eWON with Self-Hosted OpenVPN Server Setup

HMS Networks

November 22, 2019

This Application Note describing the steps required to configure the Flexy VPN Client with a self-hosted OpenVPN server. Limited testing has been performed on this solution. It is recommended customers develop a test system and verify every Flexy feature required for the customer specific application is tested prior to deploying. Anyone deploying this solution should have prior experience with OpenVPN and Linux networking as it is not officially supported at this time.

The following software packages are used, and version changes may impact the results. An effort should be made to use the identical software packages.

OpenSSL	Version 1.1.1
Ubuntu Server	Version 18.04
EasyRSA	version 3.0.4
Flexy Firmware	Version 13.2s1
OpenVPN (server)	Version 2.4.8

References

This Application Note leverages the Digital Ocean tutorial by Mark Drake, "*How To Set Up an OpenVPN Server on Ubuntu 18.04*". Points where the Flexy configuration differs are described below.

https://www.digitalocean.com/community/tutorials/how-to-set-up-an-openvpn-server-on-ubuntu-18-04

Step 0: Test Environment Setup

It is strongly recommended to use a separate CA server from your OpenVPN server, so we will be using two different Ubuntu 18.04 servers for this example. (To better distinguish commands for each server, the OpenVPN Server commands have been colored BLUE and the CA Server commands have been colored ORANGE.) A production system may want to use a corporate CA or 3rd party CA. If a CA other than the one defined in this document is used it is critical to use the CA configuration parameters as defined. Failure to do so will result in a certificate that is not usable in the Flexy system.

Step 1: Installing OpenVPN and Easy RSA

This section describes how to create both server and client keys and requests. A user generated CA will be used to sign the certification requests.

Execute the steps show in the Digital Ocean OpenVPN Setup Step 1 to install the EasyRSA and OpenVPN software packages on the VM.

$ sudo apt update

$ sudo apt install openvpn

$ wget -P ~/ https://github.com/OpenVPN/easy-rsa/releases/download/v3.0.4/EasyRSA-3.0.4.tgz

$ cd ~

$ tar xvf EasyRSA-3.0.4.tgz

If for development purposes the CA exists on the same machine, ensure there is a separate EasyRSA install directory for the CA.

Step 2: Configuring EasyRSA and Building the CA

On your **CA machine**, also download EasyRSA and navigate to the directory:

$ wget -P ~/ https://github.com/OpenVPN/easy-rsa/releases/download/v3.0.4/EasyRSA-3.0.4.tgz

$ cd ~

$ tar xvf EasyRSA-3.0.4.tgz

$ cd ~/EasyRSA-3.0.4/

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 and uncomment the following lines and then enter your own information:

```
#set_var EASYRSA_REQ_COUNTRY    "US"
#set_var EASYRSA_REQ_PROVINCE   "New Hampshire"
#set_var EASYRSA_REQ_CITY       "Manchester"
#set_var EASYRSA_REQ_ORG        "My Certificate Org"
#set_var EASYRSA_REQ_EMAIL      "me@example.net"
#set_var EASYRSA_REQ_OU         "My Organizational Unit"
```

The following adjustments *must* be made to the vars file. The vars file must include the following lines uncommented.

set_var EASYRSA_KEY_SIZE 2048

set_var EASYRSA_NS_SUPPORT "yes"

set_var EASYRSA_NS_COMMENT "Easy-RSA Generated Certificate"

The EASYRSA_NS commands enable the certificate variable ns-cert-type. This variable is a legacy configuration variable that is embedded in the Flexy core configuration. It is used to fix the certificate usage to either a client only certificate or a server only certificate. There is no way to disable this feature in the Flexy firmware so it must be used. There is no security risk associated with using this legacy feature.

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 nopass** 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. Including the **nopass** option will ensure you will not be prompted for a password each time you interact with your CA.

$ ./easyrsa build-ca nopass

Step 3: Creating the Certificates

Now that you have a CA ready to go, you can generate a private key and certificate request from your OpenVPN server and then transfer the request over to your CA Server to be signed, creating the required certificate. (If executing this whole process on a single VM, the EasyRSA software must be installed in a separate directory from the CA EasyRSA install. Create a new directory for your key generator and copy the EasyRSA tar file to the new directory and extract the software.)

Copy the vars file from the CA to this install of EasyRSA to ensure the configuration parameters are correct. (Not 100% sure this is required; it is possible the vars file is not used when generating keys and certificate requests)

On the **OpenVPN Server** create a private key for the server and a certificate request file called `server.req`:

$ cd EasyRSA-3.0.4/

$ ./easyrsa init-pki

$ ./easyrsa gen-req server nopass

Copy the server key to the `/etc/openvpn/` directory:

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

Using a secure method (like SCP, in our example below), transfer the `server.req` file to your CA machine:

$ scp ~/EasyRSA-3.0.4/pki/reqs/server.req user@your_CA_ip:/tmp

Next, on your **CA machine**, navigate to the EasyRSA directory:

$ cd EasyRSA-3.0.4/

Import the request:

$ ./easyrsa import-req /tmp/server.req server

Sign the request with the `sign-req` option followed by the request type (client or server) and common name.

$ ./easyrsa sign-req server server

Next, transfer the signed certificate and ca.crt file back to your VPN server using a secure method:

$ scp pki/issued/server.crt user@your_server_ip:/tmp

$ scp pki/ca.crt user@your_server_ip:/tmp

Next, log back into your **OpenVPN** server and copy the `server.crt` and `ca.crt` files into your `/etc/openvpn/server/` directory:

$ sudo cp /tmp/{server.crt,ca.crt} /etc/openvpn/server/

Navigate to the EasyRSA directory and create a Diffie-Hellman key to use during key exchange:

$ ./easyrsa gen-dh

$ sudo cp ~/EasyRSA-3.0.4/pki/dh.pem /etc/openvpn/server/

Step 4: Generating a Client Certificate and Key Pair

The same EasyRSA install location used to generate the server request and key can be used to generate the client request and key. We will generate a single client key and certificate pair for client1.

Get started by creating a directory structure within your home directory to store the client certificate and key files and lock down permissions:

$ mkdir -p ~/client-configs/keys

$ chmod -R 700 ~/client-configs

Use EasyRSA to generate the client signing request on the OpenVPN server:

$ cd ~/EasyRSA-3.0.4/

$ ./easyrsa gen-req client1 nopass

$ cp pki/private/client1.key ~/client-configs/keys/

Next, transfer the `client1.req` file to your CA machine using a secure method:

$ scp pki/reqs/client1.req username@your_CA_ip:/tmp

Log in to your **CA machine**, navigate to the EasyRSA directory, and import the certificate request:

$ cd EasyRSA-3.0.4/

$ ./easyrsa import-req /tmp/client1.req client1

Sign the request and transfer the `client1.crt` back to the OpenVPN server:

$ ./easyrsa sign-req client client1

$ scp pki/issued/client1.crt username@your_server_ip:/tmp

Back on your **OpenVPN server**, copy `client1.crt` and `ca.crt` to the `/client-configs/keys/` directory:

$ sudo cp /tmp/{client1.crt,ca.crt} ~/client-configs/keys/

With that, your server and client's certificates and keys have all been generated and are stored in the appropriate directories on your server.

Step 5: Configuring the OpenVPN Service

The OpenVPN server is configured using the text file `/etc/openvpn/server.conf`. The conf files used during testing are included in appendix A of this document. The HMAC firewall feature was not enabled during testing. It is possible that an alternative set of options will work with this system, the following line is required:

ns-cert-type client

Step 6: Adjusting the Server Networking Configuration

This step was not executed during the development of this application note. This section address network address routing at the server level. Customers are left to configure this portion of the system to meet their specific system needs.

Step 7: Starting and Enabling the OpenVPN Service

Start the OpenVPN server by specifying your configuration file name as an instance variable after the systemd unit file name. The configuration file for your server is called `/etc/openvpn/server.conf`, so add @server to end of your unit file when calling it:

$ sudo systemctl start openvpn@server

You can check the status of the service by typing:

$ sudo systemctl status openvpn@server

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

$ ip addr show tun0

After starting the service, enable it so that it starts automatically at boot:

$ sudo systemctl enable openvpn@server

It is useful to verify the OpenVPN service is accessible from another linux system or VM on the network. The following command will return success if the server is visible on the network. Ensure the IP address and port are updated to match the system under test.

$ nc -vu 172.16.0.45 1194

It may be useful during debug to start and stop the VPN server. Starting or restarting the server will cause the OpenVPN server to reread the server.conf file. The commands to do so are as follows:

sudo systemctl status openvpn@servername

sudo systemctl stop openvpn@servername

sudo systemctl start openvpn@servername

sudo systemctl restart openvpn@servername

Step 8: Configuring Flexy as a generic OpenVPN Client

The GUI interface can be used to configure many of the OpenVPN Client's setting but not all. Most of the configuration will be done using configuration files.

In the Flexy GUI confirm the following settings:

1) Setup > System > Communications > Networking > VPN Connection > Main setup
    a. Ensure Establish outgoing VPN to server is selected
    b. No other options should be selected
2) Setup > System > Communications > Networking > VPN Connection > Global
    a. Internet connection proxy – No proxy
    b. Talk2M Account Name – Customers Account Name

    c. Talk2M Access Server – talk2m_pro
    d. Advanced settings Diagnosis level – High
    e. Advanced settings Port In – 0
    f. Advanced settings Port Out – This should be the port of the OpenVPN Server
    g. Advanced settings 'keep alive' interval – 120
    h. Advanced settings VPN Driver Mode – TUN
    i. Advanced settings VPN Protocol – UDP
3) Setup > System > Communications > Networking > VPN Connection > Incoming
    a. Leave as default
4) Setup > System > Communications > Networking > VPN Connection > Outgoing
    a. Establish VPN connection should be checked
    b. Primary server – IP address of OpenVPN Server
    c. Secondary server – IP address of backup OpenVPN Server
    d. Connect to…
        i. Select VPN Server in the drop down
        ii. The Private key, Certificate and CA certificate can be left unchanged or blank

The complete Flexy configuration is stored in a series of text files at the root of the Flexy device. We will create a separate config file, `client.ovpn`, for our VPN settings and link to it in the `comcfg.txt` file.

1) FTP into the Flexy
2) Copy the `/comcfg.txt` file to a host PC
3) Edit the file and add this line

    VPNCfgFile:+/usr/client.ovpn

4) Edit the client.ovpn file used in this app note (attached)
    a. Different cipher files can be used, for example, auth SHA1 or SHA256
    b. It is critical that the auth used matches the setting in server.conf
    c. ns-cert-type server is not included in the settings because it is automatically set by the Flexy firmware

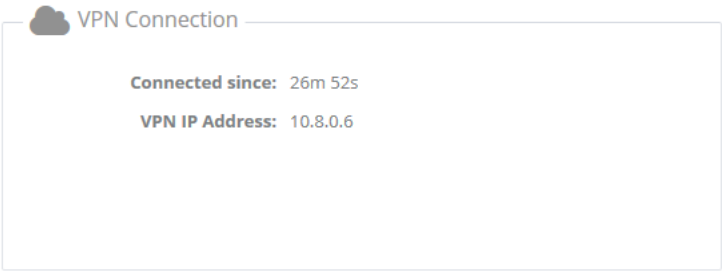
FTP the following files over to the Flexy /usr directory:

1) ca.crt
2) client.ovpn
3) client1.crt
4) client1.key.

Reboot the Flexy.

Verify the System is Working

If properly configured using the settings defined is this document the OpenVPN server will assign a IP address to Flexy in the 10.8.0.xxx range. In the Flexy GUI the top-level summary should show the following info in the VPN Connection section.



Useful tools

During the development of this application note the following commands and tools were used to debug the system. End users may find these tools useful in debugging the system.

1) Checking a Key, Certification pair. If the output of these two functions match the CRT and Key are a pair.
   $ openssl rsa -noout -modulus -in client1.key

   $ openssl x509 -noout -modulus -in client1.crt

2) Determining the version of ssl
   $ Openssl version -a

3) Command line reboot Linux box
   $ sudo shutdown -r now

Appendix A

server.conf:

```
port 1194
port udp
dev tun
ca /etc/openvpn/server/ca.crt
cert /etc/openvpn/server/server.crt
key /etc/openvpn/server/server.key
dh /etc/openvpn/server/dh.pem
server 10.8.0.0 255.255.255.0
ifconfig-pool-persist /var/log/openvpn/ipp.txt
keepalive 10 120
mode server
tls-server
cipher AES-256-CBC
auth SHA256
comp-lzo
user nobody
group nogroup
persist-key
persist-tun
status /var/log/openvpn/openvpn-status.log
log /var/log/openvpn/openvpn.log
verb 5
mute 25
explicit-exit-notify 1
ns-cert-type client
```

client.ovpn:

```
client
dev tun
proto udp
remote 10.10.135.80
port 1194
resolv-retry infinite
comp-lzo
nobind
persist-key
persist-tun
keepalive 15 180
tls-client
cipher AES-256-CBC
auth SHA256
ca /usr/ca.crt
cert /usr/client1.crt
key /usr/client1.key
```