How to Setup and Configure an OpenVPN Server on CentOS 6

Authored by: ***ASPHostServer Administrator*** *[asphostserver@gmail.com]*
Saved From: http://faq.asphosthelpdesk.com/article.php?id=234

This article will guide you through the setup and configuration of OpenVPN server on your CentOS 6 cloud server. We will also cover how to configure your Windows, OS X, or Linux client to connect to your newly installed OpenVPN server.

Before we begin, you'll need to have the Extra Packages for Enterprise Linux (EPEL) Repository enabled on your cloud server. This is a third party repository offered by the Fedora Project which will provide the OpenVPN package.

```
wget http://dl.fedoraproject.org/pub/epel/6/i386/epel-release-6-8.noarch.rpm
rpm -Uvh epel-release-6-8.noarch.rpm
```

Initial OpenVPN Configuration

First, install the OpenVPN package from EPEL:

```
yum install openvpn -y
```

OpenVPN ships with only a sample configuration, so we will copy the configuration file to its destination:

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

Now that we have the file in the proper location, open it for editing:

```
nano -w /etc/openvpn/server.conf
```

Our first change will be to uncomment the "push" parameter which causes traffic on our client systems to be routed through OpenVPN.

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

We'll also want to change the section that immediately follows route DNS queries to Google's Public DNS servers.

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

In addition, to enhance security, make sure OpenVPN drops privileges after startup. Uncomment the relevant "user" and "group" lines.

```
user nobody
```

```
group nobody
```

Generating Keys and Certificates Using easy-rsa

Now that we've finished modifying the configuration file, we'll generate the required keys and certificates. As with the configuration file, OpenVPN places the required scripts in the documentation folder by default. Create the required folder and copy the files over.

```
mkdir -p /etc/openvpn/easy-rsa/keys
cp -rf /usr/share/openvpn/easy-rsa/2.0/* /etc/openvpn/easy-rsa
```

With the files in the desired location, we'll edit the "vars" file which provides the easy-rsa scripts with required information.

```
nano -w /etc/openvpn/easy-rsa/vars
```

We're looking to modify the "KEY_" variables, located at the bottom of the file. The variable names are fairly descriptive and should be filled out with the applicable information.

Once completed, the bottom of your "vars" file should appear similar to the following:

```
export KEY_COUNTRY="US"
export KEY_PROVINCE="NY"
export KEY_CITY="New York"
export KEY_ORG="Organization Name"
export KEY_EMAIL="administrator@example.com"
export KEY_CN=server.example.com
export KEY_NAME=server
export KEY_OU=server
```

OpenVPN might fail to properly detect the OpenSSL version on CentOS 6. As a precaution, manually copy the required OpenSSL configuration file.

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

We'll now change into our working directory and build our Certificate Authority, or CA, based on the information provided above.

```
cd /etc/openvpn/easy-rsa
source ./vars
./clean-all
./build-ca
```

Now that we have our CA, we'll create our certificate for the OpenVPN server. When asked by build-key-server, answer yes to commit.

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

We're also going to need to generate our Diffie Hellman key exchange files using the build-dh script and copy all of our files into /etc/openvpn as follows:

```
./build-dh
cd /etc/openvpn/easy-rsa/keys
cp dh1024.pem ca.crt server.crt server.key /etc/openvpn
```

In order to allow clients to authenticate, we'll need to create client certificates. You can repeat this as necessary to generate a unique certificate and key for each client or device. If you plan to have more than a couple certificate pairs be sure to use descriptive filenames.

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

Routing Configuration and Starting OpenVPN Server

Create an iptables rule to allow proper routing of our VPN subnet.

```
iptables -t nat -A POSTROUTING -s 10.8.0.0/24 -o eth0 -j MASQUERADE
service iptables save
```

Then, enable IP Forwarding in sysctl:

```
nano -w /etc/sysctl.conf
# Controls IP packet forwarding
net.ipv4.ip_forward = 1
```

Finally, apply our new sysctl settings. Start the server and assure that it starts automatically on boot:

```
sysctl -p
service openvpn start
chkconfig openvpn on
```

You now have a working OpenVPN server. In the following steps, we'll discuss how to properly configure your client.

Configuring OpenVPN Client

Now that your OpenVPN server is online, lets configure your client to connect. The steps are largely the same regardless of what operating system you have.

In order to proceed, we will need to retrieve the ca.crt, client.crt and client.key files from the remote server.

Simply use your favorite SFTP/SCP (Secure File Transfer Protocol/Secure Copy) client and move them to a local directory. You can alternatively open the files in nano and copy the contents to local files manually. Be aware that the client.crt and client.key files will are automatically named based on the parameters used with "./build-key" earlier. All of the necessary files are located in /etc/openvpn/easy-rsa/keys

```
nano -w /etc/openvpn/easy-rsa/keys/ca.crt
nano -w /etc/openvpn/easy-rsa/keys/client.crt
nano -w /etc/openvpn/easy-rsa/keys/client.key
```

With our certificates now on our client system, we'll create another new file called client.ovpn, where "client" should match the name of the client being deployed (from build-key), the contents should be as follows, substituting "x.x.x.x" with your cloud servers IP address, and with the appropriate files pasted into the designated areas. Include only the contents starting from the "BEGIN" header line, to the "END" line, as demonstrated below. Be sure to keep these files as confidential as you would any authentication token.

```
client
dev tun
proto udp
remote x.x.x.x 1194
resolv-retry infinite
nobind
persist-key
persist-tun
comp-lzo
verb 3
<ca>
Contents of ca.crt
</ca>
<cert>
Contents of client.crt
</cert>
<key>
Contents of client.key
</key>
```

As all of the required information to establish a connection is now centralized in the .ovpn file, we can now deploy it on our client system. On Windows, regardless of edition, you will need the official OpenVPN Community Edition binaries which come prepackaged with a GUI. The only step required post-installation is to 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.

On Mac OS X, the open source application "Tunnelblick" provides an interface similar to OpenVPN GUI on Windows, and comes prepackagd with OpenVPN and 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.

On Linux, you should install OpenVPN from your distributions official repositories. You can then invoke OpenVPN by simply executing:

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

Congratulations! If you made it this far you should now have a fully operational VPN running on your cloud server. You can verify that your traffic is being routed through the VPN by checking Google to reveal your public IP.
