How to set up your own Raspberry Pi powered VPN

This document is based on a paper by Kate Russell, BBC Click presenter, 16 July 2015.

The original was annotated by G8KBB to correct some details. This was to reflect the packages that the article used, because as of 2019, these had changed since the original publication by the BBC.

You can find that mark-up on the Warrington Amateur Radio Club website (http://warc.org.uk).

This version is an update as of April 2020 as the packages have changed again – notably Easy RSA. The original article carried markups showing the changes to the BBC article, these have not been included this time as the document was getting too messy. This is just the April 2020 details.

This first section is reproduced verbatim from BBC Click.

Eyes are everywhere online.

The websites you visit often track where you came from and watch where you head off to next.

A VPN - or virtual private network - helps you browse the internet more anonymously by routing your traffic through a server that is not your point of origin.

It is a bit like switching cars to shake off someone who is tailing you.

There are plenty of companies offering services with varying degrees of security and varying degrees of cost, but if you are willing to roll your sleeves up and get technical with some basic coding and a £30 Raspberry Pi computer, you can build your own VPN server at home.

It won't give you the option of appearing to be from somewhere else but you can use it to connect external devices like a smartphone to browse the internet more securely through your home network, and access shared files and media on your home computer.

Make no mistake, this is not a quick and easy process.
On **BBC Click** I have shared some tips from my own experience setting up a DIY VPN server.

Below is a step-by-step guide you will need to follow to the letter, symbol and space if you want to follow in my footsteps.

To follow this guide you will need:

- 1 x Raspberry Pi/Pi 2
- 1 x 8GB micro SD card
- 1 x SD card reader
- 1 x 5 volt mini USB power supply (a suitable phone charger will do)
- 1 x HDMI monitor (your TV or computer monitor)
- 1 x USB keyboard

- 1 x Ethernet network cable

Prepare to install your operating system

Insert the micro SD card in your card reader.
If you are reusing an old SD card make sure it is fully formatted to remove any old files using the free tool at **http://sdcard.org**

Install Raspbian on your Raspberry Pi

Download NOOBS (New Out Of the Box Software) from the Raspberry Pi website (**https://www.raspberrypi.org/downloads**/). This is an easy operating system installation manager.

Open the .zip you downloaded and select all files, then just drag and drop them onto your SD card.

Insert the SD card in the Raspberry Pi then connect a monitor, keyboard and power cable.

Connecting the power will cause the Raspberry Pi to boot up and the green and red LEDs on the board should light up.

If the files are copied properly onto the SD card the green light will start flashing as the computer reads the data.

After a few seconds you will see a window open on the monitor with a range of operating systems to install - use the arrow keys on the keyboard to choose Raspbian and hit ENTER to install.
N.B. If you have trouble getting the NOOBS installation manager to work, you can also install Raspbian by copying the disk image of the operating system onto your micro SD card. Follow the instructions at **https://www.raspberrypi.org/downloads/** to do this.

Change the default password

Before you go any further, make sure you change the default password, or anyone who knows the default will be able to access your home network.

You can do this from the options screen you are shown the first time you boot up your Raspberry Pi after Raspbian is installed.

When you next reboot your Raspberry Pi the login will be "pi" and the password whatever you have set.

Also set locale etc. and change the pi logon and admin passwords

Give your Raspberry Pi a static IP address

The IP address is what tells devices where to find your Raspberry Pi on your home network.

Networks usually issue a dynamic IP address, which can change each time you power up the device. If you want to be able to consistently connect to your Raspberry Pi from outside your home network you need to fix its IP address so that it is always the same - a static IP address.

Connect your Raspberry Pi to your router with an Ethernet cable.

At command prompt type:

ifconfig

A bunch of information will come up and you need to note down what it says for your set against the following:

inet addr [Current IP Address]

bcast [Broadcast Range]

mask [Subnet Mask]

Next at the command prompt type:

sudo route -n

This tells you information about your router. Note down:

Gateway

Destination

You now have all the information you need about your current IP set up and can edit the network configuration file to make the IP static.

At command prompt type:
sudo nano /etc/dhcpcd.conf

update the following lines with your IP, router and domain name server addresses, for example:
static ip_address=192.168.1.51/24

static routers=192.168.1.254

static domain_name_servers=192.168.1.254

To save the file press CTRL and X together, when prompted to save type "y" and hit Enter to accept the file name without changing it.

At the command prompt type:
sudo reboot

Your Raspberry Pi will now restart with the new, static IP address.

Set up an easy control system

To save switching around cables if you do not have a spare HDMI monitor and keyboard you can download a free utility that lets you control your Raspberry Pi through a pop up window on another computer.
This is called an SSH. The tool is called PuTTY (**j.mp/DLPutty**).

In order to use Putty you need to enable SSH on the pi. To do this run raspi-config as root and enable ssh

Double click the PuTTY.exe file you download and it opens a dialogue box where you can enter the new static IP address you have given your Raspberry Pi. The first time you do this it will ask you to confirm accessing the device.

You can now login and do everything you need to through this dialogue box on your computer, which means your Raspberry Pi never needs a monitor or keyboard to keep running. This is known as running it "headless".

Update your Raspberry Pi

One last piece of housekeeping to ensure you are running the latest software and drivers.

At the command prompt type:
sudo apt-get update

Wait for the updates to finish downloading and then type:
sudo apt-get upgrade

Wait until the upgrade completes.

After this, cleanup by executing:
sudo apt-get clean
sudo apt-get autoremove -y

You are now ready to make your VPN

The Raspbian operating system we just installed comes with OpenVPN ready to unpack, which is the software we will be using to make our VPN.

At command prompt type:
sudo apt-get install openvpn

You will be asked to confirm your instruction then the software will be unpacked and installed.
Generating keys

Just like the unique key that unlocks your front door, your VPN needs keys generated to make sure that only authorised devices can connect to it.

OpenVPN comes with Easy_RSA, a simple package for using the RSA encryption method to generate your unique keys.

The next series of commands need to be done in the root directory. You will notice at the moment the command prompt sits in a directory labelled as 'pi@raspberrypi: ~ $'.

Typing "sudo" before a command tells the operating system to execute the command in the root directory, but if you want to save yourself some typing you can go ahead and type:
sudo -s

You will now see your command prompt sits at 'root@raspberrypi:'

Now, at the command type on one line:
cp -r /usr/share/easy-rsa /etc/openvpn/easy-rsa

Make sure you have spaces in the right places (before /usr and /etc). This instruction copies all of the files from the easy-rsa directory into a directory in your openvpn installation.

N.B. You can copy lines of text using right-click and then when you right click inside the PuTTY window it should paste, saving you a lot of typing. Be aware though, some formatting errors can occur when copying and pasting large blocks of text so if you do not get the result you are expecting, resort to typing the details in by hand.

Next type:

cd /etc/openvpn/easy-rsa

This changes the directory your command prompt sits at to your openvpn/easy-rsa directory.

Now make a copy of the file vars.example and call it vars. To do that execute this:

cp vars.example vars

You now need to edit the text in the file we just copied. Nano is a simple text editor in Raspbian you are going to see a lot of over the next few pages.

To open the file inside this text editor type:

nano /etc/openvpn/easy-rsa/vars

You do not need to edit the vars file as most of the defaults are fine, but I suggest you change the key size line to ensure 2048 bits RSA is used. To do this, find the line:

#set_var EASYRSA_KEY_SIZE 2048

And remove the '#' from the start of the line

Type CTRL and X then Y then ENTER to save this file.

Build your certificates

You are now set up to build the certificates your VPN will use to grant authority to devices you want to connect with.

Enter and run these three commands. The first one initialises the PKI data files. If you make a mess of any of the following steps, come pack to this point and rerun init-pki. It will erase everything that you have done with keys and certificates and you can start again.

./easyrsa init-pki
dd if=/dev/urandom of=pki/.rnd bs=256 count=1
./easyrsa build-ca

When prompted, choose a good strong passphrase and write it down somewhere safe

When prompted for common name enter something innocuous like "dummy.com"

Now build the key server. By all means replace "MyHomeCA" with something you prefer for the server name

./easyrsa build-server-full MyHomeCA

When prompted choose a PEM passphrase (this is NOT the one you entered earlier – choose another one and write it down).

It will then ask for the ca key passphrase – this is the one you created earlier. Enter it

Build keys for each user

Your server is now set up and you need to build keys for all the devices you want to be able to connect.

You can cut corners here and just build one key to use on all devices. Only one device can connect using each key at a time though, so if you want simultaneous connections you will need a different key for each one. I suggest separate keys.

To assign a user a key, type:

./easyrsa build-client-full username

Where “username” is whatever name you want to use – for example ‘DaveNexus”

You will be prompted for a PEM passphrase; this one is a passphrase for the user for whom you are generating the certificate – choose something that is easy to remember (but not weak).

You will then be prompted for the CA key passphrase – enter that too

You can then add an additional layer of security.

Change directory to the private keys folder and encipher the key with a passphrase

cd ./pki/private

You now encipher the message using 3DES

openssl rsa -in KateAndroid.key -des3 -out KateAndroid.3des.key

you will then be prompted for a passphrase for the user for whom you are generating the certificate – repeat the one you used for them above, and again for the new PEM pass phrase.

Repeat the client certificate instructions for each client (remember to change directory back to easy-rsa folder each time)

You have now created your "client certificates".

Change directory back to the easy-rsa folder.

Generate the Diffie-Hellman key exchange.

This is the code that lets two entities with no prior knowledge of one another share secret keys over a public server. Type:

./easyrsa gen-dh

The screen will slowly fill with dots as the key is built from random numbers. It will take at least an hour if you upped your encryption to 2048-bit. If you left it at 1024-bit it could take as little as five minutes on a raspberry pi 1. For a Pi4 it’s quite quick in comparison.

Denial of Service (DoS) attack protection

OpenVPN protects against this kind of attack by generating a static pre-shared hash-based message authentication code (HMAC) key. This means the server will not try to authenticate an access request if it does not detect this key. To generate the static HMAC key type:

Execute this:

openvpn --genkey --secret pki/private/ta.key

N.B. If you are using copy and paste it probably will not work on this line as the double "-" seems not to translate in the same way if you do not type it in.

Configuring your server

Now you have created all the locks and keys you need to tell your Raspberry Pi where you want to put the doors and who you want to give the keys to - essentially instructing the OpenVPN which keys to use, where you are going to be connecting from and which IP address and port to use.

To do this you must create a server configuration file. At command prompt type:

nano /etc/openvpn/server.conf

This opens an empty file called “server.conf”.

Fill it with this text, taking care to change the details where indicated with a comment in # CAPS LOCK. (Placing a "#" in front of a sentence in the code like this tells the system it is a comment and to ignore it when building the program). Also when changing the YOUR SERVER NAME sections I refer to the server name that was given to the 'build-key-server' command earlier on.

```
dev tun
proto udp
port 1194
ca /etc/openvpn/easy-rsa/pki/ca.crt
cert /etc/openvpn/easy-rsa/pki/issued/XX.crt # SWAP XX WITH YOUR SERVER NAME
key /etc/openvpn/easy-rsa/pki/private/XX.key # SWAP XX WITH YOUR SERVER NAME
dh /etc/openvpn/easy-rsa/pki/dh.pem
server 10.8.0.0 255.255.255.0
# server and remote endpoints
ifconfig 10.8.0.1 10.8.0.2
# Add route to Client routing table for the OpenVPN Server
push "route 10.8.0.1 255.255.255.255"
# Add route to Client routing table for the OpenVPN Subnet
push "route 10.8.0.0 255.255.255.0"
# your local subnet
push "route 192.168.0.10 255.255.255.255" # SWAP THE IP NUMBER WITH YOUR
RASPBERRY PI IP
ADDRESS
# Set primary domain name server address to the SOHO Router
# If your router does not do DNS, you can use Google DNS 8.8.8.8
push "dhcp-option DNS 192.168.0.1" # THIS SHOULD MATCH YOUR OWN ROUTER
ADDRESS
# Override the Client default gateway by using 0.0.0.0/1 and
# 128.0.0.0/1 rather than 0.0.0.0/0. This has the benefit of
# overriding but not wiping out the original default gateway.
push "redirect-gateway def1"
client-to-client
duplicate-cn
keepalive 10 120
tls-auth /etc/openvpn/easy-rsa/pki/private/ta.key 0
cipher AES-128-CBC
comp-lzo
```

user nobody
group nogroup
persist-key
persist-tun
status /var/log/openvpn-status.log 20
log /var/log/openvpn.log
verb 1

Hit CTRL and X then Y and ENTER to save.

There is one last edit to make in the server configuration files to make sure your Raspberry Pi knows you want it to forward Internet traffic through our new network.

Type:
nano /etc/sysctl.conf

Near the top it says, "Uncomment the next line to enable packet forwarding for IPv4."

You want to remove the "#" from the start of the next line to inform OpenVPN you want it to take that text into consideration.

The line should then read:
net.ipv4.ip_forward=1

Hit CTRL and X, then Y and ENTER to save.

Finally you need to action the change you just made in the sysctl.conf file. To do this type:
sysctl -p

You have now made a functioning server that can access the internet.

Pass through the firewall

Raspbian has a built-in firewall that will block incoming connections, so we need to tell it to allow traffic from OpenVPN to pass through.

To create a file that will run each time you start up your Raspberry Pi issuing this permission type:
nano /etc/firewall-openvpn-rules.sh

Inside this new file type:

#!/bin/sh
iptables -t nat -A POSTROUTING -s 10.8.0.0/24 -o eth0 -j SNAT --to-source 192.168.0.10
 # SWAP THE IP NUMBER WITH YOUR RASPBERRY PI IP ADDRESS

CTRL and X then Y and ENTER to save.

Newly created files are not executable by default, so we will need to change the permissions and ownership of this file you just created. To do this type:
chmod 700 /etc/firewall-openvpn-rules.sh
chown root /etc/firewall-openvpn-rules.sh

This script gives OpenVPN permission to breach the firewall and we now need to add it into the interfaces setup code so it runs on boot. To do this…

Create a file in '/etc/network/if-pre-up.d' called 'openvpn-firewall' by executing

cd /etc/network/if-pre-up.d
sudo nano openvpn-firewall

And set the contents to:

#!/bin/sh
/etc/firewall-openvpn-rules.sh

And finally make the file executable by typing:

sudo chmod 755 openvpn-firewall

Finally, reboot your Raspberry Pi by typing:

reboot

N.B. Each time you reboot your Raspberry Pi you will need to relaunch PuTTY to connect to it.
Ensure you have a static public IP address

We have created locks and keys for devices to use to connect to your VPN, but before we hand those keys out we need to tell them where to find the front door. This is your public IP address, which should be kept a secret as it identifies your location on the internet.

You can find out your public IP by asking Google. Just type "what's my IP address?" into the search box.

If this address changes each time you log on you do not have a static IP address so will need to use a dynamic domain name system (DDNS) service to give yourself a domain name to put in place of the IP address.
There is a free service at **https://www.changeip.com** Then on your Raspberry Pi, you need to run something called DDclient to update your DDNS registry automatically.

At the command prompt type:

sudo apt-get install ddclient

This will launch a wizard for configuring ddclient. Don't worry too much about what you enter here as we will be entering the config file straight away.

To edit the DDClient configuration with the correct setting type:

sudo nano /etc/ddclient.conf

Every service will have slightly different configuration, - if you are using changeip.com this blog post will tell you how to edit your settings successfully:

https://blogdotmegajasondotcom.wordpress.com/2011/03/14/use-ddclient-with-changeip-com/

FOR CHANGEIP:
use=web, web=ip.changeip.com
protocol=changeip
ssl=yes
server=nic.changeip.com/nic/update
login=<your login name>

password=<your password>
AND ENTER YOUR DOMAIN NAME

CTRL and X then Y and ENTER to save.

Finally, to set this program running type:
sudo ddclient
sudo systemctl start ddclient.service

N.B. If you reboot your Raspberry Pi you'll need to type "*sudo ddclient*" to start running it again. In order to make sure that the dynamic dns entry is kept up to date, edit /etc/default/ddclient and set run-daemon='true'

then for good measure, to force a weekly update, edit /etc/cron.weekly/ddclient and set the contents to:
#!/bin/sh
/use/sbin/ddclient -force

then make the script executable by typing
sudo chmod +x /etc/cron.weekly/ddclient

Note: to debug & check execute this:
sudo ddclient -debug -verbose -noquiet

Note: to check status do this:
sudo service ddclient status

Create profile scripts for the devices you want to connect

We have created keys for clients (computers and devices) to use to connect to your VPN, but we have not told the clients where to find the server, how to connect, or which key to use.

If you created several different client keys for each of the devices you want to grant access, it would be a lot of trouble to generate a new configuration file for each client from scratch.

Luckily Eric Jodoin of the SANS institute has written a script to generate them automatically. This is his script modified for the file system structure of easyrsa 3.

First type:
sudo nano /etc/openvpn/easy-rsa/pki/Default.txt

Fill in the blank text file with the following:
client
dev tun
proto udp
remote [YOUR PUBLIC IP ADDRESS] 1194 #REPLACE YOUR DYNAMIC DNS VALUE FROM CHANGEIP.COM
resolv-retry infinite
nobind
persist-key
persist-tun
mute-replay-warnings
ns-cert-type server

key-direction 1
cipher AES-128-CBC
comp-lzo
verb 1
mute 20

CTRL and X then Y and ENTER to save.

Next, to create the script that makes your profile keys type:
nano /etc/openvpn/easy-rsa/pki/MakeOVPN.sh

In this file you need to add the text that Jodoin wrote to create the script:

```
#!/bin/bash
# Default Variable Declarations
DEFAULT="Default.txt"
FILEEXT=".ovpn"
CRT=".crt"
KEY=".3des.key"
CA="ca.crt"
TA="ta.key"
CLIENTDIR="../../client/"
CERTDIR="./issued/"
KEYDIR="./private/"
#Ask for a Client name
echo "Please enter an existing Client Name:"
read NAME
#1st Verify that client's Public Key Exists
if [ ! -f $CERTDIR$NAME$CRT ]; then
echo "[ERROR]: Client Public Key Certificate not found: $CERTDIR$NAME$CRT"
exit
fi
echo "Client's cert found: $CERTDIR$NAME$CR"
#Then, verify that there is a private key for that client
if [ ! -f $KEYDIR$NAME$KEY ]; then
echo "[ERROR]: Client 3des Private Key not found: $KEYDIR$NAME$KEY"
exit
fi
echo "Client's Private Key found: $KEYDIR$NAME$KEY"
#Confirm the CA public key exists
if [ ! -f $CA ]; then
echo "[ERROR]: CA Public Key not found: $CA"
exit
fi
echo "CA public Key found: $CA"
#Confirm the tls-auth ta key file exists
if [ ! -f $KEYDIR$TA ]; then
echo "[ERROR]: tls-auth Key not found: $KEYDIR$TA"
exit
fi
echo "tls-auth Private Key found: $KEYDIR$TA"
#Ready to make a new .opvn file - Start by populating with the default file
cat $DEFAULT > $CLIENTDIR$NAME$FILEEXT
#Now, append the CA Public Cert
echo "<ca>" >> $CLIENTDIR$NAME$FILEEXT
cat $CA >> $CLIENTDIR$NAME$FILEEXT
```

```
echo "</ca>" >> $CLIENTDIR$NAME$FILEEXT
#Next append the client Public Cert
echo "<cert>" >> $CLIENTDIR$NAME$FILEEXT
cat $CERTDIR$NAME$CRT | sed -ne '/-BEGIN CERTIFICATE-/,/-END CERTIFICATE-/p' >>
$CLIENTDIR$NAME$FILEEXT
echo "</cert>" >> $CLIENTDIR$NAME$FILEEXT
#Then, append the client Private Key
echo "<key>" >> $CLIENTDIR$NAME$FILEEXT
cat $KEYDIR$NAME$KEY >> $CLIENTDIR$NAME$FILEEXT
echo "</key>" >> $CLIENTDIR$NAME$FILEEXT
#Finally, append the TA Private Key
echo "<tls-auth>" >> $CLIENTDIR$NAME$FILEEXT
cat $KEYDIR$TA >> $CLIENTDIR$NAME$FILEEXT
echo "</tls-auth>" >> $NAME$FILEEXT
echo "Done! $CLIENTDIR$NAME$FILEEXT Successfully Created."
#Script written by Eric Jodoin
```

Note – there should be No newline at the end of the file

CTRL and X then Y and ENTER to save.

N.B. I was not able to successfully copy and paste the entire script accurately in one go, but taking it one section at a time worked no problem).

Next you need to give this script permission to run. Type:

cd /etc/openvpn/easy-rsa/pki

Then to give it root privileges type:

chmod 700 MakeOVPN.sh

Finally, execute the script with:

./MakeOVPN.sh

As it runs, it will ask you to input the usernames names of the clients for you generated keys for earlier (in my case KateAndroid). Type that when prompted and you should see the line:

Done! KateAndroid.ovpn Successfully Created.

Repeat this step for each additional username you added client.

Export your client keys for use on the connecting devices

You now need to copy those keys onto the devices you want to use them. If you are using PuTTY on a Windows machine you can use a software package called WinSCP to do this. For Mac, try Fugu.

First, to grant yourself read/write access to the folder at the command prompt type the following, naoting that for easy-rsa the files are in /etc/openvpn/easy-rsa/client and not in /etc/openvpn/easy-rsa/keys

```
chmod 777 /etc/openvpn

chmod 777 /etc/openvpn/easy-rsa

chmod 777 /etc/openvpn/easy-rsa/keys
```

chmod 777 /etc/openvpn/easy-rsa/keys/[ClientName].ovpn

Be sure to undo this when you're done copying files by typing:

chmod 600 /etc/openvpn

and repeating for each step with the chmod 600 command, which removes read/write access again.

You can now launch the software you are using to copy the files off your Raspberry Pi to navigate to the openvpn folder and copy the files labelled "KateAndroid.ovpn" etc.

You can also open the command prompt on the machine in your network you would like to copy the files to and type:

scp pi@[ip-address-of-your-pi]:/etc/openvpn/easy-rsa/keys/[ClientName].ovpn [ClientName].ovpn

Install the OpenVPN Connect app on your device

You are now ready to download and install the OpenVPN Connect app on your Android or iPhone - they are available through the stores as a free download. You will need to import the profile keys you just made as the final piece of the VPN connection puzzle.

When prompted for a pass phrase here it is the 3des.key one you will need to enter.

For iOS

Use iTunes to add the .ovpn file to the OpenVPN Connect app. When you launch the app on your phone you will now get the option of installing that profile and making the connection.

For Android

Connect your android device to your computer with a USB cable. Navigate to the Downloads folder on your handset and paste the .ovpn file there.

When you launch the app on your handset you can now tap the menu dropdown in the top right corner, select Import>Import profile from SD card then navigate to the downloads folder and choose to import the file and make the connection.

One more thing

After all this is done, if your phone still can't connect to the OpenVPN server you might need to adjust the firewall on your router to do port-forwarding of port 1194 to the Raspberry Pi. You'll need to follow the instructions from your ISP to access the router and complete this step.

OTHER STUFF:

TO ROTATE THE OPENVPN LOG FILES ADD OPENVPN.LOG TO /etc/logrotate.d/rsyslog :

```
/var/log/openvpn.log
{
        daily
        rotate 7
        missingok
        notifempty
```

```
        postrotate
                invoke-rc.d rsyslog rotate > /dev/null
                systemctl restart openvpn
        endscript
}
```

TO DO A WEEKLY REBOOT ADD THE FOLLOWING TO CRONTAB BY EXECUTING THE COMMAND ‘crontab -e’

```
0 4 * * 0 /sbin/shutdown -r now
```

THE ‘0 4’ MEANS 4:00 AM AND THE SECOND 0 MEANS SUNDAY