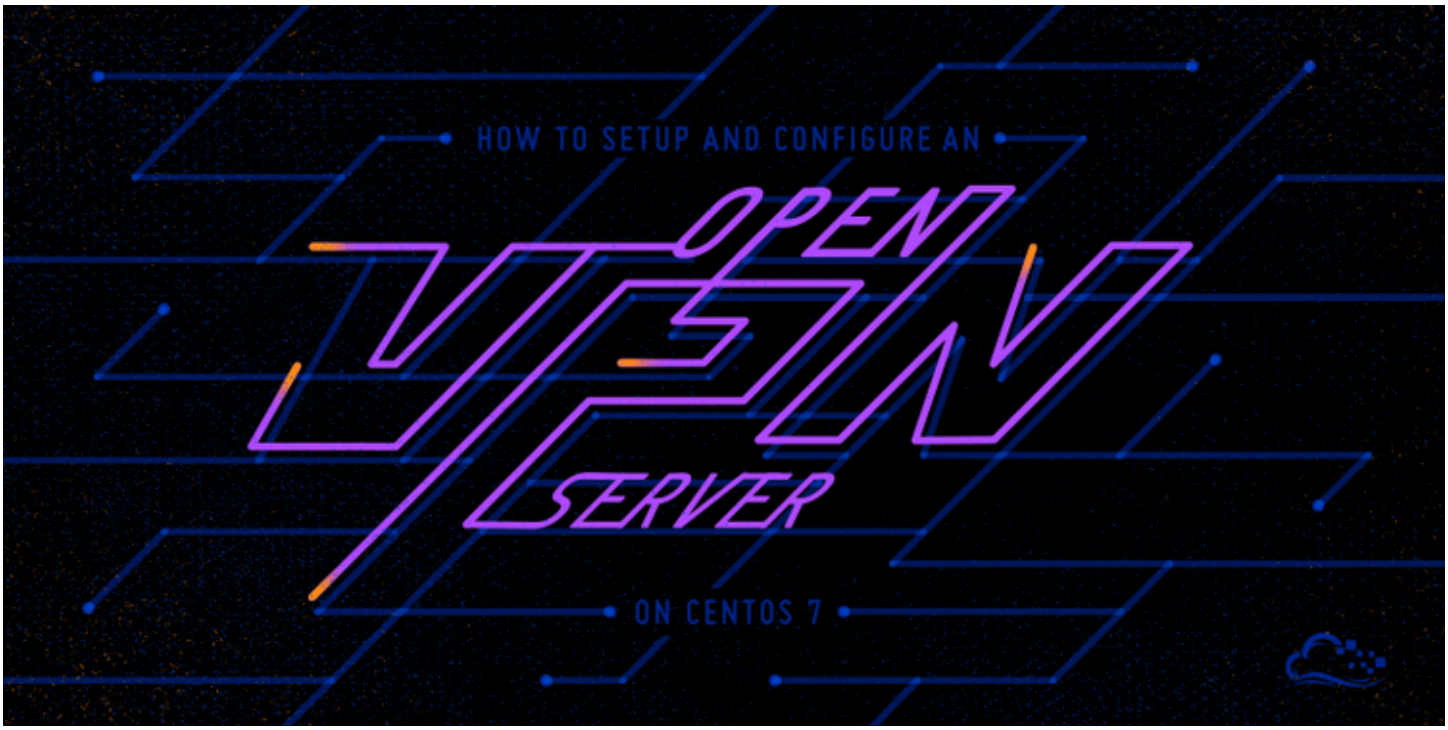
DigitalOcean

Subscribe



How To Set Up and Configure an OpenVPN Server on CentOS 7

56

Updated March 19, 2018 559.5k VPN SECURITY FIREWALL CENTOS

By: Jacob Tomlinson

Not using **CentOS 7**? Choose a different version:

Introduction

A Virtual Private Network (VPN) allows you to traverse untrusted networks as if you were on a private network. It gives you the freedom to access the internet safely and securely from your smartphone or laptop when connected to an untrusted network, like the WiFi at a hotel or coffee shop.

When combined with HTTPS connections, this setup allows you to secure your wireless logins and transactions. You can circumvent geographical restrictions and censorship, and shield your location and any unencrypted HTTP traffic from the untrusted network.

OpenVPN is a full featured, open-source Secure Socket Layer (SSL) VPN solution that accommodates a wide range of configurations. In this tutorial, you will set up OpenVPN on a CentOS 7 server, and then configure it to be accessible from a client machine.

Note: If you plan to set up an OpenVPN server on a DigitalOcean Droplet, be aware that we, like many hosting providers, charge for bandwidth overages. For this reason, please be mindful of how much traffic your server is handling.

See this page for more info.

Prerequisites

To follow this tutorial, you will need:

- One CentOS 7 server with a sudo non-root user and a firewall set up with firewalld, which you can achieve with our Initial Server Setup with CentOS 7 guide and the Additional Recommended Steps for New CentOS 7 Servers.
- A domain or subdomain that resolves to your server that you can use for the certificates. To set this up, you will first need to register a domain name and then add a DNS record via the DigitalOcean Control Panel. Note that just adding an A record will meet the requirements of this tutorial.
- A client machine which you will use to connect to your OpenVPN server. For the purposes of this tutorial, it’s recommend that you use your local machine as the OpenVPN client.

With these prerequisites in place, you are ready to begin setting up and configuring an OpenVPN server on CentOS 7.

Step 1 — Installing OpenVPN

To start, we will install OpenVPN on the server. We'll also install Easy RSA, a public key infrastructure management tool which will help us set up an internal certificate authority (CA) for use with our VPN. We'll also use Easy RSA to generate our SSL key pairs later on to secure the VPN connections.

Log in to the server as the non-root sudo user, and update the package lists to make sure you have all the latest versions.

```
$ sudo yum update -y
```

The Extra Packages for Enterprise Linux (EPEL) repository is an additional repository managed by the Fedora Project containing non-standard but popular packages. OpenVPN isn't available in the default CentOS repositories but it is available in EPEL, so install EPEL:

```
$ sudo yum install epel-release -y
```

Then update your package lists once more:

```
$ sudo yum update -y
```

Next, install OpenVPN and `wget`, which we will use to install Easy RSA:

```
$ sudo yum install -y openvpn wget
```

Using `wget`, download Easy RSA. For the purposes of this tutorial, we recommend using easy-rsa-2 because there's more available documentation for this version. You can find the download link for the latest version of easy-rsa-2 on the project's Releases page:

```
$ wget -O /tmp/easyrsa https://github.com/OpenVPN/easy-rsa-old/archive/2.3.3.tar.gz
```

Next, extract the compressed file with `tar`:

```
$ tar xfz /tmp/easyrsa
```

This will create a new directory on your server called `easy-rsa-old-2.3.3`. Make a new subdirectory under `/etc/openvpn` and name it `easy-rsa`:

```
$ sudo mkdir /etc/openvpn/easy-rsa
```

Copy the extracted Easy RSA files over to the new directory:

```
$ sudo cp -rf easy-rsa-old-2.3.3/easy-rsa/2.0/* /etc/openvpn/easy-rsa
```

Then change the directory's owner to your non-root sudo user:

```
$ sudo chown sammy /etc/openvpn/easy-rsa/
```

Once these programs are installed and have been moved to the right locations on your system, the next step is to customize the server-side configuration of OpenVPN.

Step 2 — Configuring OpenVPN

Like many other widely-used open-source tools, there are dozens of configuration options available to you. In this section, we will provide instructions on how to set up a basic OpenVPN server configuration.

OpenVPN has several example configuration files in its documentation directory. First, copy the sample `server.conf` file as a starting point for your own configuration file.

```
$ sudo cp /usr/share/doc/openvpn-2.4.4/sample/sample-config-files/server.conf /etc/
```

Open the new file for editing with the text editor of your choice. We'll use nano in our example, which you can download with the `yum install nano` command if you don't have it on your server already:

```
$ sudo nano /etc/openvpn/server.conf
```

There are a few lines we need to change in this file, most of which just need to be uncommented by removing the semicolon, `;`, at the beginning of the line. The functions of these lines, and the other lines not mentioned in this tutorial, are explained in-depth in the comments above each one.

To get started, find and uncomment the line containing `push "redirect-gateway def1 bypass-dhcp"`. Doing this will tell your client to redirect all of its traffic through your OpenVPN server. Be aware that enabling this functionality can cause connectivity issues with other network services, like SSH:

/etc/openvpn/server.conf

```
push "redirect-gateway def1 bypass-dhcp"
```

Because your client will not be able to use the default DNS servers provided by your ISP (as its traffic will be rerouted), you need to tell it which DNS servers it can use to connect to OpenVPN. You can pick different DNS servers, but here we'll use Google's public DNS servers which have the IPs of `8.8.8.8` and `8.8.4.4`.

Set this by uncommenting both `push "dhcp-option DNS ..."` lines and updating the IP addresses:

/etc/openvpn/server.conf

```
push "dhcp-option DNS 8.8.8.8"
push "dhcp-option DNS 8.8.4.4"
```

We want OpenVPN to run with no privileges once it has started, so we need to tell it to run with a user and group of **nobody**. To enable this, uncomment the `user nobody` and `group nobody` lines:

/etc/openvpn/server.conf

```
user nobody
group nobody
```

Next, uncomment the `topology subnet` line. This, along with the `server 10.8.0.0 255.255.255.0` line below it, configures your OpenVPN installation to function as a subnetwork and tells the client machine which IP address it should use. In this case, the server will become `10.8.0.1` and the first client will become `10.8.0.2`:

/etc/openvpn/server.conf

```
topology subnet
```

It's also recommended that you add the following line to your server configuration file. This double checks that any incoming client certificates are truly coming from a client, hardening the security parameters we will establish in later steps:

/etc/openvpn/server.conf

```
remote-cert-eku "TLS Web Client Authentication"
```

Lastly, OpenVPN strongly recommends that users enable TLS Authentication, a cryptographic protocol that ensures secure communications over a computer network. To do this, you will need to generate a static encryption key (named in our example as `myvpn.tlsauth`, although you can choose any name you like). Before creating this key, comment the line in the configuration file containing `tls-auth ta.key 0` by prepending it with a semicolon. Then, add `tls-crypt myvpn.tlsauth` to the line below it:

/etc/openvpn/server.conf

```
;tls-auth ta.key 0
tls-crypt myvpn.tlsauth
```

Save and exit the OpenVPN server configuration file (in nano, press `CTRL - X`, `Y`, then `ENTER` to do so), and then generate the static encryption key with the following command:

```
$ sudo openvpn --genkey --secret /etc/openvpn/myvpn.tlsauth
```

Now that your server is configured, you can move on to setting up the SSL keys and certificates needed to securely connect to your VPN connection.

Step 3 — Generating Keys and Certificates

Easy RSA uses a set of scripts that come installed with the program to generate keys and certificates. In order to avoid re-configuring every time you need to generate a certificate, you can modify Easy RSA's configuration to define the default values it will use for the certificate fields, including your country, city, and preferred email address.

We'll begin our process of generating keys and certificates by creating a directory where Easy RSA will store any keys and certs you generate:

```
$ sudo mkdir /etc/openvpn/easy-rsa/keys
```

The default certificate variables are set in the `vars` file in `/etc/openvpn/easy-rsa`, so open that file for editing:

```
$ sudo nano /etc/openvpn/easy-rsa/vars
```

Scroll to the bottom of the file and change the values that start with `export KEY_` to match your information. The ones that matter the most are:

- `KEY_CN`: Here, enter the domain or subdomain that resolves to your server.
- `KEY_NAME`: You should enter `server` here. If you enter something else, you would also have to update the configuration files that reference `server.key` and `server.crt`.

The other variables in this file that you may want to change are:

- `KEY_COUNTRY`: For this variable, enter the two-letter abbreviation of the country of your residence.
- `KEY_PROVINCE`: This should be the name or abbreviation of the state of your residence.
- `KEY_CITY`: Here, enter the name of the city you live in.

- `KEY_ORG`: This should be the name of your organization or company.
- `KEY_EMAIL`: Enter the email address that you want to be connected to the security certificate.
- `KEY_OU`: This should be the name of the "Organizational Unit" to which you belong, typically either the name of your department or team.

The rest of the variables can be safely ignored outside of specific use cases. After you've made your changes, the file should look like this:

/etc/openvpn/easy-rsa/vars

```
. . .

# 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="US"
export KEY_PROVINCE="NY"
export KEY_CITY="New York"
export KEY_ORG="DigitalOcean"
export KEY_EMAIL="sammy@example.com"
export KEY_EMAIL=sammy@example.com
export KEY_CN=openvpn.example.com
export KEY_NAME="server"
export KEY_OU="Community"
. . .
```

Save and close the file.

To start generating the keys and certificates, move into the `easy-rsa` directory and `source` in the new variables you set in the `vars` file:

```
$ cd /etc/openvpn/easy-rsa
$ source ./vars
```

Run Easy RSA's `clean-all` script to remove any keys and certificates already in the folder and generate the certificate authority:

```
$ ./clean-all
```

Next, build the certificate authority with the `build-ca` script. You'll be prompted to enter values for the certificate fields, but if you set the variables in the `vars` file earlier, all of your options will already be set as the defaults. You can press `ENTER` to accept the defaults for each one:

```
$ ./build-ca
```

This script generates a file called `ca.key`. This is the private key used to sign your server and clients' certificates. If it is lost, you can no longer trust any certificates from this certificate authority, and if anyone is able to access this file they can sign new certificates and access your VPN without your knowledge. For this reason, OpenVPN recommends storing `ca.key` in a location that can be offline as much as possible, and it should only be activated when creating new certificates.

Next, create a key and certificate for the server using the `build-key-server` script:

```
$ ./build-key-server server
```

As with building the CA, you'll see the values you've set as the defaults so you can hit `ENTER` at these prompts. Additionally, you'll be prompted to enter a challenge password and an optional company name. If you enter a challenge password, you will be asked for it when connecting to the VPN from your client. If you don't want to set a challenge password, just leave this line blank and press `ENTER`. At the end, enter `Y` to commit the changes.

The last part of creating the server keys and certificates is generating a Diffie-Hellman key exchange file. Use the `build-dh` script to do this:

```
$ ./build-dh
```

This may take a few minutes to complete.

Once your server is finished generating the key exchange file, copy the server keys and certificates from the `keys` directory into the `openvpn` directory:

```
$ cd /etc/openvpn/easy-rsa/keys
$ sudo cp dh2048.pem ca.crt server.crt server.key /etc/openvpn
```

Each client will also need a certificate in order for the OpenVPN server to authenticate it. These keys and certificates will be created on the server and then you will have to copy them over to your clients, which we will do in a later step. It's advised that you generate separate keys and certificates for each client you intend to connect to your VPN.

Because we'll only set up one client here, we called it `client`, but you can change this to a more descriptive name if you'd like:

```
$ cd /etc/openvpn/easy-rsa
$ ./build-key client
```

Finally, copy the versioned OpenSSL configuration file, `openssl-1.0.0.cnf`, to a versionless name, `openssl.cnf`. Failing to do so could result in an error where OpenSSL is unable to load the configuration because it cannot detect its version:

```
$ cp /etc/openvpn/easy-rsa/openssl-1.0.0.cnf /etc/openvpn/easy-rsa/openssl.cnf
```

Now that all the necessary keys and certificates have been generated for your server and client, you can move on to setting up routing between the two machines.

Step 4 — Routing

So far, you've installed OpenVPN on your server, configured it, and generated the keys and certificates needed for your client to access the VPN. However, you have not yet provided OpenVPN with any instructions on where to send incoming web traffic from clients. You can stipulate how the server should handle client traffic by establishing some firewall rules and routing configurations.

Assuming you followed the prerequisites at the start of this tutorial, you should already have firewalld installed and running on your server. To allow OpenVPN through the firewall, you'll need to know what your active firewalld zone is. Find this with the following command:

```
$ sudo firewall-cmd --get-active-zones
```

```
Output
trusted
  Interfaces: tun0
```

Next, add the `openvpn` service to the list of services allowed by firewalld within your active zone, and then make that setting permanent by running the command again but with the `--permanent` option added:

```
$ sudo firewall-cmd --zone=trusted --add-service openvpn
$ sudo firewall-cmd --zone=trusted --add-service openvpn --permanent
```

You can check that the service was added correctly with the following command:

```
$ sudo firewall-cmd --list-services --zone=trusted
```

```
Output
openvpn
```

Next, add a masquerade to the current runtime instance, and then add it again with the `--permanent` option to add the masquerade to all future instances:

```
$ sudo firewall-cmd --add-masquerade
$ sudo firewall-cmd --permanent --add-masquerade
```

You can check that the masquerade was added correctly with this command:

```
$ sudo firewall-cmd --query-masquerade
```

```
Output
yes
```

Next, forward routing to your OpenVPN subnet. You can do this by first creating a variable (`SHARK` in our example) which will represent the primary network interface used by your server, and then using that variable to permanently add the routing rule:

```
$ SHARK=$(ip route get 8.8.8.8 | awk 'NR==1 {print $(NF-2)}')
$ sudo firewall-cmd --permanent --direct --passthrough ipv4 -t nat -A POSTROUTING -
```

Be sure to implement these changes to your firewall rules by reloading firewalld:

```
$ sudo firewall-cmd --reload
```

Next, enable IP forwarding. This will route all web traffic from your client to your server's IP address, and your client's public IP address will effectively be hidden.

Open `sysctl.conf` for editing:

```
$ sudo nano /etc/sysctl.conf
```

Then add the following line at the top of the file:

/etc/sysctl.conf

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

Finally, restart the network service so the IP forwarding will take effect:

```
$ sudo systemctl restart network.service
```

With the routing and firewall rules in place, we can start the OpenVPN service on the server.

Step 5 — Starting OpenVPN

OpenVPN is managed as a systemd service using `systemctl`. We will configure OpenVPN to start up at boot so you can connect to your VPN at any time as long as your server is running. To do this, enable the OpenVPN server by adding it to `systemctl`:

```
$ sudo systemctl -f enable openvpn@server.service
```

Then start the OpenVPN service:

```
$ sudo systemctl start openvpn@server.service
```

Double check that the OpenVPN service is active with the following command. You should see `active (running)` in the output:

```
$ sudo systemctl status openvpn@server.service
```

Output

```
● openvpn@server.service - OpenVPN Robust And Highly Flexible Tunneling Application
   Loaded: loaded (/usr/lib/systemd/system/openvpn@.service; enabled; vendor preset
   Active: **active (running)** since Wed 2018-03-14 15:20:11 EDT; 7s ago
 Main PID: 2824 (openvpn)
   Status: "Initialization Sequence Completed"
   CGroup: /system.slice/system-openvpn.slice/openvpn@server.service
           └─2824 /usr/sbin/openvpn --cd /etc/openvpn/ --config server.conf
. . .
```

We've now completed the server-side configuration for OpenVPN. Next, you will configure your client machine and connect to the OpenVPN server.

Step 6 — Configuring a Client

Regardless of your client machine's operating system, it will need a locally-saved copy of the CA certificate and the client key and certificate generated in Step 3, as well as the static encryption key you generated at the end of Step 2.

Locate the following files **on your server**. If you generated multiple client keys with unique, descriptive names, then the key and certificate names will be different. In this article we used `client`.

```
/etc/openvpn/easy-rsa/keys/ca.crt
/etc/openvpn/easy-rsa/keys/client.crt
/etc/openvpn/easy-rsa/keys/client.key
/etc/openvpn/myvpn.tlsauth
```

Copy these files to your **client machine**. You can use SFTP or your preferred method. You could even just open the files in your text editor and copy and paste the contents into new files on your client machine. Regardless of which method you use, be sure to note where you save these files.

Next, create a file called `client.ovpn` **on your client machine**. This is a configuration file for an OpenVPN client, telling it how to connect to the server:

```
$ sudo nano client.ovpn
```

Then add the following lines to `client.ovpn`. Notice that many of these lines reflect those which we uncommented or added to the `server.conf` file, or were already in it by default:

client.ovpn

```
client
tls-client
ca /path/to/ca.crt
cert /path/to/client.crt
key /path/to/client.key
tls-crypt /path/to/myvpn.tlsauth
remote-cert-eku "TLS Web Client Authentication"
proto udp
remote your_server_ip 1194 udp
dev tun
topology subnet
pull
user nobody
group nobody
```

When adding these lines, please note the following:

- You'll need to change the first line to reflect the name you gave the client in your key and certificate; in our case, this is just `client`
- You also need to update the IP address from `your_server_ip` to the IP address of your server; port `1194` can stay the same
- Make sure the paths to your key and certificate files are correct

This file can now be used by any OpenVPN client to connect to your server. Below are OS-specific instructions for how to connect your client:

Windows:

On Windows, you will need the official OpenVPN Community Edition binaries which come with a GUI. Place your `.ovpn` configuration file into the proper directory, `C:\Program Files\OpenVPN\config`, and click **Connect** in the GUI. OpenVPN GUI on Windows must be executed with administrative privileges.

macOS:

On macOS, the open source application Tunnelblick provides an interface similar to the OpenVPN GUI on Windows, and comes with OpenVPN and the required TUN/TAP drivers. As with Windows, the only step required is to place your `.ovpn` configuration file into the `~/Library/Application Support/Tunnelblick/Configurations` directory. Alternatively, you can double-click on your `.ovpn` file.

Linux:

On Linux, you should install OpenVPN from your distribution's official repositories. You can then invoke OpenVPN by executing:

```
$ sudo openvpn --config ~/path/to/client.ovpn
```

After you establish a successful client connection, you can verify that your traffic is being routed through the VPN by checking Google to reveal your public IP.

Conclusion

You should now have a fully operational virtual private network running on your OpenVPN server. You can browse the web and download content without worrying about malicious actors tracking your activity.

There are several steps you could take to customize your OpenVPN installation even further, such as configuring your client to connect to the VPN automatically or configuring client-specific rules and access policies. For these and other OpenVPN customizations, you should consult the official OpenVPN documentation. If you're interested in other ways you can protect yourself and your machines on the internet, check out our article on 7 Security Measures to Protect Your Servers.

By: Jacob Tomlinson

Editor:
Sharon Campbell

Related Tutorials

How To Set Up an OpenVPN Server on Debian 9

How To Create a Self-Signed SSL Certificate for Nginx on Debian 9

How To Create a Self-Signed SSL Certificate for Apache in Debian 9

How To Set Up vsftpd for a User's Directory on Debian 9

How To Set Up a Firewall with UFW on Debian 9

92 Comments

Leave a comment...

Log In to Comment

nickf *December 8, 2014*

0 Thanks for the guide, Jacob.

I am having issues connecting clients to the openvpn server. I have checked that the necessary udp ports are open and forwarded. Do you have any suggestions on what else I can try? Unfortunately the documentation for CentOS 7 is pretty limited. Thanks!

Here are the errors:
TLS Error: TLS key negotiation failed to occur within 60 seconds (check your network connectivity)
TLS Error: TLS handshake failed
WARNING: No server certificate verification method has been enabled. See http://openvpn.net/howto.html#mitm for more info.

rkovacic *December 10, 2014*

0 Is your client.ovpn file configured correctly? Did you replace the values?

nickf *December 10, 2014*

0 I actually got it working. For some reason I had to change the protocol from udp to tcp to get the firewall to forward the packets correctly.

Now if I can just figure out the ethernet bridging...

tegansnyder *October 23, 2015*

0 I had the same issue. Switching the protocol from UDP in /etc/openvpn/server.conf to TCP and doing the same in the client.ovpn file by changing proto udp to proto tcp did the trick.

nickf *December 10, 2014*

0 Update--got bridging working as well. In case anyone else struggles with it, make sure your NIC or virtual nic is set to promiscuous mode. Otherwise the packets from a client won't reach the local LAN.

ndcast *May 3, 2018*

1 After I applied the verbose flag to the log in the client.ovpn , it said that the EKU value was different.

++ Certificate has EKU (oid) 1.3.6.1.5.5.7.3.1, expects TLS Web Client Authentication
VERIFY EKU ERROR

After I changed

remote-cert-eku "TLS Web Client Authentication"
for
remote-cert-eku 1.3.6.1.5.5.7.3.1

it all worked fine.

By the way, the flag is "verb 3"

louisb *September 18, 2018*

0 I had a similar problem, which was solved by changing the 'Client' to 'Server' in the client.ovpn:

```
remote-cert-eku "TLS Web Server Authentication"
```

q23 *December 11, 2014*

2 A suggestion: .ovpn files support inline certificates and keys. Instead of having to mess around with multiple files and multiple paths, you can just copy and paste everything from -----BEGIN {CERTIFICATE,RSA PRIVATE KEY}----- to -----END {CERTIFICATE,RSA PRIVATE KEY}----- in between tags for each one: <ca> for the CA public key, <cert> for the server or client's public key, <key> for the server or client's private key, and <tls-auth> for the static key if you're using it.

That way, you can have it all packaged up nicely in one .ovpn file instead of having 3-4 files. Makes it a lot easier to use the OpenVPN for Android client, too.

RishavAnand *December 19, 2014*

0 can two users connect at the same time ?
and instead of .key file can there be an option for username & password ?

giuseppelomba *December 22, 2014*

2 I followed this and installed successfully, but when i start openvpn i don't have internet access. why is this?

AlpineLakes *August 31, 2015*

0 Seems odd. If anything following the steps literally will leave your iptables in a fairly open state with a default ACCEPT policy on the OUTPUT chain I believe. What does `iptables -L --line-numbers --verbose` and `iptables -S` have to say? Also, just for grins I wonder what `systemctl status firewalld` and `systemctl status NetworkManager` have to say?

harderworkingcl *October 15, 2016*

0 I am struggling with same issue, here are the commands outputs you've asked;

```
iptables -L --line-numbers --verbose
     Chain INPUT (policy ACCEPT 807 packets, 116K bytes)
     num   pkts bytes target     prot opt in     out     source

     Chain FORWARD (policy ACCEPT 0 packets, 0 bytes)
     num   pkts bytes target     prot opt in     out     source

     Chain OUTPUT (policy ACCEPT 633 packets, 102K bytes)
     num   pkts bytes target     prot opt in     out     source

iptables -S

     -P INPUT ACCEPT
     -P FORWARD ACCEPT
     -P OUTPUT ACCEPT

systemctl status firewalld`
    firewalld.service
    Loaded: masked (/dev/null)
     Active: inactive (dead)
```

```
systemctl status  NetworkManage
```

```
● NetworkManager.service - Network Manager
Loaded: loaded (/usr/lib/systemd/system/NetworkManager.service; enabled; vendor preset:
enabled)
Active: active (running) since Sat 2016-10-15 18:10:40 UTC; 15min ago
Main PID: 418 (NetworkManager)
CGroup: /system.slice/NetworkManager.service
└─418 /usr/sbin/NetworkManager --no-daemon

Oct 15 18:21:40 takinardi01 dhclient[2059]: DHCPDISCOVER on eth1 to 255.255.255.255
port 67 interval 16 (xid=0...ad1d)
Oct 15 18:21:40 takinardi01 NetworkManager[418]: <warn> (eth1): DHCPv4 request timed out.
Oct 15 18:21:40 takinardi01 NetworkManager[418]: <info> (eth1): DHCPv4 state changed
unknown -> timeout
Oct 15 18:21:40 takinardi01 NetworkManager[418]: <info> (eth1): canceled DHCP transaction,
DHCP client pid 2059
Oct 15 18:21:40 takinardi01 NetworkManager[418]: <info> (eth1): DHCPv4 state changed
timeout -> done
Oct 15 18:21:40 takinardi01 NetworkManager[418]: <info> (eth1): device state change: ip-
config -> failed (reas...0 5]
Oct 15 18:21:40 takinardi01 NetworkManager[418]: <info> Disabling autoconnect for
connection 'Wired connection 1'.
Oct 15 18:21:40 takinardi01 NetworkManager[418]: <warn> (eth1): Activation: failed for
connection 'Wired conne...n 1'
Oct 15 18:21:40 takinardi01 NetworkManager[418]: <info> (eth1): device state change: failed -
> disconnected (r...0 0]
Oct 15 18:26:12 takinardi01 systemd[1]: Started Network Manager.
Hint: Some lines were ellipsized, use -l to show in full.
```

By the way i am able to ping mail.google.com from digitalocean vps but i am still not able to connect internet.

---

pshinghal *December 26, 2014*

2 Thanks for the guide! I think the first line of the .ovpn file shouldn't be in red, though. I used a unique name for my client, but using that name as the first line didn't work. When I changed the first line back to "client", it worked fine. I might be missing something, though.

---

freshscaped *December 29, 2014*

0 Did you name the file the same as the first line or did you call it client.ovpn?

---

pshinghal *December 30, 2014*

0 I named it the same as the first line, *my-client-name*.ovpn

freshscaped *December 30, 2014*

0 Thanks. Did you only have to change the first line, or the name of the file too? I'm getting the impression each and every detail has to be correct and I'm not finding enough specific clear guidance so am having to use a lot of guesswork.

pshinghal *January 3, 2015*

1 When I "opened" my *my-client-name*.ovpn file with Tunnelblick, it installed the configuration in the app support directory. There, it renamed the ovpn file to config.ovpn
The key and certs retained their old names. In config.ovpn, I had to change the first line from *my-client-name* to *client*

StarShine *September 8, 2016*

1 yes,the first line of the .ovnp file should be client static.Because the first line tell OpenVpn the role of openvpn,the choise should be server or client.

freshscaped *December 28, 2014*

0 Why Google's DDNS servers? Why not DigitalOcean's or anybody else's?

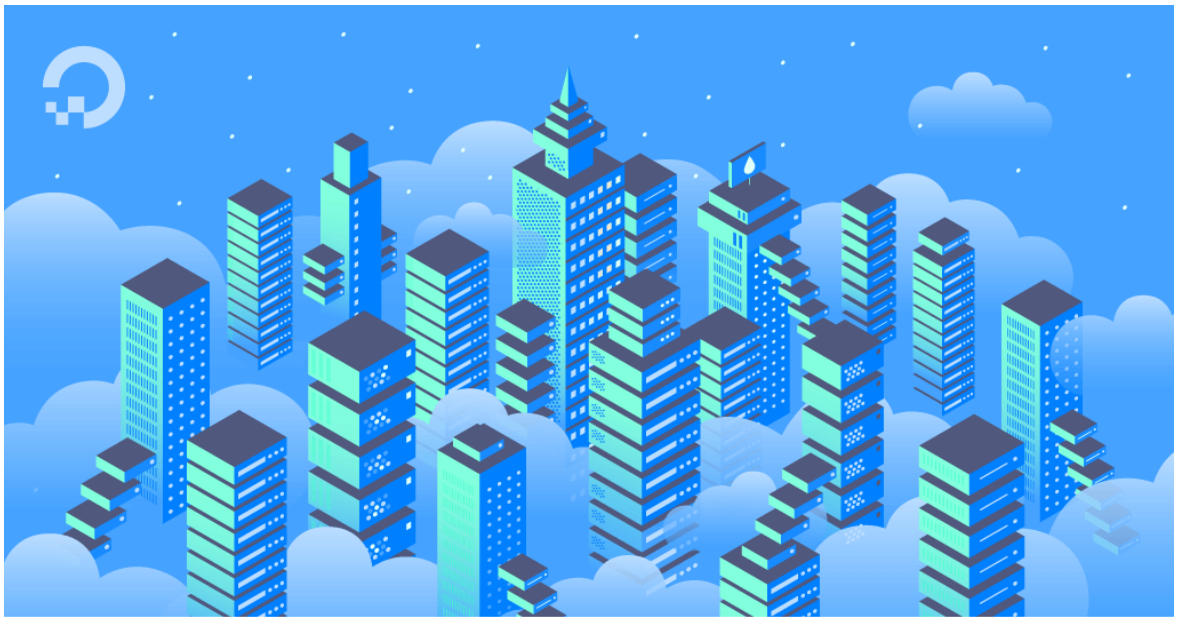
freshscaped *December 29, 2014*

0 There appears to be some confusion here. You say change KEY_CN in /etc/openvpn/easy-rsa/vars but the comments within the file say only uncomment this if you wish to use the same common name for all clients. The advice would appear to be that it is better to have a separate name and key pair for each client. Which is correct, please?

AlpineLakes *August 26, 2015*

0 I have the same question. Note that in How To Secure Traffic Between VPS Using OpenVPN Sep 26, 2013 VPN, Security, Networking, System Tools Ubuntu (published a year earlier) `KEY_CN` seems to be omitted from similar directions.

I'm going to proceed under the assumption that `KEY_CN` should not be defined. If I don't reply to this comment, assume it worked :)



How To Secure Traffic Between VPS Using OpenVPN

by Mason Gravitt

OpenVPN is a great tool to ensure traffic is not eavesdropped. You can use this to ensure a secure connection from your laptop to your DigitalOcean VPS (droplet) as well as between cloud servers. This article is to help get you started on your way to setting

---

LinuxFreakazoid *January 8, 2015*

1 The gui says im connected but my ip does not change. Help please!

---

eNkrypt *January 14, 2015*

0 Hey, thanks for the tutorial. I seem to be having a problem though. I am unable to connect to the server from the client. It simply times out. I assume it's some sort of firewall rule that is blocking it. However, when I run a "nmap -sU -p 1194" i get a "1194/udp open|filtered openvpn" so it seems to be open. When I run a "systemctl status openvpn@server.service" I get:
openvpn@server.service - OpenVPN Robust And Highly Flexible Tunneling Application On server

```
   Loaded: loaded (/usr/lib/systemd/system/openvpn@.service; enabled)
   Active: active (running) since Wed 2015-01-14 18:26:05 UTC; 1h 31min ago
  Process: 7020 ExecStart=/usr/sbin/openvpn --daemon --writepid /var/run/ope
 Main PID: 7027 (openvpn)
   CGroup: /system.slice/system-openvpn.slice/openvpn@server.service
           └─7027 /usr/sbin/openvpn --daemon --writepid /var/run/openvpn/ser

Jan 14 18:26:05 <REDACTED>.net systemd[1]: Started OpenVPN Robust And Highly
Hint: Some lines were ellipsized, use -l to show in full.
```

It appears to be up and running. I tried it both from my phone (which has no firewalls) and my work computer - both don't work. Any idea what could be causing the client to timeout when trying to connect?

Thanks again!

---

camhart *April 21, 2015*

0 I'm having this same issue... did you ever get it figured out?

---

pin *February 28, 2015*

0 I notice in article 'https://www.digitalocean.com/community/tutorials/how-to-set-up-an-openvpn-server-on-ubuntu-14-04', you can find some additional writeup about how to connect to OpenVPN with iOS. Pretty interesting.

ALKateb *March 1, 2015*

2 "You'll need to change the first line to reflect the name you gave the client in your key and certificate; in our case, this is just client"

I do not think this part is true, "client" here is a keyword, to let the software know the following is client configuration

Great tutrorial, thanks a lot

Oniled *March 16, 2015*

0 so I tried these instructions with CentOS7 and they didn't work for me. When I chose to use firewalld instead of IP Tables, I got it working. Here's my steps, using mostly the instructions in this article.

- Log into the server as root
- Prereqs - same as instructions
- Step 1 - same as instructions
- Step 2 - same as instructions
- Step 3 - same as instructions
- Step 4 - completely different instructions. we'll use the built in firewalld with CentOS instead of using IPTables
- open up shell
- run "iptables -t nat -A POSTROUTING -s 10.8.0.0/24 -o eth0 -j MASQUERADE"

- run "iptables-save > /etc/sysconfig/iptables"
- vi /etc/sysctl.conf
- Add the following line at the top of the file "net.ipv4.ip_forward = 1"
- save and exit the file
- run "systemctl restart network.service" so the IP forwarding will take effect
- Step 5 - same steps and then
- after OpenVPN service is started
    - run "firewall-cmd --add-service openvpn"
    - run "firewall-cmd --permanent --add-service openvpn" to confirm it worked, run "firewall-cmd --list-services"
    - run "firewall-cmd --add-masquerade"
    - run "firewall-cmd --permanent --add-masquerade"
    - to confirm it worked, run "firewall-cmd --query-masquerade"
- Step 6 - same as instructions

AlpineLakes *August 27, 2015*

3 If like me you wonder where **10.8.0.0/24** magically appeared from the answer is `/etc/openvpn/server.conf` :

```
# Configure server mode and supply a VPN subnet
# for OpenVPN to draw client addresses from.
# The server will take 10.8.0.1 for itself,
# the rest will be made available to clients.
# Each client will be able to reach the server
# on 10.8.0.1. Comment this line out if you are
# ethernet bridging. See the man page for more info.
server 10.8.0.0 255.255.255.0
```

---

ayjtwfx *November 11, 2016*

0 hi,I tried what you said,I make it and I can connect to VPN from my windows,but every time after I disconnected from VPN,then I can't connect to it forever.
The log said:MANAGEMENT: >STATE:1478892675,RECONNECTING,connection-reset
And I type this: systemctl status openvpn

output like this:

openvpn.service
Loaded: not-found (Reason: No such file or directory)
Active: inactive (dead)

---

vivacarlie *January 23, 2017*

0 the firewall-cmd commands seem to be the firewalld equivalent to the IPTables commands. If you decide to use Firewalld, do you still need to use the IPtables steps to configure a NAT? https://docs.fedoraproject.org/en-US/Fedora/19/html/Security_Guide/sec-Configure_Port_Forwarding-CLI.html

---

wmpr *March 17, 2015*

0 you seem to know what you're doing. if you could, help me understand what to do in step three. how do i find my domain thing?

---

crawfishmedia *March 25, 2015*

0 Worked right on first try of the tutorial. Have a great day sir.

---

yenquan *March 31, 2015*

0 When I followed tut, I got errror below. Please tell me why?

<^>[root@vps openvpn]# systemctl start openvpn@server.service
Job for openvpn@server.service failed. See 'systemctl status openvpn@server.serv
ice' and 'journalctl -xn' for details.
[root@vps openvpn]# sudo systemctl status openvpn@server.service
openvpn@server.service - OpenVPN Robust And Highly Flexible Tunneling Applicatio
n On server
Loaded: loaded (/usr/lib/systemd/system/openvpn@.service; enabled)
Active: failed (Result: exit-code) since Tue 2015-03-31 00:08:51 EDT; 8s ago
Process: 24119 ExecStart=/usr/sbin/openvpn --daemon --writepid /var/run/openvp
n/%i.pid --cd /etc/openvpn/ --config %i.conf (code=exited, status=1/FAILURE)

Mar 31 00:08:51 vps.server.com openvpn[24119]: OpenVPN 2.3.6 x86*64-redhat-l...4*
Mar 31 00:08:51 vps.server.com openvpn[24119]: library versions: OpenSSL 1.0...6
Mar 31 00:08:51 vps.server.com openvpn[24119]: Diffie-Hellman initialized wi...y
Mar 31 00:08:51 vps.server.com openvpn[24119]: Socket Buffers: R=[133120->13...]
*Mar 31 00:08:51 vps.server.com openvpn[24119]: ROUTE*GATEWAY ON_LINK IFACE=v...0
Mar 31 00:08:51 vps.server.com openvpn[24119]: ERROR: Cannot open TUN/TAP de...)
Mar 31 00:08:51 vps.server.com openvpn[24119]: Exiting due to fatal error
Mar 31 00:08:51 vps.server.com systemd[1]: openvpn@server.service: control p...1
Mar 31 00:08:52 vps.server.com systemd[1]: Failed to start OpenVPN Robust An....

Mar 31 00:08:52 vps.server.com systemd[1]: Unit openvpn@server.service enter....
Hint: Some lines were ellipsized, use -l to show in full.<^>

---

sunsiyue *April 2, 2015*

0 Great tutorial!
I managed to setup my VPN server, but the clients are only able to connect for a while, about 5 minutes, before losing all internet connection. I could manually reconnect, however, the problem appears again.
Did anyone else encounter this problem? Did I miss any step during configuration?

---

t.wengerd *April 3, 2015*

3 Thanks for the tutorial! A couple of changes I had to make:

1. I customized the **client** part in all of the instances that it came up (**client.key** became **tyler.key**, etc.), but I still had to include **client** instead of **tyler** at the top of the .ovpn file created in Step 6 (which was named **tyler.ovpn**). OpenVPN documentation states that the word "client" indicates a client connection and isn't specific to the key used (as ALKateb also mentioned above).
2. For step 4, I was using firewalld and didn't have to touch iptables at all - the following worked as a complete replacement for step 4 for me:

```
firewall-cmd --permanent --add-service openvpn
firewall-cmd --permanent --add-masquerade

(Add the following line at the top of /etc/sysctl.conf:)
net.ipv4.ip_forward = 1

systemctl restart firewalld
systemctl restart network.service
```

---

pyotruk *April 13, 2015*

0 If you get error: "no such file or directory" after "systemctl -f enable openvpn@server.service" in STEP 5, call:

```
ln -s /lib/systemd/system/openvpn\@.service /etc/systemd/system/multi-user.t
```

Helps for me.
For more info read https://fedoraproject.org/wiki/Openvpn

kyl191 *April 16, 2015*

0 The equivalent systemctl command is `systemctl enable openvpn@.service`

Note the lack of config filename - systemd doesn't appear to allow you to selectively enable individual OpenVPN servers to start on boot.

dthommail *October 8, 2015*

0 Yup. This seems to have worked for me too. Thanks!

JeffersonG *May 17, 2015*

0 helo everyone. I repeatedly followed and check if I missed something in configuration and verified there is nothing wrong on my configuration but I still can't connect to the openvpn server. This is my set up. I created a vm in virtualbox for centos server with eth0=192.168.15.6(bridge adapter) and eth1=192.168.10.1(host only). I also created another centos which has eth0=192.168.15.3(bridge adapter). What I am trying to do is to reach eth1=192.168.10.1 of the server and ping it but to no success. please help me.

AlpineLakes *August 27, 2015*

0 **careful** - if you have iptables rule set in place before starting this procedure, flushing the rule set is likely not what you want to do. At least make a backup before flushing:

```
iptables -L --line-numbers > ~/iptables-rules-backup
```

Load More Comments