# RouterTech Firmware OpenVPN "HowTo"

## What is VPN?

VPN is the abbreviation for "Virtual Private Network". With a VPN one can build a secure "private" network encapsulated within another ("public") network.

## What is OpenVPN?

OpenVPN is an open source VPN solution available for Linux and Windows.

## Do I need a VPN?

Normally a VPN consists of a VPN server and at least one VPN client. The server listens on a specific network port for VPN requests. A VPN client can then connect to this VPN server, thereby gaining full access to the server's network.

*Imagine the following scenario*:
At home you have a physical server and a NAS (Network Attached Storage device). Now you are somewhere else, and you need access to a file on your NAS. If there is a VPN running on this server you can connect via the Internet to this server and have full access to your NAS. This is a typical VPN scenario.

## Do I need OpenVPN on my router?

If you have a physical server on which you can install a VPN solution, then the answer is "No".

However, if you don't have a server, your router can be the VPN server. It is cheap (i.e., you already have the router) and it is normally running all day long.

*Example*:
On my network at home there are four computers and a NAS with a database (MySQL) running on it. When nobody is at home the computers are all switched off. The only machines running all day long are the router and the NAS. Now, wherever I may be,  I can take my laptop, connect it to the internet and then connect to my VPN at home. I have full access to my network at home, I can see all network shares and have access to my database. If in need I can start one of my machines via Wake On Lan and have access to this machine. It's like you are physically connected to your home's network even if you are far away from home.

## Why do I need to generate "certificates" and "keys"?

Without certificates and keys your VPN is insecure. Everybody can have access to your network, read, write and delete files and everything else. You can of course configure your VPN so that it uses simple username/password authentication. This is however not very secure.

For this reason it is recommended that you generate certificates and keys. If you do this, then only clients with a valid certificate will be able to connect to your network. All data will be encrypted with the keys, so that nobody can read your data stream while you are connected.

See http://openvpn.net/index.php/open-source/documentation/howto.html#pki for a discussion about what this certificate/key stuff is, and how to set it up.

## Why can't the certificates and the keys be downloaded from the RouterTech repository as with the openvpn binary?

The certificates and the keys are “private” files per user. If the RouterTech repository provides certificates and keys, this would be the same as having no certificates and keys (because everybody can download these files, with them, connect to your computer).

## How do I create certificate and key files?

The first thing to do is to generate a *server certificate* and a *private key* file. When you have a valid client installation (see the next question) there is a subdirectory in the OpenVPN installation named **easy-rsa**. To use it you need write permission on this directory.

The simplest way is to copy the whole easy-rsa directory into another directory with write permission.  Open a command prompt and change the directory to the newly created easy-rsa directory and run these batch/script files:

*Windows*:
1) **vars.bat**

2) **clean-all.bat**
This will erase all existing certificates, keys and pem files. Do **NOT** run this batch file if you only want to add clients (see step 6)

3) **build-ca.bat**
This will create a ca.crt file and a ca.key file. We only need ca.crt

4) **build-dh.bat**
This will create a dh1024.pem file

5) **build-key-server.bat <servername>**
This will create <servername>.crt,<servername>.csr and <servername>.key. We only need the <servername>.crt file and the <servername>.key file.

For the **router** (which will be the *server*) we need the **ca.crt** file from step 3, the **dh1024.pem** file from step 4, the **<servername>.crt** and **<servername>.key** files from step 5.

*Now we need to create the files for the client*:
6) **build-key <clientname>**
This will create a <clientname>.crt, <clientname>.csr and <clientname>.key file. Again we only need the <clientname>.crt and the <clientname>.key file.

For the **client**, we need the **ca.crt** file from step 3 and the **<clientname>.crt** and **<clientname>.key** files from step 6.

*Linux*:
1) ./**vars**

2) ./**clean-all**
This will erase all existing certificates, keys and pem files. Do **NOT** run this batch file if you only want to add clients (see step 6)

3) ./**build-ca**
This will create a ca.crt file and a ca.key file. We only need ca.crt

4) ./**build-dh**
This will create a dh1024.pem file

5) ./**build-key-server <servername>**
This will create <servername>.crt,<servername>.csr and <servername>.key. We only need the <servername>.crt file and the <servername>.key file.

For the **router** (which will be the *server*) we need the **ca.crt** file from step 3, the **dh1024.pem** file from step 4, the **<servername>.crt** and **<servername>.key** files from step 5.

*Now we need to create the files for the client*:
6) ./**build-key <clientname>**
This will create a <clientname>.crt, <clientname>.csr and <clientname>.key file. Again we only need the <clientname>.crt and the <clientname>.key file.

For the **client**, we need the **ca.crt** file from step 3 and the **<clientname>.crt** and **<clientname>.key** files from step 6.

For more "professional" instructions see http://openvpn.net/index.php/open-source/documentation/howto.html#pki

**NOTE**:
It is a good idea to keep these files (in fact the whole easy-rsa directory) because later - and this can be one minute, one day or one year later - we can add as many clients as we want by simply repeating step 6 for every client. No change to the server is needed, because we already have the ca.crt file, which is valid for 10 years. Whenever we need a new client, we simply call **vars.bat** (*Windows*) or **./vars** (*Linux*) and then **build-key.bat <clientname>** (*Windows*), or ./**build-key <clientname>** (*Linux*) and copy the generated files to the client.

## OK, I have the certificate and key files. How do I set up the OpenVPN server on my RouterTech enabled router?

At you need at least RouterTech Firmware **v2.94 or later** (with a wireless or "4mb-flash" firmware) to do this. The RouterTech firmware provides a script named **openvpn.sh** which accepts several parameters. Most of them have default values and can be set as firmware environment variables, or provided manually at the commandline to openvpn.sh.

For those with a **"1350A" wireless firmware**, because the 1350A wireless routers do not have sufficient space on their flash memory to store the OpenVPN binary, the binary must be stored somewhere else. Preferably, you can store it on a **samba**-compatible share on your network (mounted on an **smbfs** mountpoint on the router (e.g., **/sbmfs/**), or, it can be downloaded from the RouterTech repository into the router's **/var/** directory (this should be used as a last resort, so as not to overload the RouterTech server).

The memory overhead for running OpenVPN on a **1350A wireless** firmware is significant. The OpenVPN binary itself (1.1mb in size) has to reside on the router's *ram-disk*, thereby using at least 1.1mb extra memory, and it also needs memory to run. You may wish to ensure that you are not also running too many other services at the same time. With respect to the "**standard" wireless (1130)** and **4mb-flash non-wireless firmwares**, the OpenVPN binary is already in the firmware, and so this issue does not arise.

## A. STANDARD WIRELESS/NON-WIRELESS "4mb-flash" FIRMWARES:

Here, the binaries are already included in the firmware, so the instructions below ("*Downloading the binaries from the RouterTech repository"* and "*Copying the binaries from a samba share on your network"*) do NOT apply. For these firmwares, the main concern is the location of the certificate and key files (see **section C** on that below). If you **don't** have a 1350A wireless firmware, you can skip section B below.

## B. 1350A WIRELESS FIRMWARES:

Here, the binaries must be downloaded or copied from somewhere else:

*i. Downloading the binaries from the RouterTech repository* (**last resort option**):

To instruct the firmware to download the OpenVPN binaries from the repository, call the **openvpn.sh** script with the parameter **"–bin_dir=inet**" (or set the **openvpn_bin_dir** environment variable to "inet").

**Example1 (needs to be done every time)**:

```
openvpn.sh –bin_dir=inet .....
```

**Example2 (the "setenv" command only needs to be run once)**:

```
setenv "openvpn_bin_dir" "inet"
openvpn.sh ....
```

In either of these cases, the openvpn.sh will download the binaries from the RouterTech repository, extract the files, and then run the relevant binary.

*ii. Copying the binaries from a samba share on your network* (**preferred option**):

To copy the OpenVPN binaries from a **samba**-compatible share on your network, the "**--bin_dir"** parameter (or its corresponding environment variable "**openvpn_bin_dir**") must point to a network shared directory/folder where the binary can be copied from. This shared network directory must be mountable on the router via the **smbfs** feature, and must contain **only the OpenVPN binary and server files** that you want to use on the router (**ALL** the files in this directory will be copied to the router – and if there are extraneous files that will not all fit on the router's ram-disk, then your router **WILL** CRASH!!).

If neither **–bin_dir** nor **openvpn_bin_dir** is used, then the script will use a default of "**/smbfs/openvpn**".

Some other parameters may be required:

**--sharepoint** – this should point to the relevant network share (to be mounted automatically by the script)

**--user** - the network username that is required to mount your network share, and

**–password** - the network password required to connect to the network share.

**Example**:

```
openvpn.sh –sharepoint=192.168.1.45/vpn –user=Joe –password=Bloggs –bin_dir=/smbfs/Joe ....
```

In this case, the script will try to mount the sharepoint **192.168.1.45/vpn** to a mountpoint of **/smbfs/** on the router, using the supplied username and password. If successful, then it will copy the OpenVPN binary (and all other files) from **/smbfs/Joe/**, and then unmount the mountpoint. If you wish to mount the sharepoint to a different mountpoint on the router (i.e., different from **/smbfs**) then you need to specify that mountpoint with "**--smbfs**".

If you had already mounted the sharepoint to a sharepoint on the router yourself (e.g., to /smbfs), then all you need to do is to specify the location of the OpenVPN binaries on that mountpoint with **--bin_dir**. In this case, you do **not** use **–sharepoint**, or **--user**, or **–password**. So, for example, if you had already mounted the network directory containing these files to **/var/testing** on the router, you would just need to run:

```
openvpn.sh –bin_dir=/var/testing ....
```

In this case, the script will not try to mount (or unmount) anything, but will simply try to copy the files from **/var/testing**, and give up if it doesn't succeed.

## C. CERTIFICATE AND KEY FILES (all firmwares):

At the barest minimum, the **server** certificate and the key files created in steps 3 and 5 (above) must go somewhere on your **network**, and must be accessible. You can either store them on a network **samba** share, or on a **minix partition** on your router. The "**–cert_dir**" parameter (or the "**openvpn_cert_dir**" environment variable) should point to the location of these files on the **minix partition** or the connected **samba** share. If neither of these is used, then the script will default to looking for the certificate and key files in **/smbfs/openvpn/**. **All** the **.crt**, **.key**, **.ovpn**, and **.pem** files in this directory will be copied to the router. It is probably best to have them on a **minix** partition (created with "***makemtd.sh***").

### Example 1 (complex – 1350A wireless firmwares):

The network share is \\192.168.1.100\Router, the username is "routertech" and the password is "routertech". The OpenVPN binary is located in **/ovpn/bin** and the configuration files (including the required crt and key files) are located in **/ovpn/config**.

The **public crt** file is **ca.crt** (*same as default*), the **private crt and key** files are **MyServer.crt** (*default=server.crt*) and **MyServer.key** (*default=server.key*). The **port** to use is **443** (*default=1194*). The **proto**col to use is **tcp** (*default=udp*).

Then you can start openvpn.sh like this:

```
openvpn.sh --sharepoint \\192.168.42.100\Router --user=routertech --password=routertech
--bin_dir=/smbfs/ovpn/bin --cert_dir=/smbfs/ovpn/config --port=443 --key=./MyServer.key –
cert=./MyServer.crt --proto=tcp
```

### Example 2 (simple – 1350A wireless firmwares):

You wish to fetch the OpenVPN binaries from the RouterTech repository, and you have kept your server key and certificate files on a **minix** partition (mounted on **/nvram)**. You prefer to use the default port (*1194*), and the default names for the server key and certificate files (*server.key* and *server.crt*).

Then you can start **openvpn.sh** like this:

```
openvpn.sh --bin_dir=inet --cert_dir=/nvram
```

**Example 3 (simplest – standard wireless/non-wireless firmwares)**:

Here, the OpenVPN binary is already in the firmware on your router. You have kept your server key and certificate files on a **minix** partition (mounted on **/nvram)**. You prefer to use the default port (*1194*), and the default names for the server key and certificate files (*server.key* and *server.crt*).

Then you can start **openvpn.sh** like this:

```
openvpn.sh --cert_dir=/nvram
```

## How do I set up an OpenVPN client on my PC?

Simply download the OpenVPN client software from http://openvpn.net/ and install it. It is important that the client is the same version as the server on the router (currently v2.1.1). A graphical Windows client is available here: http://www.openvpn.net/index.php/open-source/downloads.html

**NOTE**: you need administrative privileges to install this Windows client.

You will need to create a **client.ovpn** file (there is a sample in the RouterTech firmware documentation). The client.ovpn is the configuration file for the client (on Windows, it must be in <OpenVpnInstallDir>\config together with the ca.crt, the <client>.crt and the <client>.key files). An easy way to run an OpenVPN client is to use the "**--config**" parameter to specify the client configuration file.

**Example**:

```
openvpn –config client.ovpn
```

The Windows GUI client will do this automatically if the client.ovpn file is in the client program's "config" directory. should look like this:

The client.ovpn configuration file should look like this:

```
# Sample OpenVPN client config
# General settings
client
port 1194
#proto tcp-client
proto udp
dev tap

# Client settings
tls-client
ns-cert-type server
remote <myhome.homedns.org> 1194

# Certificate and keys
ca ca.crt
cert <clientname>.crt
key <clientname>.key
auth SHA1
cipher BF-CBC
```

```
# Misc
pull
verb 3
persist-key
persist-tun
# -------------------- end ----------------
```

<u>**NOTES**</u>:
Every line beginning with a **#** is only a comment line, and is ignored by OpenVPN.

The following settings are important:

**port**
Normally OpenVPN listens at port 1194. On some networks this port is not open. So you have to ask the networks admin to open the port for you. But in most cases port 443 is open by default. But keep in mind that 443 is the https port and using this port can lead to other problems when using secured http connections on your client. On the router it shouldn't cause any problem.

**Note**: the OpenVPN port on the server and the client must match.

**remote <IP Address> <port>**
**remote <DNS Name> <port>**
The “remote” keyword in the configuration file specifies your client's DNS name or its IP address in the internet. It is important to understand that this is **NOT** the router's LAN IP Address (e.g., 192.168.1.1), but the IP address that was assigned to you by your Internet Service Provider (ISP). In some cases this is a permanent IP address but in most cases for the standard home internet connection it is a **dynamically assigned address**. In this case you can register to a web service like **DynDNS** (www.dyndns.com). Then you should set up your router under “Advanced”->”DDNS” that it connects itself to the DynDNS service. In your client's “remote” keyword then you have to specify the assigned DNS name, for example “**myhome.homedns.org**”. What now happens is: Every time your router gets a new IP address from your ISP, it connects to the DynDNS service and then provides its new IP address. When start OpenVPN on your client, it searches for the DNS name “**myhome.homedns.org**”. It gets its IP address and then connects to your router.

**ca**
Use the name of your ca.crt file here. Usually this is simply the **ca.crt** created in step 3 in “**How do I create certificate and key files?**”. We need a copy of this certificate in the OpenVPN's config directory.

**cert <clientname>.crt**
**key <clientname>.key**
These are the certificates and the keys for the client created in step 6 in “**How do I create certificate and key files?**”. Replace <clientname> with the real file names. Of course we need these files too and they should go into the OpenVPN's config directory.

**proto**
This identifies the protocol to use. The default is udp. You can use “**tcp-client**” here, if you also pass “**--proto=tcp”** to the OpenVPN server.

All other settings should be as provided and should only be changed when you know what you are doing. Especially don't miss the “**pull**” keyword which simply pulls the right commands to set up a route on your client so that it can use the VPN.

## How do I run OpenVPN on my RouterTech router?

1. The first point to note is that you need to have a place on your network to store certificate and keys files. To put it simply, you need either a **minix partition**, or a shared directory on your PC or NAS, that can be mounted via **smbfs**. The certificate and key files must be stored here. Without one of these, you will not be able to use OpenVPN on a RouterTech router.

2. After configuring the server and client sides, you need to run **openvpn.sh** – either **manually** (from a telnet or ssh login session – but *never* from the “Run Command” feature of the web interface), or **automatically** by setting the **openvpn** environment variable.

3. This is the (optional) parameter list of **openvpn.sh**:

**--sharepoint**=<smbfs sharepoint>

**--user**=<smbfs username>

**--password**=<smbfs password>

**--path**=<openvpn install path>  [*default = /var/tmp/openvpn*]

**--smbfs**=<smbfs mountpoint>     [*default = /smbfs*]

**--url**=<full URL for openvpn binary tarball> [*default = the RouterTech repository*]

**--cert_dir**=<location of openvpn private certificate and key files> [*default = the value of "***openvpn_cert_dir***" in the environment, or /smbfs/openvpn*]

**--bin_dir**=<location of openvpn binaries on a network (samba) share> [*default = the value of "***openvpn_bin_dir***" in the environment, or /smbfs/openvpn]*; supply "inet" for this or as the value of the "openvpn_bin_dir" to fetch the openvpn binaries from the internet (i.e., the *RouterTech* repository, or the value in "**--url**")

**--slog**=<name of file for startup logs> [*default = none*]

**--port**=<port number> [*default=1194*]

**--vnet**=<virtual net IP>  [*default=10.0.0.0*]

**--ca**=<root certificate file>  [*default=./ca.crt*]

**--cert**=<local certificate file> [*default=./server.crt*]

**--key**=<local private key> [*default=./server.key*]

**--dh**=<file containing *Diffie-Hellman* parameters> [*default=./dh1024.pem*]

**--proto**=<protocol> [*default=udp*]

You can run **openvpn.sh** [a] with *default parameters*, [b] with *parameters supplied manually* at the command line, or [c] with *parameters specified in the environment and config.xml.*

A. <u>*Using default parameters*</u>:
This is the easiest, if things are set up properly. The default parameters are listed above. To use the defaults, share a directory (your **sharepoint**) on your PC/NAS (to be mounted on **/smbfs**), create a subdirectory in it, called "**openvpn**", and put all your OpenVPN files in that subdirectory (the openvpn binary (if your have a 1350A wireless firmware), and the **dh, certificate** and **key** files). The server **certificate** and **key** files must be called **ca.crt**, **server.crt** and **server.key**. The **dh** file must be called **dh1024.pem**.

With these, you are set. You can simply run something like this:

```
openvpn.sh –sharepoint=192.168.1.45/vpn –user=Joe –password=Bloggs --proto=tcp
```

If you want this to run automatically every time the router boots up, you just need to run this:

```
setenv "openvpn.sh" "–sharepoint=192.168.1.45/vpn –user=Joe –password=Bloggs" --proto=tcp
```

That is all!

If you prefer to **mount your network share yourself** (mounted on **/smbfs**), then **after** the mounting, you just need to run this:

```
openvpn.sh
```

If you want this to run automatically every time the router boots up, you just need to run this:

```
setenv "openvpn" "1"
```

B. <u>*Parameters supplied manually at the command line*</u>:
This gives you all the freedom you need to configure, name, and place things exactly as you wish. However, you will need to run **openvpn.sh** manually every time, and give it a long list of parameters.

C. <u>P*arameters specified in the environment and config.xml*</u>:
This is similar to "**B**" above, but, instead of always having to type all the parameters manually (you can of course put them in a script of your own), you can store the parameters in the firmware. To do this, set the environment variable "**openvpn**" to "**--config**" (using *setenv*), then store all the required parameters in the "**VPN**" entry in the firmware's config.xml (using *setconf* – note that any entry in config.xml with *setconf* will be cleared every time the router is reset to defaults).

<u>Example</u>:

```
setenv "openvpn" "--config"
setconf "VPN" "--sharepoint=//192.168.1.9/vpn --user=Joe --password=@2K --bin_dir=/smbfs/
--cert_dir=/nvram/ --key=./n1.key –cert=./n1.crt"
```

Reboot the router, and the firmware will run OpenVPN with those parameters every time the router boots up.

Here, we have kept the OpenVPN binaries at the root of the sharepoint **//192.168.1.9/vpn** (mounted on **/smbfs**), and we have kept the certificate files on a **minix** partition (mounted on **/nvram**). The server key and certificate files are **n1.key** and **n1.crt** respectively.

**NOTE**:
You only need to do this once (unless the firmware is reset to defaults, in which case you will need to do it again).

**Authors**:

- Stefan Fröhlich
- TheChief

2 January 2011