Install and Setup OpenVPN Server on Ubuntu 20.04

In this guide, we are going to learn how to install and setup OpenVPN Server on Ubuntu 20.04. OpenVPN is a robust and highly flexible open-source VPN software that uses all of the encryption, authentication, and certification features of the OpenSSL library to securely tunnel IP networks over a single UDP or TCP port.

It facilitates the extension of private network across a public network, access remote sites, make secure point-to-point connections, while maintaining security that would be achieved in a private network.

Install and Setup OpenVPN Server on Ubuntu 20.04

Run system update

```
apt update
apt upgrade
```

Install OpenVPN on Ubuntu 20.04

OpenVPN package is available on the default Ubuntu 20.04 repos. Thus the installation is as simple as running the command below;

```
apt install openvpn
```

Install Easy-RSA CA Utility on Ubuntu 20.04

Easy-RSA package provides utilities for generating SSL key-pairs that is used to secure VPN connections.

```
apt install easy-rsa
```

Create OpenVPN Public Key Infrastructure

Once you have installed easy-rsa, you need to initialize the OpenVPN PKI. The PKI consists of:

- a public key and private key for the server and each client
- a master Certificate Authority (CA) certificate and key which is used to sign each of the server and client certificates.

Before you can proceed, copy the easy-rsa configuration directory to a different location to ensure that that future OpenVPN package upgrades won't overwrite your modifications.

```
cp -r /usr/share/easy-rsa /etc/
```

Next, initialize the PKI.

```
cd /etc/easy-rsa/
./easyrsa init-pki
```

Once the PKI is initialized, **`/etc/easy-rsa/pki`** is created.

Generate the Certificate Authority (CA) Certificate and Key

Next, generate the CA certificate and key for signing OpenVPN server and client certificates.

```
cd /etc/easy-rsa/
./easyrsa build-ca
```

This will prompt you for the CA key passphrase and the server common name.

```
Using SSL: openssl OpenSSL 1.1.1f  31 Mar 2020

Enter New CA Key Passphrase: ENTER PASSWORD
Re-Enter New CA Key Passphrase: RE-ENTER PASSWORD
Generating RSA private key, 2048 bit long modulus (2 primes)
...................+++++
.............+++++
```

```
e is 65537 (0x010001)
Can't load /etc/easy-rsa/pki/.rnd into RNG
139840045897024:error:2406F079:random number generator:RAND_load_file:Cannot open file:../crypto/rand/randfile
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]:Kifarunix-demo CA

CA creation complete and you may now import and sign cert requests.
Your new CA certificate file for publishing is at:
/etc/easy-rsa/pki/ca.crt
```

The CA certificate is generated and stored at `/etc/easy-rsa/pki/ca.crt`.

Generate Diffie Hellman Parameters

Generate Diffie-Hellman keys used for key exchange during the TLS handshake between OpenVPN server and the connecting clients. This command has be executed within the Easy-RSA directory;

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

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

Generate OpenVPN Server Certificate and Key

To generate a certificate and private key for the OpenVPN server, run the command below;

```
cd /etc/easy-rsa
./easyrsa build-server-full server nopass
```

Enter the CA key passphrase create above to generate the certificates and keys.

`nopass` disables the use of passphrase.

Generate Hash-based Message Authentication Code (HMAC) key

TLS/SSL pre-shared authentication key is used as an additional HMAC signature on all SSL/TLS handshake packets to avoid DoS attack and UDP port flooding. This can be generated using the command;

```
openvpn --genkey --secret /etc/easy-rsa/pki/ta.key
```

Generate OpenVPN Revocation Certificate

To invalidate a previously signed certificate, you need to generate a revocation certificate. Run the script within the Easy-RSA directory;

```
./easyrsa gen-crl
```

The revocation certificate is generated and stored at `/etc/easy-rsa/pki/crl.pem`.

Copy Server Certificates and Keys to Server Config Directory

Copy all generated server certificates/keys to OpenVPN server configuration directory.

```
cp -rp /etc/easy-rsa/pki/{ca.crt,dh.pem,ta.key,crl.pem,issued,private} /etc/openvpn/server/
```

Generate OpenVPN Client Certificates and Keys

OpenVPN clients certificates and private keys can be generated as follows

```
cd /etc/easy-rsa
./easyrsa build-client-full koromicha nopass
```

- where **`koromicha`** is the name of the client for which the certificate and keys are generated.
- Always use a unique common name for each client that you are generating certificate and keys for.

To generate for the second client,

```
./easyrsa build-client-full janedoe nopass
```

You can see how to use `easyrsa` command with `./easyrsa --help`.

Copy Client Certificates and Keys to Client Directory

Create OpenVPN clients directories. For example, we have generated certificates and key files for two clients, koromicha and janedoe, hence we create directories as;

```
mkdir /etc/openvpn/client/{koromicha,janedoe}
```

After that, copy the client generated certificates/keys and server CA certificate to OpenVPN client configuration directory. You can

```
cp -rp /etc/easy-rsa/pki/{ca.crt,issued/koromicha.crt,private/koromicha.key} /etc/openvpn/client/koromicha
cp -rp /etc/easy-rsa/pki/{ca.crt,issued/janedoe.crt,private/janedoe.key} /etc/openvpn/client/janedoe/
```

Configure OpenVPN Server on Ubuntu 20.04

The next step is to configure OpenVPN server. Copy the sample OpenVPN server configuration to `/etc/openvpn/server` directory as shown below;

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

Extract the configuration and modify it to suite your needs;

```
cd /etc/openvpn/server/
gunzip server.conf.gz
vim /etc/openvpn/server/server.conf
```

This is how our sample configurations looks like with no comments. The configuration is highly commented to help you understand various option usage.

```
port 1194
proto udp
dev tun
ca ca.crt
cert issued/server.crt
key private/server.key  # This file should be kept secret
dh dh.pem
topology subnet
server 10.8.0.0 255.255.255.0
ifconfig-pool-persist /var/log/openvpn/ipp.txt
push "redirect-gateway def1 bypass-dhcp"
push "dhcp-option DNS 208.67.222.222"
push "dhcp-option DNS 192.168.2.11"
client-to-client
keepalive 10 120
tls-auth ta.key 0 # This file is secret
cipher AES-256-CBC
comp-lzo
persist-key
persist-tun
status /var/log/openvpn/openvpn-status.log
log-append  /var/log/openvpn/openvpn.log
verb 3
explicit-exit-notify 1
auth SHA512
```

Save and exit the config once done editing.

Configure OpenVPN IP Forwarding

To ensure that traffic from the client is routed through the OpenVPN server's IP address (helps masks the the client IP address), you need to enable IP forwarding on the OpenVPN server.

Uncomment the line, **`net.ipv4.ip_forward=1`**, on **`/etc/sysctl.conf`** to enable packet forwarding for IPv4

```
sed -i 's/#net.ipv4.ip_forward=1/net.ipv4.ip_forward=1/' /etc/sysctl.conf
```

Apply the changes without rebooting the server.

```
sysctl --system
```

Allow OpenVPN service port through firewall;

```
ufw allow 1194/udp
```

Configure IP Masquerading on UFW

Find your default interface through which your packets are sent.

```
ip route get 8.8.8.8
8.8.8.8 via 10.0.2.2 dev enp0s3 src 10.0.2.15 uid 0
```

Next, update UFW rules;

```
vim /etc/ufw/before.rules
```

Add the following highlighted lines just before the **`*filter`** table settings. Note the interface used shoud match the interface name above.

```
...
*nat
:POSTROUTING ACCEPT [0:0]
-A POSTROUTING -s 10.8.0.0/8 -o enp0s3 -j MASQUERADE
COMMIT
# Don't delete these required lines, otherwise there will be errors
*filter
...
```

Save and exit the config.

Enable UFW packet forwarding;

```
sed -i 's/DEFAULT_FORWARD_POLICY="DROP"/DEFAULT_FORWARD_POLICY="ACCEPT"/' /etc/default/ufw
```

Reload UFW;

```
ufw reload
```

Running OpenVPN Server on Ubuntu 20.04

Start and enable OpenVPN server to run on system boot;

```
systemctl enable --now openvpn-server@server
```

Checking the status;

```
systemctl status openvpn-server@server
● openvpn-server@server.service - OpenVPN service for server
     Loaded: loaded (/lib/systemd/system/openvpn-server@.service; enabled; vendor preset: enabled)
     Active: active (running) since Fri 2020-05-01 16:07:33 UTC; 3s ago
       Docs: man:openvpn(8)
             https://community.openvpn.net/openvpn/wiki/Openvpn24ManPage
             https://community.openvpn.net/openvpn/wiki/HOWTO
   Main PID: 11980 (openvpn)
     Status: "Initialization Sequence Completed"
      Tasks: 1 (limit: 2281)
     Memory: 1.0M
     CGroup: /system.slice/system-openvpn\x2dserver.slice/openvpn-server@server.service
             └─11980 /usr/sbin/openvpn --status /run/openvpn-server/status-server.log --status-version 2 --sup

May 01 16:07:33 vpn.kifarunix-demo.com systemd[1]: Starting OpenVPN service for server...
May 01 16:07:33 vpn.kifarunix-demo.com systemd[1]: Started OpenVPN service for server.
```

When OpenVPN service runs, it will create a tunnelling interface, tun0;

```
ip add s
...
4: tun0: <POINTOPOINT,MULTICAST,NOARP,UP,LOWER_UP> mtu 1500 qdisc fq_codel state UNKNOWN group default qlen 10
    link/none
    inet 10.8.0.1/24 brd 10.8.0.255 scope global tun0
       valid_lft forever preferred_lft forever
    inet6 fe80::1989:2bf2:1e7f:7415/64 scope link stable-privacy
       valid_lft forever preferred_lft forever
```

Also, be sure to check the logs;

```
tail /var/log/openvpn/openvpn.log
/sbin/ip link set dev tun0 up mtu 1500
/sbin/ip addr add dev tun0 10.8.0.1/24 broadcast 10.8.0.255
Could not determine IPv4/IPv6 protocol. Using AF_INET
Socket Buffers: R=[212992->212992] S=[212992->212992]
UDPv4 link local (bound): [AF_INET][undef]:1194
UDPv4 link remote: [AF_UNSPEC]
MULTI: multi_init called, r=256 v=256
IFCONFIG POOL: base=10.8.0.2 size=252, ipv6=0
IFCONFIG POOL LIST
Initialization Sequence Completed
```