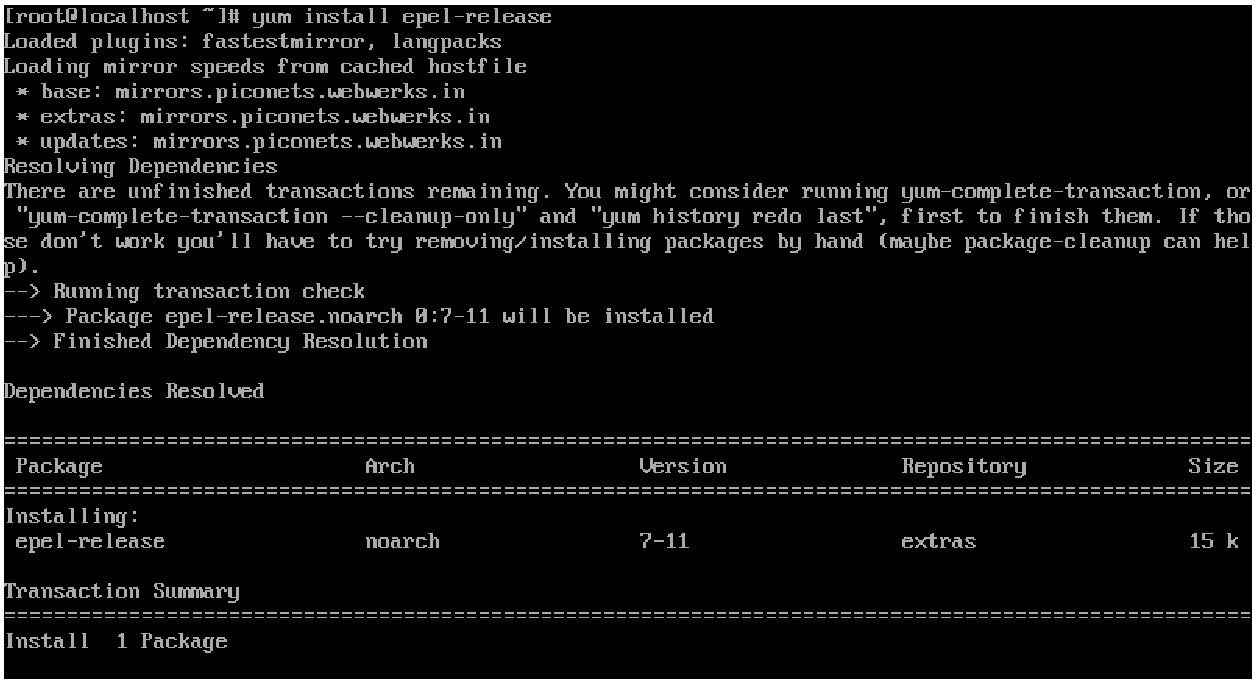
Qno1

Setup a VPN server in VM1(CentOS), we use OpenVPN for this purpose.

To install OpenVPN we will need the EPEL repository, we install EPEL repositories using the following command:

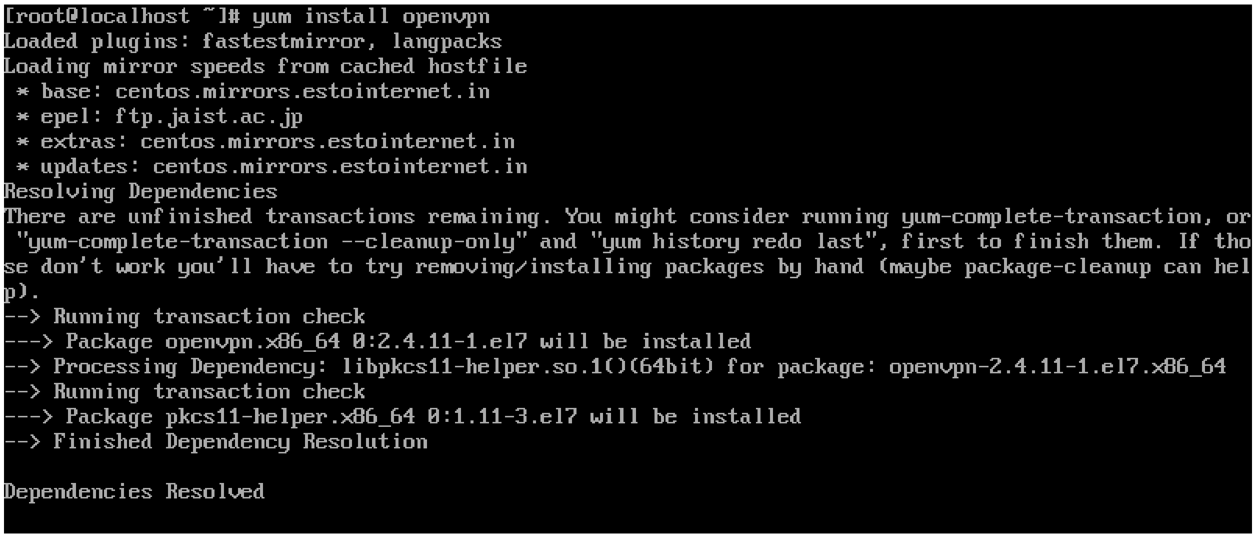
- yum install epel-release



```
[root@localhost ~]# yum install epel-release
Loaded plugins: fastestmirror, langpacks
Loading mirror speeds from cached hostfile
 * base: mirrors.piconets.webwerks.in
 * extras: mirrors.piconets.webwerks.in
 * updates: mirrors.piconets.webwerks.in
Resolving Dependencies
There are unfinished transactions remaining. You might consider running yum-complete-transaction, or
 "yum-complete-transaction --cleanup-only" and "yum history redo last", first to finish them. If tho
se don't work you'll have to try removing/installing packages by hand (maybe package-cleanup can hel
p).
--> Running transaction check
---> Package epel-release.noarch 0:7-11 will be installed
--> Finished Dependency Resolution

Dependencies Resolved

====================================================================================================
 Package                     Arch                  Version              Repository            Size
====================================================================================================
Installing:
 epel-release                noarch                7-11                 extras                15 k

Transaction Summary
====================================================================================================
Install  1 Package
```

To install OpenVPN server, we install OpenVPN using the following command:

- yum install openvpn

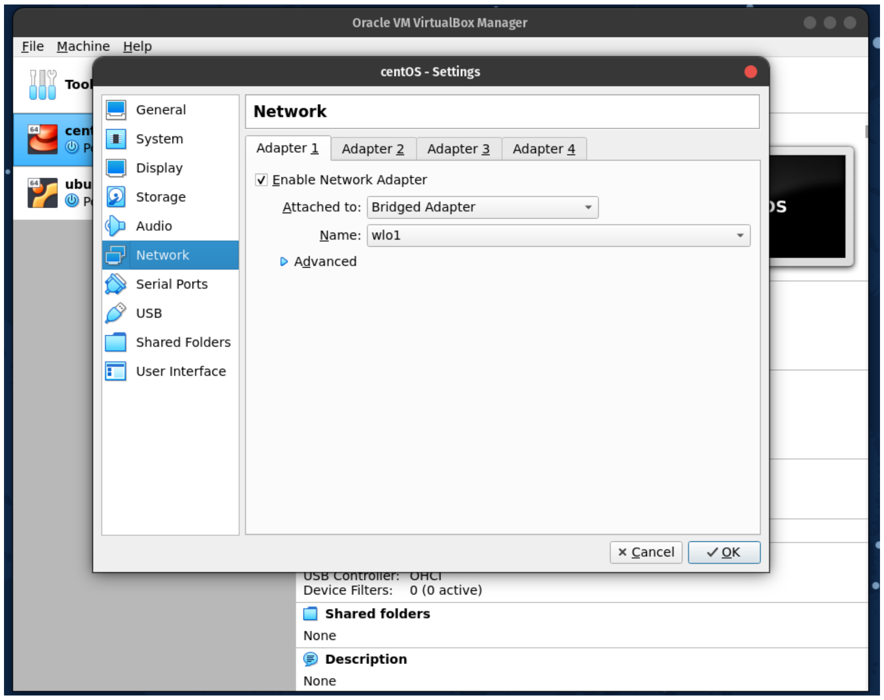


```
[root@localhost ~]# yum install openvpn
Loaded plugins: fastestmirror, langpacks
Loading mirror speeds from cached hostfile
 * base: centos.mirrors.estointernet.in
 * epel: ftp.jaist.ac.jp
 * extras: centos.mirrors.estointernet.in
 * updates: centos.mirrors.estointernet.in
Resolving Dependencies
There are unfinished transactions remaining. You might consider running yum-complete-transaction, or
 "yum-complete-transaction --cleanup-only" and "yum history redo last", first to finish them. If tho
se don't work you'll have to try removing/installing packages by hand (maybe package-cleanup can hel
p).
--> Running transaction check
---> Package openvpn.x86_64 0:2.4.11-1.el7 will be installed
--> Processing Dependency: libpkcs11-helper.so.1()(64bit) for package: openvpn-2.4.11-1.el7.x86_64
--> Running transaction check
---> Package pkcs11-helper.x86_64 0:1.11-3.el7 will be installed
--> Finished Dependency Resolution

Dependencies Resolved
```

Qno1(a)

First VM has two Network Interface one for WAN and another for LAN.

- Adaptor 1


Oracle VM VirtualBox Manager

centOS - Settings

Network

Adapter 1 | Adapter 2 | Adapter 3 | Adapter 4

☑ Enable Network Adapter

Attached to: Bridged Adapter

Name: wlo1

Advanced

USB Controller: OHCI
Device Filters: 0 (0 active)

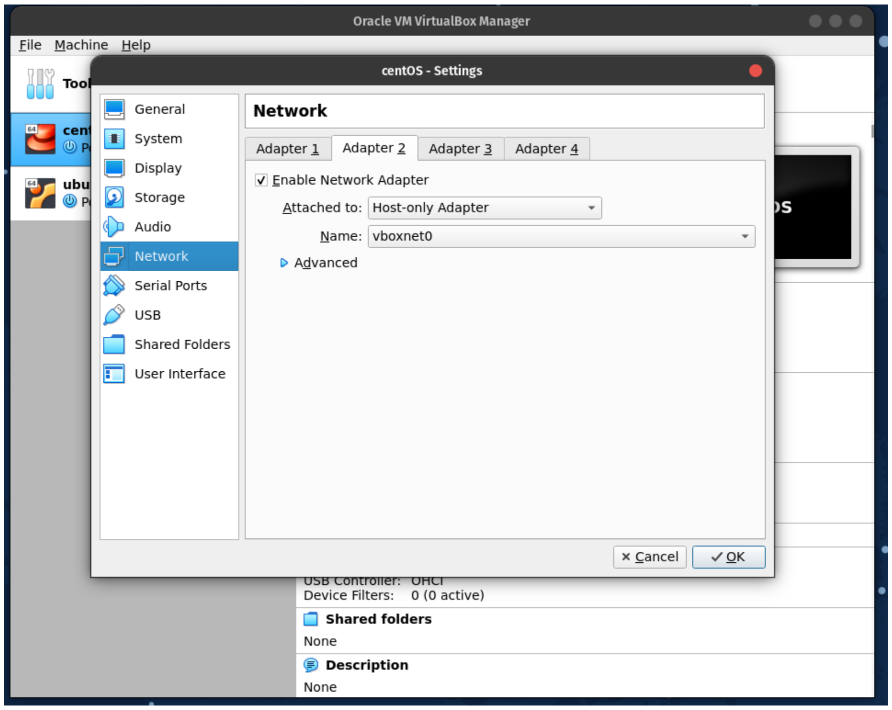
Shared folders

None

Description

None

- Adaptor 2


Oracle VM VirtualBox Manager

centOS - Settings

Network

Adapter 1 | Adapter 2 | Adapter 3 | Adapter 4

☑ Enable Network Adapter

Attached to: Host-only Adapter

Name: vboxnet0

Advanced

USB Controller: OHCI
Device Filters: 0 (0 active)

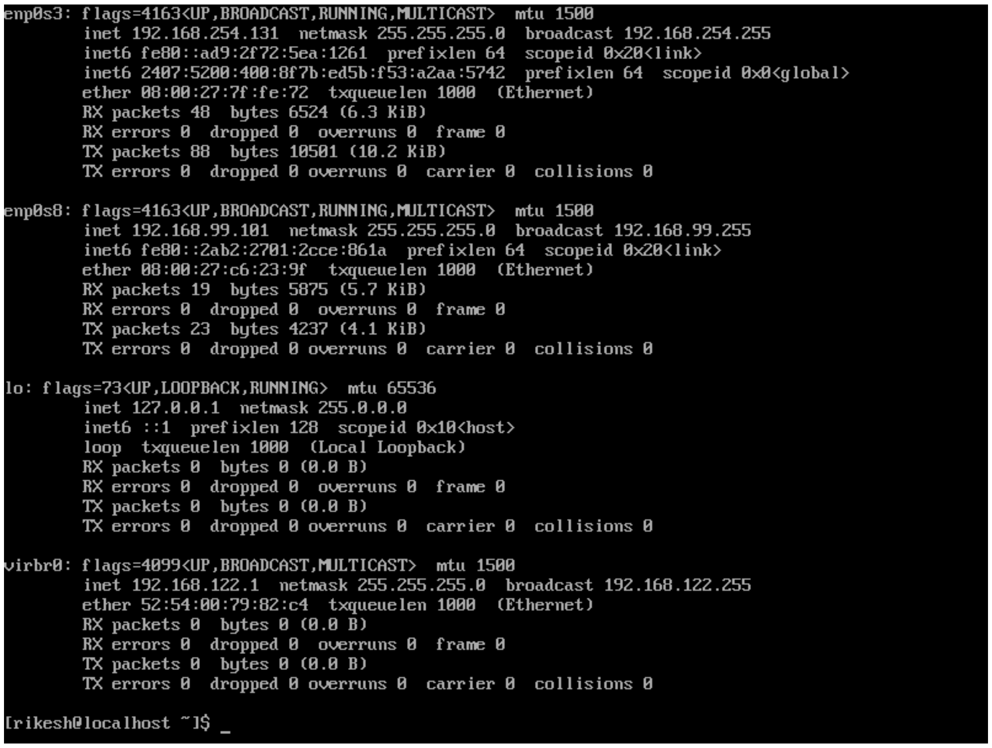
Shared folders

None

Description

None

the configuration/ information of all the network interfaces currently in operation on the system is checked using ifconfig

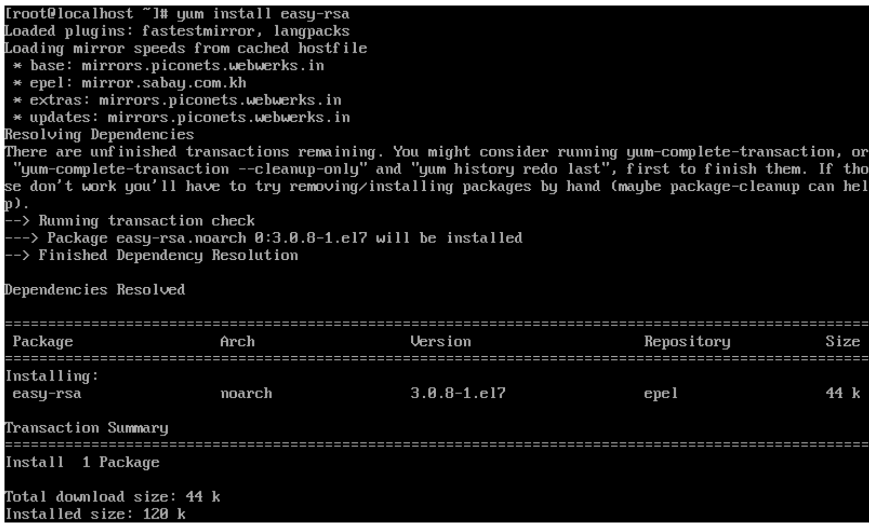


```
enp0s3: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
        inet 192.168.254.131  netmask 255.255.255.0  broadcast 192.168.254.255
        inet6 fe80::ad9:2f72:5ea:1261  prefixlen 64  scopeid 0x20<link>
        inet6 2407:5200:400:8f7b:ed5b:f53:a2aa:5742  prefixlen 64  scopeid 0x0<global>
        ether 08:00:27:7f:fe:72  txqueuelen 1000  (Ethernet)
        RX packets 48  bytes 6524 (6.3 KiB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 88  bytes 10501 (10.2 KiB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

enp0s8: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
        inet 192.168.99.101  netmask 255.255.255.0  broadcast 192.168.99.255
        inet6 fe80::2ab2:2701:2cce:861a  prefixlen 64  scopeid 0x20<link>
        ether 08:00:27:c6:23:9f  txqueuelen 1000  (Ethernet)
        RX packets 19  bytes 5875 (5.7 KiB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 23  bytes 4237 (4.1 KiB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

lo: flags=73<UP,LOOPBACK,RUNNING>  mtu 65536
        inet 127.0.0.1  netmask 255.0.0.0
        inet6 ::1  prefixlen 128  scopeid 0x10<host>
        loop  txqueuelen 1000  (Local Loopback)
        RX packets 0  bytes 0 (0.0 B)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 0  bytes 0 (0.0 B)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

virbr0: flags=4099<UP,BROADCAST,MULTICAST>  mtu 1500
        inet 192.168.122.1  netmask 255.255.255.0  broadcast 192.168.122.255
        ether 52:54:00:79:82:c4  txqueuelen 1000  (Ethernet)
        RX packets 0  bytes 0 (0.0 B)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 0  bytes 0 (0.0 B)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

[rikesh@localhost ~]$ _
```

<u>Qno1(b)</u>

To create certificates files for both server and client to connect to server and export client certificates to the client vm.
Install easy-rsa first to create certificates using the command:

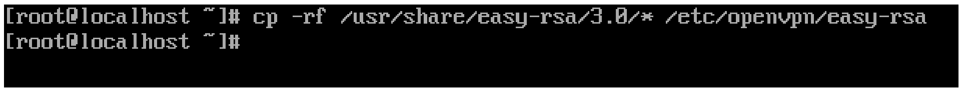
- yum install easy-rsa



```
[root@localhost ~]# yum install easy-rsa
Loaded plugins: fastestmirror, langpacks
Loading mirror speeds from cached hostfile
 * base: mirrors.piconets.webwerks.in
 * epel: mirror.sabay.com.kh
 * extras: mirrors.piconets.webwerks.in
 * updates: mirrors.piconets.webwerks.in
Resolving Dependencies
There are unfinished transactions remaining. You might consider running yum-complete-transaction, or
 "yum-complete-transaction --cleanup-only" and "yum history redo last", first to finish them. If tho
se don't work you'll have to try removing/installing packages by hand (maybe package-cleanup can hel
p).
--> Running transaction check
---> Package easy-rsa.noarch 0:3.0.8-1.el7 will be installed
--> Finished Dependency Resolution

Dependencies Resolved

====================================================================================================
 Package                Arch                 Version                    Repository          Size
====================================================================================================
Installing:
 easy-rsa               noarch               3.0.8-1.el7                epel                44 k

Transaction Summary
====================================================================================================
Install  1 Package

Total download size: 44 k
Installed size: 120 k
```

Now make a new directory for easy-rsa to store the certificates, keys:

- mkdir -p /etc/openvpn/easy-rsa/keys
  - *p*- make required directories
- cp -rf /usr/share/easy-rsa/2.0/* /etc/openvpn/easy-rsa
  - *r*- copy recursively
  - *f*- force this/ accept

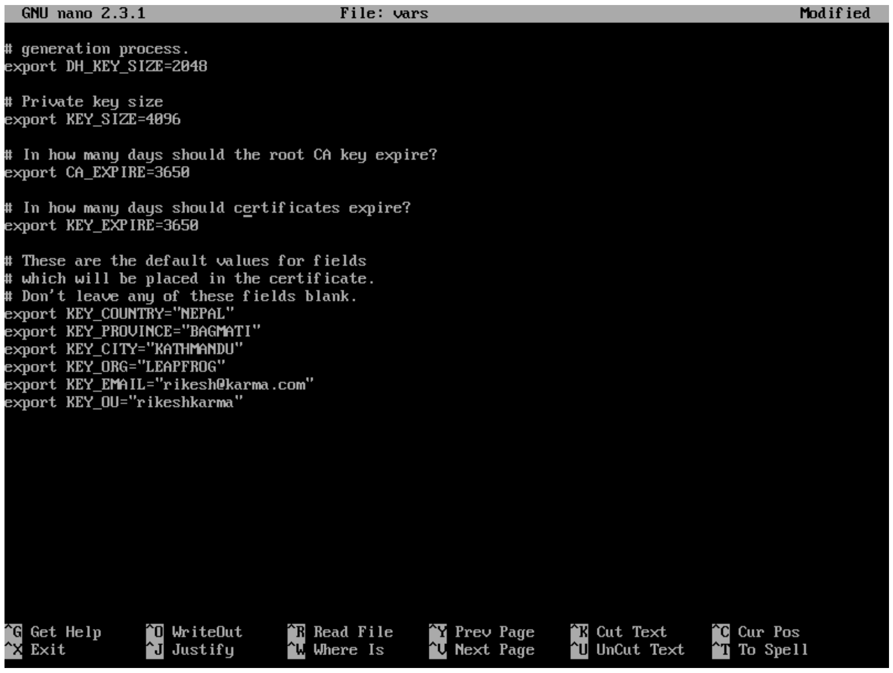


```
[root@localhost ~]# cp -rf /usr/share/easy-rsa/3.0/* /etc/openvpn/easy-rsa
[root@localhost ~]#
```

Change the variables file in the easy-rsa folder

- nano /etc/openvpn/easy-rsa/vars

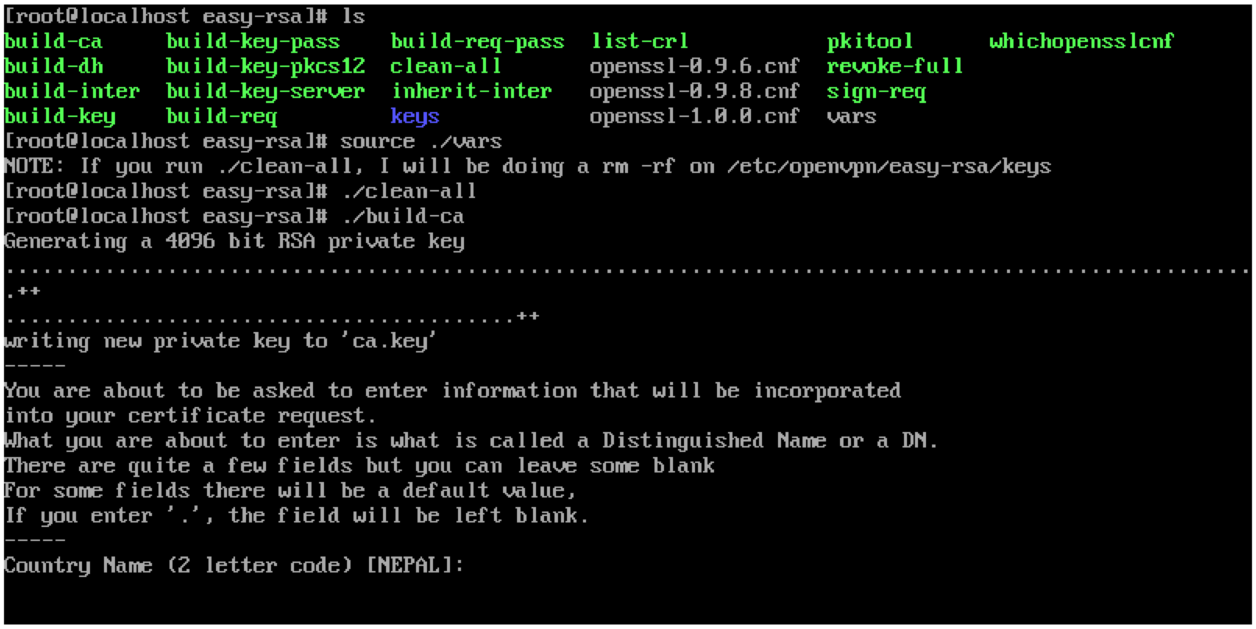
Edit as per requirement



```
  GNU nano 2.3.1                  File: vars                                  Modified

# generation process.
export DH_KEY_SIZE=2048

# Private key size
export KEY_SIZE=4096

# In how many days should the root CA key expire?
export CA_EXPIRE=3650

# In how many days should certificates expire?
export KEY_EXPIRE=3650

# These are the default values for fields
# which will be placed in the certificate.
# Don't leave any of these fields blank.
export KEY_COUNTRY="NEPAL"
export KEY_PROVINCE="BAGMATI"
export KEY_CITY="KATHMANDU"
export KEY_ORG="LEAPFROG"
export KEY_EMAIL="rikesh@karma.com"
export KEY_OU="rikeshkarma"

^G Get Help    ^O WriteOut    ^R Read File   ^Y Prev Page   ^K Cut Text    ^C Cur Pos
^X Exit        ^J Justify     ^W Where Is    ^V Next Page   ^U UnCut Text  ^T To Spell
```

Now we build the security of our server security certificates and keys. while on the /etc/openvpn/easy-rsa, we use following commands:

- source ./vars
  - to build our certificate of authority
- ./clean-all
  - clean if any keys are already existing
- /build-ca
  - signing our server and client's certificates
- ./build-key-server $(hostname)
  - add our host name to the script file

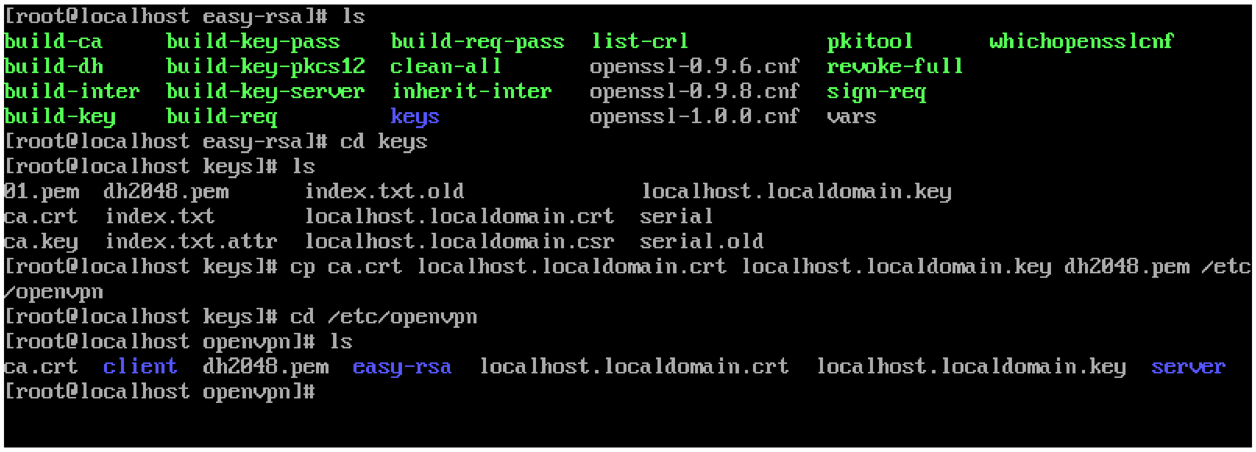


```
[root@localhost easy-rsa]# ls
build-ca     build-key-pass    build-req-pass  list-crl          pkitool      whichopensslcnf
build-dh     build-key-pkcs12  clean-all       openssl-0.9.6.cnf  revoke-full
build-inter  build-key-server  inherit-inter   openssl-0.9.8.cnf  sign-req
build-key    build-req         keys            openssl-1.0.0.cnf  vars
[root@localhost easy-rsa]# source ./vars
NOTE: If you run ./clean-all, I will be doing a rm -rf on /etc/openvpn/easy-rsa/keys
[root@localhost easy-rsa]# ./clean-all
[root@localhost easy-rsa]# ./build-ca
Generating a 4096 bit RSA private key
.....................................................................................................
.++
........................................++
writing new private key to 'ca.key'
-----
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.
-----
Country Name (2 letter code) [NEPAL]:
```

- ./build-dh
  - building our diffie-hellman
  - function to exchange keys securely over the internet or over a network

Now we copy the server certificates and keys to the openvpn folder using the command:

- cd /etc/openvpn/easy-rsa/keys
- cp ca.crt localhost.localdomain.crt localhost.localdomain.key dh2048.pem /etc/openvpn

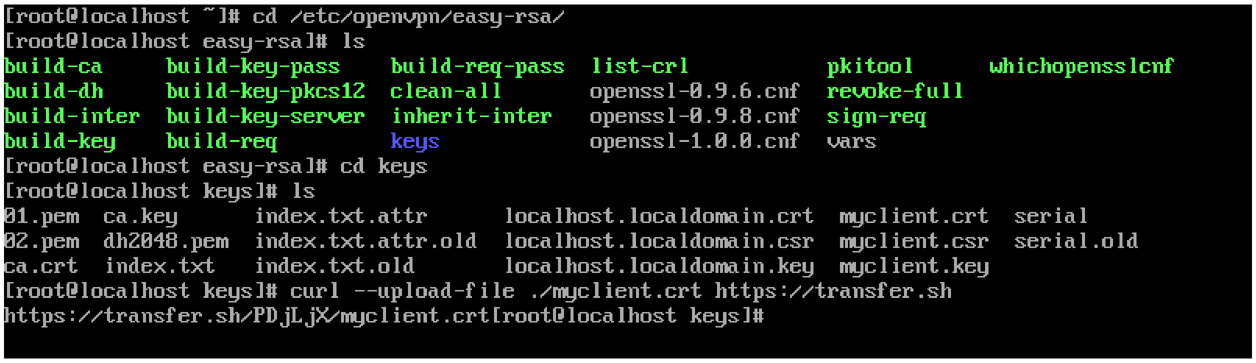


```
[root@localhost easy-rsa]# ls
build-ca     build-key-pass    build-req-pass  list-crl          pkitool      whichopensslcnf
build-dh     build-key-pkcs12  clean-all       openssl-0.9.6.cnf  revoke-full
build-inter  build-key-server  inherit-inter   openssl-0.9.8.cnf  sign-req
build-key    build-req         keys            openssl-1.0.0.cnf  vars
[root@localhost easy-rsa]# cd keys
[root@localhost keys]# ls
01.pem  dh2048.pem      index.txt.old            localhost.localdomain.key
ca.crt  index.txt       localhost.localdomain.crt  serial
ca.key  index.txt.attr  localhost.localdomain.csr  serial.old
[root@localhost keys]# cp ca.crt localhost.localdomain.crt localhost.localdomain.key dh2048.pem /etc
/openvpn
[root@localhost keys]# cd /etc/openvpn
[root@localhost openvpn]# ls
ca.crt  client  dh2048.pem  easy-rsa  localhost.localdomain.crt  localhost.localdomain.key  server
[root@localhost openvpn]#
```

Generating Client Keys

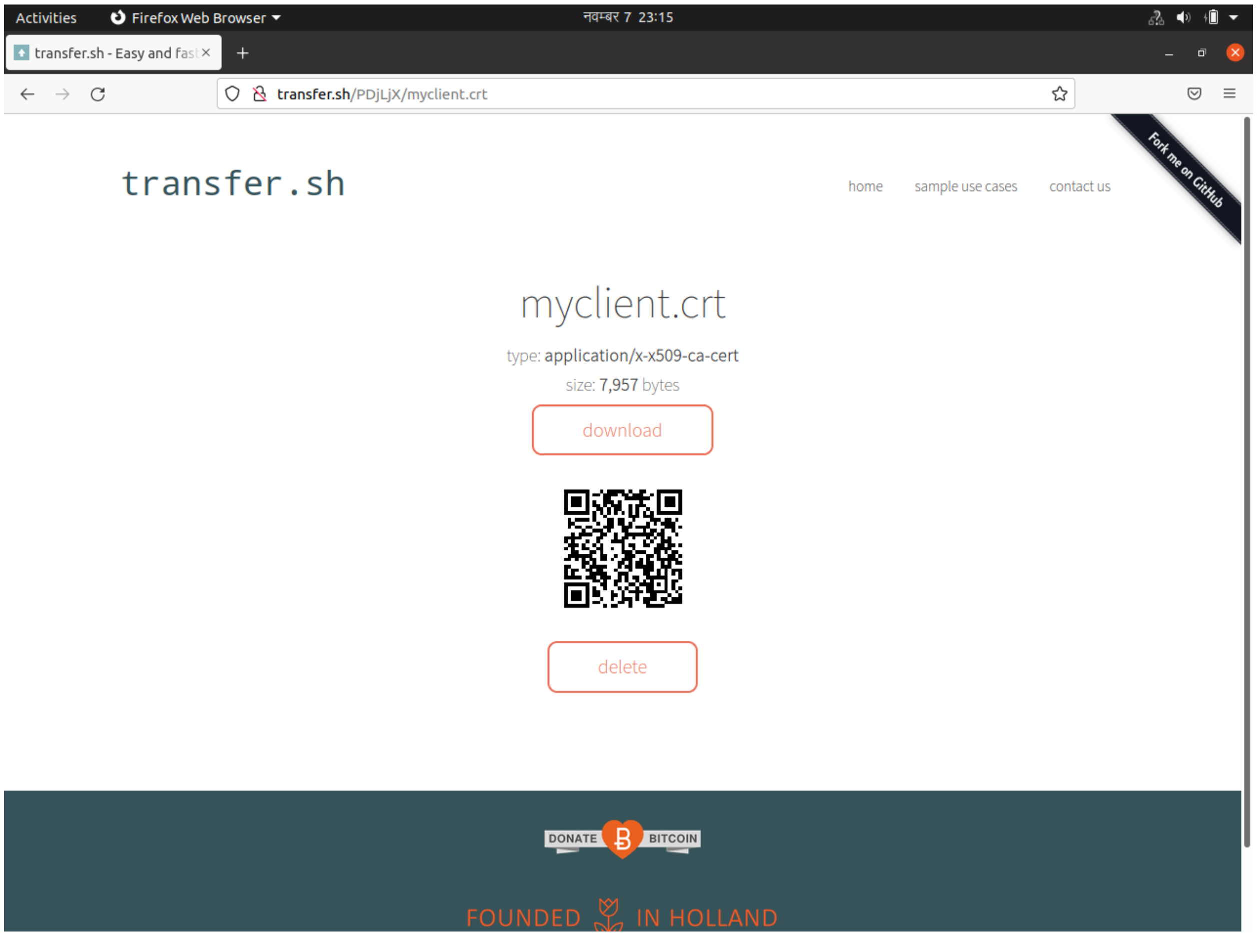
To build the client keys we navigate to `/etc/openvpn/easy-rsa` and run the command

- source ./vars
- ./build-key client


```
[root@localhost ~]# cd /etc/openvpn/easy-rsa/
[root@localhost easy-rsa]# ls
build-ca      build-key-pass     build-req-pass  list-crl          pkitool      whichopensslcnf
build-dh      build-key-pkcs12   clean-all       openssl-0.9.6.cnf  revoke-full
build-inter   build-key-server   inherit-inter   openssl-0.9.8.cnf  sign-req
build-key     build-req          keys            openssl-1.0.0.cnf  vars
[root@localhost easy-rsa]# cd keys
[root@localhost keys]# ls
01.pem  ca.key      index.txt.attr      localhost.localdomain.crt  myclient.crt  serial
02.pem  dh2048.pem  index.txt.attr.old  localhost.localdomain.csr  myclient.csr  serial.old
ca.crt  index.txt   index.txt.old       localhost.localdomain.key  myclient.key
[root@localhost keys]# curl --upload-file ./myclient.crt https://transfer.sh
https://transfer.sh/PDjLjX/myclient.crt[root@localhost keys]#
```

Now change the directory to `keys` folder and verify myclient keys, we should see `myclient.crt myclient.key`.
To export client certificates to the client vm we can use flash drive, email or SSH/SCP client like Filezilla.

1. let's send the file via cURL
2. navigate to the `/etc/openvpn/easy-rsa/keys`
3. use the following command:
   a. curl --upload-file ./myclient.crt https://transfer.sh



myclient.crt

type: application/x-x509-ca-cert

size: 7,957 bytes

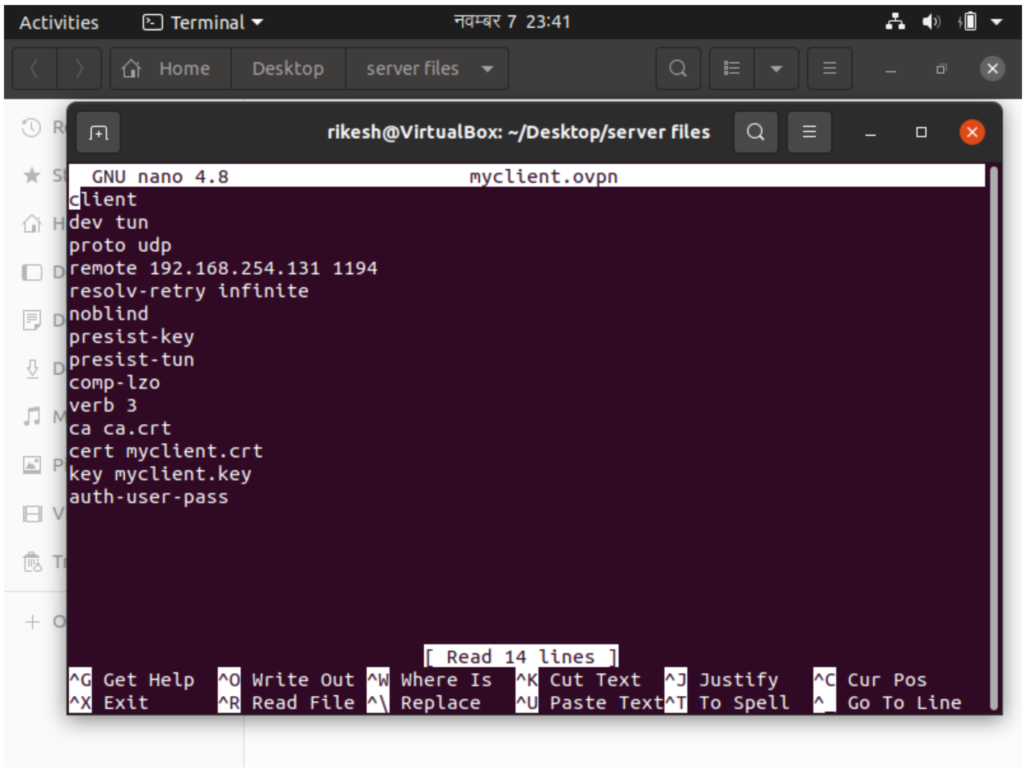
<u>Qno2</u>

Create a OpenVPN client connect configuration file used to connect to the server

Created a text file myclient.ovpn using `touch myclient.ovpn` in the same directory as the files received from the server machine/ VM.

Add the following lines which helps to connect to the server and this also provides the certificates:

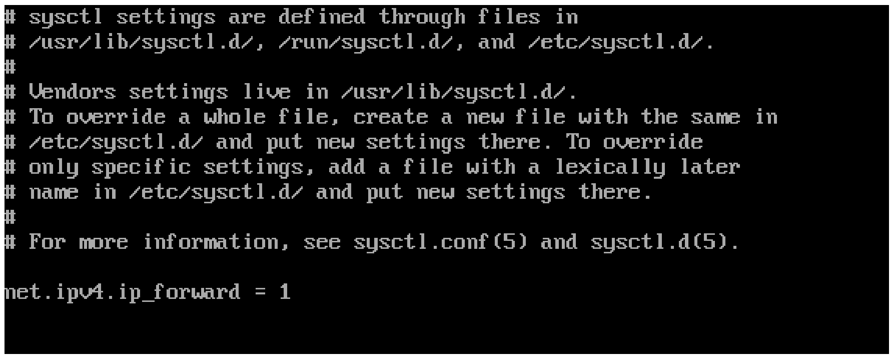
```
client
dev tun
proto udp
remote 192.168.254.131 1194
resolv-retry infinite
nobind
persist-key
persist-tun
comp-lzo
verb 3
ca ca.crt
cert myclient.crt
key myclient.key
auth-user-pass
```



```
  GNU nano 4.8                    myclient.ovpn
client
dev tun
proto udp
remote 192.168.254.131 1194
resolv-retry infinite
noblind
presist-key
presist-tun
comp-lzo
verb 3
ca ca.crt
cert myclient.crt
key myclient.key
auth-user-pass

                         [ Read 14 lines ]
^G Get Help  ^O Write Out ^W Where Is  ^K Cut Text  ^J Justify   ^C Cur Pos
^X Exit      ^R Read File ^\ Replace   ^U Paste Text^T To Spell  ^_ Go To Line
```

On the server, the first VM(CentOS) enable to forward packets through its network interface
Use sysctl to allow IP packet forwarding. Add the following line to the _sysctl.conf_ file

- nano /etc/sysctl.conf
    - net.ipv4.ip_forward = 1



```
# sysctl settings are defined through files in
# /usr/lib/sysctl.d/, /run/sysctl.d/, and /etc/sysctl.d/.
#
# Vendors settings live in /usr/lib/sysctl.d/.
# To override a whole file, create a new file with the same in
# /etc/sysctl.d/ and put new settings there. To override
# only specific settings, add a file with a lexically later
# name in /etc/sysctl.d/ and put new settings there.
#
# For more information, see sysctl.conf(5) and sysctl.d(5).

net.ipv4.ip_forward = 1
```

- sysctl -p

Enable OpenVPN pam authentication module to add user authentication
Using the OpenVPN auth-pam module the OpenVPN server can authenticate using the Linux system users. For this we need to create a PAM service file using the following commands:

- touch /etc/pam.d/openvpn
- nano /etc/pam.d/openvpn

Then we need to add the following two lines to the file:

```
auth required  pam_unix.so shadow  nodelay
account required pam_unix.so
```

Now restart the OpenVPN server

- systemctl stop openvpn@server.service
- systemctl start openvpn@server.service
- systemctl status openvpn@server.service`

Install and configure OpenVPN on the client's machine/ VM2.
Install openvpn-client on VM2(Ubuntu) using the command:

- sudo apt install openvpn

Connect the VM2 to the server/ VM1 using the command:

- sudo openvpn --config client.ovpn

After this OpenVPN will be running in the terminal window.

Qno3