As of DD-WRT v.24 SP1, it is now possible to set up DD-WRT as an OpenVPN appliance using only the web-based GUI. It is no longer necessary to issue shell commands, or to echo quoted certificates and config files using a shell script.

This Tutorial shows how to set up an OpenVPN Server on DD-WRT and his clients on either Desktop PCs or another DD-WRT box.

Contents

Getting Started - Flashing the Router

To flash a brand new WRT54GL:

First, install the "mini" version of DD-WRT. (Current filename: dd-wrt.v24_mini_generic.bin) Then, install the "vpn" version of DD-WRT that has OpenVPN support. (Current filename: dd-wrt.v24_vpn_generic.bin) For other routers, use the appropriate bin files and installation procedure, as per the DD-WRT website.

We have more detailed instructions on this for example at Installation.

Enough NVRAM storage space?

All the data from the web-GUI is permanently stored in the NVRAM area. **Overfilling the NVRAM area is likely to brick your router**.

Using a KEY_SIZE of 1024 you need about 5200 bytes available in NVRAM on the server-side before you push SAVE in the web-GUI, **or you might brick your router**.
Using a KEY_SIZE of 2048 you need about 6000 bytes available in NVRAM on the server-side.

To test how much NVRAM space is left (and used) telnet or ssh into your router and type:

```
nvram show | grep size
```

If you do not have enough NVRAM space available, you cannot use the web-GUI method that is outlined below. You must use the Script method to store the certificates and activate VPN. Doing a factory reset may free up NVRAM, however, you will lose your existing configuration.

Creating Certificates

Once you have verified you have enough nvram space, you need the OpenVPN software installed on your computer, as it is used to create all the needed certificates.

See steps below for "how to" download/install/use OpenVPN on your computer, or visit http://openvpn.net/index.php/documentation/howto.html for the general official guide.

Creating Certificates Using Easy RSA in Windows

PRECAUTION: When generating certificates using Easy RSA in Windows, the certificate will be signed using GMT time, not your local time. This will result in the certificated not being valid until your local time equals that of GMT at the time of the certificate signing. For instance, if on the West Coast of USA, your local time is GMT-8. Your certificates will not be valid until 8 hours after generation, assuming that you have your time set to GMT-8 on the router and are using a NTP time server to manage the router time setting. You will need to set your router to use GMT until the time difference elapses or you will receive TLS Auth errors when tring to connect until the time difference passes. After then, you can set your time to GMT-8 or what ever time zone you are in.

Easy RSA is installed with the OpenVPN package on Windows. Open up a Command Prompt and cd to C:\<<Program Files>>\OpenVPN\easy-rsa. Run the following batch file to copy configuration files into place (this will overwrite any preexisting vars.bat and openssl.cnf files):

```
init-config   ->ENTER
```

Now edit the vars file (called vars.bat on Windows) and set the KEY_COUNTRY, KEY_PROVINCE, KEY_CITY, KEY_ORG, and KEY_EMAIL parameters. Don't leave any of these parameters blank. Save the file and return to the CMD Prompt. All of the following build actions produce files that will be placed in the "keys" directory under C:\<<Program Files>>\OpenVPN\easy-rsa\.

In the CMD Prompt, type:

```
vars ->ENTER

clean-all ->ENTER

build-ca  ->ENTER
```

The final command (build-ca) will build the certificate authority (CA) certificate and key by invoking the interactive openssl command:

```
ai:easy-rsa # ./build-ca
Generating a 1024 bit RSA private key
............++++++
...........++++++
writing new private key to 'ca.key'
-----
You are about to be asked to enter information that will be incorporated
into your certificate request.
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) [NewYork]:
Organization Name (eg, company) [MyORG]:
Organizational Unit Name (eg, section) []:MyUNIT
Common Name (eg, your name or your server's hostname) []:OpenVPN-CA
Email Address [me@myhost.mydomain]:
```

Note that in the above sequence, most queried parameters were defaulted to the values set in the vars or vars.bat files. The only parameter which must be explicitly entered is the Common Name. In the example above, I used "OpenVPN-CA". Generate certificate & key for server

Next, we will generate a certificate and private key for the server. Type:

```
vars   ->ENTER

build-key-server server   ->ENTER
```

As in the previous step, most parameters can be defaulted. When the Common Name is queried, enter "server". Two other queries require positive responses, "Sign the certificate? [y/n]" and "1 out of 1 certificate requests certified, commit? [y/n]". Generate certificates & keys for 3 clients

Generating client certificates is very similar to the previous step. Create as many client certs as needed, naming each individually. Type:

```
vars   ->ENTER
build-key client1   ->ENTER
```

If you would like to password-protect your client keys, substitute the build-key-pass script.

Remember that for each client, make sure to type the appropriate Common Name when prompted, i.e. "client1", "client2", or "client3". Always use a unique common name for each client. Generate Diffie Hellman parameters

Diffie Hellman parameters must be generated for the OpenVPN server. Type:

```
vars   ->ENTER
build-dh   ->ENTER
```

Output:

```
ai:easy-rsa # ./build-dh
Generating DH parameters, 1024 bit long safe prime, generator 2
This is going to take a long time
..................+.........................................
...................+.............+..................+.........
.....................................
```

Creating Certificates using Ubuntu Linux

Creating certificates only requires the easy-rsa package. Easy-rsa is now managed as an independent package

```
sudo apt-get install easy-rsa
```

You may want to use TLS pre-authentication on your VPN. Then you could need to install openvpn in this ubuntu machine, even this machine won't be a VPN tunnel peer, just to generate a *ta.key* file (you could generate this file from a client computer where the openvpn package is mandatory). Then the command above becomes

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

Recent easy-rsa package in Ubuntu features a single *make-cadir* command that will deal with all the stuff below about copying templates files and setting permissions into your own $HOME/<certification authority directory> or elsewhere. No need to sudo. Just run

```
make-cadir path
cd path
```

then edit *vars* as explained below, then go on with

```
. ./vars
```

then go on beginning with *./clean-all* in the code below.

Install OpenVPN package (with easy-rsa inside)

```
sudo apt-get install openvpn openssl
```

You must execute the following commands as root. **Either type *sudo* in front of every command (ie *sudo ./clean-all*) or do *sudo su* once and see the last character on the command prompt change to #.**

```
# Move to the OpenVPN script folder
cd /usr/share/doc/openvpn/examples/easy-rsa/2.0/

# Before anything else you may want to make a backup copy of the vars script
cp vars vars-org

# You can now edit some default values in the vars script if you want to, just saves you some typ
# there is no actual need to edit the vars script at any point during the use of  this guide
# gedit vars

# The following are the actual certificate building commands
source ./vars
./clean-all
./build-ca
./build-key-server server
./build-key client1
./build-key client2   #Etc, for other clients
./build-dh
```

If you plan to use TLS authentication (see openvpn manual , install openvpn now if not yet done. Create the shared key:

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

You can uninstall openvpn now if this machine won't be a vpn peer.

At this point, you have created the certificates which you will need to pass out to the server and clients. You find them in the new directory *keys*. KEEP THEM IN A SAFE PLACE.

Notes about the above commands:

- Before you run the *source vars* command you may want to edit some of the *export* lines in the *vars* file. You might do: *cp vars vars-org* before you edit the vars script. One of the following commands will allow you to edit the vars file in Ubuntu Linux *gedit vars* or *nano vars* or *vi vars*. KEY_SIZE, KEY_COUNTRY, KEY_PROVINCE, KEY_CITY, KEY_ORG, KEY_EMAIL are probably the only export variables you should mess with. Please note: if you change KEY_SIZE it must be done before running *source vars*.
- **source vars** - will run the vars script and export the vars variables all the way to the command prompt. Type *sudo set* in the Linux terminal window to see all global variables, or just *set* if you have already done *sudo su*. The rest of the commands above depend on and use the global variables exported from the *vars* script
- **./clean-all** - makes sure no old keys are stored in the "keys" directory. All the .crt and .key files you create make up ONE set of mutually dependent keys, that all store parts from one another.
- **./build-ca** - creates the ca.crt and ca.key files - ./build-ca will ask you to enter some parameters. Here is an example of what I might use (living in Denmark)

```
Country Name (2 letter code) [US]:DK
State or Province Name (full name) [CA]:DK
Locality Name (eg, city) [SanFrancisco]:AtHome
Organization Name (eg, company) [Fort-Funston]:AtHome
Organizational Unit Name (eg, section) []:AtHome
Common Name (eg, your name or your server's hostname) [Fort-Funston CA]:AtHome CA
Email Address [me@myhost.mydomain]:vpn@mydomain.dk
```

- **./build-key-server server** - creates the server.crt and server.key files. Here is an example of what I

might use (living in Denmark)

```
Country Name (2 letter code) [US]:DK
State or Province Name (full name) [CA]:DK
Locality Name (eg, city) [SanFrancisco]:AtHome
Organization Name (eg, company) [Fort-Funston]:AtHome
Organizational Unit Name (eg, section) []:AtHome
Common Name (eg, your name or your server's hostname) [server]:server
Email Address [me@myhost.mydomain]:vpn@mydomain.dk

Please enter the following 'extra' attributes
to be sent with your certificate request
A challenge password []:password
An optional company name []:AJAX Inc.
```

- **./build-key client1** - creates the client1.crt and client1.key files

```
Country Name (2 letter code) [US]:DK
State or Province Name (full name) [CA]:DK
Locality Name (eg, city) [SanFrancisco]:AtHome
Organization Name (eg, company) [Fort-Funston]:AtHome
Organizational Unit Name (eg, section) []:AtHome
Common Name (eg, your name or your server's hostname) [client1]:client1
Email Address [me@myhost.mydomain]:vpn@mydomain.dk

Please enter the following 'extra' attributes
to be sent with your certificate request
A challenge password []:password
An optional company name []:AJAX Inc.
```

  So long as you change nothing in the vars file, you can come back and run the ./build-key clientX command at any time, in order to create keys for one more client to connect to your OpenVPN (DD-WRT) server.

- **./build-dh** - creates the dh1024.pem or dh2048.pem files, depending on KEY-SIZE variable. Please note: if you change KEY_SIZE you *must* re-do all steps above begining with *source vars*

edit: easy-rsa is at version 3 and some of the commands have changed. These are the new ones. I use no password and require certificates. I believe this works closely with the rest of the instructions.

```
easyrsa build-ca nopass
easyrsa init-pki
easyrsa gen-req OVPN_server_name nopass
easyrsa gen-req OVPN_client_name nopass
openvpn --genkey --secret /etc/easy-rsa/ta.key
openssl dhparam -out /etc/easy-rsa/pki/dh.pem 2048
```

Setting up the Gateway

In the Web Interface of your DD-WRT loaded router, go to *Services* > *OpenVPN Daemon*.

The server will reject certificates unless the server clock is set correctly. To fix this, enable NTP.

- First, set "Start OpenVPN: Enable".

- Then you can either choose "System" or "WAN Up" as "Start Type". The first choice launches OpenVPN on system startup whereas the second runs OpenVPN whenever the WAN interface goes up

NOTE: Be carefull! If your openvpn.conf has to resolve some Domain Names, you may face issues with "System" start type. Especially if your openvpn.conf contains the command *local domain*.

- Second, paste the certificate files created above into the boxes in the DD-WRT web interface as follows:

Box	File to insert
Public Server Cert (CA Cert)	ca.crt
Certificate Revoke List	(blank)
Public Server Cert	server.crt
Private Server Key	server.key
DH PEM	dh1024.pem
OpenVPN Config	(see below)
OpenVPN TLS Auth	blank (or ta.key file content)

NOTE: Only paste the sections of text starting with (and including):
-----BEGIN CERTIFICATE-----
and ending with (and including):
-----END CERTIFICATE-----
in the text files. That is, include the two ---BEGIN/END CERTIFICATE--- lines. Do not paste all the descriptive stuff above that section.

NOTE on File Access in Ubuntu (for recent Ubuntu see Note** further): The following commands will make it easier to use the GUI tools in Ubuntu to copy text from the files stored in */usr/share/doc/openvpn/examples/easy-rsa/2.0/keys* to the web-GUI of DD-WRT.

```
# When you have issued  ./build-dh, then do:
chmod 755 keys
cd keys
chmod 755 *.*
```

Then open any File browser in Ubuntu menu "Places" and paste

```
/usr/share/doc/openvpn/examples/easy-rsa/2.0/keys
```

into the "Location:" text field and you can easily open (display/edit) the files in order to copy-and-paste the certificate and key sections into DD-WRT web-GUI.

Note** In recent Ubuntu, after you ran make-cadir with some target in your $HOME (e.g. My-OpenVPN-CA), source vars, clean-all and build-ca as explain above, the key folder ($KEY_DIR) has permissions set to 700 (fine). In it you can read/write all files. *.keys are set to permissions 600 (fine again), and all others are 644 (fine too). Don't open key/crt/pem files from the graphical file browser as a graphical utility ViewFile will popup, either puzzeling you with what to copy/paste or failing to show anything. Instead, in a terminal, use:

```
cd $KEY_DIR
# source ~/My-OpenVPN-CA/vars  (if the command above failed)
# cd $KEY_DIR (again)
```

```
cat ca.crt
cat server.crt
cat server.key
cat dh2048.pem
cat ta.key
```

Copy paste **only** the useful data to limit nvram space needed (including headers/footers -----BEGIN .....END PRIVATE KEY|CERTIFICATE|OpenVPN Static key V1-----

Sample Setup with Routing

The following example config file uses OpenVPN in routed mode. It is also possible to set up OpenVPN in bridged mode, this is explained below.

The Server Config File

In routed mode, there are three networks to consider:

1. The LAN (192.168.54.0 here)
2. The WAN (Internet)
3. The OpenVPN private routing network (192.168.66.0 here)

The OpenVPN private routing network is used by the OpenVPN software. The OpenVPN server and clients will be on this "private" subnet, and OpenVPN will route packets between your LAN subnet (192.168.54.0) and the OpenVPN subnet (192.168.66.0). Change the LAN subnet addresses to whatever you like. By default the LAN address is 192.168.1.0, but I changed it to 192.168.54.0 here so I could test these devices under my pre-existing LAN.

Note that the port used in this example is 1194, and the OpenVPN subnet is 192.168.66.0. You can change those, but then you must change the firewall commands to match your new settings. I chose 192.168.66.0 simply because it doesn't conflict with anything else on my network.

My Server Config File:

```
push "route 192.168.54.0 255.255.255.0"
push "dhcp-option DNS 192.168.66.1"
server 192.168.66.0 255.255.255.0

dev tun0
proto udp
keepalive 10 120
dh /tmp/openvpn/dh.pem
ca /tmp/openvpn/ca.crt
cert /tmp/openvpn/cert.pem
key /tmp/openvpn/key.pem

# Only use crl-verify if you are using the revoke list - otherwise leave it commented out
# crl-verify /tmp/openvpn/ca.crl

# management parameter allows DD-WRT's OpenVPN Status web page to access the server's management
# port must be 5001 for scripts embedded in firmware to work
management localhost 5001
```

In this example, all the computers behind this appliance will have 192.168.54.* IP addresses (The network is 192.168.54.0/24).

The Server Firewall Script

Go to Administration > Commands.

Type in these text "commands" for the Firewall, replace 1194 with your OpenVPN port number:

```
iptables -I INPUT 1 -p udp --dport 1194 -j ACCEPT
```

Replace 192.168.66.0/24 with your OpenVPN server subnet:

```
iptables -I FORWARD 1 --source 192.168.66.0/24 -j ACCEPT

 # These next two lines may or may not be necessary.
 # I (dereks) did not need them, but bmatthewshea did.
 # Thus, we include them so that this works for more people:
iptables -I FORWARD -i br0 -o tun0 -j ACCEPT
iptables -I FORWARD -i tun0 -o br0 -j ACCEPT
```

Then click *Save Firewall*.

The first two lines allow external clients to connect to the OpenVPN software (on port 1194). The last line allows packets to flow to/from the OpenVPN private network, and thus may not be necessary on bridged configurations.

Client Config File - Desktop

This is my configuration file for a Desktop OpenVPN Client to connect to the server we just set up. It was tested from a laptop with Ubuntu Linux.

Client Config File:

```
remote XXXXserver.dyndns.org 1194

client
remote-cert-tls server
dev tun0
proto udp
resolv-retry infinite
nobind
persist-key
persist-tun
float

#If the pushed routes appear not to be added on windows hosts, add the following:
route-delay 30

ca ca.crt
cert client1.crt
key client1.key
```

(Note: for a routed OpenVPN, the "float" option is necessary.)

You could prefer to use the sample client.conf file provided by openvpn installation: copy /usr/share/doc/openvpn/examples/sample-config-files/client.conf (or equivalent client.openvpn for Windows) into your $HOME (or *config* folder for Windows) then rename the copy to *your-client-name*, keeping the extension as is. Adapt it according your needs, e.g. : Set the line(s)

```
remote my-server-1 1194
;remote my-server-2 1194
```

according to the server name/ip port.

For linux you can uncomment lines

```
;user nobody
;group nogroup
```

for security, but if the tun/tap goes down, you'll need to re run the *sudo openvpn my.conf;* command.

Change the lines

```
cert client.crt
key client.key
```

to

```
cert your-client-name.crt
key your-client-name.key
```

as issued by certificates generation.

Uncomment the line

```
;tls-auth ta.key 1
```

if you use it (ddwrt server config is set to *tls-auth ta.key 0* , which is what we want), and so, copy the ta.key file into all clients **config** folder.

Just below, add

```
auth md5
```

if you want to match with ddwrt default, which is **not** openvpn *sha1* default. If you set ddwrt server to *sha1* you don't (may not) need this line.

Uncomment/change the line

```
;cipher x
```

to

```
cipher bf-cbc
```

which is the openvpn recommended ciphering algorithm, and set ddwrt openvpn server to use it instead of the default *none*.

Uncomment the line

```
;comp-lzo
```

which, with no more argument, means adaptive compression (match with ddwrt default)

Run openvpn as said above for linux. For Windows, right-clic the icon OpenVPN-GUI and select Run as administrator.

Note: the first statement in the config file, *client*, implies the *--pull* option which allows the server to push some other parameters to the clients once the connection is established.

Client Configuration - DD-WRT

This is the configuration for an OpenVPN Client running on another DD-WRT box.

Just set the OpenVPN server name (its WAN address or name) and port (1194) using the GUI, and then put in the certs similar to the procedure on the server:

Box	File to insert
Public Server Cert	ca.crt
Public Client Cert	client1.crt
Private Client Key	client1.key

Performance

Using Linksys WRT54GL v1.1 boxes as both client and server OpenVPN appliances.
CPU Model: Broadcom BCM5352 chip rev 0
SCP File transfer with CPU at 200 MHz: 313 KB/s
SCP File transfer with CPU at 250 MHz: 423 KB/s

Thus, estimated OpenVPN User Capacity (with CPU at 250 MHz):
For users with 768 Kbit DSL:
About 4-5 users (100% usage, like big downloads)
About 10-20 users (intermittent usage, like web or shell traffic)

For users with 128 Kbit dial-up modems:
About 25 users (100% usage, like big downloads)
About 50-100 users (intermittant usage, like web or shell traffic)

I tested an OpenVPN connection for about 24 hours in my lab. I transferred 525 MB of files. I also did two power cycles on both client and server appliances, and the tunnel re-established itself correctly.

Instructions for Bridging

Bridging is mainly needed if you need to exchange Broadcasts (needed for many network games and for windows shares) or if you have other protocols than IP. With a bridge the Clients get a part of the servers "other-side-network", normally the LAN on DD-WRT boxes.

In this example, the network (which is the LAN **and** the network where the VPN Clients are located) is 10.22.0.0/16 or 10.22.0.0 with a subnet mask of 255.255.0.0, the *normal* DD-WRTs DHCP-Server assigns no addresses under 10.22.0.100 (to set under Setup > Basic Setup > Network Address Server Settings (DHCP)).

The server config file for bridging

```
mode server
proto udp
port 1194
dev tap0
server-bridge 10.22.0.1 255.255.0.0 10.22.0.50 10.22.0.100
 # Gateway (VPN Server)   Subnetmask   Start-IP   End-IP
keepalive 10 120
daemon
verb 5
client-to-client
dh /tmp/openvpn/dh.pem
ca /tmp/openvpn/ca.crt
cert /tmp/openvpn/cert.pem
key /tmp/openvpn/key.pem

# Only use crl-verify if you are using the revoke list - otherwise leave it commented out
# crl-verify /tmp/openvpn/ca.crl
```

`verb 5` isn't necessary, but good for troubleshooting.

The Start-IP and End-IP in the `server-bridge` statement define the IP address range from where the Client get their addresses assigned. **It must not overlap with the DHCP Servers address range** (see above, before the file).

The Startup Script

Goto Administration > Commands, paste

```
openvpn --mktun --dev tap0
brctl addif br0 tap0
ifconfig tap0 0.0.0.0 promisc up
```

then hit *Save Startup*.

The Firewall Script

Clear the box (if it doesn't automatically do so) and enter

```
iptables -A INPUT -i tap0 -j ACCEPT
iptables -I INPUT -p udp --dport 1194 -j ACCEPT
```

Then hit *Save Firewall*.

Client config file for bridging

You will also need to modify your client config file(s) to match your server config file. In particular, the dev setting needs to match what is configured on the server.

Client Config File:

```
remote XXXXserver.dyndns.org 1194

client
dev tap0
proto udp
resolv-retry infinite
nobind
persist-key
persist-tun
float

ca ca.crt
cert client1.crt
key client1.key

ns-cert-type server
```

Now restart your router and try.

Troubleshooting

Prerequisites

Running commands and watching logs. Use telnet, SSH, or Administration > Commands to run commands.

To troubleshoot, you should turn on logging, and then watch the log file using this command:

```
 tail -f /var/log/messages
```

To turn on logging, do this:

1. Turn on syslog with setting Services > System Log > Syslogd" to *Enabled*,
2. Add a line that says "verb 5" to the OpenVPN config file

Steps

- First, is **OpenVPN running**?

```
ps | grep openvpn
```

If you don't see the OpenVPN process listed, then that is the first problem you need to solve.

Make sure you are running it by setting "Start type: WAN Up".

- Maybe there is an error in your **config file**. If so, OpenVPN will log an error message and then die.

Here is an example error I got when my config file had a bad setting in it:

```
root@WRT54GL:~# cat /var/log/messages | grep openvpn
Jul 31 11:55:13 WRT54GL daemon.err openvpn[1686]: Options error: --server directive network/netma
Jul 31 11:55:13 WRT54GL daemon.warn openvpn[1686]: Use --help for more information.
root@WRT54GL:~#
```

See if OpenVPN is logging an error message for you, to tell you what is wrong. (See "Logging" above; you need syslogd on and "verb 5" in your OpenVPN config file.)

- Next, is your **firewall** blocking OpenVPN?

If your firewall settings are wrong, OpenVPN's packets will be blocked by the DD-WRT firewall software.

First, turn off your firewall altogether, as a test to see if that makes things work. "Security > Firewall > SPI Firewall: Disable". If that fixes your problem, you need to tweak your firewall rules.

You can review your Linux IPTABLES firewall rules with this command:

```
iptables -L -v -n --line-numbers
```

Also, the firewall can log any DROPped or REJECTed packets. Examine these log messages and compare the DROPped packets to your iptables rules, and then tweak as necessary.

"Security > Firewall > Log: Enable" "Security > Firewall > Log > Log Level: Medium" "Security > Firewall > Log > Options > Dropped: Enable; Rejected: Enable"

There are many web sites that explain Linux iptables rules and commands.

- If you are still having trouble, make sure the cert and config **files are saved** correctly on the DD-WRT by looking in the directory /tmp/openvpn/ (for server) and /tmp/openvpncl/ (for client).

TLS Configuration Issues

There may be problems that occur due to TLS Errors, the following in particular "cannot locate HMAC in incoming packet." Here is an evolution/combination of the server and client configurations from this wiki and

the startup script listed in the other VPN configuration page:

Server:

```
push "route 192.168.1.0 255.255.255.0"
server 192.168.66.0 255.255.255.0

dev tun0
proto udp
port 1194

keepalive 15 60
daemon
verb 3
comp-lzo

client-to-client
duplicate-cn

tls-server
dh /tmp/openvpn/dh.pem
ca /tmp/openvpn/ca.crt
cert /tmp/openvpn/cert.pem
key /tmp/openvpn/key.pem
```

and client:

```
client
dev tun0
proto udp
remote 192.168.1.1 1194
resolv-retry infinite
nobind
persist-key
persist-tun
ns-cert-type server
comp-lzo
verb 3
float

ca ca.crt
cert client2.crt
key client2.key
```

Many thanks to all the contributors below, especially bmatthewshea who figured out the correct locations for the certificates.

--Derek

Windows Certificate Creation

For those who wish to create the certificates on their Windows machine, please reference the documentation from the OpenVPN site.

http://openvpn.net/index.php/open-source/documentation/howto.html#pki

ERROR MESSAGE: "OpenVPN: TLS_ERROR: BIO read tls_read_plaintext error: error:14090086:SSL routines:SSL3_GET_SERVER_CERTIFICATE:certificate verify failed"

In Windows there is a bug with easy-rsa that causes this error at any connection attempt if you added the line *remote-cert-tls server* to your client configuration file. The following fix was provided by a user on the OpenVPN mailing list.

In file *easy-rsa/openssl.cnf.sample* go to section [server] and add the following two lines:

```
extendedKeyUsage=serverAuth
keyUsage = digitalSignature, keyEncipherment
```

Then remake certificates from the beginning. You even have to run *init-config* again in order to erase the old openssl.cnf by the sample with the the right settings. Now everything should work properly :) .

Unix version is fine.

2010-08-14 --- I have installed OpenVPN 2.1 for Windows and faces this issue. The fix works very well!

Comments

- 2009-03-25: Posted to this wiki after looking everywhere and not finding this =P
  The original post which this article was copied & pasted from: Forum thread
- 2009-06-25: The server will reject certificates unless the server clock is set correctly. To fix this, enable NTP.
  Other note: The server seems to reject Server certificates where the common name on the certificate isn't 'server'
- 2009-07-16: revised the whole site --Azaël
- 2009-08-08: added TLS Configuration Issues section --zamodeo
- 2009-09-19: Followed the wiki using Ubuntu 9.04 and made minor clarification changes to the wiki --MrAlvin
- 2009-11-15: I suggest to remove last line of config file related to server config in bridging mode ("crl-verify /tmp/openvpn/ca.crl"). There is no "Certificate Revoke List" section filled in and no file created, so when it starts the first connection attempt, the server process goes in error and ends --bfg9000it
- 2009-12-18: I second the last suggestion regarding the config file for the server config in bridging mode. My setup was non-functional until I removed that line "crl-verify /tmp/openvpn/ca.crl" as I was not using a revoke list. I imagine that when one is instated, the line will be necessary to reference it; however I do not know. --arriflex

- 2010-01-21: To configure routed openvpn in linksys<->linksys mode with subnets behind each router talking to each other through the tunnel, it seems it's still necessary (dd-wrt v24-sp2 build 13064) to script the ccd directory and populate it with the proper iroute statement in a file named for the common name in the certificate of the connecting client router. Also, I found I needed "iptables -t nat -I POSTROUTING -o tun0 -j ACCEPT" in my firewall script and another line similar for the PREROUTING table. I used this on server and client because these rules turn off the NATing of the vpn tunnel between endpoints, which you want if you want your tunnel to be truly routed. Otherwise, that NAT does bad things like one-way audio for SIP traffic, e.g. --mulderlr

- 2010-02-10: I have set up a pfSense server and dd-wrt client (build 13064) which I'm tunneling VoIP traffic. I had to set up client specific iroutes on pfSense through the gui (there is a tab for this under OpenVPN), and make sure that the proper networks were routed and pushed in the server config. I did not find I need the PREROUTE and POSTROUTE of mulderlr, however, i did have to add a forward rule: "iptables -I FORWARD -i tun0 -o br0 -j ACCEPT" --Edjusted 15:52, 10 February 2010 (CET)

- 2010-02-15: As a nice addition to this how-to, I have an Asus WL500W with 4GB memory stick attached to it and Optware installed on it (although not necessary) I mount it as /opt and created an "openvpn" directory on it then copied all of the files from a running OpenVPN setup from a real server I have a chroot jail, ccd/ directory and certs on this directory, and as clients connect they get their respective settings from my server DHCP, DNS and WINS (samba) information so road-warriors or sites with more computers behind them can connect and it creates a really nice network. I'm testing this with five clients connected right now and works flawlessly.

- 2010-06-11: Had some Problems to get the openvpn daemon start with (v24-sp2 SVN revision: 13064). Reason was an **unreadable ca.crt certificate** although I pasted it via the webinterface. If the daemon isn't running (ssh -> ps | grep vpn) after you finished to follow the instructions from above, try starting it by hand (openvpn /tmp/openvpn/openvpn.conf) and watch for error messages. In my case I had to copy the ca.crt via scp to the router. --Dude 11:00, 11 June 2010 (CEST)

- 2010-06-11: If the connection to your vpn-server can be established and pinging some IPs in your "home-net" also works but the **communications freezes** after some seconds try "proto tcp-server" (Server-conf) and "proto tcp-client" (Client-conf) as alternative protocol. Don't forget to open the tcp-protocol in your firewall! This worked fine for me!--Dude 11:00, 11 June 2010 (CEST)

- 2010-08-30: I used Windows to generate the certificates and tried for over an hour to get it to work before I figured out that the certificates were generated for UTC time so the certificates would not become valid until 5 hours after creating them. I changed the router time to UTC for the time being, then changed it back to -500 the next day. It worked like a charm.

I'm using eko-s big build with OpenVPN on it, I set it up according to this how-to, but I added some bits from my own.

- Bridged mode setup
- Certificate based auth
- static key auth
- Client config dir
- Chroot jail for the vpn process
- Network details taken from the router

My config looks like this:

```
daemon
cd                      /opt/etc/openvpn
mode                   server
port                    443
proto                   tcp-server
dev                     tap0
chroot                  /opt/etc/openvpn/chroot/
ca                      /tmp/openvpn/ca.crt
cert                    /tmp/openvpn/cert.pem
key                     /tmp/openvpn/key.pem
dh                      /tmp/openvpn/dh.pem
tls-auth                /tmp/openvpn/ta.key 0

client-config-dir       ccd
ccd-exclusive

status                  logs/openvpn-status.log
log                      logs//openvpn.log
cipher                  AES-256-CBC
verb                    4
mute                   20
max-clients           15
management         127.0.0.1 7505
keepalive              10 120
tls-server
client-to-client
comp-lzo
persist-key
persist-tun
push "route 192.168.100.0 255.255.255.0 192.168.254.1"
# Adding a route for LAN2 on site A to the connecting VPN clients

push "dhcp-option WINS 192.168.254.100"
push "dhcp-option WINS 192.168.254.110"
push "dhcp-option DNS 192.168.254.110"
push "dhcp-option DNS 192.168.254.100"
push "dhcp-option DOMAIN dghvoip.lan"
```

As you see I do some "cool" things with my setup, I made some videos I posted on youtube about this, now with dd-wrt it's even better as many of these services can be centralized on the router.

Hope this comes in handy for someone.

If you need a bit of help with this PM me at the forum, I'd gladly try to help.

Great job, dd-wrt rocks,

--Dgonzalezh 00:41, 15 February 2010 (CET)

- 2009-05-11: By user icmp: To allow clients in the server side LAN to access clients in the client LAN i had to do some additional configuring.

Server side startup commands:

```
mkdir -p /tmp/openvpn/ccd
echo "iroute 192.168.24.0 255.255.255.0" > /tmp/openvpn/ccd/client1
echo "iroute 192.168.25.0 255.255.255.0" > /tmp/openvpn/ccd/client2
```

(Where 192.168.24.0 is the network on client1 and 192.168.25.0 is the network on client 2)

I added the following to the server side OpenVPN config:

```
push "route 192.168.24.0 255.255.255.0"
push "route 192.168.25.0 255.255.255.0"
client-config-dir /tmp/openvpn/ccd
route 192.168.24.0 255.255.255.0
route 192.168.25.0 255.255.255.0
client-to-client
```

Firewall commands on each client:

```
# Allow server-side and client-side hosts to ping routers endpoint
iptables -I INPUT 3 -i tun0 -p icmp -j ACCEPT
# Allow internal remote administration through HTTP
iptables -I INPUT 1 -i tun0 -p tcp --dport 80 -j ACCEPT
# Allow forwarding to other clients in the LAN
iptables -I FORWARD -i br0 -o tun0 -j ACCEPT
iptables -I FORWARD -i tun0 -o br0 -j ACCEPT
```

To save some memory I don't start httpd automatically. I disabled HTTPD in the web interface. If for any reason I should need the web interface, I simply connect with SSH to the router and issue the command: httpd If you consider doing the same, make sure first that you can access your router with telnet or ssh. Otherwise you'll lock yourself out.

Thank You very much for this article. Great job!

- 2011-03-13: Regarding comments above dated 2009-11-15 and 2009-12-18, it's worth pointing out that if you do use a Certificate Revocation List, when you add the directive crl-verify <crl_file>, be sure to point to the full path of <crl_file>. The OpenVPN documentation does not indicate this, and it appears that clients cannot connect if openvpn cannot find the CRL file. --jmaher

---

--Keamas 11:50, 27 October 2010 (CEST)

Can anyone post how to route the whole traffic through the VPN tunnel with the Gateway scenario ?

--Krissi 19:00, 25 June 2011 (CEST)

Public Server Cert and CA Cert
This article assimilates that the "Public Server Cert" and "CA Cert" are the same. That is seen under the Server Configuration. It is actually described as just one value to insert.
Fact is though that the OpenVPN setup needs input for both **CA Cert** and **Public Server Cert**. Somebody should update the documentation to reflect this. As far as I understand the values should be as follows:
CA Cert = ca.crt
Public Server Cert = server.crt
Private Server Key = server.key
DH PEM = dh1024.pem

Public Client Cert = client.crt
Private Client Key = client.key