**DigitalOcean**                                                                    ≡
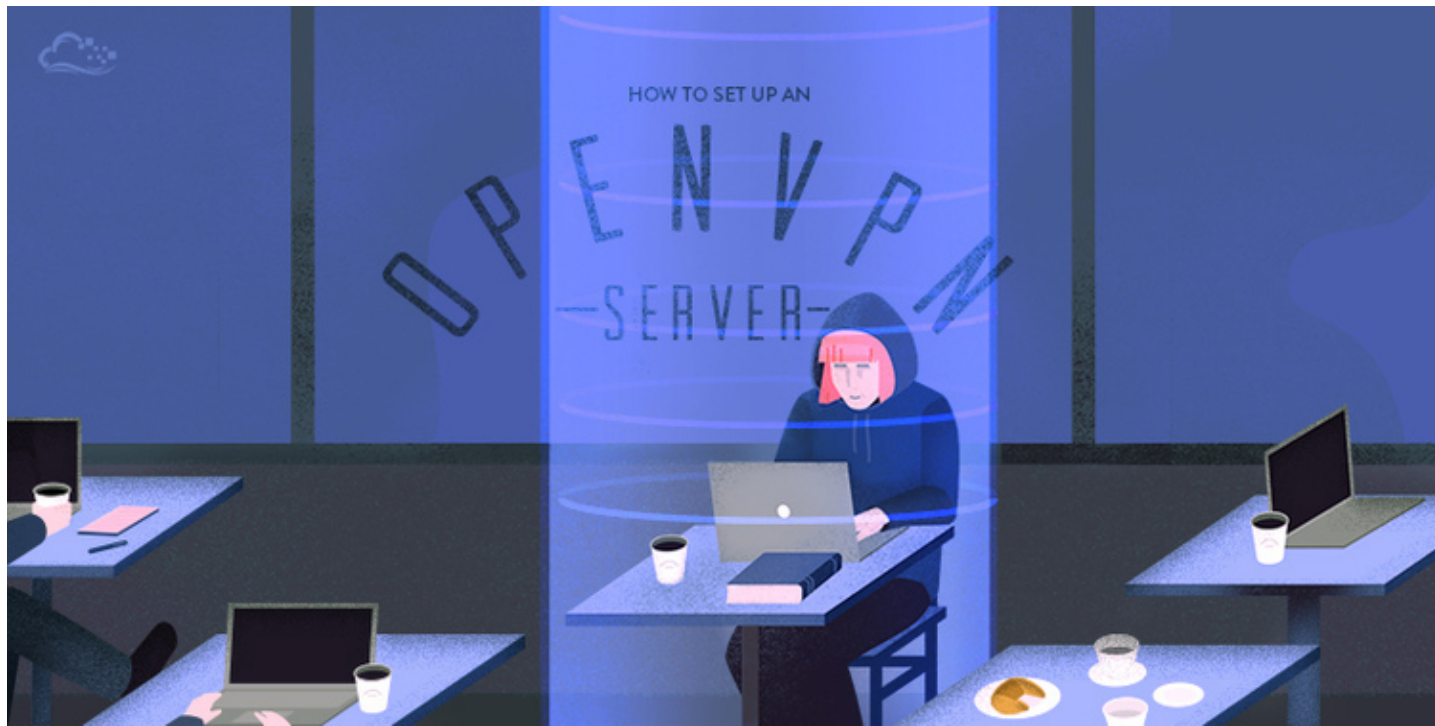
By: Justin Ellingwood                                          ⌄          ⌐+ Subscribe



 How To Set Up an OpenVPN Server on Ubuntu
16.04

                                                                            ᴧ
                                                                            ♡
                                                                           197

Posted May 4, 2016      👁 1.2m     VPN      UBUNTU      UBUNTU 16.04

## Introduction

Want to access the Internet safely and securely from your smartphone or laptop when connected
to an untrusted network such as the WiFi of a hotel or coffee shop? A Virtual Private Network
(VPN) allows you to traverse untrusted networks privately and securely as if you were on a
private network. The traffic emerges from the VPN server and continues its journey to the
destination.

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, we'll set up an OpenVPN server on
                                                                                    tutorial will

# Prerequisites

To complete this tutorial, you will need access to an Ubuntu 16.04 server.

You will need to configure a non-root user with `sudo` privileges before you start this guide. You can follow our Ubuntu 16.04 initial server setup guide to set up a user with appropriate permissions. The linked tutorial will also set up a **firewall**, which we will assume is in place during this guide.

When you are ready to begin, log into your Ubuntu server as your `sudo` user and continue below.

# Step 1: Install OpenVPN

To start off, we will install OpenVPN onto our server. OpenVPN is available in Ubuntu's default repositories, so we can use `apt` for the installation. We will also be installing the `easy-rsa` package, which will help us set up an internal CA (certificate authority) for use with our VPN.

To update your server's package index and install the necessary packages type:

```
$ sudo apt-get update
$ sudo apt-get install openvpn easy-rsa
```

The needed software is now on the server, ready to be configured.

# Step 2: Set Up the CA Directory

OpenVPN is an TLS/SSL VPN. This means that it utilizes certificates in order to encrypt traffic between the server and clients. In order to issue trusted certificates, we will need to set up our own simple certificate authority (CA).

To begin, we can copy the `easy-rsa` template directory into our home directory with the `make-cadir` command:

```
$ make-cadir ~/openvpn-ca
```

Move into the newly created directory to begin configuring the CA:

```
$ cd ~/openvpn-ca
```

# Step 3: Configure the CA Variables

To configure the values our CA will use, we need to edit the `vars` file within the directory. Open that file now in your text editor:

```
$ nano vars
```

Inside, you will find some variables that can be adjusted to determine how your certificates will be created. We only need to worry about a few of these.

Towards the bottom of the file, find the settings that set field defaults for new certificates. It should look something like this:

~/openvpn-ca/vars

```
. . .

export KEY_COUNTRY="US"
export KEY_PROVINCE="CA"
export KEY_CITY="SanFrancisco"
export KEY_ORG="Fort-Funston"
export KEY_EMAIL="me@myhost.mydomain"
export KEY_OU="MyOrganizationalUnit"

. . .
```

Edit the values in red to whatever you'd prefer, but do not leave them blank:

~/openvpn-ca/vars

```
. . .

export KEY_COUNTRY="US"
export KEY_PROVINCE="NY"
export KEY_CITY="New York City"
export KEY_ORG="DigitalOcean"
export KEY_EMAIL="admin@example.com"
export KEY_OU="Community"

. . .
```

While we are here, we will also edit the `KEY_NAME` value just below this section, which populates

~/openvpn-ca/vars

```
export KEY_NAME="server"
```

When you are finished, save and close the file.

# Step 4: Build the Certificate Authority

Now, we can use the variables we set and the `easy-rsa` utilities to build our certificate authority.

Ensure you are in your CA directory, and then source the `vars` file you just edited:

```
$ cd ~/openvpn-ca
$ source vars
```

You should see the following if it was sourced correctly:

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

Make sure we're operating in a clean environment by typing:

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

Now, we can build our root CA by typing:

```
$ ./build-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:

```
Output
Generating a 2048 bit RSA private key
.............................................................................+++
.............................+++
writing new private key to 'ca.key'
-----
```

```
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) [US]:
State or Province Name (full name) [NY]:
Locality Name (eg, city) [New York City]:
Organization Name (eg, company) [DigitalOcean]:
Organizational Unit Name (eg, section) [Community]:
Common Name (eg, your name or your server's hostname) [DigitalOcean CA]:
Name [server]:
Email Address [admin@email.com]:
```

We now have a CA that can be used to create the rest of the files we need.

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

Next, we will generate our server certificate and key pair, as well as some additional files used during the encryption process.

Start by generating the OpenVPN server certificate and key pair. We can do this by typing:

> **Note**: If you choose a name other than `server` here, you will have to adjust some of the
> instructions below. For instance, when copying the generated files to the `/etc/openvpn` directroy,
> you will have to substitute the correct names. You will also have to modify the
> `/etc/openvpn/server.conf` file later to point to the correct `.crt` and `.key` files.

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

Once again, the prompts will have default values based on the argument we just passed in (`server`) and the contents of our `vars` file we sourced.

Feel free to accept the default values by pressing **ENTER**. Do *not* enter a challenge password for this setup. Towards the end, you will have to enter **y** to two questions to sign and commit the certificate:

2018/4/18 How To Set Up an OpenVPN Server on Ubuntu 16.04 | DigitalOcean

```
Certificate is to be certified until May  1 17:51:16 2026 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
```

Next, we'll generate a few other items. We can generate a strong Diffie-Hellman keys to use during key exchange by typing:

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

This might take a few minutes to complete.

Afterwards, we can generate an HMAC signature to strengthen the server's TLS integrity verification capabilities:

```
$ openvpn --genkey --secret keys/ta.key
```

# Step 6: Generate a Client Certificate and Key Pair

Next, we can generate a client certificate and key pair. Although this can be done on the client machine and then signed by the server/CA for security purposes, for this guide we will generate the signed key on the server for the sake of simplicity.

We will generate a single client key/certificate for this guide, but if you have more than one client, you can repeat this process as many times as you'd like. Pass in a unique value to the script for each client.

Because you may come back to this step at a later time, we'll re-source the `vars` file. We will use `client1` as the value for our first certificate/key pair for this guide.

To produce credentials without a password, to aid in automated connections, use the `build-key` command like this:

```
$ cd ~/openvpn-ca
$ source vars
$ ./build-key client1
```

If instead, you wish to create a password-protected set of credentials, use the `build-key-pass` command:

```
$ cd ~/openvpn-ca
$ source vars
$ ./build-key-pass client1
```

Again, the defaults should be populated, so you can just hit **ENTER** to continue. Leave the challenge password blank and make sure to enter **y** for the prompts that ask whether to sign and commit the certificate.

# Step 7: Configure the OpenVPN Service

Next, we can begin configuring the OpenVPN service using the credentials and files we've generated.

## Copy the Files to the OpenVPN Directory

To begin, we need to copy the files we need to the `/etc/openvpn` configuration directory.

We can start with all of the files that we just generated. These were placed within the `~/openvpn-ca/keys` directory as they were created. We need to move our CA cert, our server cert and key, the HMAC signature, and the Diffie-Hellman file:

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

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

```
$ gunzip -c /usr/share/doc/openvpn/examples/sample-config-files/server.conf.gz | sudo tee /etc/open
```

◄ ▐▐▐▐▐▐▐▐▐▐▐▐▐▐▐▐▐▐▐▐▐▐▐▐▐▐▐▐▐▐▐▐▐▐▐▐▐▐▐▐▐▐▐▐▐ ►

## Adjust the OpenVPN Configuration

Now that our files are in place, we can modify the server configuration file:

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

First, find the HMAC section by looking for the `tls-auth` directive. Remove the ";" to uncomment the `tls-auth` line. Below this, add the `key-direction` parameter set to "0":

/etc/openvpn/server.conf

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

Next, find the section on cryptographic ciphers by looking for the commented out `cipher` lines. The `AES-128-CBC` cipher offers a good level of encryption and is well supported. Remove the ";" to uncomment the `cipher AES-128-CBC` line:

/etc/openvpn/server.conf

```
cipher AES-128-CBC
```

Below this, add an `auth` line to select the HMAC message digest algorithm. For this, `SHA256` is a good choice:

/etc/openvpn/server.conf

```
auth SHA256
```

Finally, find the `user` and `group` settings and remove the ";" at the beginning of to uncomment those lines:

/etc/openvpn/server.conf

```
user nobody
group nogroup
```

## (Optional) Push DNS Changes to Redirect All Traffic Through the VPN

The settings above will create the VPN connection between the two machines, but will not force any connections to use the tunnel. If you wish to use the VPN to route all of your traffic, you will likely want to push the DNS settings to the client computers.

You can do this, uncomment a few directives that will configure client machines to redirect all web traffic through the VPN. Find the `redirect-gateway` section and remove the semicolon ";" from the beginning of the `redirect-gateway` line to uncomment it:

/etc/openvpn/server.conf

Just below this, find the `dhcp-option` section. Again, remove the ";" from in front of both of the lines to uncomment them:

<div align="center">/etc/openvpn/server.conf</div>

```
push "dhcp-option DNS 208.67.222.222"
push "dhcp-option DNS 208.67.220.220"
```

This should assist clients in reconfiguring their DNS settings to use the VPN tunnel for as the default gateway.

## (Optional) Adjust the Port and Protocol

By default, the OpenVPN server uses port 1194 and the UDP protocol to accept client connections. If you need to use a different port because of restrictive network environments that your clients might be in, you can change the `port` option. If you are not hosting web content your OpenVPN server, port 443 is a popular choice since this is usually allowed through firewall rules.

<div align="center">/etc/openvpn/server.conf</div>

```
# Optional!
port 443
```

Often if the protocol will be restricted to that port as well. If so, change `proto` from UDP to TCP:

<div align="center">/etc/openvpn/server.conf</div>

```
# Optional!
proto tcp
```

If you have no need to use a different port, it is best to leave these two settings as their default.

## (Optional) Point to Non-Default Credentials

If you selected a different name during the `./build-key-server` command earlier, modify the `cert` and `key` lines that you see to point to the appropriate `.crt` and `.key` files. If you used the default `server`, this should already be set correctly:

<div align="center">/etc/openvpn/server.conf</div>

```
cert server.crt
key server.key
```

# Step 8: Adjust the Server Networking Configuration

Next, we need to adjust some aspects of the server's networking so that OpenVPN can correctly route traffic.

## Allow IP Forwarding

First, we need to allow the server to forward traffic. This is fairly essential to the functionality we want our VPN server to provide.

We can adjust this setting by modifying the `/etc/sysctl.conf` file:

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

Inside, look for the line that sets `net.ipv4.ip_forward`. Remove the "#" character from the beginning of the line to uncomment that setting:

/etc/sysctl.conf

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

Save and close the file when you are finished.

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

```
$ sudo sysctl -p
```

## Adjust the UFW Rules to Masquerade Client Connections

If you followed the Ubuntu 16.04 initial server setup guide in the prerequisites, you should have the UFW firewall in place. Regardless of whether you use the firewall to block unwanted traffic (which you almost always should do), we need the firewall in this guide to manipulate some of the traffic coming into the server. We need to modify the rules file to set up masquerading, an `iptables` concept that provides on-the-fly dynamic NAT to correctly route client connections.

Before we open the firewall configuration file to add masquerading, we need to find the public network interface of our machine. To do this, type:

```
$ ip route | grep default
```

Your public interface should follow the word "dev". For example, this result shows the interface named `wlp11s0`, which is highlighted below:

```
Output
default via 203.0.113.1 dev wlp11s0  proto static  metric 600
```

When you have the interface associated with your default route, open the `/etc/ufw/before.rules` file to add the relevant configuration:

```
$ sudo nano /etc/ufw/before.rules
```

This file handles configuration that should be put into place before the conventional UFW rules are loaded. Towards the top of the file, add the highlighted lines below. This will set the default policy for the `POSTROUTING` chain in the `nat` table and masquerade any traffic coming from the VPN:

> **Note**: Remember to replace `wlp11s0` in the `-A POSTROUTING` line below with the interface you found in the above command.

/etc/ufw/before.rules

```
#
# rules.before
#
# 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 wlp11s0 (change to the interface you discovered!)
-A POSTROUTING -s 10.8.0.0/8 -o wlp11s0 -j MASQUERADE
COMMIT
# END OPENVPN RULES
```

Save and close the file when you are finished.

We need to tell UFW to allow forwarded packets by default as well. To do this, we will open the `/etc/default/ufw` file:

```
$ sudo nano /etc/default/ufw
```

Inside, find the `DEFAULT_FORWARD_POLICY` directive. We will change the value from `DROP` to `ACCEPT`:

<div align="center">/etc/default/ufw</div>

```
DEFAULT_FORWARD_POLICY="ACCEPT"
```

Save and close the file when you are finished.

## Open the OpenVPN Port and Enable the Changes

Next, we'll adjust the firewall itself to allow traffic to OpenVPN.

If you did not change the port and protocol in the `/etc/openvpn/server.conf` file, you will need to open up UDP traffic to port 1194. If you modified the port and/or protocol, substitute the values you selected here.

We'll also add the SSH port in case you forgot to add it when following the prerequisite tutorial:

```
$ sudo ufw allow 1194/udp
$ sudo ufw allow OpenSSH
```

Now, we can disable and re-enable UFW to load the changes from all of the files we've modified:

```
$ sudo ufw disable
$ sudo ufw enable
```

Our server is now configured to correctly handle OpenVPN traffic.

# Step 9: Start and Enable the OpenVPN Service

We're finally ready to start the OpenVPN service on our server. We can do this using systemd.

We need to start the OpenVPN server by specifying our configuration file name as an instance variable after the systemd unit file name. Our configuration file for our server is called /etc/openvpn/server.conf , so we will add @server to end of our unit file when calling it:

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

Double-check that the service has started successfully by typing:

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

If everything went well, your output should look something that looks like this:

```
Output
● openvpn@server.service - OpenVPN connection to server
   Loaded: loaded (/lib/systemd/system/openvpn@.service; disabled; vendor preset: enabled)
   Active: active (running) since Tue 2016-05-03 15:30:05 EDT; 47s ago
     Docs: man:openvpn(8)
           https://community.openvpn.net/openvpn/wiki/Openvpn23ManPage
           https://community.openvpn.net/openvpn/wiki/HOWTO
  Process: 5852 ExecStart=/usr/sbin/openvpn --daemon ovpn-%i --status /run/openvpn/%i.status 10 --c
 Main PID: 5856 (openvpn)
    Tasks: 1 (limit: 512)
   CGroup: /system.slice/system-openvpn.slice/openvpn@server.service
           └─5856 /usr/sbin/openvpn --daemon ovpn-server --status /run/openvpn/server.status 10 --

May 03 15:30:05 openvpn2 ovpn-server[5856]: /sbin/ip addr add dev tun0 local 10.8.0.1 peer 10.8.0.2
May 03 15:30:05 openvpn2 ovpn-server[5856]: /sbin/ip route add 10.8.0.0/24 via 10.8.0.2
May 03 15:30:05 openvpn2 ovpn-server[5856]: GID set to nogroup
May 03 15:30:05 openvpn2 ovpn-server[5856]: UID set to nobody
May 03 15:30:05 openvpn2 ovpn-server[5856]: UDPv4 link local (bound): [undef]
May 03 15:30:05 openvpn2 ovpn-server[5856]: UDPv4 link remote: [undef]
May 03 15:30:05 openvpn2 ovpn-server[5856]: MULTI: multi_init called, r=256 v=256
May 03 15:30:05 openvpn2 ovpn-server[5856]: IFCONFIG POOL: base=10.8.0.4 size=62, ipv6=0
May 03 15:30:05 openvpn2 ovpn-server[5856]: IFCONFIG POOL LIST
May 03 15:30:05 openvpn2 ovpn-server[5856]: Initialization Sequence Completed
```

You can also check that the OpenVPN tun0 interface is available by typing:

```
$ ip addr show tun0
```

```
Output
4: tun0: <POINTOPOINT,MULTICAST,NOARP,UP,LOWER_UP> mtu 1500 qdisc noqueue state UNKNOWN group defaul
    link/none
    inet 10.8.0.1 peer 10.8.0.2/32 scope global tun0
        valid_lft forever preferred_lft forever
```

If everything went well, enable the service so that it starts automatically at boot:

```
$ sudo systemctl enable openvpn@server
```

# Step 10: Create Client Configuration Infrastructure

Next, we need to set up a system that will allow us to create client configuration files easily.

## Creating the Client Config Directory Structure

Create a directory structure within your home directory to store the files:

```
$ mkdir -p ~/client-configs/files
```

Since our client configuration files will have the client keys embedded, we should lock down permissions on our inner directory:

```
$ chmod 700 ~/client-configs/files
```

## Creating a Base Configuration

Next, let's copy an example client configuration into our directory to use as our base configuration:

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

Open this new file in your text editor:

```
$ nano ~/client-configs/base.conf
```

First, locate the `remote` directive. This points the client to our OpenVPN server address. This should be the public IP address of your OpenVPN server. If you changed the port that the OpenVPN server is listening on, change `1194` to the port you selected:

~/client-configs/base.conf

```
. . .
# The hostname/IP and port of the server.
# You can have multiple remote entries
# to load balance between the servers.
remote server_IP_address 1194
. . .
```

Be sure that the protocol matches the value you are using in the server configuration:

~/client-configs/base.conf

```
proto udp
```

Next, uncomment the `user` and `group` directives by removing the ";":

~/client-configs/base.conf

```
# Downgrade privileges after initialization (non-Windows only)
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:

~/client-configs/base.conf

```
# SSL/TLS parms.
# See the server config file for more
# description.  It's best to use
# a separate .crt/.key file pair
# for each client.  A single ca
# file can be used for all clients.
#ca ca.crt
#cert client.crt
#key client.key
```

Mirror the `cipher` and `auth` settings that we set in the `/etc/openvpn/server.conf` file:

Sign up for our newsletter. Get the latest tutorials on SysAdmin and open source topics.  ✕

Enter your email address                                    Sign Up

```
cipher AES-128-CBC
auth SHA256
```

Next, add the `key-direction` directive somewhere in the file. This **must** be set to "1" to work with the server:

<div align="center">~/client-configs/base.conf</div>

```
key-direction 1
```

Finally, add a few **commented out** lines. We want to include these with every config, but should only enable them for Linux clients that ship with a `/etc/openvpn/update-resolv-conf` file. This script uses the `resolvconf` utility to update DNS information for Linux clients.

<div align="center">~/client-configs/base.conf</div>

```
# script-security 2
# up /etc/openvpn/update-resolv-conf
# down /etc/openvpn/update-resolv-conf
```

If your client is running Linux and has an `/etc/openvpn/update-resolv-conf` file, you should uncomment these lines from the generated OpenVPN client configuration file.

Save the file when you are finished.

## Creating a Configuration Generation Script

Next, we will create a simple script to compile our base configuration with the relevant certificate, key, and encryption files. This will place the generated configuration in the `~/client-configs/files` directory.

Create and open a file called `make_config.sh` within the `~/client-configs` directory:

```
$ nano ~/client-configs/make_config.sh
```

Inside, paste the following script:

<div align="center">~/client-configs/make_config.sh</div>

```
#!/bin/bash
```

```
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
```

Save and close the file when you are finished.

Mark the file as executable by typing:

```
$ chmod 700 ~/client-configs/make_config.sh
```

## Step 11: Generate Client Configurations

Now, we can easily generate client configuration files.

If you followed along with the guide, you created a client certificate and key called `client1.crt` and `client1.key` respectively by running the `./build-key client1` command in step 6. We can generate a config for these credentials by moving into our `~/client-configs` directory and using the script we made:

```
$ cd ~/client-configs
$ ./make_config.sh client1
```

If everything went well, we should have a `client1.ovpn` file in our `~/client-configs/files` directory:

```
$ ls ~/client-configs/files
```

## Transferring Configuration to Client Devices

We need to transfer the client configuration file to the relevant device. For instance, this could be your local computer or a mobile device.

While the exact applications used to accomplish this transfer will depend on your choice and device's operating system, you want the application to use SFTP (SSH file transfer protocol) or SCP (Secure Copy) on the backend. This will transport your client's VPN authentication files over an encrypted connection.

Here is an example SFTP command using our client1.ovpn example. This command can be run from your local computer (OS X or Linux). It places the  .ovpn  file in your home directory:

```
local$  sftp sammy@openvpn_server_ip:client-configs/files/client1.ovpn ~/
```

Here are several tools and tutorials for securely transferring files from the server to a local computer:

- WinSCP

- How To Use SFTP to Securely Transfer Files with a Remote Server

- How To Use Filezilla to Transfer and Manage Files Securely on your VPS

# Step 12: Install the Client Configuration

Now, we'll discuss how to install a client VPN profile on Windows, OS X, iOS, and Android. None of these client instructions are dependent on one another, so feel free to skip to whichever is applicable to you.

The OpenVPN connection will be called whatever you named the  .ovpn  file. In our example, this means that the connection will be called  client1.ovpn  for the first client file we generated.

## Windows

### Installing

The OpenVPN client application for Windows can be found on OpenVPN's Downloads page. Choose the appropriate installer version for your version of Windows.

Note

Sign up for our newsletter. Get the latest tutorials on SysAdmin and open source topics.      ✕

Enter your email address                                    Sign Up

After installing OpenVPN, copy the `.ovpn` file to:

```
C:\Program Files\OpenVPN\config
```

When you launch OpenVPN, it will automatically see the profile and makes it available.

OpenVPN must be run as an administrator each time it's used, even by administrative accounts. To do this without having to right-click and select **Run as administrator** every time you use the VPN, you can preset this, but this must be done from an administrative account. This also means that standard users will need to enter the administrator's password to use OpenVPN. On the other hand, standard users can't properly connect to the server unless the OpenVPN application on the client has admin rights, so the elevated privileges are necessary.

To set the OpenVPN application to always run as an administrator, right-click on its shortcut icon and go to **Properties**. At the bottom of the **Compatibility** tab, click the button to **Change settings for all users**. In the new window, check **Run this program as an administrator**.

### Connecting

Each time you launch the OpenVPN GUI, Windows will ask if you want to allow the program to make changes to your computer. Click **Yes**. Launching the OpenVPN client application only puts the applet in the system tray so that the VPN can be connected and disconnected as needed; it does not actually make the VPN connection.

Once OpenVPN is started, initiate a connection by going into the system tray applet and right-clicking on the OpenVPN applet icon. This opens the context menu. Select **client1** at the top of the menu (that's our `client1.ovpn` profile) and choose **Connect**.

A status window will open showing the log output while the connection is established, and a message will show once the client is connected.

Disconnect from the VPN the same way: Go into the system tray applet, right-click the OpenVPN applet icon, select the client profile and click **Disconnect**.

## OS X

### Installing

Tunnelblick is a free, open source OpenVPN client for Mac OS X. You can download the latest disk image from the Tunnelblick Downloads page. Double-click the downloaded `.dmg` file and

Towards the end of the installation process, Tunnelblick will ask if you have any configuration files. It can be easier to answer **No** and let Tunnelblick finish. Open a Finder window and double-click `client1.ovpn`. Tunnelblick will install the client profile. Administrative privileges are required.

**Connecting**

Launch Tunnelblick by double-clicking Tunnelblick in the **Applications** folder. Once Tunnelblick has been launched, there will be a Tunnelblick icon in the menu bar at the top right of the screen for controlling connections. Click on the icon, and then the **Connect** menu item to initiate the VPN connection. Select the **client1** connection.

# Linux

## Installing

If you are using Linux, there are a variety of tools that you can use depending on your distribution. Your desktop environment or window manager might also include connection utilities.

The most universal way of connecting, however, is to just use the OpenVPN software.

On Ubuntu or Debian, you can install it just as you did on the server by typing:

```
client$ sudo apt-get update
client$ sudo apt-get install openvpn
```

On CentOS you can enable the EPEL repositories and then install it by typing:

```
client$ sudo yum install epel-release
client$ sudo yum install openvpn
```

## Configuring

Check to see if your distribution includes a `/etc/openvpn/update-resolv-conf` script:

```
client$ ls /etc/openvpn
```

```
Output
update-resolve-conf
```

Sign up for our newsletter. Get the latest tutorials on SysAdmin and open source topics.　　✕

Enter your email address　　　　　　　　　　　　　　Sign Up

```
client$ nano client1.ovpn
```

Uncomment the three lines we placed in to adjust the DNS settings if you were able to find an `update-resolv-conf` file:

<div align="center">client1.ovpn</div>

```
script-security 2
up /etc/openvpn/update-resolv-conf
down /etc/openvpn/update-resolv-conf
```

If you are using CentOS, change the `group` from `nogroup` to `nobody` to match the distribution's available groups:

<div align="center">client1.ovpn</div>

```
group nobody
```

Save and close the file.

Now, you can connect to the VPN by just pointing the `openvpn` command to the client configuration file:

```
client$ sudo openvpn --config client1.ovpn
```

This should connect you to your server.

## iOS

### Installing

From the iTunes App Store, search for and install OpenVPN Connect, the official iOS OpenVPN client application. To transfer your iOS client configuration onto the device, connect it directly to a computer.

Completing the transfer with iTunes will be outlined here. Open iTunes on the computer and click on **iPhone** > **apps**. Scroll down to the bottom to the **File Sharing** section and click the OpenVPN app. The blank window to the right, **OpenVPN Documents**, is for sharing files. Drag the `.ovpn` file to the OpenVPN Documents window.

---

Sign up for our newsletter. Get the latest tutorials on SysAdmin and open source topics.                    ✕
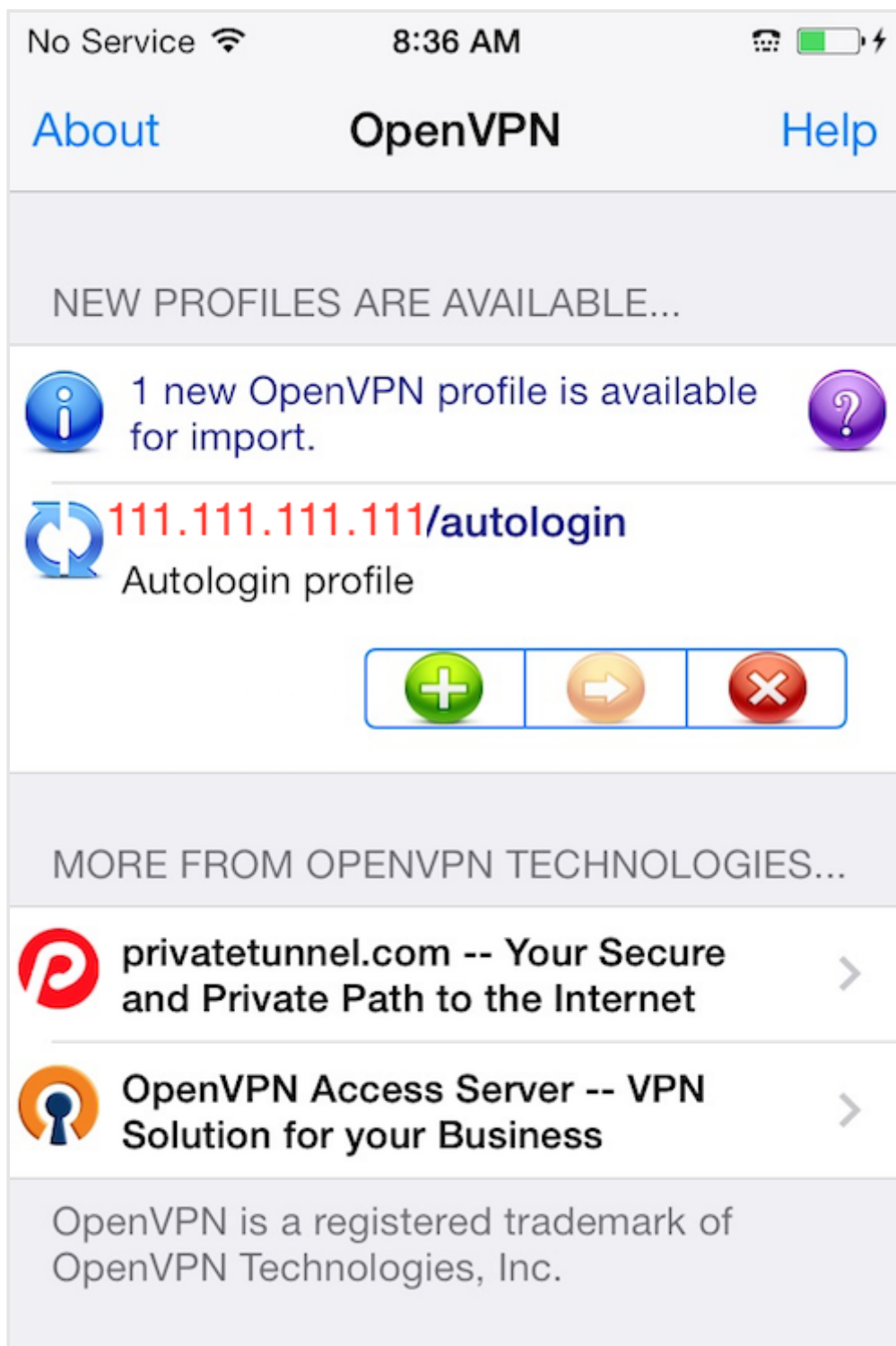
| Enter your email address | Sign Up |
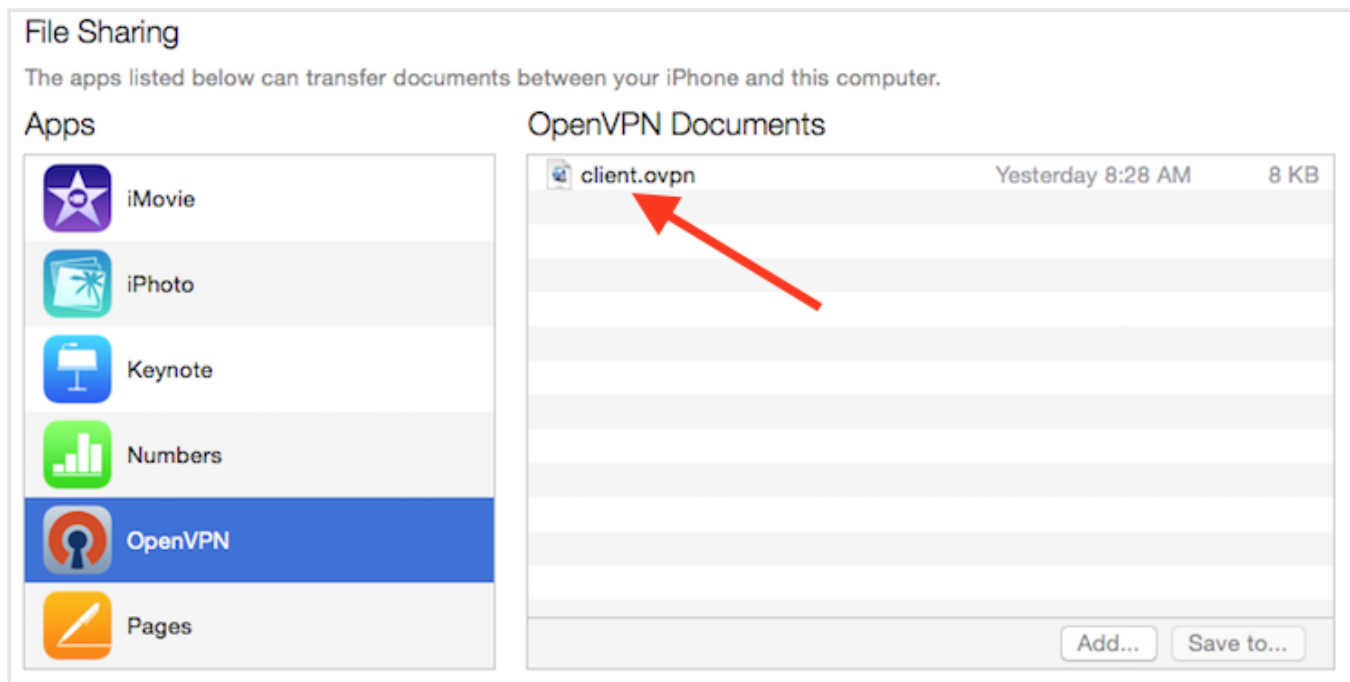
Now launch the OpenVPN app on the iPhone. There will be a notification that a new profile is ready to import. Tap the green plus sign to import it.

**Connecting**

OpenVPN is now ready to use with the new profile. Start the connection by sliding the **Connect** button to the **On** position. Disconnect by sliding the same button to **Off**.
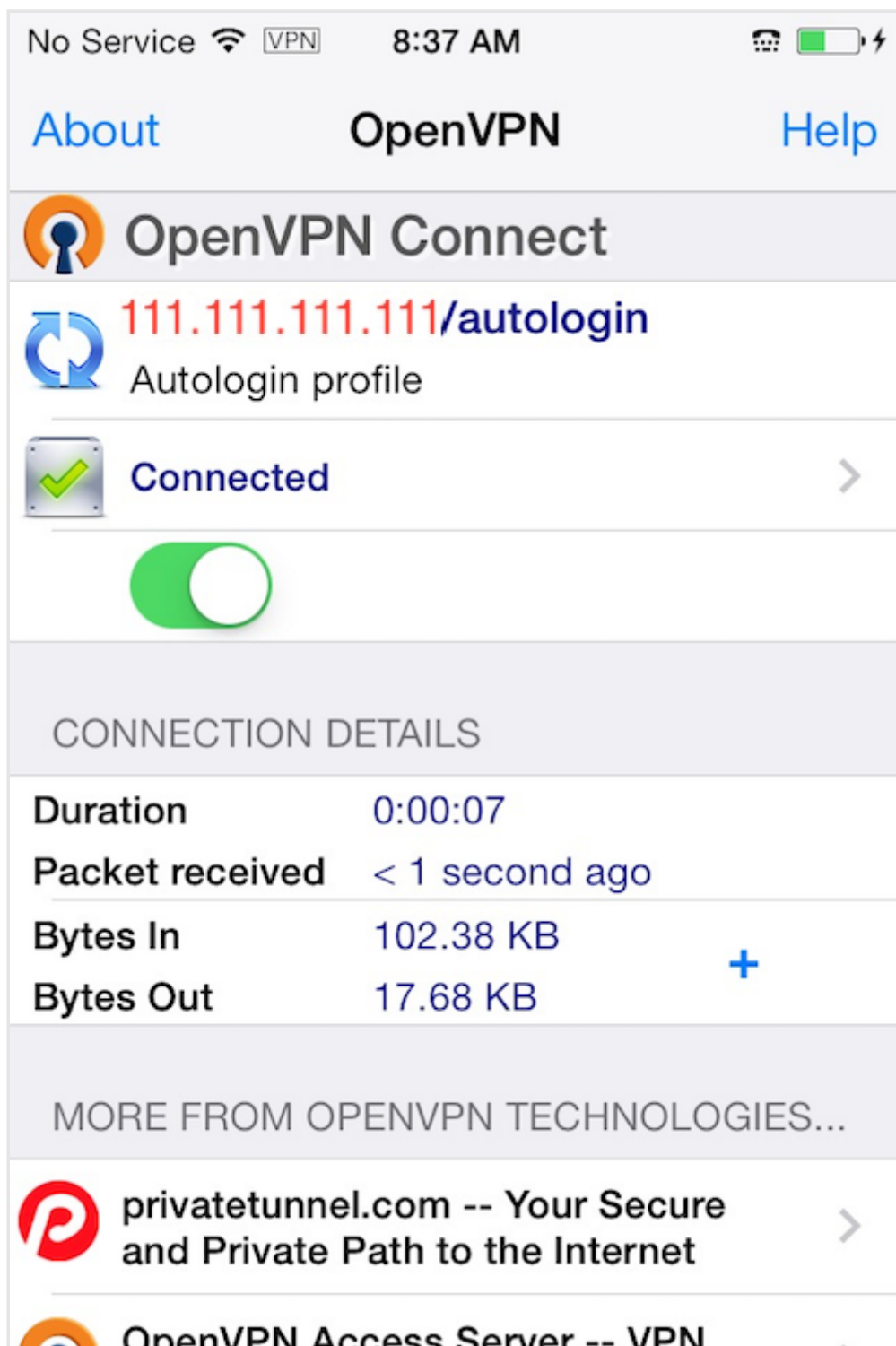
Note

The VPN switch under **Settings** cannot be used to connect to the VPN. If you try, you will receive a notice to only connect using the OpenVPN app.

## Android

**Installing**

Open the Google Play Store. Search for and install Android OpenVPN Connect, the official
Android OpenVPN client application.

The `.ovpn` profile can be transferred by connecting the Android device to your computer by USB
and copying the file over. Alternatively, if you have an SD card reader, you can remove the
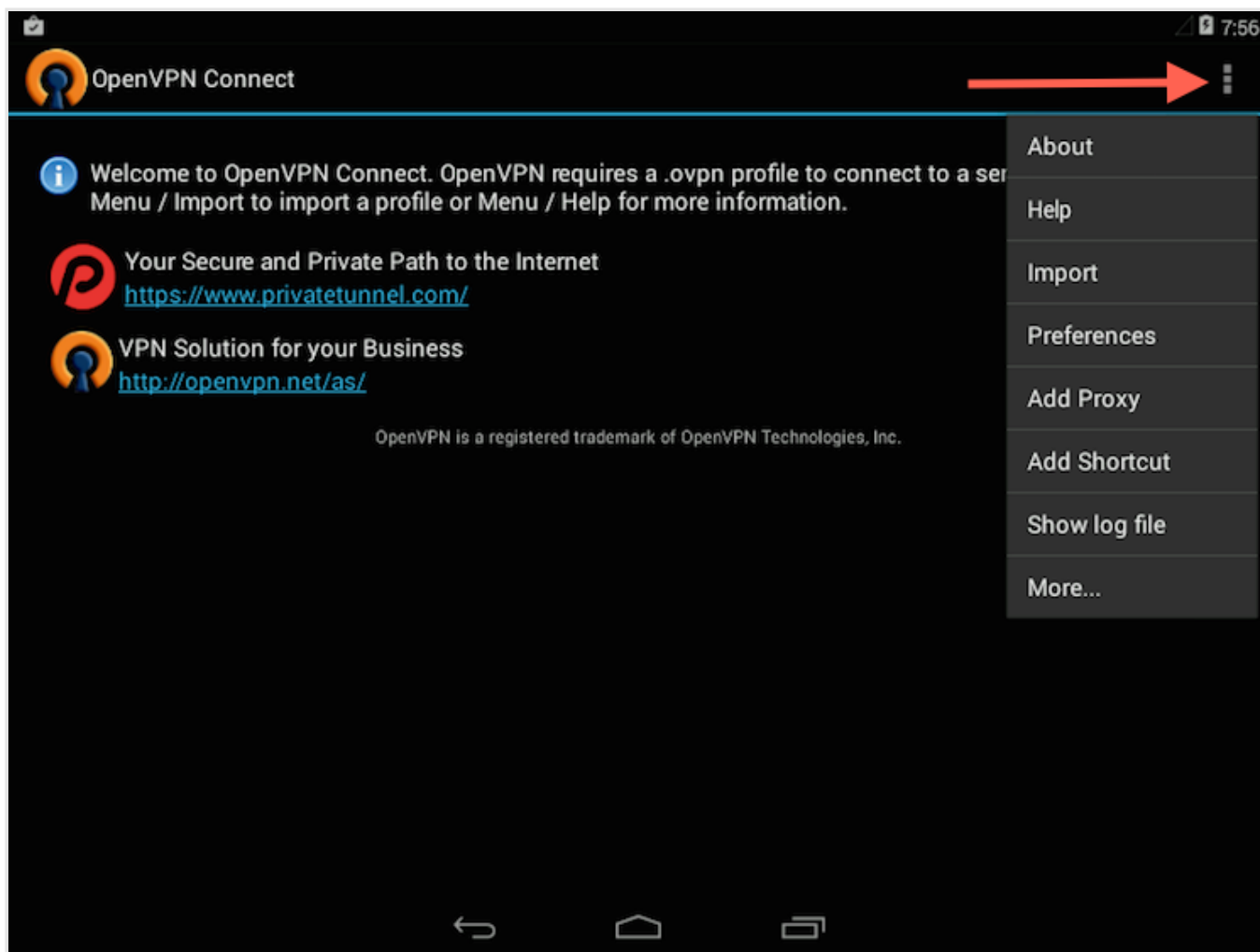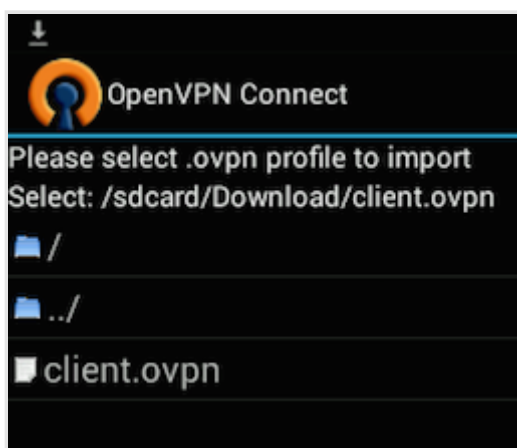
d device.

Start the OpenVPN app and tap the menu to import the profile.



Then navigate to the location of the saved profile (the screenshot uses `/sdcard/Download/`) and select the file. The app will make a note that the profile was imported.
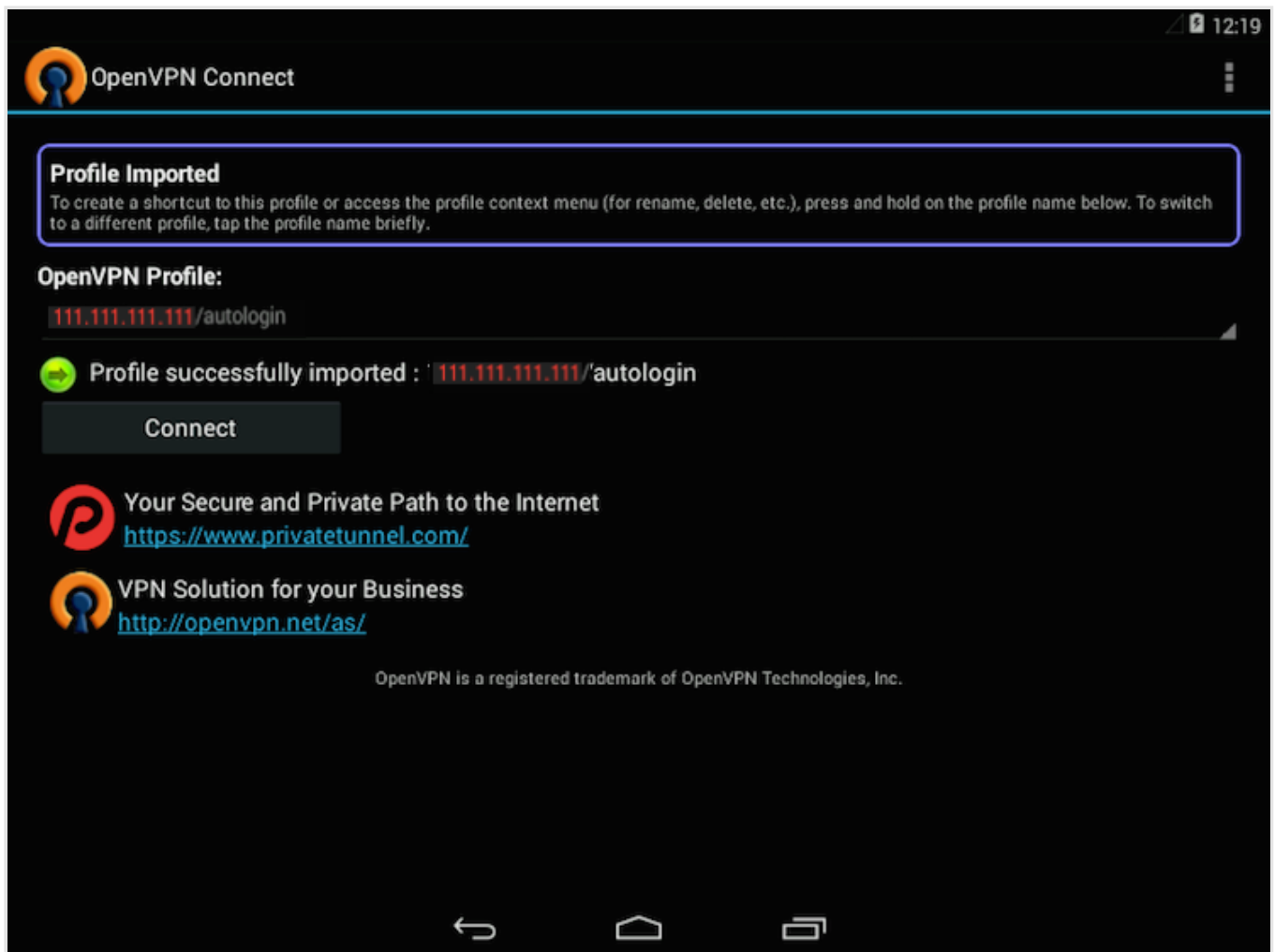


### Connecting

To connect, simply tap the **Connect** button. You'll be asked if you trust the OpenVPN application. Choose **OK** to initiate the connection. To disconnect from the VPN, go back to the OpenVPN app

## Step 13: Test Your VPN Connection

Once everything is installed, a simple check confirms everything is working properly. Without having a VPN connection enabled, open a browser and go to DNSLeakTest.

The site will return the IP address assigned by your internet service provider and as you appear to the rest of the world. To check your DNS settings through the same website, click on **Extended Test** and it will tell you which DNS servers you are using.

Now connect the OpenVPN client to your Droplet's VPN and refresh the browser. The completely different IP address of your VPN server should now appear. That is now how you appear to the world. Again, DNSLeakTest's **Extended Test** will check your DNS settings and confirm you are now using the DNS resolvers pushed by your VPN.

## Step 14: Revoking Client Certificates

Occasionally, you may need to revoke a client certificate to prevent further access to the OpenVPN server.

```
$ cd ~/openvpn-ca
$ source vars
```

Next, call the `revoke-full` command using the client name that you wish to revoke:

```
$ ./revoke-full client3
```

This will show some output, ending in `error 23`. This is normal and the process should have successfully generated the necessary revocation information, which is stored in a file called `crl.pem` within the `keys` subdirectory.

Transfer this file to the `/etc/openvpn` configuration directory:

```
$ sudo cp ~/openvpn-ca/keys/crl.pem /etc/openvpn
```

Next, open the OpenVPN server configuration file:

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

At the bottom of the file, add the `crl-verify` option, so that the OpenVPN server checks the certificate revocation list that we've created each time a connection attempt is made:

/etc/openvpn/server.conf

```
crl-verify crl.pem
```

Save and close the file.

Finally, restart OpenVPN to implement the certificate revocation:

```
$ sudo systemctl restart openvpn@server
```

The client should now longer be able to successfully connect to the server using the old credential.

To revoke additional clients, follow this process:

1.  Generate a new certificate revocation list by sourcing the `vars` file in the `~/openvpn-ca` directory

3. Restart the OpenVPN service.

This process can be used to revoke any certificates that you've previously issued for your server.

# Conclusion

Congratulations! You are now securely traversing the internet protecting your identity, location, and traffic from snoopers and censors.

To configure more clients, you only need to follow steps **6**, and **11-13** for each additional device. To revoke access to clients, follow step **14**.

By: Justin Ellingwood

♡ Upvote (197)          ⊡ Subscribe

---

## Spin up an SSD cloud server in under a minute.

Simple setup. Full root access.
Straightforward pricing.

**DEPLOY SERVER**

---

## Related Tutorials

How To Create a Point-To-Point VPN with WireGuard on Ubuntu 16.04

How To Run a Secure MongoDB Server with OpenVPN and Docker on Ubuntu 16.04

How to Set Up an IKEv2 VPN Server with StrongSwan on Ubuntu 16.04

Technical Recommendations and Best Practices for DigitalOcean's Tutorials

How To Encrypt Traffic to Redis with PeerVPN on Ubuntu 16.04

# 343 Comments

Leave a comment...

Log In to Comment

gammx1  *May 4, 2016*

20 Thanks!

BTW, if not working for some one this script may do the trick: https://github.com/Nyr/openvpn-install

Thor187  *August 6, 2016*

1 Hey, how secure is this?

How do we know it's not adding some funny logging/calling home setup in the background?

Sorry about the noob question.

skobkinru  *August 8, 2016*

1 You can view source of this script obviously.

ckoehncke  *January 22, 2017*

0 Thanks, your open-vpn installed worked flawlessly on Digital Ocean. I couldn't get the step-by-step running (my ip route | grep default kept reporting eth0 as my public interface - think that was a problem). In any event, OpenVPN on PC connected no worries.

0  yeah thanks the script works perfect for me too. I've tipped you also. I was not able to get things working by following the guide here (to the letter!), so you've been really helpful :)

---

[Metonymy](#)  *March 14, 2017*

0  I've followed this tutorial step by step and managed to install and connect to the VPN, however after I was connected I couldn't do anything except ssh to server ip. After using the script you provided everything works okay, thanks :)

(Also I'm writing this from the server)

---

[bri.amesse](#)  *March 24, 2017*

1  Thank god for this script... after doing this 100 times over and over again and it not working, tried your script worked right away lol my issue was : Mar 24 15:24:15 bradVPN ovpn-server[2236]: 152.233.21.22:13901 TLS Error: reading acknowledgement record from packet

---

[b123](#)  *March 31, 2017*

0  wow, I spent hours on above tutorial, not sure where I went wrong but it didn't work for me at step 11 (error: line 21: -----END: command not found ./make_config.sh: line 36: /dev/fd/63: Permission denied, tried sudo, same)

This script worked perfectly!! All these in less than a minute! Thank you so much!!

---

[docean887b3a3d884bad3e08d1](#)  *May 30, 2017*

0  After I started a new droplet, this worked flawlessly. I tried for three frikkin days...
Forget RTFM. **RTFC**

---

[gari](#)  *June 27, 2017*

0  There's something wrong with this tutorial somewhere. Been trying to setup using this tutorial to the letter but after countless retries, I finally gave up and used the script above. Took only seconds. Thank you!

---

[pazuso](#)  *August 2, 2017*

0  Worked PERFECTLY. After installing it, just tweak ufw, etc.

---

rteribe  *January 12, 2018*

0 how can I run the script on the server?

---

vineshkumar  *February 14, 2018*

0 hi

I run an OpenVpn service on my Ubuntu VPS by following this article.
service is activate successfully. but I can't connect to that by my client Debian machine.
the connection failed error is here:

ERROR: Private key password verification failed.
Exiting due to fatal error.

justinm87  *March 26, 2018*

0 This ran just fine. My question is, how do you then generate the file for iphone?

---

justinm87  *March 26, 2018*

0 I'm talking about the ovpn file. The VPN is running, but now how to do the ovpn generation?

---

rasmusj78  *May 10, 2016*

0 Works great, thanks !!

But if I want more than one client to connect, how do I create login for another one?

jellingwood  **MOD**  *May 11, 2016*

3 @rasmusj78: You can repeat step 6 and 11-13 for any additional clients.

rasmusj78  *May 11, 2016*

0 Thanks :-)

adilh  *March 27, 2017*

0 At first shot, i got my openvpn server running perfrectly but now i am trying to add another

again. Probably its a simple mistake that i repeat, what can it be?
Thank you.

**GomiGuchi**  *May 11, 2016*

2  Thanks for making an updated guide for the new LTS release!!

If anyone else is having trouble with iOS setup using OpenVPN Connect, you need to remove all of the [inline] options (ca, cert, key, tls-auth) and just leave the embedded certificates there. I chose to just comment the lines out in the config, but you can delete them as well.

**njwfish**  *November 21, 2016*

1  I think I'm having this issue, iOS devices "connect" to the VPN, but the IP addresses are not changed.

I can connect successfully on my laptop, but not on iOS. I tried to delete the inline options, but that seems to have broken the ovpn file. Any help would be awesome.

**njwfish**  *November 21, 2016*

2  Ok, I've fixed it. It was a different issue, I needed to route all client traffic through the VPN. Simple enough to add:

push "redirect-gateway def1"

To the server.conf.

**dkrachtus**  *May 12, 2016*

0  You could also use this, deploy an OpenVPN Endpoint with a single command.
https://github.com/ttlequals0/autovpn

**justincf**  *May 12, 2016*

0  If you want a GUI -- Pritunl is great also, as is OpenVPN Access Server.

**alimakki**  *May 12, 2016*

1  OpenVPN still defaults to BF-CBC which hasn't been broken yet but as a good practice it would be best to use AES-128-CBC; you will at least get an advantage since many CPUs hardware optimize AES, not to mention it's got more eyeballs looking at it and is 'battle hardened'.

Using the additional directive "auth SHA256" will use SHA2 instead of SHA1 for message authentication; the latter is now considered cryptographically broken.

More hardening tips: https://blog.g3rt.nl/openvpn-security-tips.html

---

⌃
♡     gertvdijkdigita  *August 29, 2016*

0   Thanks for linking my article. :)

By the way,

> OpenVPN still defaults to BF-CBC which hasn't been broken yet

BF-CBC (Blowfish) is broken too now, because Blowfish, amongst other 64-bit block ciphers, is affected by the SWEET32 attack. https://tech.slashdot.org/story/16/08/27/186213/new-sweet32-crypto-attacks-speed-up-deprecation-of-3des-blowfish

The reason for OpenVPN still using BF-CBC as default cipher (and SHA-1 for message authentication) is the compatibility with older OpenVPN software. Fortunately OpenVPN protocol will support cipher/auth negotiation which allows this while maintaining support with older releases. https://community.openvpn.net/openvpn/wiki/StatusOfOpenvpn24

---

⌃
♡     cyptex01  *May 14, 2016*

0   Great tutorial.....on my Androids I am getting "Error reading multiple files referenced by profile: [inline], [inline], [inline], [inline]"....how can I correct that?

---

⌃
♡     jellingwood   MOD    *May 16, 2016*

0   @cyptex01: Yeah, some clients don't like the `[inline]` syntax. I've remove them since most implementations don't really seem to need them. Thanks for the heads up!

---

⌃
♡     mrmjsm  *May 15, 2016*

0   is there a solution if the net provider here prevents UDP connection ?

---

⌃
♡     InspireNL  *May 16, 2016*

0   I've strictly followed this guide on a newly crested Ubuntu 16.04 system, altough I had to leave out the [inline] directives (I commented out those lines) to be able to get the ovpn profile working on iOS. I can now succesfully set up a connection from an iPad to the server, however once OpenVPN is connected, I can't actually browse the web. The syslog en ufw.log files don't seem to report anything unexpected. How can I best approach debugging this issue?

Sign up for our newsletter. Get the latest tutorials on SysAdmin and open source topics.     ✕

Enter your email address                                          Sign Up

InspireNL  *May 16, 2016*

0  I've solved this by using ens3 instead of eth0 in the before.rules config.

---

jellingwood  **MOD**    *May 16, 2016*

0  @InspireNL: Glad you figured it out! I've updated the tutorial to include finding the server's correct public interface so that people hopefully won't run into that next time. Thanks for sharing the solution.

---

Vayku  *May 20, 2016*

1  Thanks!

Can you add a Username & Password to the openvpn config? So that not everyone can logon who has the config

---

nekkidtruth  *May 23, 2016*

0  Great tutorial. However, I have a couple of suggestions that I think are very important. First, you should note that if you use a name other than "server", you will have to change information in the server's config file in order to get up and running by pointing to the correct .key and .crt files. OpenVPN will fail to start if you do not do so.

The other suggestion is related to the order in which you give instructions when setting up routing. You should be getting the interface name before even going into the ufw rules. It doesn't really make sense to open the file, edit it, exit and save the file, use the command to find interface name, go back into file, edit it and then save it again. Interface name > Edit File > Save file and done.

Thanks for taking the time to create this tutorial!!!

---

gecko40  *May 31, 2016*

1  I'm kind of confused here. Tried to setup OpenVPN on an Ubuntu server 16.04 using this guide except I used 443/tcp for the vpn port. I got the .ovpn file on my phone and it connects fine but does not load any web pages. Help is much appreciated.

---

user47583  *June 3, 2016*

1  I'm having the same problem as gecko40. Followed the instructions, but specified tcp on the default port instead of udp. I can connect fine, just can't view any web pages. Any ideas?

---

Sign up for our newsletter. Get the latest tutorials on SysAdmin and open source topics.    ✕

Enter your email address                                    Sign Up

Check forwarding rules by UFW and sysctl. And of course check logs.

---

⌃ jwallace  *June 8, 2016*

♡

2  On OS X with an iOS device you can also AirDrop the `client1.ovpn` file and open it in OpenVPN Client on the phone/tablet. A little easier than iTunes transfer

---

⌃ zenofob  *June 9, 2016*

♡

1  For those experiencing dnsleaks, if found appending these three lines into the .ovpn file fixed it:
script-security 2
up /etc/openvpn/update-resolv-conf
down /etc/openvpn/update-resolv-conf

---

⌃ MrBuggySan  *August 4, 2016*

♡

0  Thank you for your comment, I had some DNS leaks and this was the exact solution to my problems

---

⌃ willieaames  *June 10, 2016*

♡

0  I am in China at the moment. It's working mostly as expected: I can access youtube, facebook, etc. If I go to Wolfram Alpha and ask "where am I" it would give me the VPN server location, as expected. But if I go to skype.com, I'd get forwarded to their Chinese site. I am on a Mac and I have tried several browsers. They all forwarded me to the Chinese site. How does skype know that I am in China? I thought all traffic in VPN goes through the VPN server.
Using Pure OpenVPN: http://www.bestvpnprovider.com/china-vpn/

---

⌃ ipeacocks  *January 3, 2017*

♡

0  Maybe it redirects you based on info from your account? Try to create new one.

---

⌃ vmalep  *June 11, 2016*

♡

0  This is a very nice howto and I could configure the server and client without many trouble. However, I do not want the client traffics be redirected through the VPN, but only the client to be able to connect to each other through the VPN, while accessing internet directly from their respective connections.

I commented back the "push" instructions and uncommented the "client-to-client" one, but to no avail...

⌃ ipeacocks  *January 17, 2017*
♡
0  Comment:

push "redirect-gateway def1 bypass-dhcp"

But even w/o it some GUI clients can add vpn-route as default one.

⌃ jul514  *June 11, 2016*
♡
1  I have TLS problem when I try to connect Windows 10 client to my ubuntu openvpn server.

See my problem there: http://askubuntu.com/questions/785537/openvpn-tls-handshake-failed-with-linux-server-windows-client

Can someone help me?

⌃ Vedeyn  *June 14, 2016*
♡
0  Thank you for the incredible guide! It was easy to get my OpenVPN server up and running.

Quick question.. Is there a configuration somewhere to have OpenVPN prevent clients from accessing the local network? I would like to forward internet traffic only.

Load More Comments

Sign up for our newsletter. Get the latest tutorials on SysAdmin and open source topics.           ✕

Enter your email address                                 Sign Up

Copyright © 2018 DigitalOcean™ Inc.

Community　Tutorials　Questions　Projects　Tags　Newsletter　RSS 🔊

Distros & One-Click Apps　Terms, Privacy, & Copyright　Security　Report a Bug　Write for DOnations　Shop

Sign up for our newsletter. Get the latest tutorials on SysAdmin and open source topics.　✕

Enter your email address　　　　　　　　　　　　　　　Sign Up