OpenVPN install

Article URL
OpenVPN install

Author
SecurityHome.eu

Published: 26 November 2020

Last updated on 26 September 2023.

Â

Setting OpenVPN

There are few steps to take care off:

* Authority Server
* Server configuration
* Client
* Configure forwarding

Based on the tutorial from DigitalOcean.com

Using OpenVPN version 2.4.9 and EasyRSA version 3.0.8.

Authority Server

To sign your certificates you need a certificate authority (CA). You should set it up on a seperated secure

server.
For a home setup, you can do it on the same server as your openVPN. But for a company, separate them.

It is relative easy to setup;
Download Easy-RSA

.
On your CA machine, navigate to the EasyRSA directory:
cd EasyRSA-3.x.x/

Inside this directory is a file named vars.example. Make a copy of this file, and name the copy vars without a file extension:
cp vars.example vars

Open this new file using your preferred text editor:
nano vars

Find the settings that set field defaults for new certificates. It will look something like this:

```
#set_var EASYRSA_REQ_COUNTRY    "US"
#set_var EASYRSA_REQ_PROVINCE   "California"
#set_var EASYRSA_REQ_CITY       "San Francisco"
#set_var EASYRSA_REQ_ORG        "Copyleft Certificate Co"
#set_var EASYRSA_REQ_EMAIL      "me@example.net"
#set_var EASYRSA_REQ_OU         "My Organizational Unit"
```

Uncomment these lines and update the values (do not leave them blank)

Within the EasyRSA directory is a script called easyrsa which is called to perform a variety of tasks involved with building and managing the CA. Run this script with the init-pki option to initiate the public key infrastructure on the CA server:
./easyrsa init-pki

After this, call the easyrsa script again, following it with the build-ca option. This will build the CA and create two important files (ca.crt and ca.key) which make up the public and private sides of an SSL certificate.

* ca.crt is the CA's public certificate file which, in the context of OpenVPN, the server and the client use to inform one another that they are part of the same web of trust and not someone performing a man-in-the-middle attack. For this reason, your server and all of your clients will need a copy of the ca.crt file.

* ca.key is the private key which the CA machine uses to sign keys and certificates for servers and clients. If an attacker gains access to your CA and, in turn, your ca.key file, they will be able to sign certificate requests and gain access to your VPN, impeding its security. This is why your ca.key file should only be on your CA machine and that, ideally, your CA machine should be kept offline when not signing certificate requests as an extra security measure.

If you don't want to be prompted for a password every time you interact with your CA, you can run the build-ca command with the nopass option, like this:
./easyrsa build-ca nopass

In the output, you'll be asked to confirm the common name for your CA: For simplicity, press ENTER to

accept the default name.

Signing certificates

Inside the EasyRSA-dir:
cd EasyRSA-3.x.x/
Using the easyrsa script again, import the server.req file, following the file path with its common name:
./easyrsa import-req /tmp/file.req name

Then sign the request by running the easyrsa script with the sign-req option, followed by the request type and the common name. The request type can either be *client*

or *server*

, so for the OpenVPN server's certificate request, be sure to use the server request type:
./easyrsa sign-req server name
or for a client:
./easyrsa sign-req client name

In the output, you'll be asked to verify that the request comes from a trusted source. Type yes then press ENTER to confirm this:

You are about to sign the following certificate.
Please check over the details shown below for accuracy. Note that this request
has not been cryptographically verified. Please be sure it came from a trusted
source or that you have verified the request checksum with the sender.

Request subject, to be signed as a server certificate for 3650 days:

subject=
    commonName   = name

Type the word 'yes' to continue, or any other input to abort.
  Confirm request details: yes

If you encrypted your CA key, you'll be prompted for your password at this point.

Next, transfer the signed certificate back to your VPN server using a secure method:
scp pki/issued/name.crt user@your_vpn_server:/tmp

Remember the certificate has a lifetime of 3 years. So you need to redo this step every 3 years.

Server configuration

Now that you have a CA ready to go, you can generate a private key and certificate request from your server and then transfer the request over to your CA to be signed, creating the required certificate.

If needed install EasyRSA on your OpenVPN-server.
cd EasyRSA-3.x.x/

From there, run the easyrsa script with the init-pki option. Although you already ran this command on the CA machine, it's necessary to run it here because your server and CA will have separate PKI directories:
./easyrsa init-pki

Then call the easyrsa script again, this time with the gen-req option followed by a common name for the machine. Again, this could be anything you like but it can be helpful to make it something descriptive. Throughout this tutorial, the OpenVPN server's common name will simply be "server". Be sure to include the nopass option as well. Failing to do so will password-protect the request file which could lead to permissions issues later on.

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

This will create a private key for the server and a certificate request file called *server.req*

. Copy the server key to the */etc/openvpn/*

directory:

```
sudo cp EasyRSA-3.0.4/pki/private/server.key /etc/openvpn/
```

The req-file needs to be send to be send to the secure CA-server, to be signed, and the you get a certificate back.

```
scp EasyRSA-3.0.4/pki/reqs/server.req user@CA_serve:/tmp
```

And do the signed on the CA-server (see above)

When the crt-file is back on the server (/tmp/server.crt), copy it to openVPN-dir:

```
sudo mv /tmp/{server.crt,ca.crt} /etc/openvpn/
```

Next create a strong Diffie-Hellman key:

```
cd EasyRSA-3.x.x/
./easyrsa gen-dh
```

This takes a few minutes. Once it finished, generate an HMAC signature to strengthen the server's TLS integrity verification capabilities:

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

When the command finishes, copy the two new files to your /etc/openvpn/ directory:

```
sudo cp ~/EasyRSA-3.0.4/ta.key /etc/openvpn/
sudo cp ~/EasyRSA-3.0.4/pki/dh.pem /etc/openvpn/
```

With that, all the certificate and key files needed by your server have been generated.

Configure OpenVPN server

First step: copying a sample OpenVPN configuration file into the configuration directory:

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

Now edit the config file:

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

Find the HMAC section by looking for the tls-auth directive. This line should already be uncommented, but if isn't then remove the ";" to uncomment it:

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

Find the section on cryptographic ciphers by looking for the commented out cipher lines. The AES-256-GCM cipher offers a good level of encryption and is well supported.

```
cipher AES-256-GCM
```

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

auth SHA256

Next, find the line containing a dh directive which defines the Diffie-Hellman parameters. Because of some recent changes made to EasyRSA, the filename for the Diffie-Hellman key may be different than what is listed in the example server configuration file. If necessary, change the file name listed here by removing the 2048 so it aligns with the key you generated in the previous step:
dh dh.pem

Finally, find the user and group settings and remove the ";" at the beginning of each to uncomment these lines:
user nobody
group nogroup

The basics are now setup. There are more setting that can be configured, for forwarding.

Start the server, add the configuration file :/etc/openvpn/server.conf, so add @server to end of your unit file when calling it:
sudo systemctl start openvpn@server

After starting the service, enable it so that it starts automatically at boot:
sudo systemctl enable openvpn@server

Client

Now we can create the corresponding certificates and keys which your client machine will use to access your OpenVPN server.
This is all done on your OpenVPN-server, only the config file (with certificates in it) is send to the client.
Create a directory to store the client certificate and key files:
mkdir -p /etc/openvpn/client-configs/keys
chmod -R 700 client-configs

Back to the EasyRSA directory on your OpenVPN-server:
cd EasyRSA-3.x.x/
./easyrsa gen-req client1 nopass
client1 is a common name for the client.

Press ENTER to confirm the common name. Then, copy the client1.key file to the *client-configs/keys/*

directory you created earlier:
cp pki/private/client1.key /etc/openvpn/client-configs/keys/

Next, transfer the client1.req file to your CA machine using a secure method and let it be signed (See *Signing certificates*

above):
scp pki/reqs/client1.req user@your_vpn_server:/tmp

Move the file from the CA-server to the directory
mv /tmp/client1.crt /etc/openvpn/client-configs/keys/

Next, copy the ca.crt and ta.key files to the client-configs/keys/ directory as well
cp /etc/openvpn/EasyRSA-3.x.x/ta.key /etc/openvpn/client-configs/keys/
sudo cp /etc/openvpn/ca.crt /etc/openvpn/client-configs/keys/

Now we can setup the client configuration-file.
mkdir -p /etc/openvpn/client-configs/files

cp /usr/share/doc/openvpn/examples/sample-config-files/client.conf /etc/openvpn/client-configs/base.conf
nano /etc/openvpn/client-configs/base.conf

Inside, locate the remote directive. This points the client to your OpenVPN server address - the public IP address of your OpenVPN server. If you decided to change the port that the OpenVPN server is listening on, you will also need to change 1194 to the port you selected:

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

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

```
proto udp
```

uncomment the user and group directives by removing the ";" at the beginning of each line:

```
# 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 you will add the certs and keys within the file itself shortly:

```
# 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
```

Similarly, comment out the tls-auth directive, as you will add ta.key directly into the client configuration file:

```
# If a tls-auth key is used on the server
# then every client must also have the key.
#tls-auth ta.key 1
```

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

```
cipher AES-256-GCM
auth SHA256
```

Next, add the key-direction directive somewhere in the file. You must set this to "1" for the VPN to function correctly on the client machine:

```
key-direction 1
```

Next you need to add the keys and the certificates in the file. Some need to be base64 encoded.

A good way is to use a script to create standard client config files.

```
nano /etc/openvpn/client-configs/make_config.sh
```

Add the code:

```
#!/bin/bash

# First argument: Client identifier

KEY_DIR=/etc/openvpn/client-configs/keys
OUTPUT_DIR=/etc/openvpn/client-configs/files
BASE_CONFIG=/etc/openvpn/client-configs/base.conf

cat ${BASE_CONFIG}  echo -e "
```