OpenVPN in ubuntu 16.

Step 1: Install OpenVPN

```
user@openvpn : ~ $ sudo apt-get update
```

```
user@openvpn : ~ $ sudo apt-get install openvpn easy-rsa
```

Step 2: Set Up the CA Directory

Copy the easy-rsa template directory into the home directory and move into the newly created directory:

```
user@openvpn : ~ $ make-cadir ~/openvpn-ca
```

```
user@openvpn : ~ $ cd ~/openvpn-ca
```

Step 3: Configure the CA Variables

Modify the default values for fields which will be placed in the certificate to look like below.

Don't leave any of these fields blank.

```
user@openvpn : ~/openvpn-ca $ vi vars
```

```
export KEY_COUNTRY="BT"
export KEY_PROVINCE="Paro"
export KEY_CITY="Paro"
export KEY_ORG="BtNOG"
export KEY_EMAIL="openvpn@btnog.bt"
export KEY_OU="BtNOG6"
```

Also change KEY_NAME value which is used to populate the subject field

```
# X509 Subject Field
export KEY_NAME="BtNOG-openvpn"
```

Step 4: Build the Certificate Authority

Build the certificate authority using the set variables and the easy-rsa utilities

```
user@openvpn : ~/openvpn-ca $ source vars
```

```
NOTE: If you run ./clean-all, I will be doing a rm -rf on /home/user/openvpn-ca/keys
```

Make sure that you are operating in a clean environment by typing:

```
user@openvpn : ~/openvpn-ca $ ./clean-all
```

Built the root CA. This will initiate the process of creating the root certificate authority key and certificate. Since we filled out the vars file, all of the values should be populated automatically.

Just press ENTER through the prompts to confirm the selections:

```
user@openvpn : ~/openvpn-ca $ ./build-ca
```

```
Generating a 2048 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) [BT]:
State or Province Name (full name) [Paro]:
Locality Name (eg, city) [Paro]:
Organization Name (eg, company) [BtNOG]:
Organizational Unit Name (eg, section) [BtNOG6]:
Common Name (eg, your name or your server's hostname) [BtNOG CA]:
Name [BtNOG-openvpn]:
Email Address [openvpn@btnog.bt]:
```

Step 5: Create the Server Certificate, Key, and Encryption Files

Create OpenVPN server certificate and key pair

```
user@openvpn : ~/openvpn-ca $ ./build-key-server BtNOG-openvpn
```

```
Generating a 2048 bit RSA private key
.....................................................................................................................................+++
........+++
writing new private key to 'BtNOG-openvpn.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) [BT]:
State or Province Name (full name) [Paro]:
Locality Name (eg, city) [Paro]:
Organization Name (eg, company) [BtNOG]:
Organizational Unit Name (eg, section) [BtNOG6]:
Common Name (eg, your name or your server's hostname) [BtNOG-openvpn]:
Name [BtNOG-openvpn]:
Email Address [openvpn@btnog.bt]:
Please enter the following 'extra' attributes
to be sent with your certificate request
A challenge password []:
An optional company name []:
Using configuration from /home/user/openvpn-ca/openssl-1.0.0.cnf
Check that the request matches the signature
Signature ok
The Subject's Distinguished Name is as follows
countryName :PRINTABLE:'BT'
stateOrProvinceName :PRINTABLE:'Paro'
localityName :PRINTABLE:'Paro'
organizationName :PRINTABLE:'BtNOG'
organizationalUnitName:PRINTABLE:'BtNOG6'
commonName :PRINTABLE:'BtNOG-openvpn'
name :PRINTABLE:'BtNOG-openvpn'
emailAddress :IA5STRING:'openvpn@btnog.bt'
Certificate is to be certified until May 17 08:10:14 2029 GMT (3650 days)
Sign the certificate? [y/n]:y
1 out of 1 certificate requests certified, commit? [y/n]y
Write out database with 1 new entries
Data Base Updated
```

Generate a Diffie-Hellman key to use during key exchange

```
user@openvpn : ~/openvpn-ca $ ./build-dh
```

```
Generating DH parameters, 2048 bit long safe prime, generator 2
This is going to take a long time
.....+.....
.....+.....
.....+.....+.....+.....
.....+.....+.....+.....
+.....+.....+.....
.....+.....+.....
.....
.....+.....
.....+.....+.....+.....
.....+.....+.....+.....
.....+.....
.....+.....+.....+++*++*
```

Generate an HMAC signature to strengthen the server's TLS integrity verification capabilities:

```
user@openvpn : ~/openvpn-ca $ openvpn --genkey --secret keys/ta.key
```

Step 6: Generate a Client Certificate and Key Pair

```
user@openvpn : ~/openvpn-ca $ source vars
```

```
NOTE: If you run ./clean-all, I will be doing a rm -rf on /home/user/openvpn-ca/keys
```

Generate a client certificate and key pair. Leave the challenge password blank and make sure to enter y for the prompts that ask whether to sign and commit the certificate.

```
user@openvpn : ~/openvpn-ca $ ./build-key BtNOG
```

```
Generating a 2048 bit RSA private key
.....+++
.....+++
writing new private key to 'BtNOG1.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) [BT]:
State or Province Name (full name) [Paro]:
Locality Name (eg, city) [Paro]:
Organization Name (eg, company) [BtNOG]:
Organizational Unit Name (eg, section) [BtNOG6]:
Common Name (eg, your name or your server's hostname) [BtNOG1]:
Name [BtNOG-openvpn]:
Email Address [openvpn@btnog.bt]:
Please enter the following 'extra' attributes
to be sent with your certificate request
A challenge password []:
An optional company name []:
Using configuration from /home/user/openvpn-ca/openssl-1.0.0.cnf
Check that the request matches the signature
Signature ok
The Subject's Distinguished Name is as follows
countryName :PRINTABLE:'BT'
stateOrProvinceName :PRINTABLE:'Paro'
localityName :PRINTABLE:'Paro'
organizationName :PRINTABLE:'BtNOG'
organizationalUnitName:PRINTABLE:'BtNOG6'
commonName :PRINTABLE:'BtNOG1'
name :PRINTABLE:'BtNOG-openvpn'
emailAddress :IA5STRING:'openvpn@btnog.bt'
Certificate is to be certified until May 17 08:12:35 2029 GMT (3650 days)
Sign the certificate? [y/n]:y
1 out of 1 certificate requests certified, commit? [y/n]y
Write out database with 1 new entries
Data Base Updated
```

Step 7: Configure the OpenVPN Service

Configure the OpenVPN service using the credentials and files we've generated. Copy the required files to the OpenVPN Directory

```
user@openvpn : ~/openvpn-ca $ cd ~/openvpn-ca/keys
```

```
user@openvpn : ~/openvpn-ca/keys $ sudo cp ca.crt BtNOG-openvpn.crt BtNOG-openvpn.key ta.key dh2048.pem /etc/openvpn
```

```
[sudo] password for user:
```

Next, copy and unzip a sample OpenVPN configuration file into configuration directory so that we can use it as a basis for our setup:

```
user@openvpn : ~/openvpn-ca/keys $ gunzip -c /usr/share/doc/openvpn/examples/sample-config-files/server.conf.gz | sudo
tee /etc/openvpn/server.conf
```

Change the default server configuration to meet your needs:

```
 user@openvpn : ~/openvpn-ca/keys $ sudo vi /etc/openvpn/server.conf
```

Point to Non-Default Credentials

```
cert BtNOG-openvpn.crt
key BtNOG-openvpn.key
```

Push DNS Changes to Redirect All Traffic Through the VPN

```
push "redirect-gateway def1 bypass-dhcp"
push "dhcp-option DNS 208.67.222.222"
push "dhcp-option DNS 208.67.220.220"
```

To enable HMAC, uncomment the tls-auth directive and add the key-direction parameter set to "0":

```
tls-auth ta.key 0 # This file is secret
key-direction 0
```

Select AES-128-CBC cipher since it offers a good level of encryption and is well supported. Remove the ";" to uncomment the cipher AES-128-CBC line:

Set HMAC message digest algorithm to SHA256.

```
cipher AES-128-CBC # AES
auth SHA
```

For users and group setting, remove the ";" at the beginning to uncomment those lines:

```
user nobody
group nogroup
```

Step 8: Adjust the Server Networking Configuration

Allow IP Forwarding by uncommenting the following line

```
user@openvpn : ~/openvpn-ca/keys $ sudo vi /etc/sysctl.conf
```

```
 net.ipv4.ip_forward= 1
```

To read the file and adjust the values for the current session, type:

```
user@openvpn : ~/openvpn-ca/keys $ sudo sysctl -p
```

```
net.ipv4.ip_forward = 1
```

Note the public network interface of your machine.

```
user@openvpn : ~/openvpn-ca/keys $ ip route | grep default
```

```
 default  via 172.16.9.1 dev ens160 onlink
```

Step 9: Start and Enable the OpenVPN Service

```
user@openvpn : ~/openvpn-ca/keys $ sudo systemctl start openvpn@server
```

```
user@openvpn : ~/openvpn-ca/keys $ sudo systemctl status openvpn@server
```

```
● openvpn@server.service - OpenVPN connection to server
Loaded: loaded (/lib/systemd/system/openvpn@.service; disabled; vendor preset
Active:  active (running)  since Mon 2019-05-20 14:26:50 +06; 8s ago
Docs: man:openvpn(8)
https://community.openvpn.net/openvpn/wiki/Openvpn23ManPage
https://community.openvpn.net/openvpn/wiki/HOWTO
Process: 4403 ExecStart=/usr/sbin/openvpn --daemon ovpn-%i --status /run/openv
Main PID: 4407 (openvpn)
CGroup: /system.slice/system-openvpn.slice/openvpn@server.service
└─4407 /usr/sbin/openvpn --daemon ovpn-server --status /run/openvpn/s
May 20 14:26:50 openvpn ovpn-server[4407]:  /sbin/ip addr add dev tun0 local 10.
May 20 14:26:50 openvpn ovpn-server[4407]:  /sbin/ip route add 10.8.0.0/24 via 10
May 20 14:26:50 openvpn ovpn-server[4407]:  GID set to nogroup
May 20 14:26:50 openvpn ovpn-server[4407]:  UID set to nobody
May 20 14:26:50 openvpn ovpn-server[4407]:  UDPv4 link local (bound): [undef]
May 20 14:26:50 openvpn ovpn-server[4407]:  UDPv4 link remote: [undef]
May 20 14:26:50 openvpn ovpn-server[4407]:  MULTI: multi_init called, r=256 v=
May 20 14:26:50 openvpn ovpn-server[4407]:  IFCONFIG POOL: base=10.8.0.4 size=62,
May 20 14:26:50 openvpn ovpn-server[4407]:  IFCONFIG POOL LIST
May 20 14:26:50 openvpn ovpn-server[4407]:  Initialization Sequence Completed

#### Check if  the OpenVPN tun0 interface is available
```

```
user@openvpn : ~/openvpn-ca/keys $ ip addr show tun
```

```
3: tun0: <POINTOPOINT,MULTICAST,NOARP,UP,LOWER_UP> mtu 1500 qdisc pfifo_fast state
UNKNOWN group default qlen 100
link/none
inet 10.8.0.1 peer 10.8.0.2/32 scope global tun
valid_lft forever preferred_lft forever
```

Enable the service so that it starts automatically at boot:

```
user@openvpn : ~/openvpn-ca/keys $ sudo systemctl enable openvpn@server
```

```
Created symlink from /etc/systemd/system/multi-user.target.wants/openvpn@server.service to
/lib/systemd/system/openvpn@.service.
```

Step 10: Create Client Configuration Infrastructure

Create the Client Config Directory Structure with proper permissions:

```
user@openvpn : ~/openvpn-ca/keys $ mkdir -p ~/client-configs/files
```

```
user@openvpn : ~/openvpn-ca/keys $ chmod 700 ~/client-configs/files
```

Copy an example client configuration into your directory to use as base configuration:

```
user@openvpn : ~/openvpn-ca/keys $ cp
```

```
/usr/share/doc/openvpn/examples/sample-config-files/client.conf
~/client-configs/base.conf
```

Make few changes to meet your requirement

```
user@openvpn : ~/openvpn-ca/keys $ vi ~/client-configs/base.conf
```

Change the ip to global openvpn server IP

```
remote 202.144.144.82 1194
```

uncomment the user and group directives by removing the ";":

```
 user nobody

 group nogroup
```

Find the directives that set the ca, cert, and key. Comment out these directives since we will be adding the certs and keys within the file itself:

```
#ca ca.crt
#cert client.crt
#key client.key
```

Set cipher and auth settings as in server.conf file. Add the key-direction directive and set to "1" to work with the server:

```
cipher AES-128-CBC
auth SHA
key-direction 1
```

Create a Configuration Generation Script

```
user@openvpn : ~/openvpn-ca/keys $ vi ~/client-configs/make_config.sh
```

```
#!/bin/bash

# First argument: Client identifier

 KEY_DIR=~/openvpn-ca/keys

 OUTPUT_DIR=~/client-configs/files

 BASE_CONFIG=~/client-configs/base.conf

 cat  ${BASE_CONFIG}  \

 <( echo  -e '<ca>') \

  ${KEY_DIR} /ca.crt \

 <( echo  -e '</ca>\n<cert>') \

  ${KEY_DIR} / ${1} .crt \

 <( echo  -e '</cert>\n<key>') \
```

```
 ${KEY_DIR} / ${1} .key \

<( echo  -e '</key>\n<tls-auth>') \

 ${KEY_DIR} /ta.key \

<( echo  -e '</tls-auth>') \

>  ${OUTPUT_DIR} / ${1} .ovpn
```

```
user@openvpn : ~/openvpn-ca/keys $ chmod 700 ~/client-configs/make_config.sh
```

Step 11: Generate Client Configurations

```
user@openvpn : ~/openvpn-ca/keys $ cd ~/client-configs
```

```
user@openvpn : ~/client-configs $ ./make_config.sh BtNOG
```

```
user@openvpn : ~/client-configs $ ls ~/client-configs/files BtNOG1.ovpn
```

Transfer Configuration to Client Devices

```
user@openvpn : ~/client-configs $ scp /home/user/client-configs/files/BtNOG1.ovpn admin@172.16.8.200:/Users/admin/Desktop
```

```
The authenticity of host '172.16.8.200 (172.16.8.200)' can't be established.
ECDSA key fingerprint is SHA256:Ocy7bBMdYSQnAuyKmpXsA5cY1FkyFYwIM0DcF1Q7M+k.
Are you sure you want to continue connecting (yes/no)? yes
Warning: Permanently added '172.16.8.200' (ECDSA) to the list of known hosts.
Password:
BtNOG1.ovpn 100% 13KB 12.8KB/s 00:
```

```
user@openvpn : ~/client-configs $
```

Client for MAC

https://openvpn.net/vpn-server-resources/connecting-to-access-server-with-macos/

Once installed import the ovpn file copied from server and connect to server.

Worked??? Did firewall block your connection?

Firewall rules if using UFW

NAT the VPN client traffic to the Internet. Change the ip address and interface the results while running "ip a " command. Add the lines marked red.

```
user@openvpn : ~/openvpn-ca/keys $ sudo vi /etc/ufw/before.rules
```

```
# Rules that should be run before the ufw command line added rules. Custom
# rules should be added to one of these chains:
# ufw-before-input
# ufw-before-output

# ufw-before-forward
#

# START OPENVPN RULES
# NAT table rules
*nat
:POSTROUTING ACCEPT [0:0]
# Allow traffic from OpenVPN client to ens160 (change to the interface you discovered!)
-A POSTROUTING -s 10.8.0.0/24 -o ens160 -j MASQUERADE
```

```
COMMIT
# END OPENVPN RULES

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

To allow forwarded packets by default:

```
user@openvpn : ~/openvpn-ca/keys $ sudo vi /etc/default/ufw
```

```
DEFAULT_FORWARD_POLICY="ACCEPT"
```

Set rules to allow openvpn port and protocol

```
user@openvpn : ~/openvpn-ca/keys $ sudo ufw allow 1194/udp
```

```
Rule added
Rule added (v6)
```

```
user@openvpn : ~/openvpn-ca/keys $ sudo ufw disable
```

```
Firewall stopped and disabled on system startup
```

```
user@openvpn : ~/openvpn-ca/keys $ sudo ufw enable
```

```
Command may disrupt existing ssh connections. Proceed with operation (y|n)? y
Firewall is active and enabled on system startup
```

Firewall rules if using iptables

NAT the VPN client traffic to the Internet. Change the ip address and interface the results while running "ip a " command.

```
iptables -t nat -A POSTROUTING -s 10.8.0.0/24 -o ens160 -j MASQUERADE
```

Set rules to allow openvpn port and protocol

```
iptables -A INPUT -i ens160 -m state --state NEW -p udp --dport 1194 -j ACCEPT
```

Allow TUN interface connections to OpenVPN server

```
iptables -A INPUT -i tun+ -j ACCEPT
```

Allow TUN interface connections to be forwarded through other interfaces.

```
iptables -A FORWARD -i tun+ -j ACCEPT
iptables -A FORWARD -i tun+ -o ens160 -m state --state RELATED,ESTABLISHED -j ACCEPT
iptables -A FORWARD -i ens160 -o tun+ -m state --state RELATED,ESTABLISHED -j ACCEPT
```

If your default iptables doesnt allow all outgoing traffic originating from our server you will also have to allow it from tun interface:

```
iptables -A OUTPUT -o tun+ -j ACCEPT
```