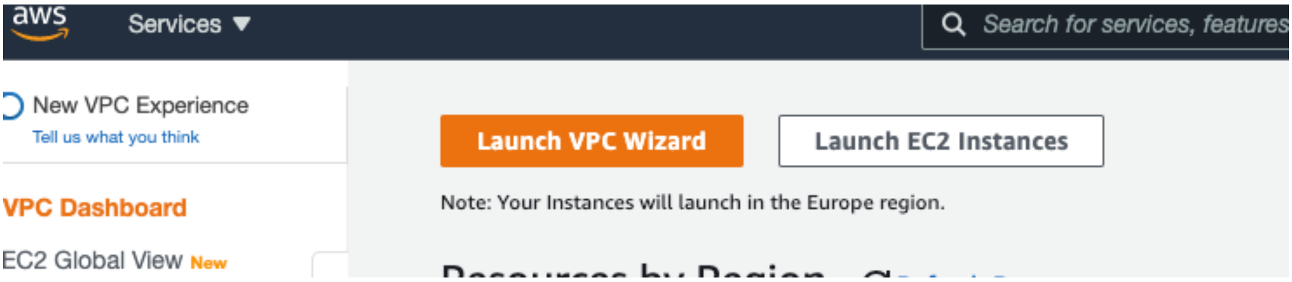
Setup VPN Server in AWS

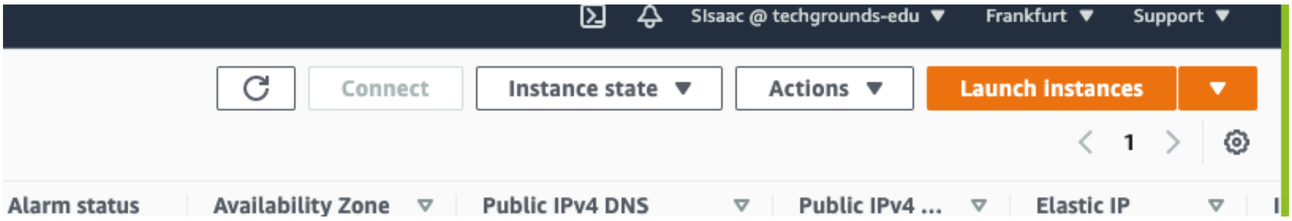
AWS Exercise 381 Open VPN Server

1. Launching VPC wizard on AWS4

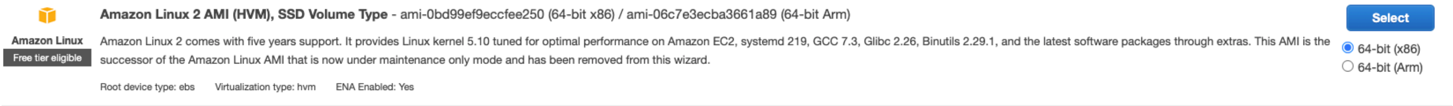


Note: Your Instances will launch in the Europe region.

2. A VPC is created with a public subnet and an Internet gateway
3. Creating a EC2 instance



4. Select Amazon Linux 2 AMI



Amazon Linux 2 AMI (HVM), SSD Volume Type - ami-0bd99ef9eccfee250 (64-bit x86) / ami-06c7e3ecba3661a89 (64-bit Arm)

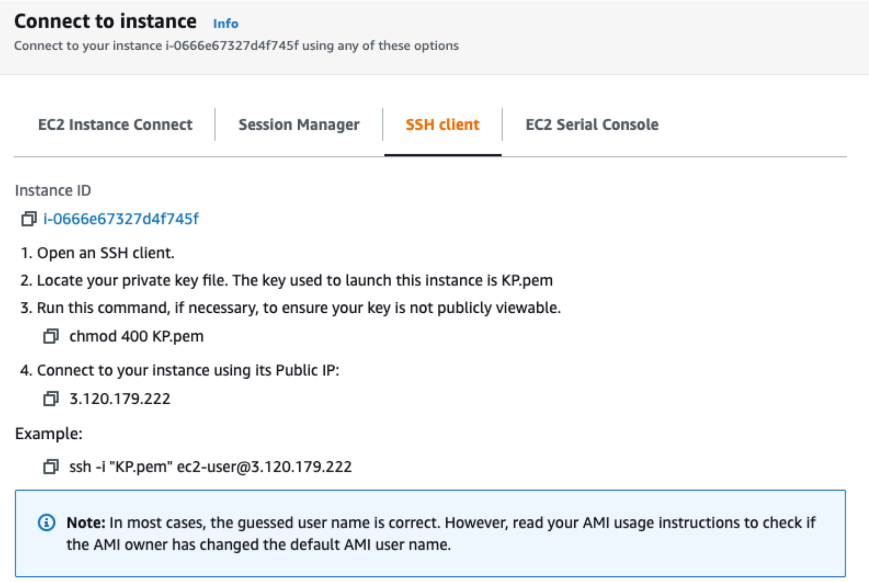
Amazon Linux 2 comes with five years support. It provides Linux kernel 5.10 tuned for optimal performance on Amazon EC2, systemd 219, GCC 7.3, Glibc 2.26, Binutils 2.29.1, and the latest software packages through extras. This AMI is the successor of the Amazon Linux AMI that is now under maintenance only mode and has been removed from this wizard.

Root device type: ebs Virtualization type: hvm ENA Enabled: Yes

● 64-bit (x86)
○ 64-bit (Arm)

5. Select t2.micro (free tier eligible)
6. Enable Auto assign Public IP and select your VPN network
7. Configure your security group and finally launch your EC2 instance

8. Next step will be connecting your SSH with your instance. See image below.



Connect to instance Info

Connect to your instance i-0666e67327d4f745f using any of these options

Instance ID

i-0666e67327d4f745f

1. Open an SSH client.
2. Locate your private key file. The key used to launch this instance is KP.pem
3. Run this command, if necessary, to ensure your key is not publicly viewable.

```
chmod 400 KP.pem
```

4. Connect to your instance using its Public IP:

```
3.120.179.222
```

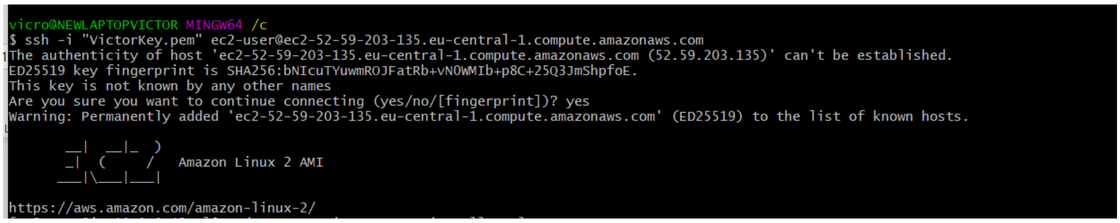
Example:

```
ssh -i "KP.pem" ec2-user@3.120.179.222
```

Note: In most cases, the guessed user name is correct. However, read your AMI usage instructions to check if the AMI owner has changed the default AMI user name.

1. Make Security Keys

2. Make ssh connection with EC2 Server



```
vicro@NEWLAPTOPVICTOR MINGW64 /c
$ ssh -i "VictorKey.pem" ec2-user@ec2-52-59-203-135.eu-central-1.compute.amazonaws.com
The authenticity of host 'ec2-52-59-203-135.eu-central-1.compute.amazonaws.com (52.59.203.135)' can't be established.
ED25519 key fingerprint is SHA256:bNIcuTYuwmROJFatRb+vNOWMIb+p8C+25Q3JmShpfoE.
This key is not known by any other names
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes
Warning: Permanently added 'ec2-52-59-203-135.eu-central-1.compute.amazonaws.com' (ED25519) to the list of known hosts.

       __|  __|_  )
       _|  (     /   Amazon Linux 2 AMI
      ___|\___|___|

https://aws.amazon.com/amazon-linux-2/
```

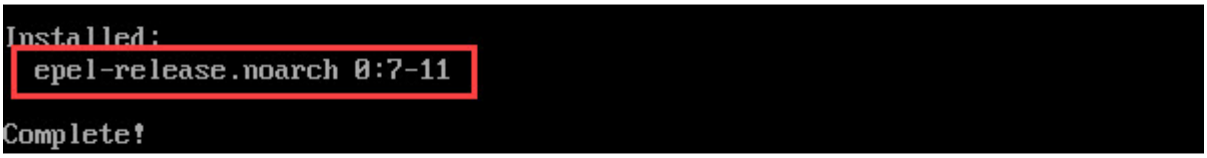
Step 1: Install OpenVPN

1. Update the CentOS repositories and packages by running:

```
yum update -y
```

2. You cannot download the OpenVPN package from the default CentOS repositories. However, OpenVPN is available in the Extra Packages for Enterprise Linux (EPEL) repository. To enable the EPEL repository, run the command:

```
yum install epel-release -y
```



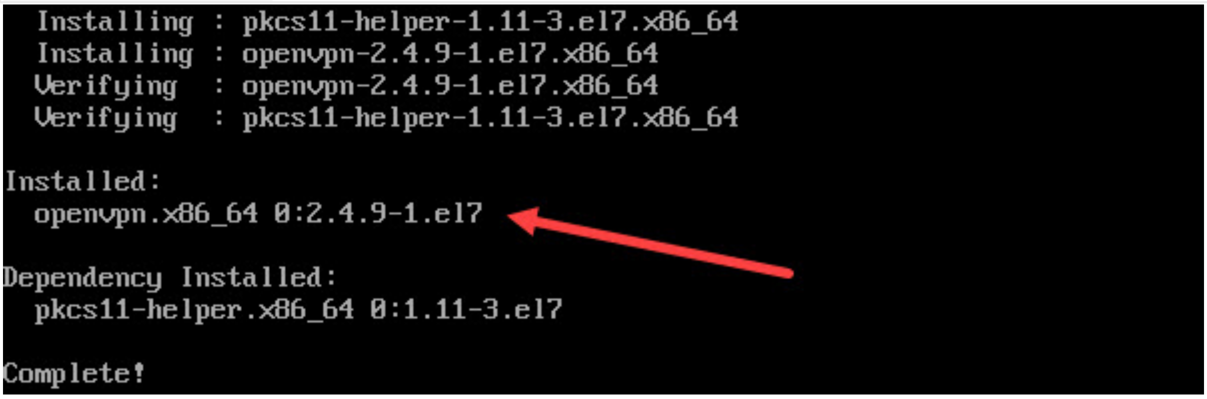
```
Installed:
  epel-release.noarch 0:7-11

Complete!
```

3. Update the repositories again:

```
yum update -y
```

4. You can now install OpenVPN with the command:

```
yum install -y openvpn
```



```
  Installing : pkcs11-helper-1.11-3.el7.x86_64
  Installing : openvpn-2.4.9-1.el7.x86_64
  Verifying  : openvpn-2.4.9-1.el7.x86_64
  Verifying  : pkcs11-helper-1.11-3.el7.x86_64

Installed:
  openvpn.x86_64 0:2.4.9-1.el7

Dependency Installed:
  pkcs11-helper.x86_64 0:1.11-3.el7

Complete!
```

Step 2: Install Easy RSA

The next step is to build a Public Key Infrastructure (PKI). To do this, you need to install **easy RSA**, a CLI utility for creating and managing a PKI Certificate Authority (CA).

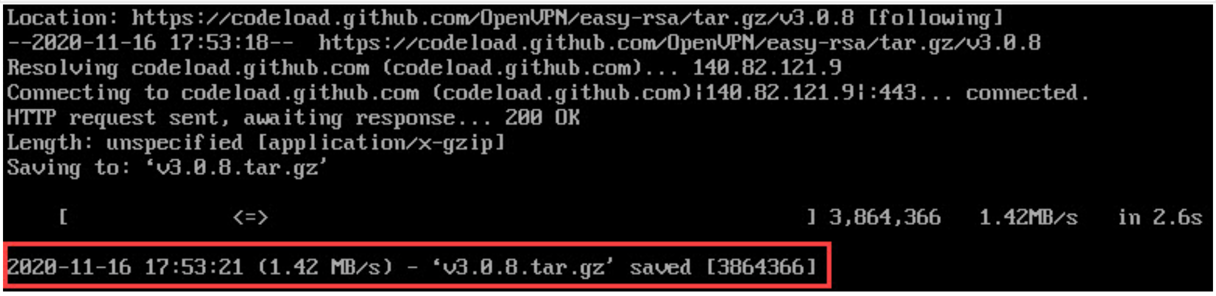
Easy RSA helps you set up an internal certificate authority (CA) and generate SSL key pairs to secure the VPN connections.

1. To download the easy RSA package, use the **wget** command. If you don't have wget on your CenOS system, install it by running:

```
yum install -y wget
```

2. At the time of writing, the latest version of the CLI utility is 3.0.8, which we will download. To use another version, check out easy RSA's release page on GitHub.

```
wget https://github.com/OpenVPN/easy-rsa/archive/v3.0.8.tar.gz
```



```
Location: https://codeload.github.com/OpenVPN/easy-rsa/tar.gz/v3.0.8 [following]
--2020-11-16 17:53:18--  https://codeload.github.com/OpenVPN/easy-rsa/tar.gz/v3.0.8
Resolving codeload.github.com (codeload.github.com)... 140.82.121.9
Connecting to codeload.github.com (codeload.github.com)|140.82.121.9|:443... connected.
HTTP request sent, awaiting response... 200 OK
Length: unspecified [application/x-gzip]
Saving to: 'v3.0.8.tar.gz'

    [            <=>                                   ] 3,864,366   1.42MB/s   in 2.6s

2020-11-16 17:53:21 (1.42 MB/s) - 'v3.0.8.tar.gz' saved [3864366]
```

3. Next, extract the downloaded archive:

```
tar -xf v3.0.8.tar.gz
```

4. Create and move into a new **openvpn** directory:

```
cd /etc/openvpn/
```

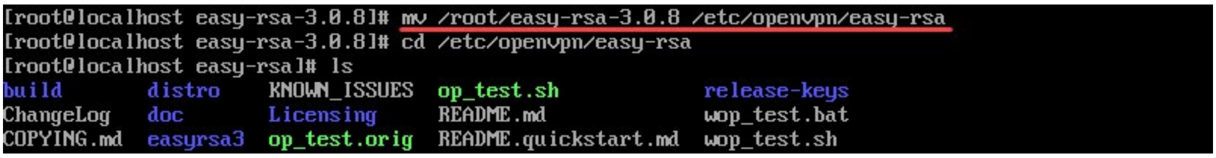
5. Then, create a subdirectory **easy-rsa** under the path **/etc/openvpn**:

```
mkdir /etc/openvpn/easy-rsa
```

6. Move the extracted directory into /etc/openvpn/easy-rsa:

```
mv /root/easy-rsa-3.0.8 /etc/openvpn/easy-rsa
```

To check whether you have successfully moved everything from the **easy-rsa-3.0.8** directory, move into easy-rsa with **cd /etc/openvpn/easy-rsa** and list the content with **ls**. You should see a list of files and folders, as in the image below.



```
[root@localhost easy-rsa-3.0.8]# mv /root/easy-rsa-3.0.8 /etc/openvpn/easy-rsa
[root@localhost easy-rsa-3.0.8]# cd /etc/openvpn/easy-rsa
[root@localhost easy-rsa]# ls
build        distro     KNOWN_ISSUES  op_test.sh             release-keys
ChangeLog    doc        Licensing     README.md              wop_test.bat
COPYING.md   easyrsa3   op_test.orig  README.quickstart.md   wop_test.sh
```

Step 3: Configure OpenVPN

Once you have installed OpenVPN and Easy RSA, you can move on to configuring the OpenVPN server.

The instructions in this section help you set up the basic configuration. You can alter it according to your needs.

Before running any of the commands, make sure to return to the root directory. To do so, type **cd** in the terminal window and hit **Enter**.

1. The first step is to copy the sample **server.conf** file from OpenVPN's documentation directory:

```
cp /usr/share/doc/openvpn-2.4.9/sample/sample-config-files/server.conf /etc/openvpn
```

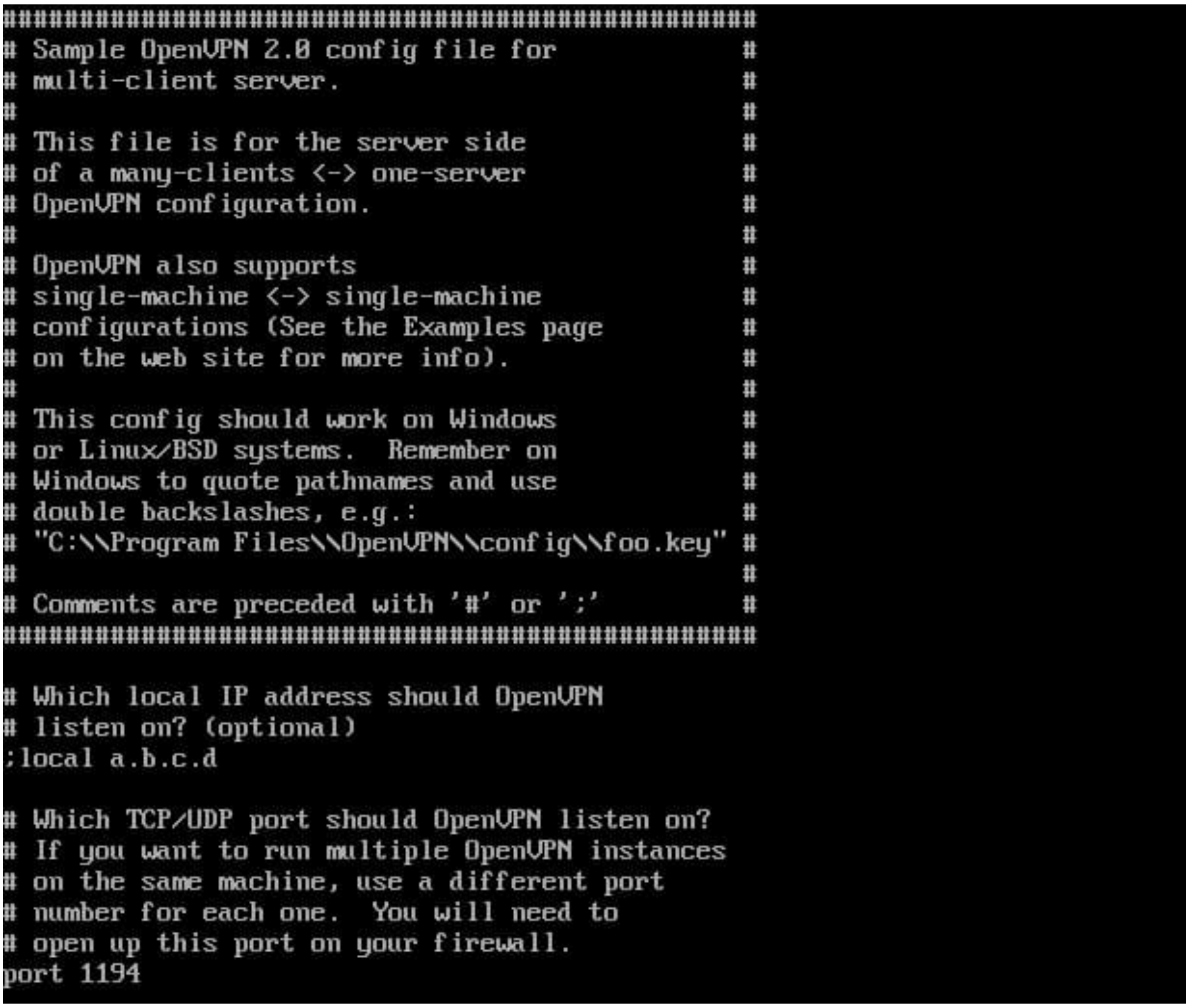
If you cannot find the OpenVPN sample configuration file, search for its location using the **find** command:

```
find / -name server.conf
```

2. Then, open the copied configuration file with a text editor of your choice:

```
nano etc/openvpn/server.conf
```

The command opens the sample OpenVPN config file. The comments in the file begin with a hashtag **#** or a semicolon **;**.



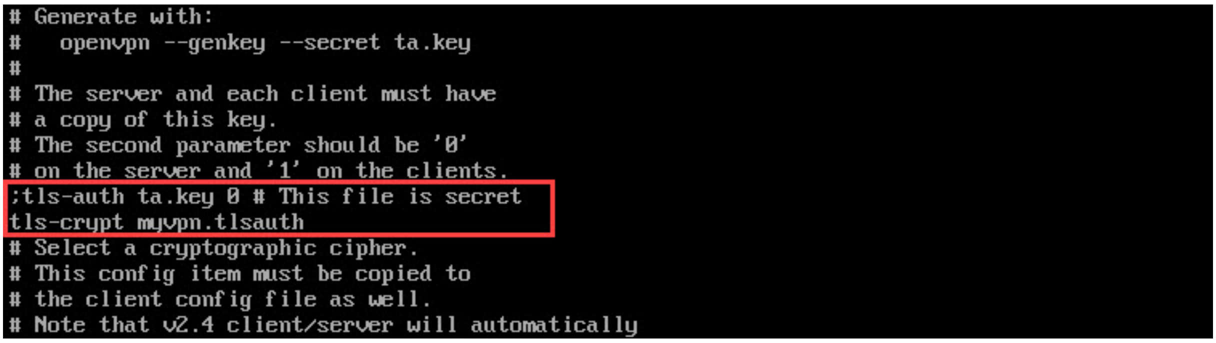
```
#################################################
# Sample OpenVPN 2.0 config file for            #
# multi-client server.                          #
#                                               #
# This file is for the server side              #
# of a many-clients <-> one-server              #
# OpenVPN configuration.                        #
#                                               #
# OpenVPN also supports                         #
# single-machine <-> single-machine             #
# configurations (See the Examples page         #
# on the web site for more info).               #
#                                               #
# This config should work on Windows            #
# or Linux/BSD systems.  Remember on            #
# Windows to quote pathnames and use            #
# double backslashes, e.g.:                     #
# "C:\\Program Files\\OpenVPN\\config\\foo.key" #
#                                               #
# Comments are preceded with '#' or ';'         #
#################################################

# Which local IP address should OpenVPN
# listen on? (optional)
;local a.b.c.d

# Which TCP/UDP port should OpenVPN listen on?
# If you want to run multiple OpenVPN instances
# on the same machine, use a different port
# number for each one.  You will need to
# open up this port on your firewall.
port 1194
```

3. To set up the basic configuration, you need to **uncomment** the following lines by removing the semicolons.

- **topology subnet** (makes the OpenVPN installation function as a subnetwork)
- **push "redirect-gateway def1 bypass-dhcp"** (instructs the client to redirect traffic through the OpenVPN server)
- **push "dhcp-option DNS 208.67.222.222"** (uses an OpenDNS resolver to connect to OpenVPN)
- **push "dhcp-option DNS 208.67.220.220"** (uses an OpenDNS resolver to connect to OpenVPN)
- **user nobody** (runs OpenVPN with no privileges)
- **group nobody** (runs OpenVPN with no privileges)

4. Then, generate a static encryption key to enable TLS authentication. To do that, locate the line **tls-auth ta.key 0** and comment it by adding **;** in front of it. Then, add a new line under it:

```
tls-crypt myvpn.tlsauth
```



```
# Generate with:
#   openvpn --genkey --secret ta.key
#
# The server and each client must have
# a copy of this key.
# The second parameter should be '0'
# on the server and '1' on the clients.
;tls-auth ta.key 0 # This file is secret
tls-crypt myvpn.tlsauth
# Select a cryptographic cipher.
# This config item must be copied to
# the client config file as well.
# Note that v2.4 client/server will automatically
```

Note: The configuration file specifies which DNS servers to use to connect to OpenVPN. By default, it is set to use OpenDNS resolvers, which is how we left it. Alternatively, you can change it to different DNS resolvers by modifying the **push "dhcp-option DNS 208.67.222.222"** and **push "dhcp-option DNS 208.67.220.220" lines**.

5. Save and exit the configuration file.

6. Finally, generate the static encryption key specified in the file with the command:

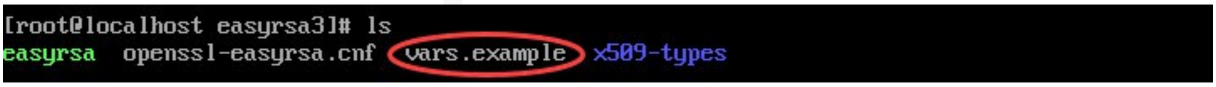
```
openvpn --genkey --secret /etc/openvpn/myvpn.tlsauth
```

Step 4: Generate Keys and Certificates

1. Create a **vars** configuration file using **vars.example** stored in the **/easy-rsa/easyrsa3** directory. Move into the mentioned directory with:

```
cd /etc/openvpn/easy-rsa/easyrsa3
```

2. You can list the contents using the **ls** command to check whether you have the **vars.example** file.

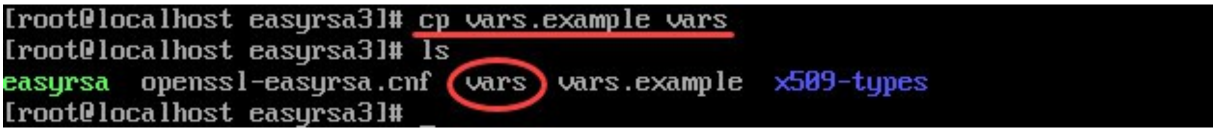


```
[root@localhost easyrsa3]# ls
easyrsa   openssl-easyrsa.cnf  vars.example  x509-types
```

3. Copy the sample file **vars.example** under the name **vars**:

```
cp vars.example vars
```

If you list the files in the directory again, you should have a separate **vars** file that you can use to configure Easy RSA.



```
[root@localhost easyrsa3]# cp vars.example vars
[root@localhost easyrsa3]# ls
easyrsa   openssl-easyrsa.cnf  vars  vars.example  x509-types
[root@localhost easyrsa3]# _
```

4. Open the **vars** file in a text editor of your choice:

```
nano vars
```

5. Scroll through the file and find the lines listed below.

```
#set_var EASYRSA_REQ_COUNTRY "US"
#set_var EASYRSA_REQ_PROVINCE "California"
#set_var EASYRSA_REQ_CITY "San Francisco"
#set_var EASYRSA_REQ_ORG "Copyleft Certificate Co"
#set_var EASYRSA_REQ_EMAIL "me@example.net"
#set_var EASYRSA_REQ_OU "My Organizational Unit"
```

Becomes

```
set_var EASYRSA_REQ_COUNTRY "NL"
set_var EASYRSA_REQ_PROVINCE "Utreg"
set_var EASYRSA_REQ_CITY "Utreg"
set_var EASYRSA_REQ_ORG "Personeelsfeest.nl"
set_var EASYRSA_REQ_EMAIL "info@Personeelsfeest.nl
set_var EASYRSA_REQ_OU "Victor’s Speciale Unit"
```

6. Uncomment the lines by removing **#** and replace the default values with your information.

7. Then, find the line specifying the **KEY_NAME** and change it to **"server"**:

The **KEY_NAME** was not present in the ‘vars’ script. Nevertheless we have added the -export KEY_NAME="server" – to the script inside the ‘vars’ file as it is required.

```
export KEY_NAME="server"
```

8. Finally, change **KEY_CN** to the domain or subdomain that resolves to your server.

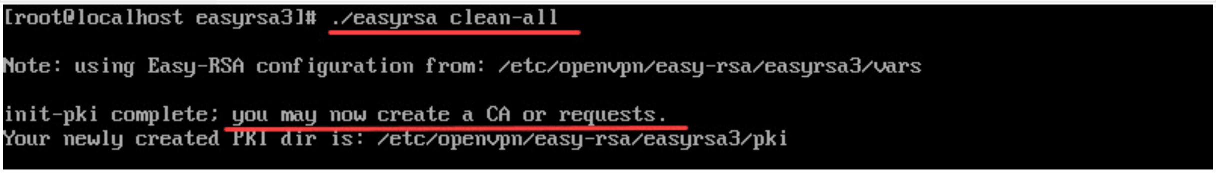
The **KEY_CN** was not present in the ‘vars’ script. Nevertheless we have added the -export KEY_NAME="server" – to the script inside the ‘vars’ file as it is required.

```
export KEY_CN=openvpn.ec2-52-59-203-135.eu-central-1.compute.amazonaws.com
```

9. Save and close the file.

10. Clean up any previous keys and generate the certificate authority:

```
./easyrsa clean-all
```

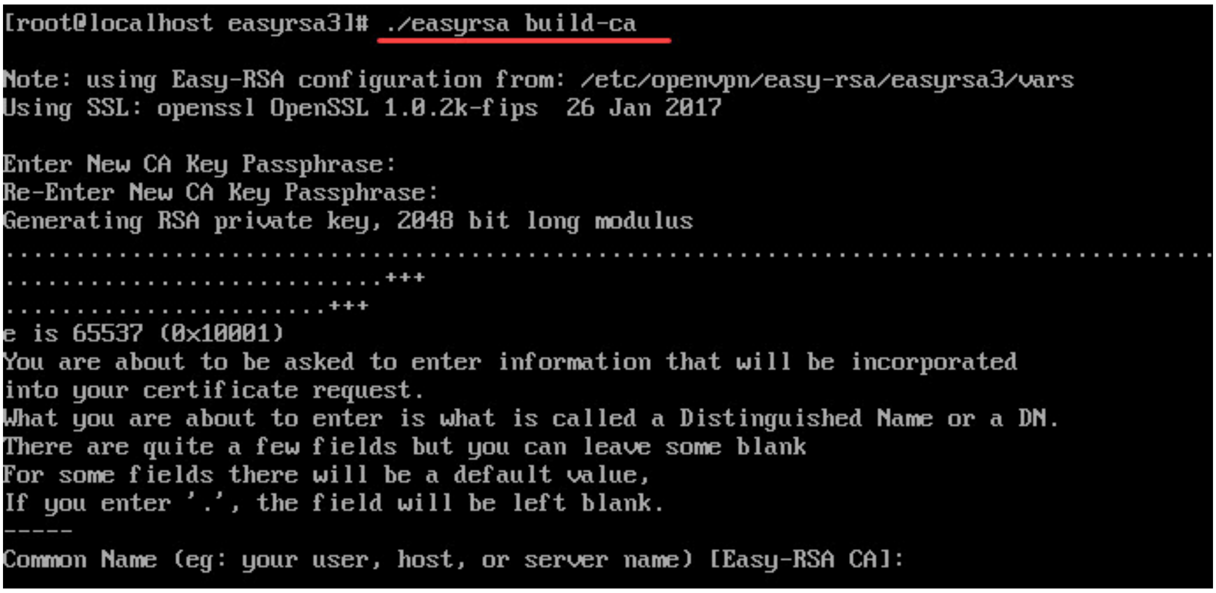


```
[root@localhost easyrsa3]# ./easyrsa clean-all

Note: using Easy-RSA configuration from: /etc/openvpn/easy-rsa/easyrsa3/vars

init-pki complete; you may now create a CA or requests.
Your newly created PKI dir is: /etc/openvpn/easy-rsa/easyrsa3/pki
```

11. Now, you can move on to building the certificate authority with the **build-ca** script. Run the command:

```
./easyrsa build-ca
```

You will be asked to set a CA Key Passphrase and a common name for your CA.

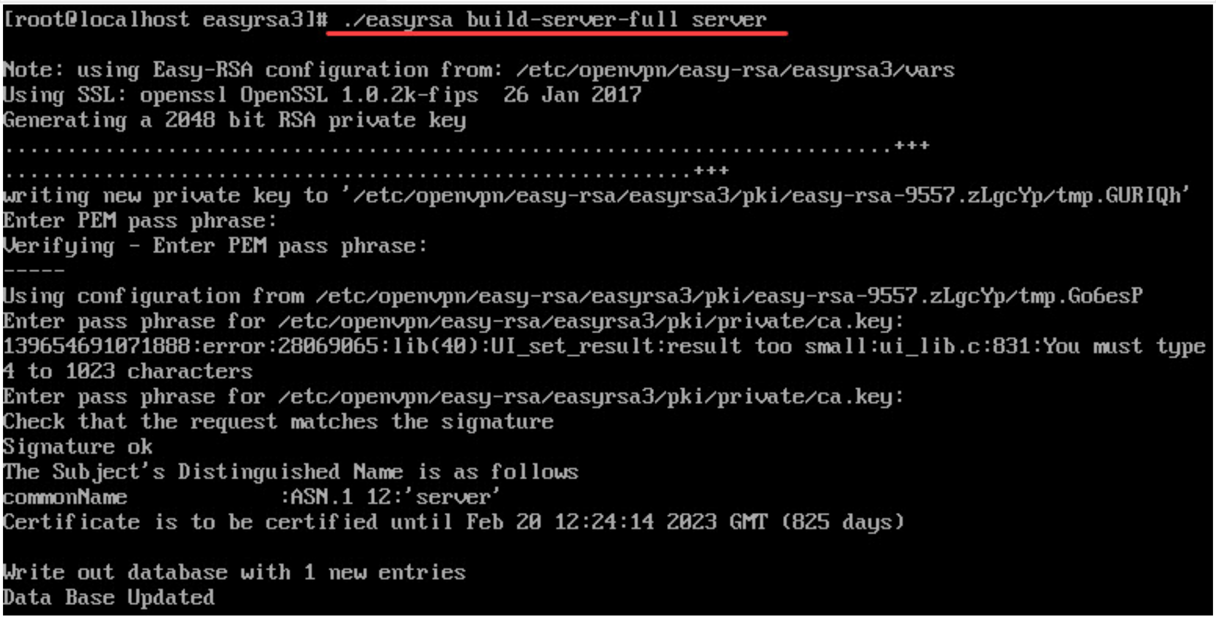


```
[root@localhost easyrsa3]# ./easyrsa build-ca

Note: using Easy-RSA configuration from: /etc/openvpn/easy-rsa/easyrsa3/vars
Using SSL: openssl OpenSSL 1.0.2k-fips  26 Jan 2017

Enter New CA Key Passphrase:
Re-Enter New CA Key Passphrase:
Generating RSA private key, 2048 bit long modulus
...........................................................................
...........................+++
........................+++
e is 65537 (0x10001)
You are about to be asked to enter information that will be incorporated
into your certificate request.
What you are about to enter is what is called a Distinguished Name or a DN.
There are quite a few fields but you can leave some blank
For some fields there will be a default value,
If you enter '.', the field will be left blank.
-----
Common Name (eg: your user, host, or server name) [Easy-RSA CA]:
```

Note: To skip password authentication each time you sign your certificates, you can use the **./easyrsa build-ca nopass** command.

12. Create a key and certificate for the server:

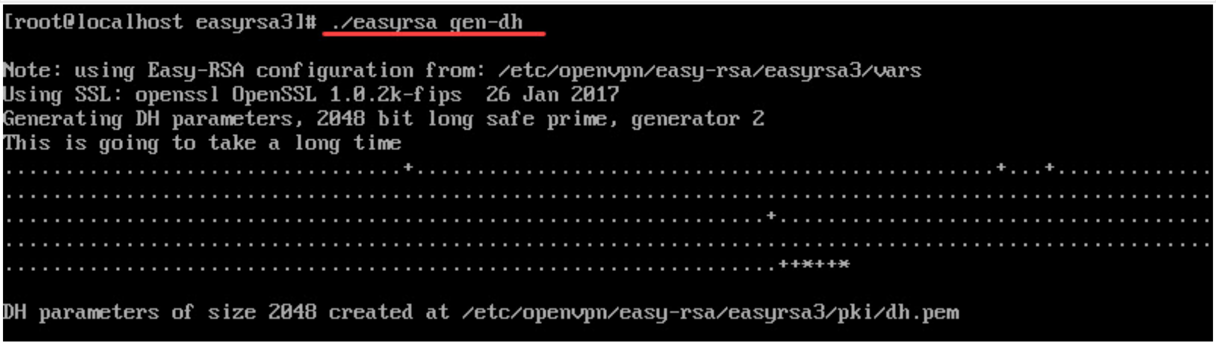
```
./easyrsa build-server-full server
```



```
[root@localhost easyrsa3]# ./easyrsa build-server-full server

Note: using Easy-RSA configuration from: /etc/openvpn/easy-rsa/easyrsa3/vars
Using SSL: openssl OpenSSL 1.0.2k-fips  26 Jan 2017
Generating a 2048 bit RSA private key
.......................................................................+++
.......................................................+++
writing new private key to '/etc/openvpn/easy-rsa/easyrsa3/pki/easy-rsa-9557.zLgcYp/tmp.GURIQh'
Enter PEM pass phrase:
Verifying - Enter PEM pass phrase:
-----
Using configuration from /etc/openvpn/easy-rsa/easyrsa3/pki/easy-rsa-9557.zLgcYp/tmp.Go6esP
Enter pass phrase for /etc/openvpn/easy-rsa/easyrsa3/pki/private/ca.key:
139654691071888:error:28069065:lib(40):UI_set_result:result too small:ui_lib.c:831:You must type
4 to 1023 characters
Enter pass phrase for /etc/openvpn/easy-rsa/easyrsa3/pki/private/ca.key:
Check that the request matches the signature
Signature ok
The Subject's Distinguished Name is as follows
commonName            :ASN.1 12:'server'
Certificate is to be certified until Feb 20 12:24:14 2023 GMT (825 days)

Write out database with 1 new entries
Data Base Updated
```

13. Next, generate a Diffie-Hellman key exchange file by running:

```
./easyrsa gen-dh
```

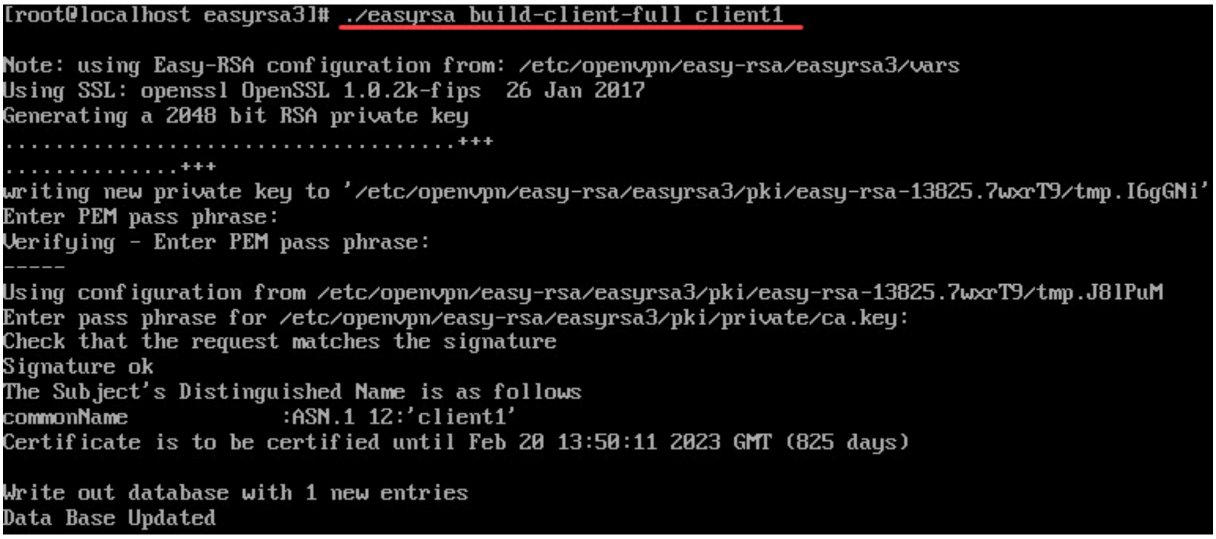


```
[root@localhost easyrsa3]# ./easyrsa gen-dh

Note: using Easy-RSA configuration from: /etc/openvpn/easy-rsa/easyrsa3/vars
Using SSL: openssl OpenSSL 1.0.2k-fips  26 Jan 2017
Generating DH parameters, 2048 bit long safe prime, generator 2
This is going to take a long time
................................+..............................................+...+............
................................................................................................
.............................................................+..................................
................................................................................................
..............................................................++*++*

DH parameters of size 2048 created at /etc/openvpn/easy-rsa/easyrsa3/pki/dh.pem
```

14. You also need a certificate for each client. Generate them on the server and then copy them on the client machine.

With the following command, we create a certificate and key for **client1**. You can modify the command by using a name of your choice.

```
./easyrsa build-client-full client1
```



```
[root@localhost easyrsa3]# ./easyrsa build-client-full client1

Note: using Easy-RSA configuration from: /etc/openvpn/easy-rsa/easyrsa3/vars
Using SSL: openssl OpenSSL 1.0.2k-fips  26 Jan 2017
Generating a 2048 bit RSA private key
....................................+++
..............+++
writing new private key to '/etc/openvpn/easy-rsa/easyrsa3/pki/easy-rsa-13825.7wxrT9/tmp.I6gGNi'
Enter PEM pass phrase:
Verifying - Enter PEM pass phrase:
-----
Using configuration from /etc/openvpn/easy-rsa/easyrsa3/pki/easy-rsa-13825.7wxrT9/tmp.J8lPuM
Enter pass phrase for /etc/openvpn/easy-rsa/easyrsa3/pki/private/ca.key:
Check that the request matches the signature
Signature ok
The Subject's Distinguished Name is as follows
commonName            :ASN.1 12:'client1'
Certificate is to be certified until Feb 20 13:50:11 2023 GMT (825 days)

Write out database with 1 new entries
Data Base Updated
```

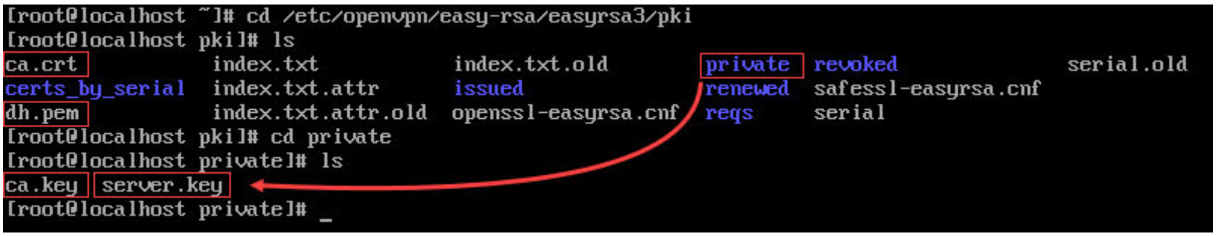
15. Once you have generated the keys and certificates, copy them from **pki** into the **openvpn** directory. To do so, navigate to the **pki** directory by running:

```
cd /etc/openvpn/easy-rsa/easyrsa3/pki
```

You need to copy four files in total:

- ca.crt
- dh.pem
- ca.key
- server.key

The first two files (**ca.crt** and **dh.pem**) are stored in the **pki** directory, while **ca.key** and **server.key** are in a subdirectory **pki/private**.



```
[root@localhost ~]# cd /etc/openvpn/easy-rsa/easyrsa3/pki
[root@localhost pki]# ls
ca.crt           index.txt           index.txt.old        private  revoked            serial.old
certs_by_serial  index.txt.attr      issued               renewed  safessl-easyrsa.cnf
dh.pem           index.txt.attr.old  openssl-easyrsa.cnf  reqs     serial
[root@localhost pki]# cd private
[root@localhost private]# ls
ca.key  server.key
[root@localhost private]# _
```

Therefore, copy **ca.crt** and **dh.pem** into the **openvpn** directory first:

```
cp ca.crt dh.pem /etc/openvpn
```

Then, move into the subdirectory **private**, and copy **ca.key** and **server.key** by running:

```
cd private
cp ca.key server.key/etc/openvpn
```

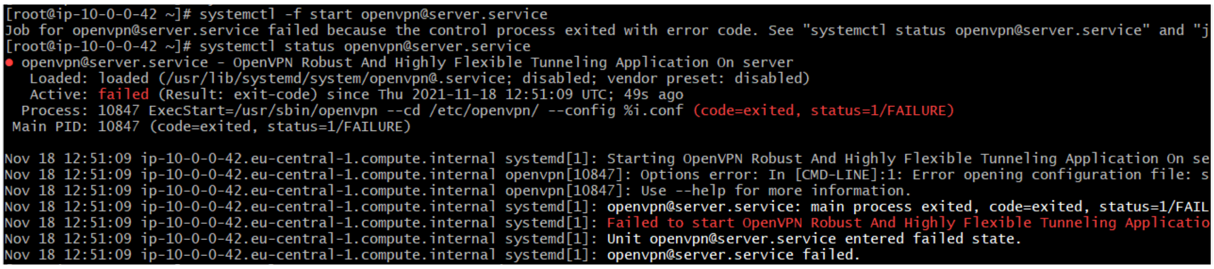
SKIP Step 5: Firewall and Routing Configuration
Set Firewall Rules

As the EC2 instance has a firewall of itself, making an extra firewall could give errors.

Step 6: Start OpenVPN

1. To start the OpenVPN service, run the command:

```
systemctl -f start openvpn@server.service
```



```
[root@ip-10-0-0-42 ~]# systemctl -f start openvpn@server.service
Job for openvpn@server.service failed because the control process exited with error code. See "systemctl status openvpn@server.service" and "j
[root@ip-10-0-0-42 ~]# systemctl status openvpn@server.service
● openvpn@server.service - OpenVPN Robust And Highly Flexible Tunneling Application On server
   Loaded: loaded (/usr/lib/systemd/system/openvpn@.service; disabled; vendor preset: disabled)
   Active: failed (Result: exit-code) since Thu 2021-11-18 12:51:09 UTC; 49s ago
  Process: 10847 ExecStart=/usr/sbin/openvpn --cd /etc/openvpn/ --config %i.conf (code=exited, status=1/FAILURE)
 Main PID: 10847 (code=exited, status=1/FAILURE)

Nov 18 12:51:09 ip-10-0-0-42.eu-central-1.compute.internal systemd[1]: Starting OpenVPN Robust And Highly Flexible Tunneling Application On se
Nov 18 12:51:09 ip-10-0-0-42.eu-central-1.compute.internal openvpn[10847]: Options error: In [CMD-LINE]:1: Error opening configuration file: s
Nov 18 12:51:09 ip-10-0-0-42.eu-central-1.compute.internal openvpn[10847]: Use --help for more information.
Nov 18 12:51:09 ip-10-0-0-42.eu-central-1.compute.internal systemd[1]: openvpn@server.service: main process exited, code=exited, status=1/FAIL
Nov 18 12:51:09 ip-10-0-0-42.eu-central-1.compute.internal systemd[1]: Failed to start OpenVPN Robust And Highly Flexible Tunneling Applicatio
Nov 18 12:51:09 ip-10-0-0-42.eu-central-1.compute.internal systemd[1]: Unit openvpn@server.service entered failed state.
Nov 18 12:51:09 ip-10-0-0-42.eu-central-1.compute.internal systemd[1]: openvpn@server.service failed.
```

Conclusion

We have spent 4 days completing this exercise. Unfortunately we did not manage to complete in successfully. Reason for this, are the errors that have occurred which can be broke down in 2 possibilities.

1. The double firewall that was installed in our process.

   As AWS already provides in a Security Group / Firewall, the extra firewall (Firewalld) installed on the server might work against the system, resulting in errors.

2. The script in the ‘vars’ file was incomplete

   We have added the missing lines as required. We did not manage to conclude if this was beneficial or not.