Your submission was sent successfully! *Close*

You have successfully unsubscribed! *Close*

Search on Server Docs

Service - OpenVPN

OpenVPN is a Virtual Private Networking (VPN) solution provided in the Ubuntu Repositories. It is flexible, reliable and secure. It belongs to the family of SSL/TLS VPN stacks (different from IPSec VPNs). This chapter will cover installing and configuring OpenVPN to create a VPN.

If you want more than just pre-shared keys OpenVPN makes it easy to set up a Public Key Infrastructure (PKI) to use SSL/TLS certificates for authentication and key exchange between the VPN server and clients. OpenVPN can be used in a routed or bridged VPN mode and can be configured to use either UDP or TCP. The port number can be configured as well, but port 1194 is the official one; this single port is used for all communication. VPN client implementations are available for almost anything including all Linux distributions, macOS, Windows and OpenWRT-based WLAN routers.

Server Installation

To install openvpn in a terminal enter:

```
sudo apt install openvpn easy-rsa
```

Public Key Infrastructure Setup

The first step in building an OpenVPN configuration is to establish a PKI (public key infrastructure). The PKI consists of:

- a separate certificate (also known as a public key) and private key for the server and each client.
- a master Certificate Authority (CA) certificate and key, used to sign the server and client certificates.

OpenVPN supports bidirectional authentication based on certificates, meaning that the client must authenticate the server certificate and the server must authenticate the client certificate before mutual trust is established.

Both server and client will authenticate the other by first verifying that the presented certificate was signed by the master certificate authority (CA), and then by testing information in the now-authenticated certificate header, such as the certificate common name or certificate type (client or server).

Certificate Authority Setup

To setup your own Certificate Authority (CA) and generate certificates and keys for an OpenVPN server and multiple clients first copy the `easy-rsa` directory to `/etc/openvpn`. This will ensure that any changes to the scripts will not be lost when the package is updated. From a terminal, run:

```
sudo make-cadir /etc/openvpn/easy-rsa
```

Note: If desired, you can alternatively edit `/etc/openvpn/easy-rsa/vars` directly, adjusting it to your needs.

As `root` user change to the newly created directory `/etc/openvpn/easy-rsa` and run:

```
./easyrsa init-pki
./easyrsa build-ca
```

Server Keys and Certificates

Next, we will generate a key pair for the server:

```
./easyrsa gen-req myservername nopass
```

Diffie Hellman parameters must be generated for the OpenVPN server. The following will place them in `pki/dh.pem`.

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

And finally a certificate for the server:

```
./easyrsa gen-req myservername nopass
./easyrsa sign-req server myservername
```

All certificates and keys have been generated in subdirectories. Common practice is to copy them to /etc/openvpn/:

```
cp pki/dh.pem pki/ca.crt pki/issued/myservername.crt pki/private/myservername.key /etc/openvpn/
```

Client Certificates

The VPN client will also need a certificate to authenticate itself to the server. Usually you create a different certificate for each client.

This can either be done on the server (as the keys and certificates above) and then securely distributed to the client. Or vice versa: the client can generate and submit a request that is sent and signed by the server.

To create the certificate, enter the following in a terminal while being user root:

```
./easyrsa gen-req myclient1 nopass
./easyrsa sign-req client myclient1
```

If the first command above was done on a remote system, then copy the .req file to the CA server. There you can then import it via `easyrsa import-req /incoming/myclient1.req myclient1`. Then you can go on with the second `sign-eq` command.

In both cases, afterwards copy the following files to the client using a secure method:

- `pki/ca.crt`
- `pki/issued/myclient1.crt`

As the client certificates and keys are only required on the client machine, you can remove them from the server.

Simple Server Configuration

Along with your OpenVPN installation you got these sample config files (and many more if you check):

```
root@server:/# ls -l /usr/share/doc/openvpn/examples/sample-config-files/
total 68
-rw-r--r-- 1 root root 3427 2011-07-04 15:09 client.conf
-rw-r--r-- 1 root root 4141 2011-07-04 15:09 server.conf.gz
```

Start with copying and unpacking server.conf.gz to /etc/openvpn/server.conf.

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

Edit `/etc/openvpn/myserver.conf` to make sure the following lines are pointing to the certificates and keys you created in the section above.

```
ca ca.crt
cert myservername.crt
key myservername.key
dh dh2048.pem
```

Complete this set with a ta key in `etc/openvpn` for tls-auth like:

```
sudo openvpn --genkey --secret ta.key
```

Edit `/etc/sysctl.conf` and uncomment the following line to enable IP forwarding.

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

Then reload sysctl.

```
sudo sysctl -p /etc/sysctl.conf
```

That is the minimum you have to configure to get a working OpenVPN server. You can use all the default settings in the sample server.conf file. Now start the server.

Be aware that the *"systemctl start openvpn"* is not starting your openvpn you just defined.
Openvpn uses templatized systemd jobs, openvpn@CONFIGFILENAME. So if for example your configuration file is `myserver.conf` your service is called openvpn@myserver. You can run all kinds of service and systemctl commands like start/stop /enable/disable/preset against a templatized service like openvpn@server.

```
$ sudo systemctl start openvpn@myserver
```

You will find logging and error messages in the journal. For example, if you started a templatized service <https://www.freedesktop.org/software/systemd/man/systemd.unit.html> openvpn@server you can filter for this particular message source with:

```
sudo journalctl -u openvpn@myserver -xe
```

The same templatized approach works for all of systemctl:

```
$ sudo systemctl status openvpn@myserver
openvpn@myserver.service - OpenVPN connection to myserver
   Loaded: loaded (/lib/systemd/system/openvpn@.service; disabled; vendor preset: enabled)
   Active: active (running) since Thu 2019-10-24 10:59:25 UTC; 10s ago
     Docs: man:openvpn(8)
           https://community.openvpn.net/openvpn/wiki/Openvpn24ManPage
           https://community.openvpn.net/openvpn/wiki/HOWTO
 Main PID: 4138 (openvpn)
   Status: "Initialization Sequence Completed"
    Tasks: 1 (limit: 533)
   Memory: 1.0M
   CGroup: /system.slice/system-openvpn.slice/openvpn@myserver.service
           └─4138 /usr/sbin/openvpn --daemon ovpn-myserver --status /run/openvpn/myserver.status 10 --cd /etc/openvpn --script-security 2 --config /etc/openvpn/myserver.conf --writepid /run/

Oct 24 10:59:26 eoan-vpn-server ovpn-myserver[4138]: /sbin/ip addr add dev tun0 local 10.8.0.1 peer 10.8.0.2
Oct 24 10:59:26 eoan-vpn-server ovpn-myserver[4138]: /sbin/ip route add 10.8.0.0/24 via 10.8.0.2
Oct 24 10:59:26 eoan-vpn-server ovpn-myserver[4138]: Could not determine IPv4/IPv6 protocol. Using AF_INET
Oct 24 10:59:26 eoan-vpn-server ovpn-myserver[4138]: Socket Buffers: R=[212992->212992] S=[212992->212992]
Oct 24 10:59:26 eoan-vpn-server ovpn-myserver[4138]: UDPv4 link local (bound): [AF_INET][undef]:1194
Oct 24 10:59:26 eoan-vpn-server ovpn-myserver[4138]: UDPv4 link remote: [AF_UNSPEC]
Oct 24 10:59:26 eoan-vpn-server ovpn-myserver[4138]: MULTI: multi_init called, r=256 v=256
Oct 24 10:59:26 eoan-vpn-server ovpn-myserver[4138]: IFCONFIG POOL: base=10.8.0.4 size=62, ipv6=0
Oct 24 10:59:26 eoan-vpn-server ovpn-myserver[4138]: IFCONFIG POOL LIST
Oct 24 10:59:26 eoan-vpn-server ovpn-myserver[4138]: Initialization Sequence Completed
```

You can enable/disable various openvpn services on one system, but you could also let Ubuntu do it for you. There is config for `AUTOSTART` in `/etc/default/openvpn`. Allowed values are "all", "none" or space separated list of names of the VPNs. If empty, "all" is assumed. The VPN name refers to the VPN configutation file name. i.e. `home` would be `/etc/openvpn/home.conf` If you're running systemd, changing this variable will require running `systemctl daemon-reload` followed by a *restart* of the openvpn service (if you removed entries you may have to stop those manually).

After "systemctl daemon-reload" a restart of the "generic" openvpn will restart all dependent services that the generator in /lib/systemd/system-generators/openvpn-generator created for your conf files when you called daemon-reload.

Now check if OpenVPN created a tun0 interface:

```
root@server:/etc/openvpn# ip addr show dev tun0
5: tun0: <POINTOPOINT,MULTICAST,NOARP,UP,LOWER_UP> mtu 1500 qdisc fq_codel state UNKNOWN group default qlen 100
    link/none
    inet 10.8.0.1 peer 10.8.0.2/32 scope global tun0
       valid_lft forever preferred_lft forever
    inet6 fe80::b5ac:7829:f31e:32c5/64 scope link stable-privacy
       valid_lft forever preferred_lft forever
```

Simple Client Configuration

There are various different OpenVPN client implementations with and without GUIs. You can read more about clients in a later section on VPN Clients. For now we use commandline/service based OpenVPN client for Ubuntu which is part of the very same package as the server. So you have to install the `openvpn` package again on the client machine:

```
sudo apt install openvpn
```

This time copy the client.conf sample config file to /etc/openvpn/:

```
sudo cp /usr/share/doc/openvpn/examples/sample-config-files/client.conf /etc/openvpn/
```

Copy the following client keys and certificate files you created in the section above to e.g. /etc/openvpn/ and edit `/etc/openvpn/client.conf` to make sure the following lines are pointing to those files. If you have the files in /etc/openvpn/ you can omit the path.

```
ca ca.crt
cert myclient1.crt
key myclient1.key
tls-auth ta.key 1
```

And you have to specify the OpenVPN server name or address. Make sure the keyword client is in the config. That's what enables client mode.

```
client
remote vpnserver.example.com 1194
```

Now start the OpenVPN client with the same templatized mechanism:

```
$ sudo systemctl start openvpn@client
```

You can check status as you did on the server:

```
$ sudo systemctl status openvpn@client
openvpn@client.service - OpenVPN connection to client
   Loaded: loaded (/lib/systemd/system/openvpn@.service; disabled; vendor preset: enabled)
   Active: active (running) since Thu 2019-10-24 11:42:35 UTC; 6s ago
     Docs: man:openvpn(8)
           https://community.openvpn.net/openvpn/wiki/Openvpn24ManPage
           https://community.openvpn.net/openvpn/wiki/HOWTO
 Main PID: 3616 (openvpn)
   Status: "Initialization Sequence Completed"
    Tasks: 1 (limit: 533)
   Memory: 1.3M
   CGroup: /system.slice/system-openvpn.slice/openvpn@client.service
           └─3616 /usr/sbin/openvpn --daemon ovpn-client --status /run/openvpn/client.status 10 --cd /etc/openvpn --script-security 2 --config /etc/openvpn/client.conf --writepid /run/openvp

Oct 24 11:42:36 eoan-vpn-client ovpn-client[3616]: Outgoing Data Channel: Cipher 'AES-256-GCM' initialized with 256 bit key
Oct 24 11:42:36 eoan-vpn-client ovpn-client[3616]: Incoming Data Channel: Cipher 'AES-256-GCM' initialized with 256 bit key
Oct 24 11:42:36 eoan-vpn-client ovpn-client[3616]: ROUTE_GATEWAY 192.168.122.1/255.255.255.0 IFACE=ens3 HWADDR=52:54:00:3c:5a:88
Oct 24 11:42:36 eoan-vpn-client ovpn-client[3616]: TUN/TAP device tun0 opened
Oct 24 11:42:36 eoan-vpn-client ovpn-client[3616]: TUN/TAP TX queue length set to 100
Oct 24 11:42:36 eoan-vpn-client ovpn-client[3616]: /sbin/ip link set dev tun0 up mtu 1500
Oct 24 11:42:36 eoan-vpn-client ovpn-client[3616]: /sbin/ip addr add dev tun0 local 10.8.0.6 peer 10.8.0.5
Oct 24 11:42:36 eoan-vpn-client ovpn-client[3616]: /sbin/ip route add 10.8.0.1/32 via 10.8.0.5
Oct 24 11:42:36 eoan-vpn-client ovpn-client[3616]: WARNING: this configuration may cache passwords in memory -- use the auth-nocache option to prevent this
Oct 24 11:42:36 eoan-vpn-client ovpn-client[3616]: Initialization Sequence Completed
```

On the server log an incoming connection looks like the following.
You can see client name and source address as well as success/failure messages.

```
ovpn-myserver[4818]: 192.168.122.114:55738 TLS: Initial packet from [AF_INET]192.168.122.114:55738, sid=5e943ab8 40ab9fed
ovpn-myserver[4818]: 192.168.122.114:55738 VERIFY OK: depth=1, CN=Easy-RSA CA
ovpn-myserver[4818]: 192.168.122.114:55738 VERIFY OK: depth=0, CN=myclient1
ovpn-myserver[4818]: 192.168.122.114:55738 peer info: IV_VER=2.4.7
ovpn-myserver[4818]: 192.168.122.114:55738 peer info: IV_PLAT=linux
ovpn-myserver[4818]: 192.168.122.114:55738 peer info: IV_PROTO=2
ovpn-myserver[4818]: 192.168.122.114:55738 peer info: IV_NCP=2
ovpn-myserver[4818]: 192.168.122.114:55738 peer info: IV_LZ4=1
ovpn-myserver[4818]: 192.168.122.114:55738 peer info: IV_LZ4v2=1
ovpn-myserver[4818]: 192.168.122.114:55738 peer info: IV_LZO=1
ovpn-myserver[4818]: 192.168.122.114:55738 peer info: IV_COMP_STUB=1
ovpn-myserver[4818]: 192.168.122.114:55738 peer info: IV_COMP_STUBv2=1
ovpn-myserver[4818]: 192.168.122.114:55738 peer info: IV_TCPNL=1
ovpn-myserver[4818]: 192.168.122.114:55738 Control Channel: TLSv1.3, cipher TLSv1.3 TLS_AES_256_GCM_SHA384, 2048 bit RSA
ovpn-myserver[4818]: 192.168.122.114:55738 [myclient1] Peer Connection Initiated with [AF_INET]192.168.122.114:55738
ovpn-myserver[4818]: myclient1/192.168.122.114:55738 MULTI_sva: pool returned IPv4=10.8.0.6, IPv6=(Not enabled)
ovpn-myserver[4818]: myclient1/192.168.122.114:55738 MULTI: Learn: 10.8.0.6 -> myclient1/192.168.122.114:55738
ovpn-myserver[4818]: myclient1/192.168.122.114:55738 MULTI: primary virtual IP for myclient1/192.168.122.114:55738: 10.8.0.6
ovpn-myserver[4818]: myclient1/192.168.122.114:55738 PUSH: Received control message: 'PUSH_REQUEST'
ovpn-myserver[4818]: myclient1/192.168.122.114:55738 SENT CONTROL [myclient1]: 'PUSH_REPLY,route 10.8.0.1,topology net30,ping 10,ping-restart 120,ifconfig 10.8.0.6 10.8.0.5,peer-id 0,cipher AES-256-GCM' (status=1)
ovpn-myserver[4818]: myclient1/192.168.122.114:55738 Data Channel: using negotiated cipher 'AES-256-GCM'
ovpn-myserver[4818]: myclient1/192.168.122.114:55738 Outgoing Data Channel: Cipher 'AES-256-GCM' initialized with 256 bit key
ovpn-myserver[4818]: myclient1/192.168.122.114:55738 Incoming Data Channel: Cipher 'AES-256-GCM' initialized with 256 bit key
```

And you can check on the client if it created a tun0 interface:

```
$ ip addr show dev tun0
4: tun0: <POINTOPOINT,MULTICAST,NOARP,UP,LOWER_UP> mtu 1500 qdisc fq_codel state UNKNOWN group default qlen 100
    link/none
    inet 10.8.0.6 peer 10.8.0.5/32 scope global tun0
       valid_lft forever preferred_lft forever
    inet6 fe80::5a94:ae12:8901:5a75/64 scope link stable-privacy
       valid_lft forever preferred_lft forever
```

Check if you can ping the OpenVPN server:

```
root@client:/etc/openvpn# ping 10.8.0.1
PING 10.8.0.1 (10.8.0.1) 56(84) bytes of data.
64 bytes from 10.8.0.1: icmp_req=1 ttl=64 time=0.920 ms
```

> Note
>
> The OpenVPN server always uses the first usable IP address in the client network and only that IP is pingable. E.g. if you configured a /24 for the client network mask, the .1 address will be used. The P-t-P address you see in the `ip addr` output above is usually not answering ping requests.

Check out your routes:

```
$ ip route
default via 192.168.122.1 dev ens3 proto dhcp src 192.168.122.114 metric 100
10.8.0.1 via 10.8.0.5 dev tun0
10.8.0.5 dev tun0 proto kernel scope link src 10.8.0.6
192.168.122.0/24 dev ens3 proto kernel scope link src 192.168.122.114
192.168.122.1 dev ens3 proto dhcp scope link src 192.168.122.114 metric 100
```

First trouble shooting

If the above didn't work for you, check this:

- Check your `journal -xe`
- Check that you have specified the keyfile names correctly in client and server conf files
- Can the client connect to the server machine? Maybe a firewall is blocking access? Check journal on server.
- Client and server must use same protocol and port, e.g. UDP port 1194, see port and proto config option
- Client and server must use same config regarding compression, see comp-lzo config option
- Client and server must use same config regarding bridged vs routed mode, see server vs server-bridge config option

Advanced configuration

Advanced routed VPN configuration on server

The above is a very simple working VPN. The client can access services on the VPN server machine through an encrypted tunnel. If you want to reach more servers or anything in other networks, push some routes to the clients. E.g. if your company's network can be summarized to the network 192.168.0.0/16, you could push this route to the clients. But you will also have to change the routing for the way back - your servers need to know a route to the VPN client-network.

The example config files that we have been using in this guide are full of all these advanced options in the form of a comment and a disabled configuration line as an example.

> Note
>
> Please read the OpenVPN hardening security guide <http://openvpn.net/index.php/open-source/documentation/howto.html#security> for further security advice.

Advanced bridged VPN configuration on server

OpenVPN can be setup for either a routed or a bridged VPN mode. Sometimes this is also referred to as OSI layer-2 versus layer-3 VPN. In a bridged VPN all layer-2 frames - e.g. all ethernet frames - are sent to the VPN partners and in a routed VPN only layer-3 packets are sent to VPN partners. In bridged mode all traffic including traffic which was traditionally LAN-local like local network broadcasts, DHCP requests, ARP requests etc. are sent to VPN partners whereas in routed mode this would be filtered.

Prepare interface config for bridging on server

First, use netplan to configure a bridge device using the desired ethernet device.

```
$ cat /etc/netplan/01-netcfg.yaml
network:
    version: 2
    renderer: networkd
    ethernets:
        enp0s31f6:
            dhcp4: no
    bridges:
        br0:
            interfaces: [enp0s31f6]
            dhcp4: no
            addresses: [10.0.1.100/24]
            gateway4: 10.0.1.1
            nameservers:
                addresses: [10.0.1.1]
```

Static IP addressing is highly suggested. DHCP addressing can also work, but you will still have to encode a static address in the OpenVPN configuration file.

The next step on the server is to configure the ethernet device for promiscuous mode on boot. To do this, ensure the networkd-dispatcher package is installed and create the following configuration script.

```
sudo apt update
sudo apt install networkd-dispatcher
sudo touch /usr/lib/networkd-dispatcher/dormant.d/promisc_bridge
sudo chmod +x /usr/lib/networkd-dispatcher/dormant.d/promisc_bridge
```

Then add the following contents.

```
#!/bin/sh
set -e
if [ "$IFACE" = br0 ]; then
    # no networkd-dispatcher event for 'carrier' on the physical interface
    ip link set enp0s31f6 up promisc on
fi
```

Prepare server config for bridging

Edit `/etc/openvpn/server.conf` to use tap rather than tun and set the server to use the server-bridge directive:

```
;dev tun
dev tap
;server 10.8.0.0 255.255.255.0
server-bridge 10.0.0.4 255.255.255.0 10.0.0.128 10.0.0.254
```

After configuring the server, restart openvpn by entering:

```
sudo systemctl restart openvpn@myserver
```

Prepare client config for bridging

The only difference on the client side for bridged mode to what was outlined above is that you need to edit `/etc/openvpn/client.conf` and set `tap` mode:

```
dev tap
;dev tun
```

Finally, restart openvpn:

```
sudo systemctl restart openvpn@client
```

You should now be able to connect to the full remote LAN through the VPN.

References

- EasyRSA <https://github.com/OpenVPN/easy-rsa/blob/master/README.quickstart.md>
- OpenVPN quick start guide <https://openvpn.net/quick-start-guide/>
- Snap'ed <https://snapcraft.io/> version of openvpn easy-openvpn

- Debian's OpenVPN Guide <https://wiki.debian.org/OpenVPN>

Previous OpenSSH Next gitolite

Last updated 3 months ago. Help improve this document in the forum <https://discourse.ubuntu.com/t/service-openvpn/11894> .