OpenVPN Community VPN daemon

START HERE

`sudo openvpn --config <file>`
Connect with a profile

`sudo openvpn --config <file> --daemon`
Run in the background

`openvpn --genkey tls-crypt <file>`
Make a control key

`openvpn --show-ciphers`
List data ciphers

`--verb 4`
More log detail

`--log-append <file>`
Log to a file

`kill -USR1 <pid>`
Restart in place

`systemctl start openvpn-client@<name>`
Run as a service

Client tun0 device → TLS handshake → **Control channel** certs, tls-crypt → keys → **Data channel** data-ciphers → UDP 1194 → **Server** tun0, routes

CONCEPTS

An option on the command line is the same as in a config file, just with the leading -- dropped: --verb 4 becomes verb 4.

dev tun carries IP packets (routed VPN); dev tap carries Ethernet frames (bridging). Both ends must match.

Prefer tls-crypt over tls-auth: it also encrypts the control channel, hiding the handshake from scanners.

RUN

`openvpn --config <file>`
Start with a config file

`openvpn --config <file> --daemon`
Detach and run in background

`openvpn --config <file> --verb 4`
Set log verbosity (0-11, default 1)

Level 3 is a good summary; above 5 logs every packet as R/W characters.

`openvpn --config <file> --log-append <file>`
Log to a file (appends)

`openvpn --config <file> --status <file> <sec>`
Write a status file every N seconds

`openvpn --version`
Show version and build options

CLIENT OPTIONS

`client`
Pull options from the server, act as TLS client

Shorthand for pull plus tls-client.

`remote <host> <port> <proto>`
Server to connect to

`dev tun`
Routed IP tunnel device

`proto udp`
Transport: udp, tcp-client, tcp-server

`remote <host> 443 tcp-client`
Server reachable over TCP 443

`nobind`
Do not bind a fixed local port

`persist-key`
Keep keys across restarts

`persist-tun`
Keep the tun device across restarts

CLIENT OPTIONS (cont.)

`auth-user-pass <file>`
Username and password from a file

The file holds the username on line one and the password on line two.

KEYS AND TLS

`ca <file>`
CA certificate to trust

`cert <file>`
Your certificate

`key <file>`
Your private key

`tls-crypt <file>`
Encrypt and authenticate control channel

`tls-auth <file> <0|1>`
Authenticate control channel with HMAC

Direction 0 on the server and 1 on the client, or use key-direction.

`remote-cert-tls server`
Require a server-purpose certificate

Guards against another client's valid certificate impersonating the server.

`data-ciphers <list>`
Colon-separated data channel ciphers

`verify-x509-name <name> name`
Require this peer certificate name

`openvpn --genkey secret <file>`
Create a static key file

`openvpn --genkey tls-crypt <file>`
Create a tls-crypt key

`openvpn --show-ciphers`
List available ciphers

SERVER OPTIONS

`server <network> <netmask>`
Server mode with a client address pool

Implies tls-server and a tun topology; 10.8.0.0 255.255.255.0 is the usual choice.

`dh none`
Use ECDH only, no DH file

`port <port>`
Local and remote port (default 1194)

`topology subnet`
One address per client, normal subnet

SERVER OPTIONS (cont.)

`push "<option>"`
Send an option to every client

`client-config-dir <dir>`
Per-client files named by common name

`ifconfig-push <ip> <netmask>`
Fixed address, set in a client file

`keepalive <interval> <timeout>`
Ping and restart on silence

keepalive 10 120 is common; the server doubles the timeout.

ROUTING AND DNS

`push "redirect-gateway def1"`
Send all client traffic via the VPN

`redirect-gateway def1`
Same, set on the client

def1 overrides the default route with two /1 routes, leaving the original route intact.

`push "route <network> <netmask>"`
Give clients a route to a subnet

`push "dns server 1 address <ip>"`
Set DNS on clients (2.6+)

Older clients and setups use push "dhcp-option DNS <ip>" instead.

`sysctl -w net.ipv4.ip_forward=1`
Let the server forward packets

A NAT rule (iptables or nftables) is also needed to reach the internet.

OPERATE

`systemctl start openvpn-client@<name>`
Start /etc/openvpn/client/<name>.conf

Unit names and paths are distro packaging; older packages use openvpn@<name> and /etc/openvpn/.

`systemctl start openvpn-server@<name>`
Start /etc/openvpn/server/<name>.conf

`journalctl -u openvpn-client@<name> -f`
Follow the service log

`kill -USR1 <pid>`
Soft restart, keeps keys and tun

`kill -TERM <pid>`
Shut down cleanly

OPERATE (cont.)

`management 127.0.0.1 7505`
Expose a management console

TROUBLESHOOT

`ip addr show tun0`
Check the tunnel got an address

`ip route`
Check the VPN routes exist

`ss -ulpn | grep 1194`
Check a server listens on UDP 1194

`openvpn --config <file> --verb 6`
Debug handshake problems

Common log messages

Message	Usually means
`TLS key negotiation failed`	Port, firewall or key
`certificate verify failed`	Bad CA, expiry, clock
`AUTH_FAILED`	Wrong login
`Initialization Sequence Completed`	Connected

RECIPES

Minimal client profile

```
client
dev tun
remote vpn.example.com 1194
nobind
remote-cert-tls server
ca ca.crt
cert client.crt
key client.key
tls-crypt ta.key
verb 3
```

Minimal server config

```
dev tun
ca ca.crt
cert server.crt
key server.key
dh none
tls-crypt ta.key
server 10.8.0.0 255.255.255.0
topology subnet
push "redirect-gateway def1"
keepalive 10 120
persist-key
persist-tun
```

Fixed address for one client

```
# in server.conf
client-config-dir /etc/openvpn/ccd
# file ccd/<common name>
ifconfig-push 10.8.0.50 255.255.255.0
push "route 192.168.10.0 255.255.255.0"
```

NAT out of the server

```
sudo sysctl -w net.ipv4.ip_forward=1
sudo iptables -t nat -A POSTROUTING -s 10.8.0.0/24 -o eth0 -j MASQUERADE
```