-A ufw-before-forward --match policy --pol ipsec --dir in --proto esp -s 10.10.10.0/24 -j ACCEPT
-A ufw-before-forward --match policy --pol ipsec --dir out --proto esp -d 10.10.10.0/24 -j ACCEPT

# allow all on loopback
```

- Save and close the file when finished.
- Before we restart the firewall, we will change some network kernel parameters to allow routing from one interface to another.
- Open UFW's kernel parameters configuration file

  Command - sudo nano /etc/ufw/sysctl.conf

```
aravind@aravind-ubuntu:/$ sudo nano /etc/ufw/sysctl.conf
aravind@aravind-ubuntu:/$
```

- In the file, do the highlighted changes (as per the image below)

```
 GNU nano 4.8                        /etc/ufw/sysctl.conf                      Modified

# Uncomment this to allow this host to route packets between interfaces
net/ipv4/ip_forward=1
#net/ipv6/conf/default/forwarding=1
#net/ipv6/conf/all/forwarding=1

# Disable ICMP redirects. ICMP redirects are rarely used but can be used in
# MITM (man-in-the-middle) attacks. Disabling ICMP may disrupt legitimate
# traffic to those sites.
net/ipv4/conf/all/accept_redirects=0
net/ipv4/conf/default/accept_redirects=0
net/ipv6/conf/all/accept_redirects=0
net/ipv6/conf/default/accept_redirects=0

# Do not send ICMP redirects (we are not a router)
# Add the following lines
net/ipv4/conf/all/send_redirects=0
net/ipv4/ip_no_pmtu_disc=1

# Ignore bogus ICMP errors
net/ipv4/icmp_echo_ignore_broadcasts=1
net/ipv4/icmp_ignore_bogus_error_responses=1
net/ipv4/icmp_echo_ignore_all=0
```

- Save the file when you are finished.
- UFW will apply these changes the next time it starts.
- Now, we can enable all of our changes by disabling and re-enabling the firewall

Commands

1. sudo ufw disable
2. sudo ufw enable

```
aravind@aravind-ubuntu:/$ sudo ufw disable
Firewall stopped and disabled on system startup
aravind@aravind-ubuntu:/$ sudo ufw enable
Firewall is active and enabled on system startup
aravind@aravind-ubuntu:/$
```

Step 7 – Testing the VPN connection on windows

- Now that we have everything set up, it's time to try it out.
- First, we will need to copy the CA certificate we created and install it on your client device(s) that will connect to the VPN.
- The easiest way to do this is to log into your server and output the contents of the certificate file

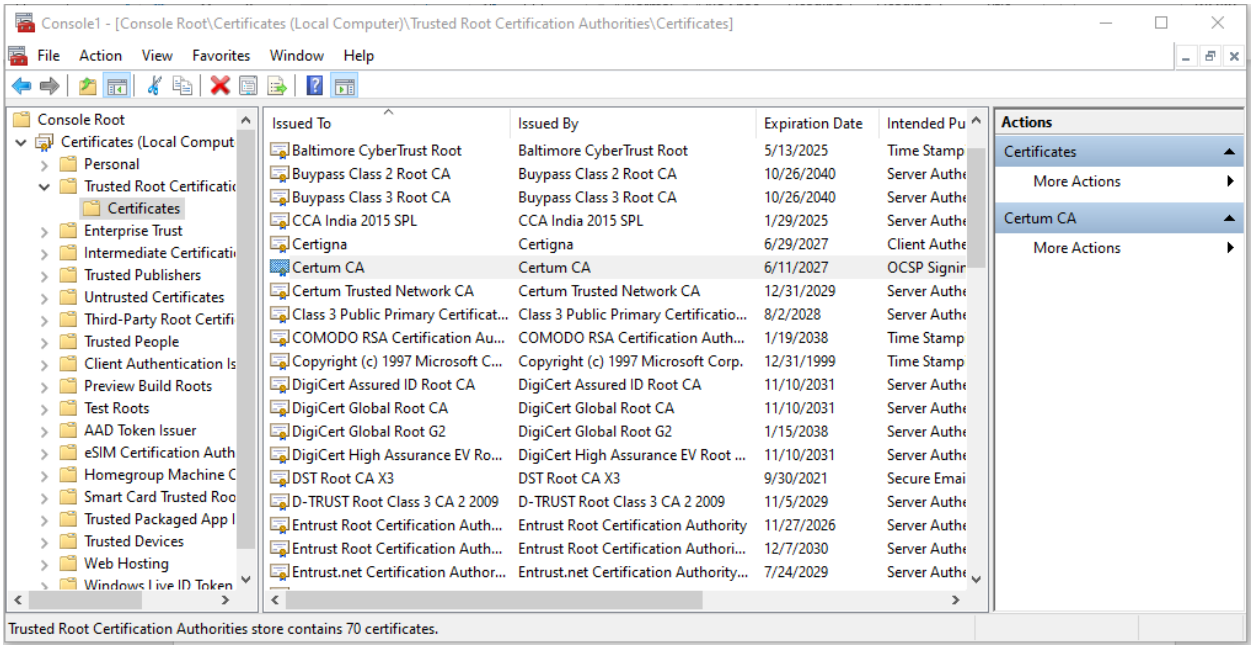
Command - cat /etc/ipsec.d/cacerts/ca-cert.pem

```
aravind@aravind-ubuntu:/$ cat /etc/ipsec.d/cacerts/ca-cert.pem
-----BEGIN CERTIFICATE-----
MIIE8DCCAtigAwIBAgIIfbuypsuPsq0wDQYJKoZIhvcNAQEMBQAwFjEUMBIGA1UE
AxMLVlBOIHJvb3QgQ0EwHhcNMjAwNTMwMjEwMDQ4WhcNMzAwNTI4MjEwMDQ4WjAW
MRQwEgYDVQQDEwtWUE4gcm9vdCBDQTCCAiIwDQYJKoZIhvcNAQEBBQADggIPADCC
AgoCggIBANL+qEMhpRmQ2qIhvvai1Z2EoiUQcFW8O0jr6PON6NI9V9imOdmzLoWc
a3utEe+XgeBuZUA1O4mZ3fzweA0ZylNzTWS86k4nqZQlTLRBqmxK5jF4VcrT8QD3
NcG86bOz1y0XpsHggKA2cDsTemwlLUemKneAoIY5DlOWUB0iKfyg2awtq2COxIqC
31F9ezqpiO3z0F3G8iSpaUMgXGlqPnq3x5G6vZqRscMgXQnsyYGdLtY0BhtdWqXV
2hu80ktT/qaEdV8V+j4912dm3wnByglgFnG6/ydIc5WysHmAyfToPCDE7NVV1OYV
lYlwqwSNC/aFORuVFhFpGsA2kaetPAO+Lo3xPZsR2W8wrcZmU+vJOwOfEKHffOH0
vkV7APi+nLKxA0IHMUh0ROePjhGkNSR8EhMKfqT+GRwUNunuODVeeWpHDjbRqloi
HX7Jh9tc3zMJzp6YTpxdZ+Xn1qxAzxfJR6gq+oI4hrN0JfYi8cufuw07u2igN9gW
lhjU5hOYOOYB4QxHwU3pZqj0PHaSU5Lr18Uli0hIYBl8BkdUhtf9TKrx81JKtRm8
cJhY94/+4TxhtEZGlMDFCCVvNgEESNJZgpoAz29koC4+ipRC34hy0Kld35hXC490
6fRv4ZGsMbs826sUlFI6aeMA2TAX72GZKyl5taNLxO6+nBovNMHzAgMBAAGjQjBA
MA8GA1UdEwEB/wQFMAMBAf8wDgYDVR0PAQH/BAQDAgAAMB0GA1UdDgQWBBSJUIHX
cPSB/BAaTOS26P9WAsxvfzANBgkqhkiG9w0BAQwFAAOCAgEAvf7EACjAX3sMRQCc
ClvorW4o69cf+Kpnwu8MYORlULTDCk2xc/4FbSE6B+K2jzZqWvBskXMF9nUo8yyC
O+kHoCEknxDjBc7M+0TgXx3m979/mJi4sNONc8LrpsnvoYSFqQY/RhfpLAhCoJzK
5f7r4NizSbN5QVDI8SkclTe1NhZiDAdkX4RiJmY2o/sTEacojBdO62TnPgMbom5l
saVFpMd7+F02KybsI1OiPi97KE416v/mz2JdxnXupIn5pfzSsRufKcMp0CT/ogUu
I0eVVS8UR9HagmPB79e9SL0GUIjIfdh/KifGfjgUmbVD2rmty1KdCyDysTsfbNXL
Dpho+3ufenEBsGUaIVeSlhbrC7e5W6dwS2UJ1S7yajietuD4+rXaOPJoC3wckV++
NK5E09cQFiIktqhnc4h9s/cxokRN2i0WiqUKAKyLr+gCASQ57PDfV7lpgZM7bTgF
```

- Copy the output to our computer, including the -----BEGIN CERTIFICATE----- and -----END CERTIFICATE----- lines, and save it to a file with a recognizable name, such as ca-cert.pem.
- Ensure the file we create has the .pem extension

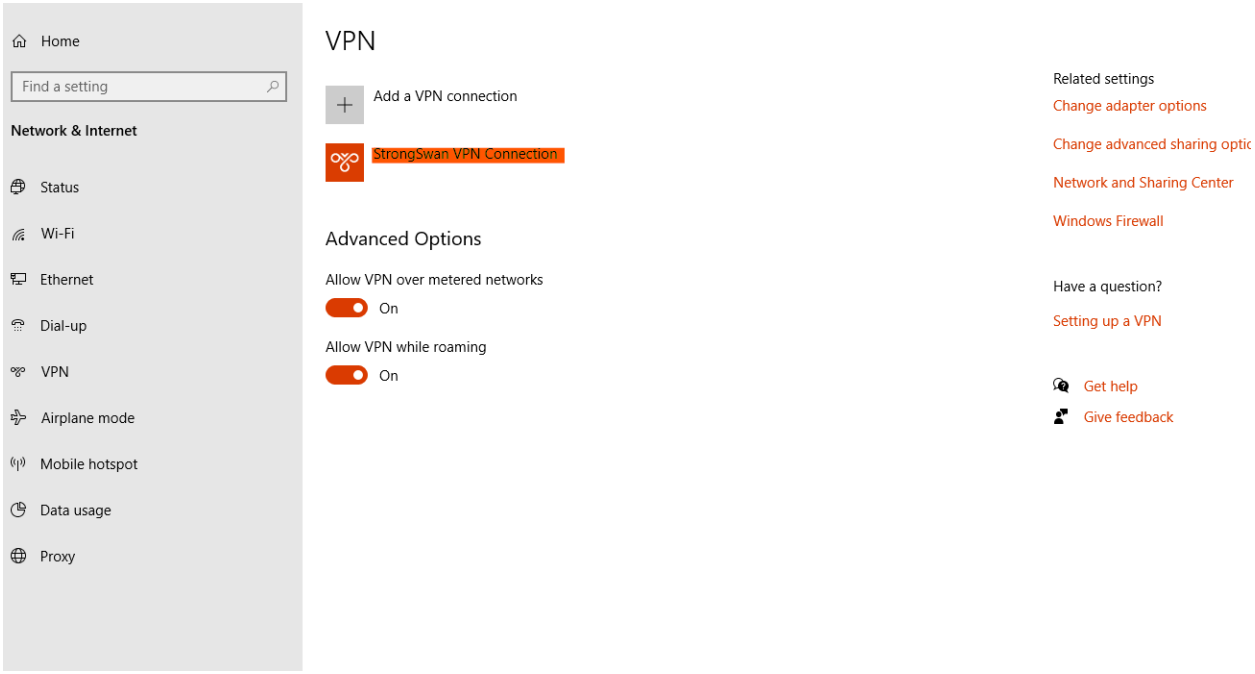
Connecting from windows

- Import the root certificate by the following steps:
    - Start run and enter mmc.exe to launch the Windows Management Console.
    - From the File -> Add or Remove Snap-in -> select Certificates from the list of available snap-ins and click **Add**.
    - We want the VPN to work with any user, so select Computer Account and click Next.
    - We are configuring things on the local computer, so select Local Computer, then click Finish.
    - Under the Console Root node, expand the Certificates (Local Computer) entry, expand Trusted Root Certification Authorities, and then select the Certificates entry:



    - From the Action menu, select All Tasks and click Import to display the Certificate Import Wizard. Click Next to move past the introduction.

    - On the File to Import screen, press the Browse button and select the certificate file that we have saved. Then click Next.
    - Ensure that the Certificate Store is set to Trusted Root Certification Authorities and click Next.
    - Click Finish to import the certificate.
- Now we are going to configure the VPN with the below steps
    - Launch Control Panel, then navigate to the Network and Sharing Center.
    - Click on Set up a new connection or network, then select Connect to a workplace.
    - Select Use my Internet connection (VPN).
    - Enter the VPN server details. Enter the server's domain name or IP address in the Internet address field, then fill in Destination name with something that describes your VPN connection. Then click Done.
- Our new VPN connection will be visible under the list of networks.
- Select the VPN and click Connect. You'll be prompted for your username and password. Type them in, click OK, and you'll be connected.



Conclusion

- We have built a VPN server that uses the IKEv2 protocol.
- Now you can be assured that your online activities will remain secure wherever you go.