Changing the VPN keys on R7000

Preface

This is a small walk-through of changing the VPN keys on the R7000 router. It is possible that these or similar steps may also work on other models. It was written because, to the best of my knowledge, there is no official way to configure or change the VPN keys on my R7000.

Some reasons you might want to change the keys:

- You want to try to install stronger keys then the device ships with
- You think your existing keys have been compromised

As a forewarning, these notes should not be too complex for a moderately technical user, but there are a lot of steps to follow. For those who have never used a console/terminal before, I've tried to write in as straightforward manner as possible, but I can't do too much hand-holding here or this doc would easily become twice as long. I would recommend people who fall into that camp get someone better qualified to help them.

I work on a Windows system and that's what I've written this doc against. If you are on another OS you'll have to make a few adaptations, but the majority of it should translate.

Disclaimer and Warnings

ANY USE OF THIS WORK IS ENTIRELY AT YOUR OWN RISK. E.g. if your router stops working, catches fire, and burns your house down, that's 100% your problem alone and you agree I am not responsible in any way for your actions. I can’t guarantee this guide will work on your particular device or even across different firmware versions. If any of this is unacceptable to you then stop reading now and delete any erase any copies of this work.

The steps in this guide have not been reviewed by Netgear, nor are they endorsed by anyone. Following the steps in this guide might void your product warranty. There is always a risk that you could experience problems on future firmware versions.

I offer no assurances that following this guide will have a positive effect on your router or security, nor any assurances that it will not have a negative effect.

Some of the tools referenced in this document provide strong crypto and/or are used as or in conjunction with VPN products. Some countries don’t like these types of tools. Please be aware of the laws in your country, and if concerned, do not follow this guide and do not follow any of the links below or download the referenced software.

Document history

Version 1.0.1

- Fixed a few typos.
- Added a suggestion to clean up the TFTP transfer folder when finishing up.
- Added a note about the version of OpenVPN this document is compatible with.

Version 1.0.0

- The first version. Nobody has even tried it out yet. Does that worry you? It should ;)

Step 1 - Enabling telnet

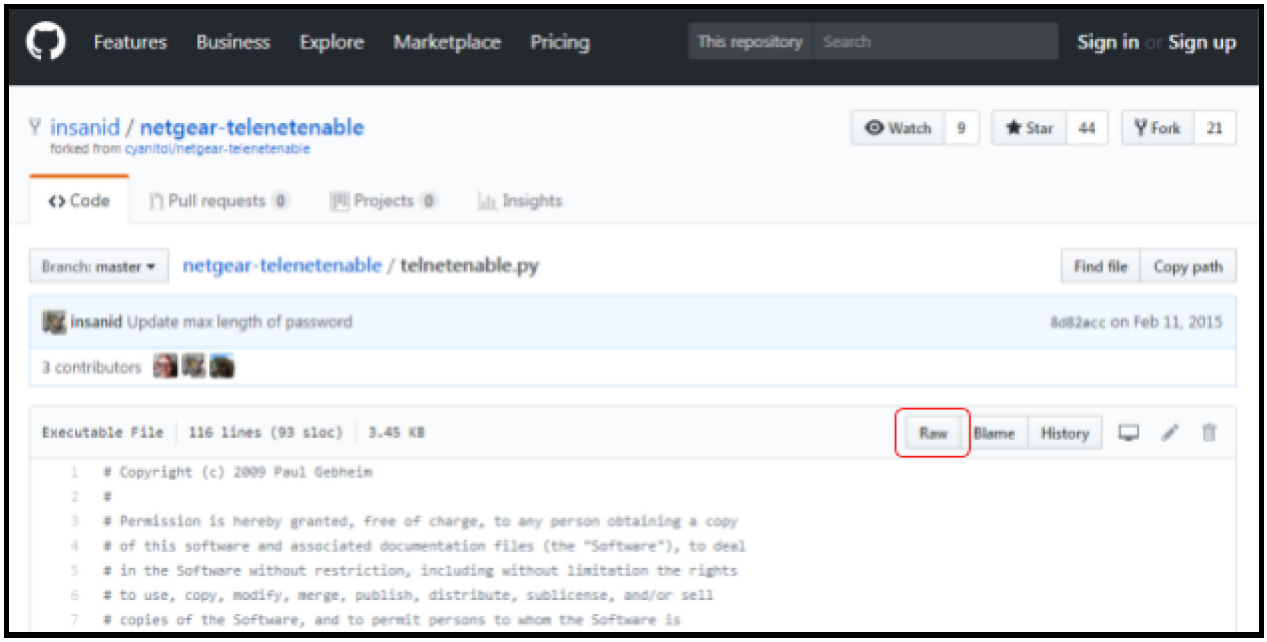
You can connect to the Nighthawk via a protocol called telnet and run various commands on it, and you'll need to do that in order to replace the VPN keys. For security, the Telnet server isn't running/open by default, and you need to send the router a "magic packet" to turn it on. The magic packet you'll need to send to your router is unique based upon its MAC address and its username and password. The packet is generated and sent via a "telnetenable" program.

There are a few "telnetenable" programs available for Netgear routers, and some versions only work with older models/firmwares, so you'll need to find one that works with the current generation.

1.a: Get a compatible telnetenable script

I chose to use a script that is written in Python since it's open source, which means you can check it's not going to do anything nasty when you run it, and it is also cross platform, i.e. it will run wherever you can find a Python interpreter.

I downloaded a script from here, which works with the R7000 and probably other recent models: https://github.com/insanid/netgear-telenetenable/blob/master/telnetenable.py

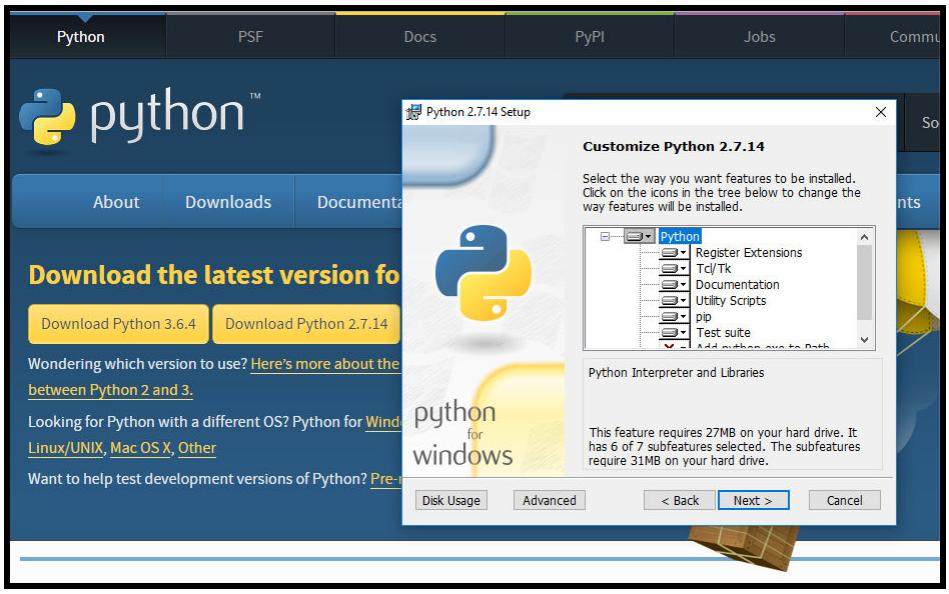


Hit the "Raw" button and save it to a plain text file with a ".py" extension.

1.b: Install Python 2.7

You need a Python interpreter to run .py scripts. You must use a version in the 2.x series because the syntax changed between Python versions 2.x and 3.x, and the telnetenable script won't work with 3.x without some modifications.

Download Python from here:
https://www.python.org/downloads/

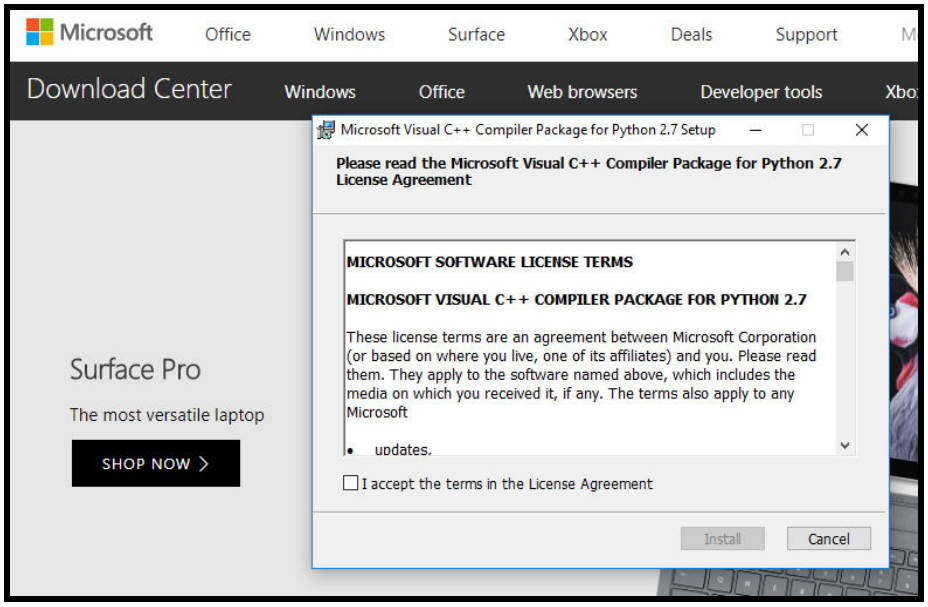


The default options should be fine for just about everybody.

1.c: Install the MSVC compiler for Python

The telnetenable script depends on a component in the pycrypto library, and before pycrypto will work you need the MSVC compiler.

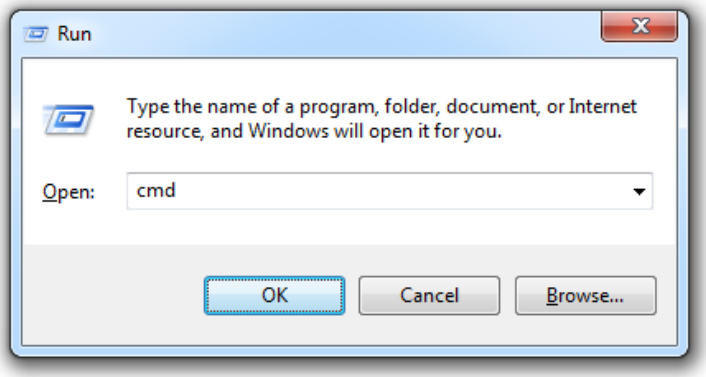
https://www.microsoft.com/en-us/download/confirmation.aspx?id=44266



1.e: Install pycrypto

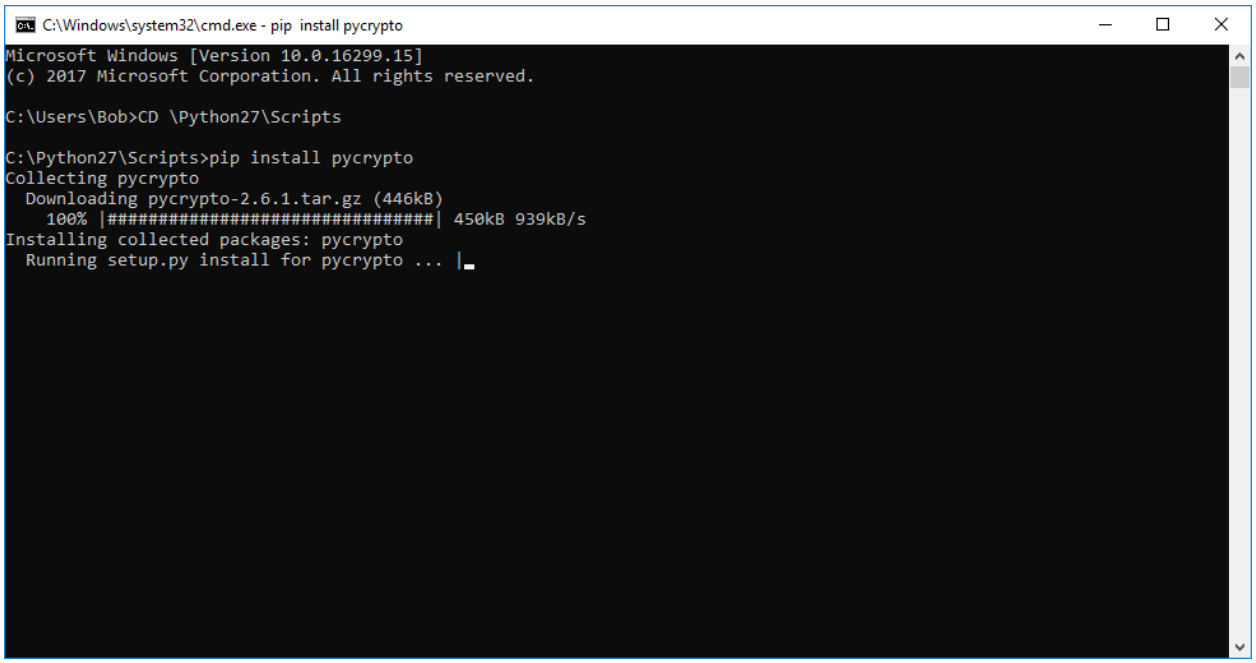
The telnetenable script needs pycrypto to calculate the magic packet you're going to send to the router.

Start a new command prompt by hitting Win+R and entering "cmd":



Assuming you are on Python 2.7 and installed it to the default location, issue these commands:

```
CD \Python27\Scripts
pip install pycrypto
```



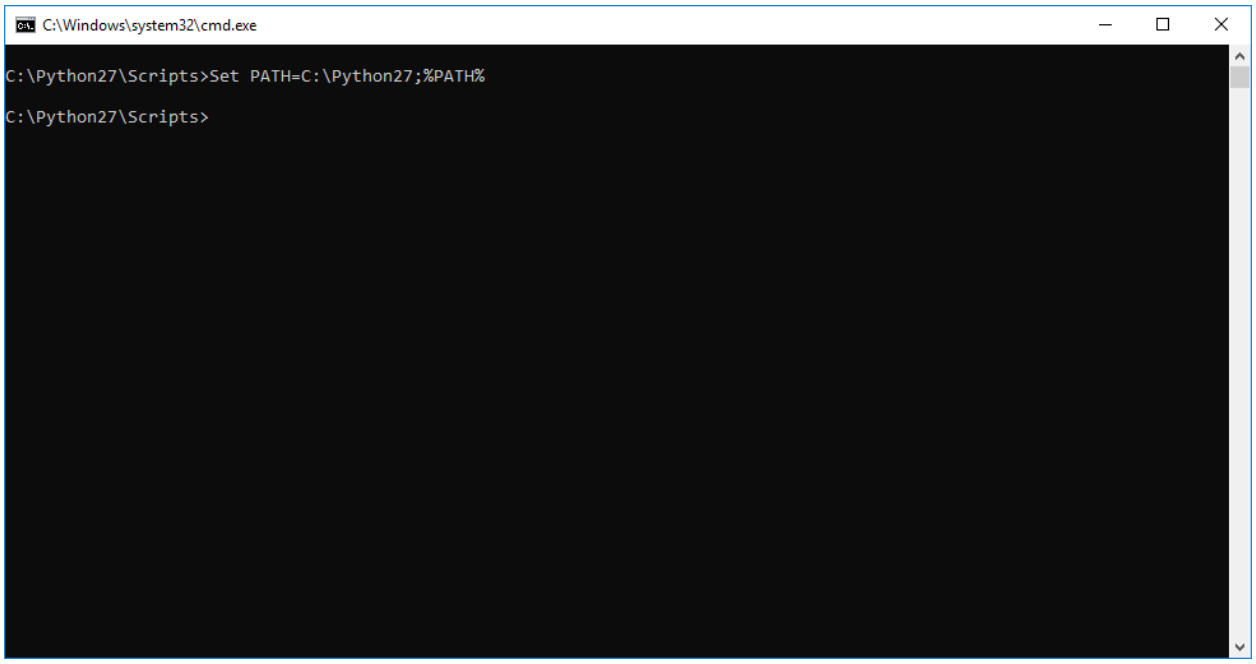
Note that you'll need to be online for this to succeed.

1.f: Set the PATH to include Python

When you try to run the .py script your system needs to be able to find the Python interpreter you just installed.

In your CMD window:

```
Set PATH=C:\Python27;%PATH%
```



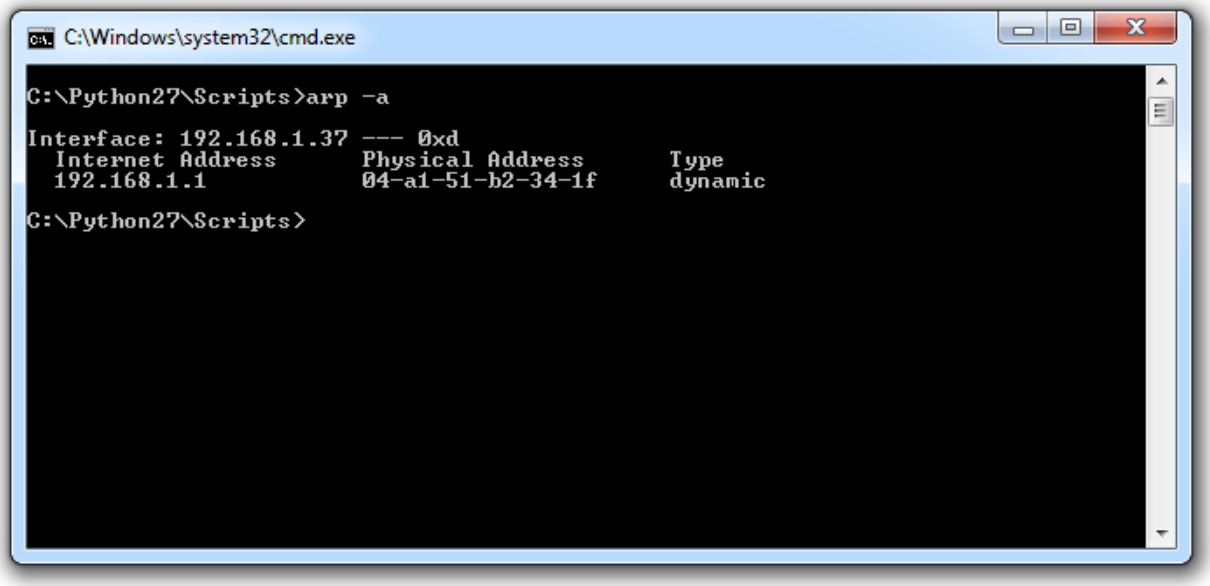
Once you've done that, you ought to be able to change into any other directory and still run python, so long as you don't close the console window.

1.g: Get the IP address and MAC address of your router

You are going to need the IP address of your router, it's MAC address, the admin username (which should be "admin") and the admin password.

Let's start with the IP/MAC. In the CMD window enter:

```
arp -a
```



You'll probably see several rows of output. Note the row which has an internet address of 192.168.1.1. That row should represent your router.

You're going to need to take the Physical Address (the MAC address) and then:

1. Remove the hyphens
2. Make all the characters upper case.

E.g. if your Physical Address shows "04-a1-51-b2-34-1f" thenn after adaptation it will be "04A151B2341F".

1.h: Enable telnet

Change directory to wherever you downloaded `telnetenable.py` to. For example, if you copied it into `C:\Python27`, you could type:

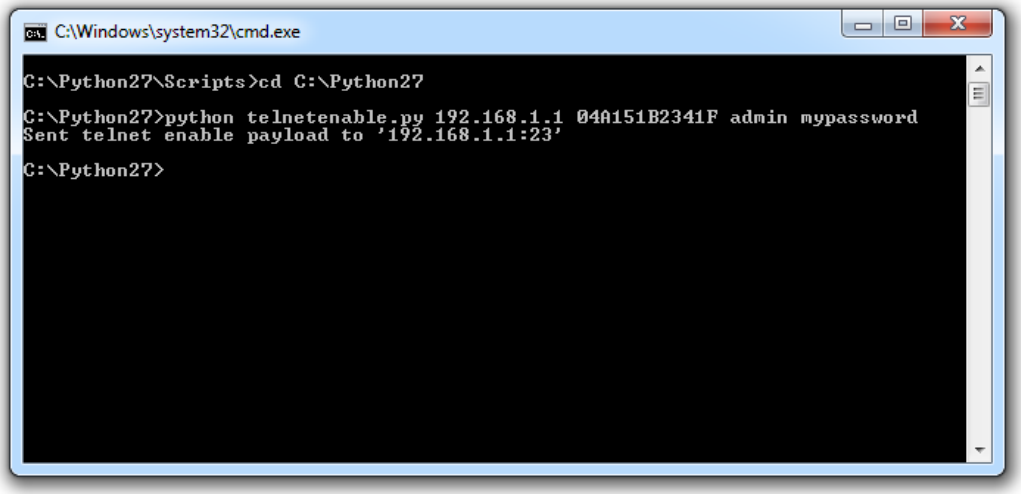
```
cd C:\Python27
```

Then you need to run the script like this:

```
python telnetenable.py <router ip> <router mac> <router username>
                       <router password>
```

e.g.:

```
python telnetenable.py 192.168.1.1 04A151B2341F admin mypassword
```



With any luck, you should see “Sent telnet enable payload to”, and this is a good sign that the telnet server is up and its port is now open for local clients to connect to.

Rebooting the router will disable telnet again, so you will need to run this python script again after any reboots.

Security warning: So long as you don't reboot the router a terminal is available on the LAN side of the router that allows manipulating all of the router’s settings with no further authentication required. You don't want to leave the router like this permanently, nor when any untrustworthy clients are connected to the local network. Always remember to reboot the router from the web interface after you're done.

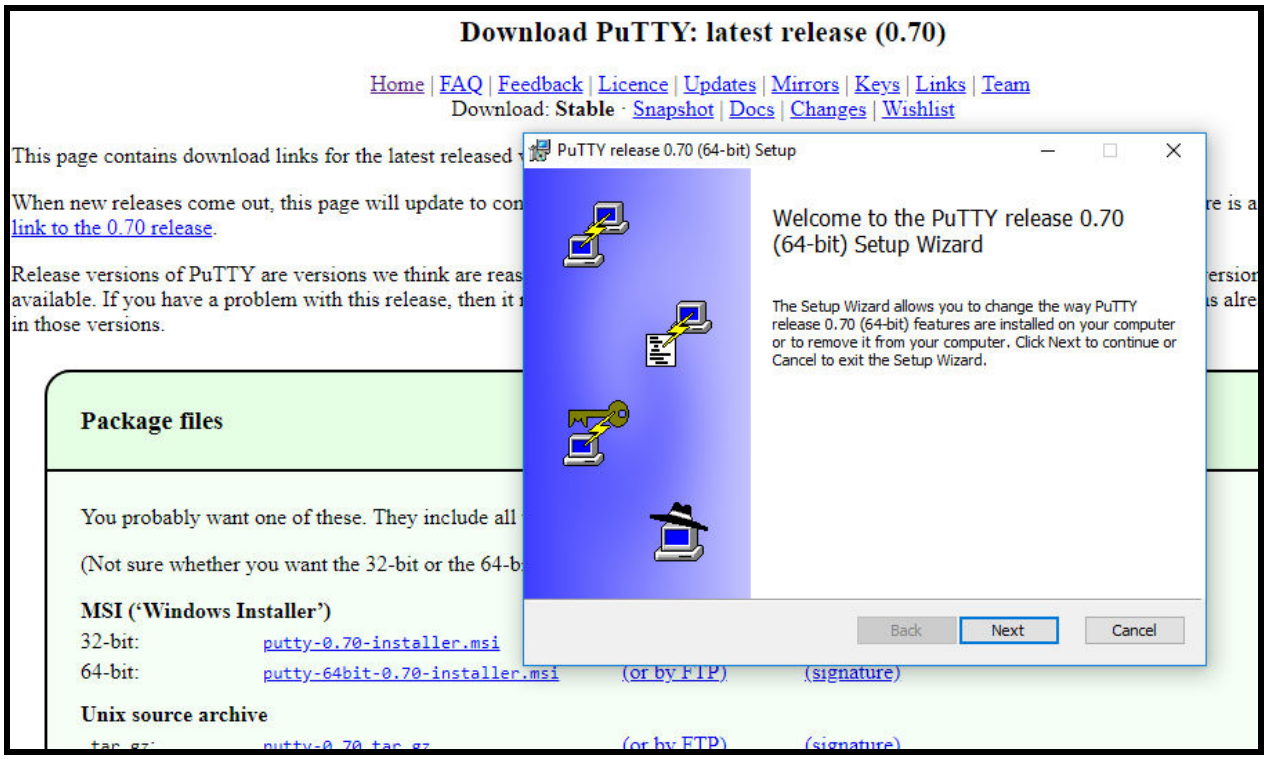
Step 2 - Telnet into the router

In Step 1 you made the router's telnet service accessible. Now you need a telnet client to connect to it so you can start issuing some commands.

2.a: Install Putty

PuTTY is one of the most popular telnet clients on Windows.

Download Putty and install it:
https://www.chiark.greenend.org.uk/~sgtatham/putty/latest.html



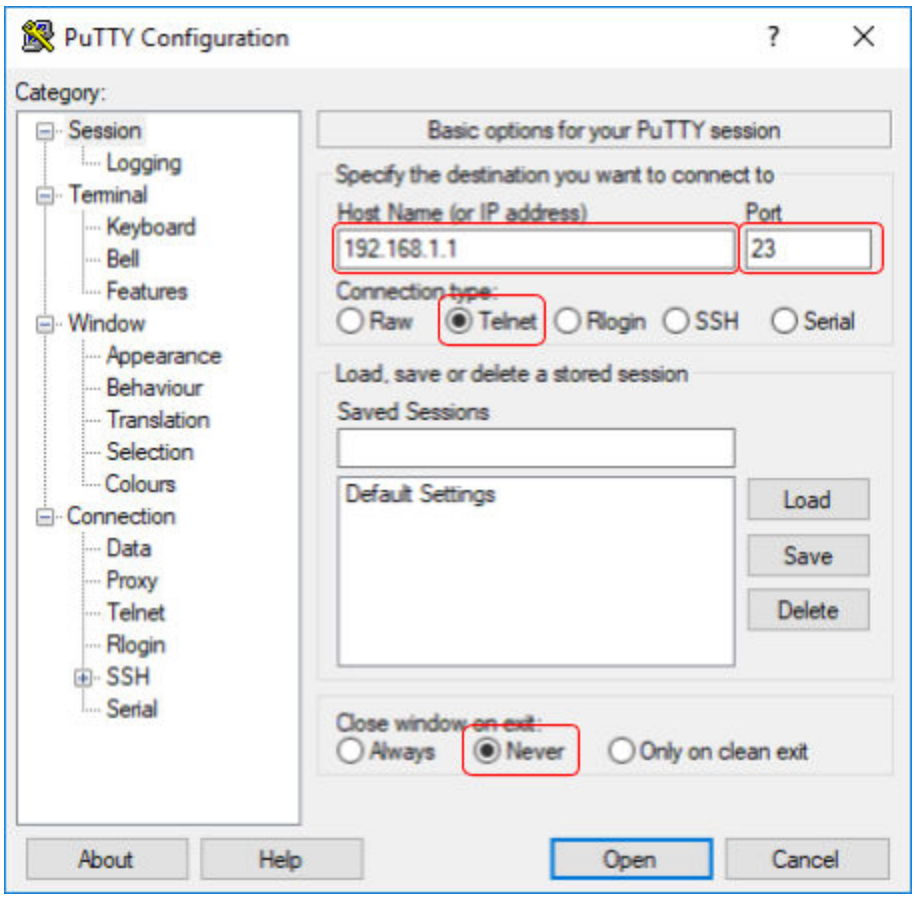
The default options should be fine for everyone. Most people will want the 64-bit installer.

2.b: Connect to the router

Run PuTTY. You'll be greeted with a configuration window.

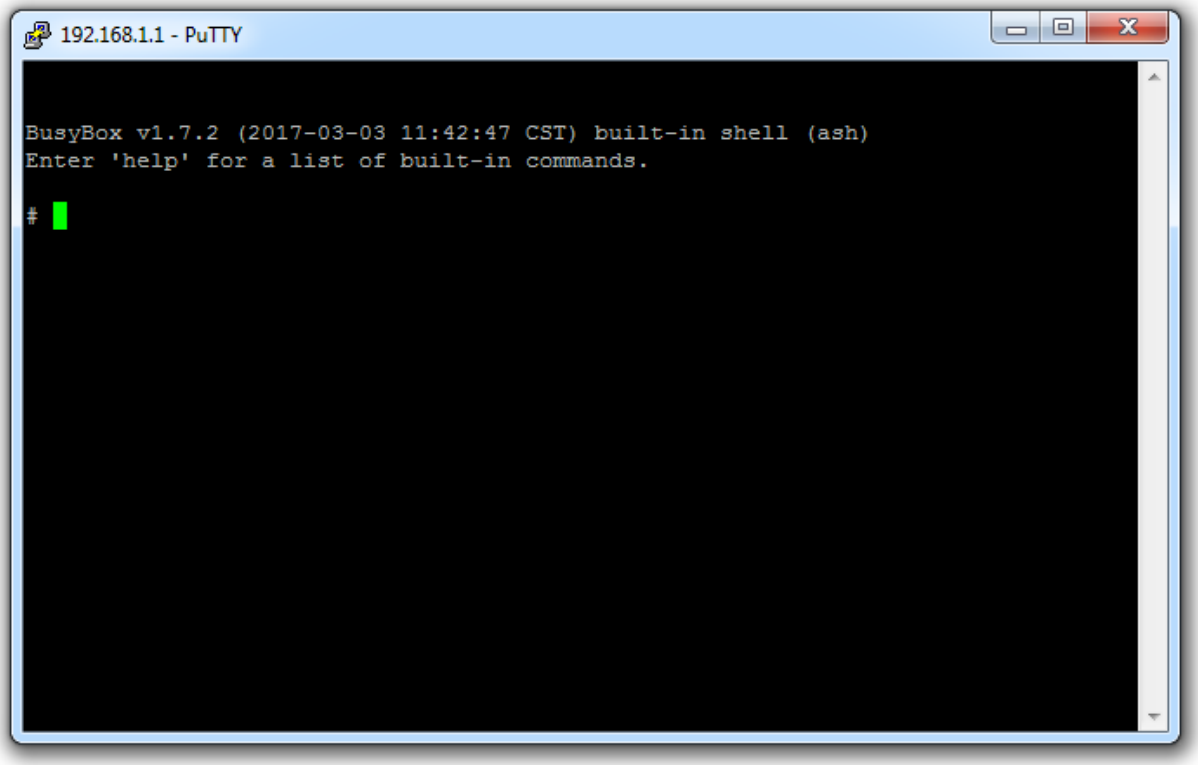
Set it as follows:

Host: `192.168.1.1`
Port: `22`
Connection type: `Telnet`
Close window on exit: `Never`



Setting *Close window on exit* to *Never* is handy because it lets you see the last output from the terminal and also reconnect if needed from the window menu.

Click *Open* and you should find yourself looking at a BusyBox console from the router:



Warning: Commands you run in this terminal operate directly on your router. This means that it is possible to change settings, alter the file system, and possible turn your device into a brick. If you aren't sure what you're doing then don't mess around.

Step 3 - Make some new VPN keys

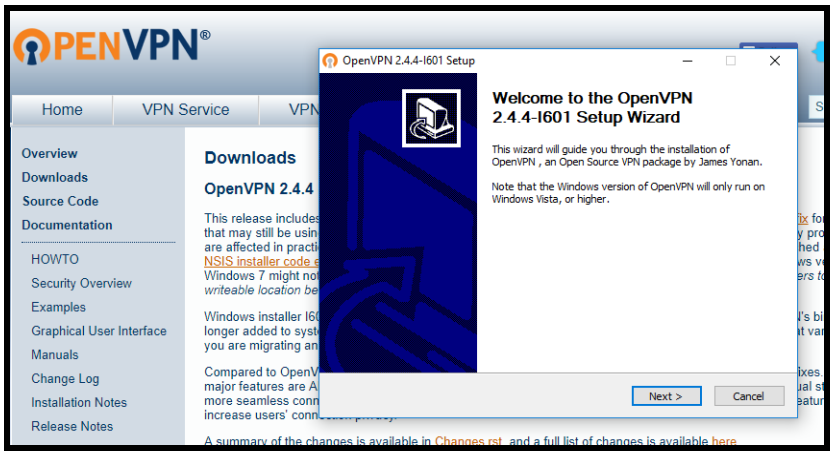
In Step 2 you opened a terminal to the router. Our ultimate goal is to set some new VPN keys, but first we need to generate some. To say that creating these can be a complex process is one hell of an understatement, but fortunately some scripts and tools that come bundled with OpenVPN make key generation significantly easier than it would be otherwise.

This step is largely based upon an OpenVPN HowTo that you can go and read if you want more detail, but I've tried to simplify it so that you don't have to. The HowTo can be viewed at: https://openvpn.net/index.php/open-source/documentation/howto.html

3.a: Get the OpenVPN tools

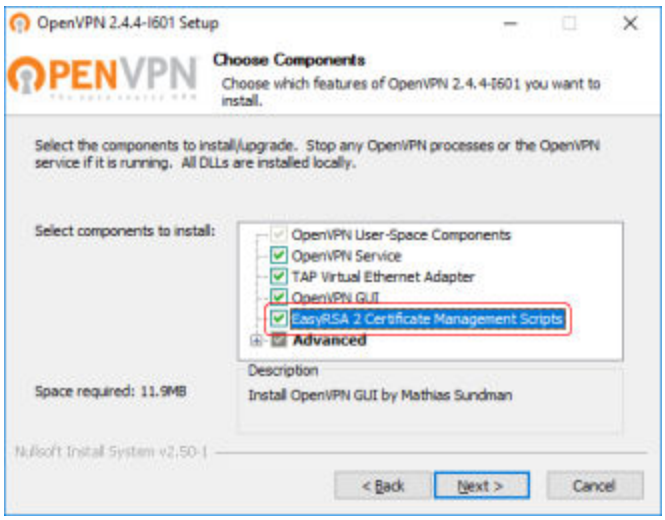
Download and install OpenVPN:
https://openvpn.net/index.php/open-source/downloads.html

Be aware that although it is generally best to use the latest version of OpenVPN, this document was written against OpenVPN version 2.4.4. Later steps in this guide walk through editing some of the scripts that shipped with version 2.4.4. If you install a different version of OpenVPN you might have problems following the instructions.



During the installation, on the components page:

1. You need to manually enable "EasyRSA 2 Certificate Management Scripts"
2. You don't have to install the service/tap driver to generate the new certificates, but you might want them so you can VPN into the router from this computer.

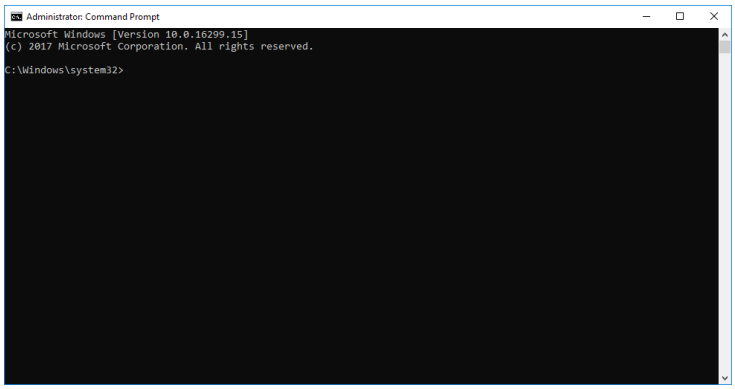


3.b: Prepare to generate keys

You will need a UAC elevated/Admin command prompt to continue. This is because OpenVPN installs under `Program Files`, a protected location.

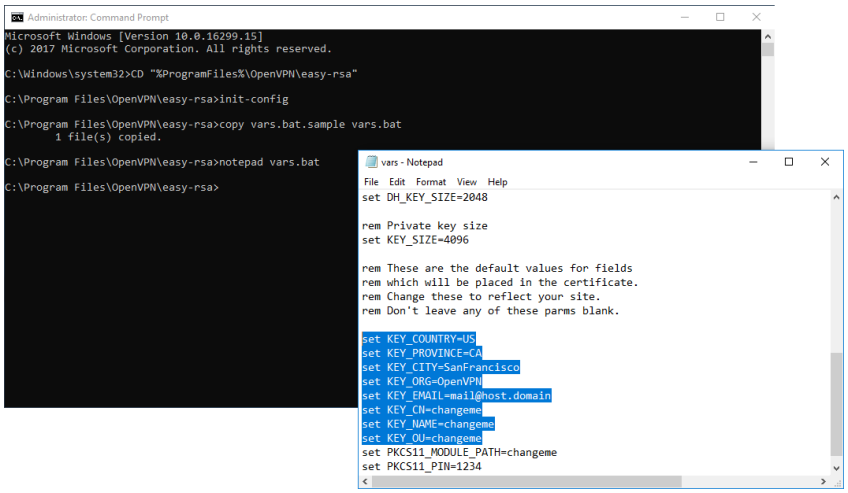
Open an Administrator command prompt by clicking *Start* (or pressing the Windows key), typing "cmd", and then when *Command Prompt* appears in the list of choices right-click it to see the context menu and choose *Run as Administrator*.

If you did that correctly, the window title should begin “Administrator:”



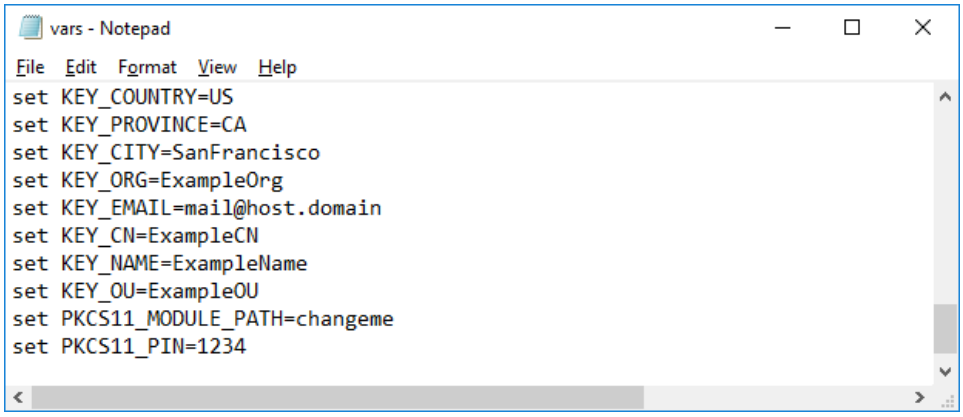
Enter each of these lines:

```
CD "%ProgramFiles%\OpenVPN\easy-rsa"
init-config
notepad vars.bat
```



Towards the bottom of the file there are some lines beginning with the command "set" that set the default key properties. Fill in the ones starting with “KEY_” with something sensible. Most of the fields aren't important and you can write what you like because you're basically filling out your own certificate. You do need to fill in KEY_CN (Common Name).

For example, these are perfectly valid, albeit not very creative settings that will work just fine:



You may notice that "DH_KEY_SIZE" is set to 2048. I recommend against increasing this value unless you are willing to wait half a day for your DH param to be calculated (I’m not kidding).

Save your changes and close Notepad to return to the command prompt.

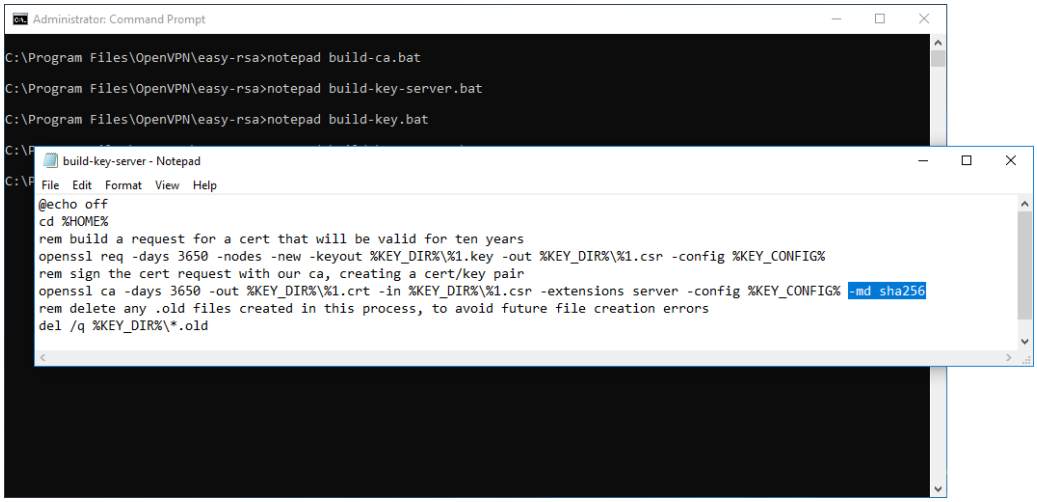
3.c: Using stronger digest algorithms

If you want, you can use stronger digest algorithms, like SHA256 instead of the default, SHA1. It is possible that SHA1 might be more compatible with some clients and possibly also with older firmware versions. If you just want to accept the defaults proceed to Step 3.d now.

Enter:
`notepad build-ca.bat`

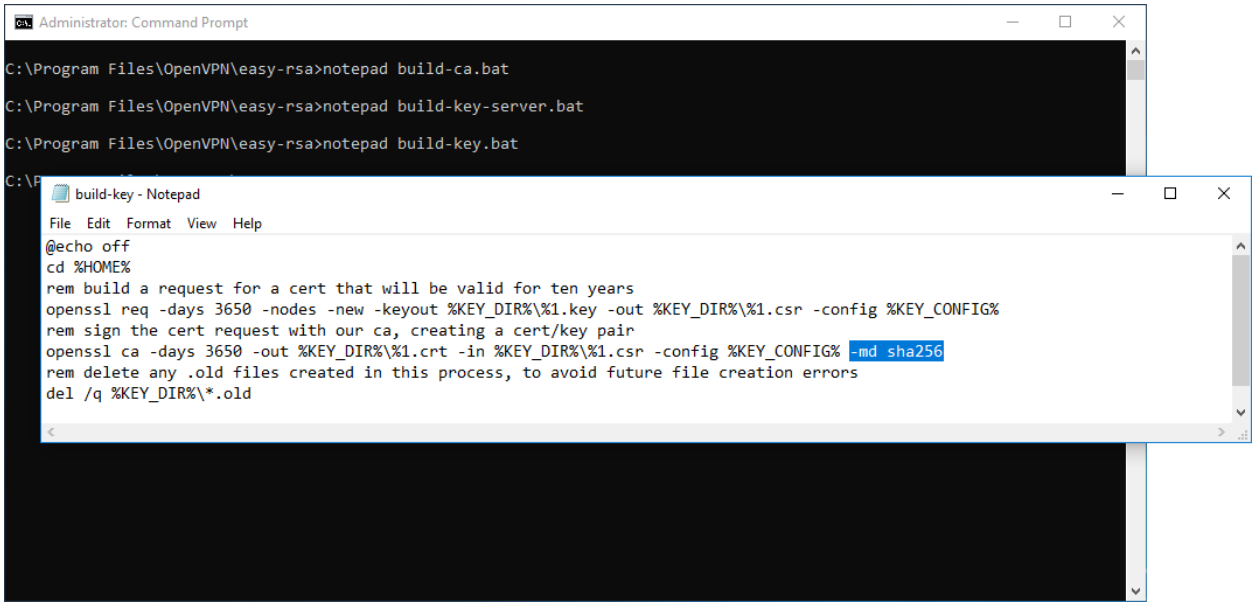
Find the end of the line where it says "`-config %KEY_CONFIG%`" and add "` -sha256`" (without the quotes), then save the file.

Enter:
`notepad build-key-server.bat`



Find the end of the line that starts "`openssl ca `" and add "` -md sha256`" (without the quotes), then save the file.

Enter:
`notepad build-key.bat`

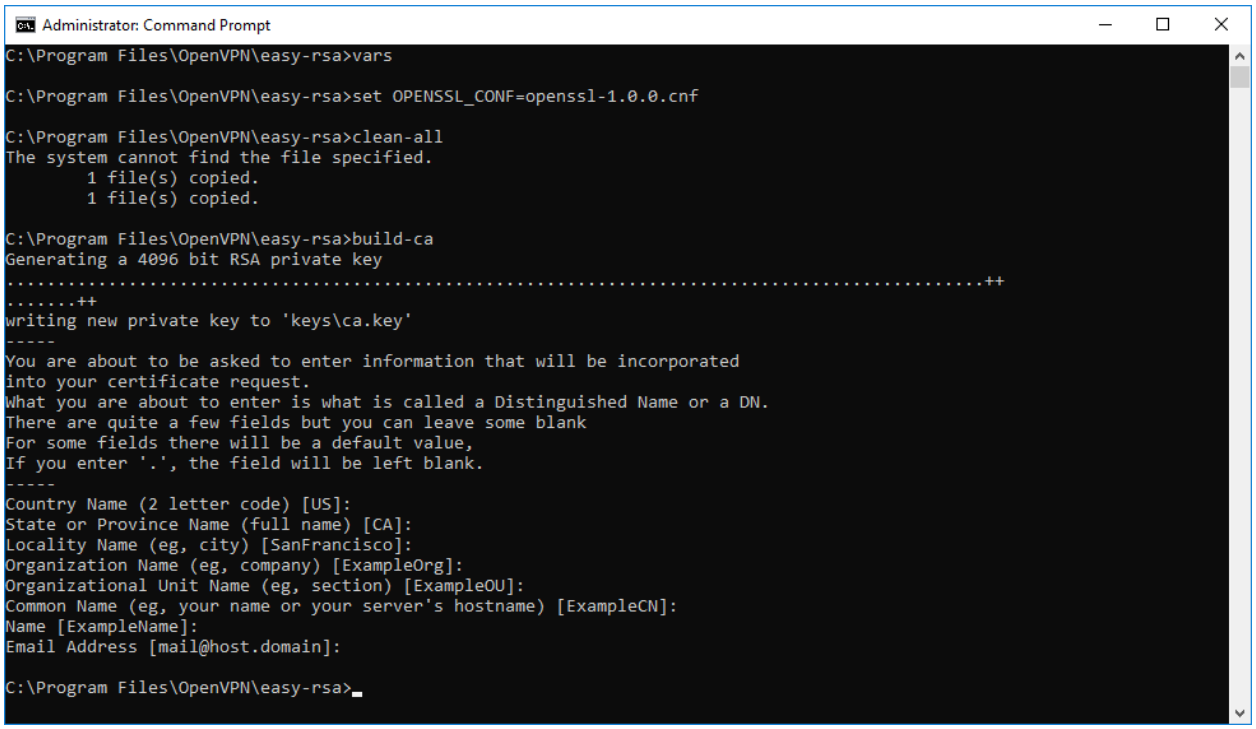


Find the end of the line that starts "`openssl ca `" and add "`-md sha256`" (without the quotes), then save the file.

3.d: Generate keys

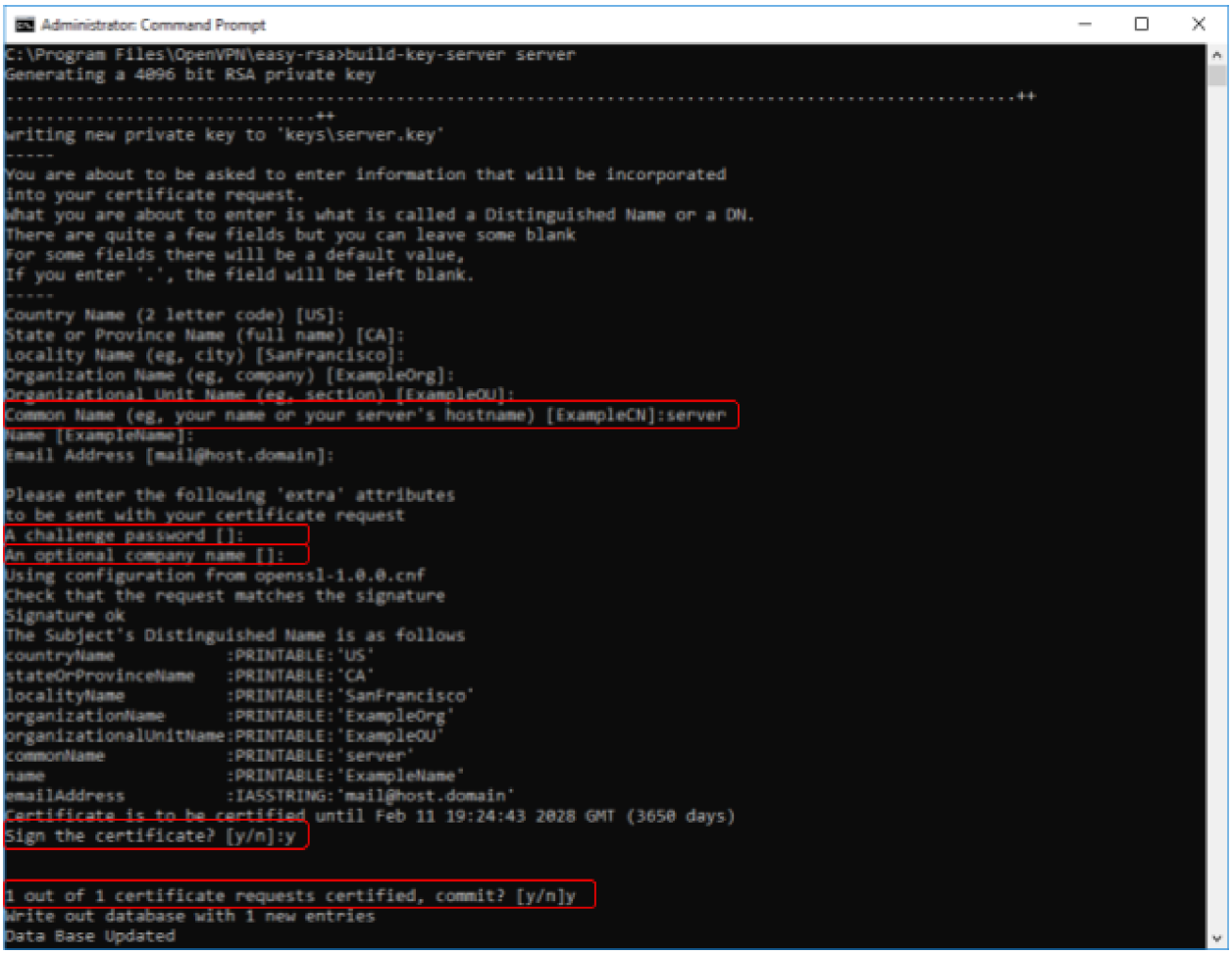
In the command prompt, enter each of these lines:

```
vars
set OPENSSL_CONF=openssl-1.0.0.cnf
clean-all
build-ca
```



Press enter at each prompt to accept the proposed answer, which should be the values you customized earlier during Step 3.b.

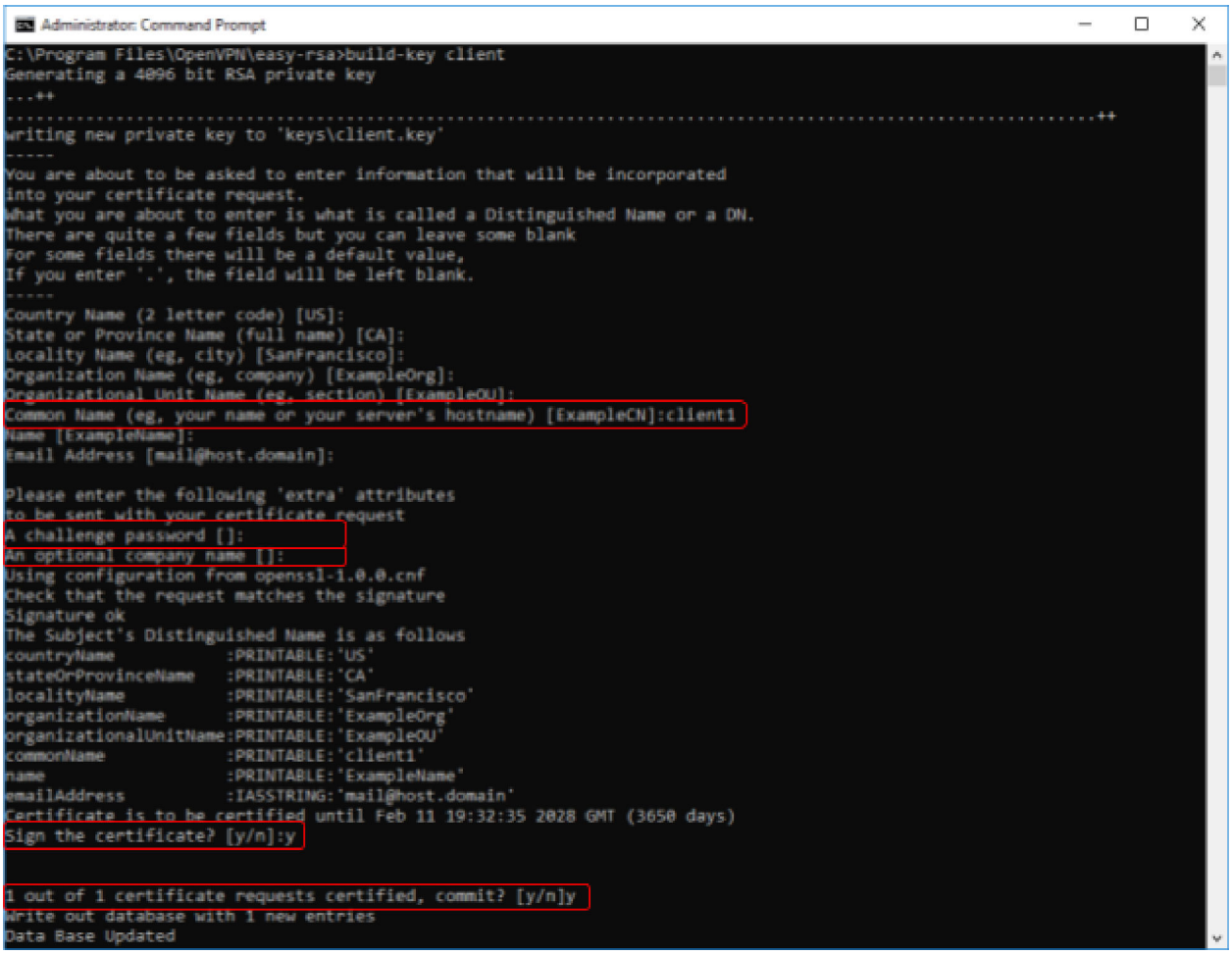
Now enter:
`build-key-server server`



Follow these steps:

1. When asked for "Common Name", enter "server"
2. Don't enter a challenge password
3. Don’t enter an optional company name
4. Choose to sign the certificate
5. Choose to commit

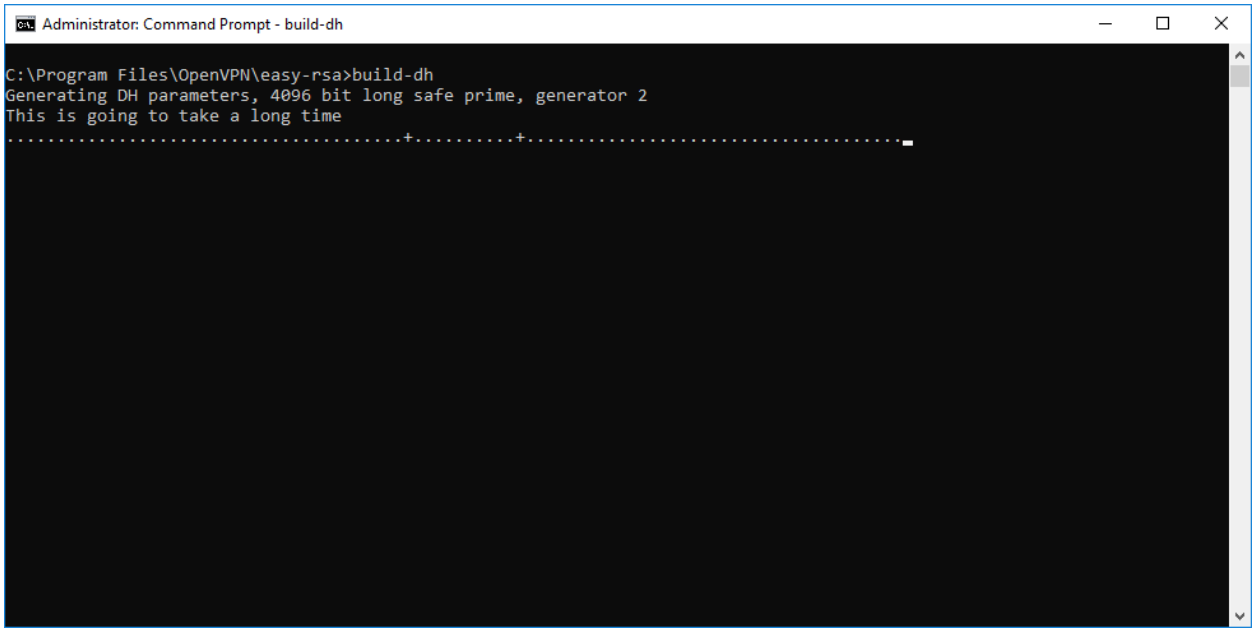
Now enter:
`build-key client`



Follow these steps:

1. When asked for “Common Name”, enter "client1"
2. Don't enter a challenge password
3. Don’t enter an optional company name
4. Choose to sign the certificate
5. Choose to commit

Now enter:
`build-dh`



Go and make a cup of tea. Generating the dh parameters takes a long time.

3.e: Prepare the keys for the router

Now that we have our new certificates and keys, we want to prepare them for the router. There's a couple of significant considerations here:

1. We have likely generated stronger keys than the router ships with (at least the unit I have), and some of the output files are named as such (e.g. "dh4096.pem"). If we are clever and align the filenames with what the router expects then everything should just work without having to change the OpenVPN configs on the router, and we should still be able to download our new OpenVPN keys through the router's web interface just like before, which is a nice touch.

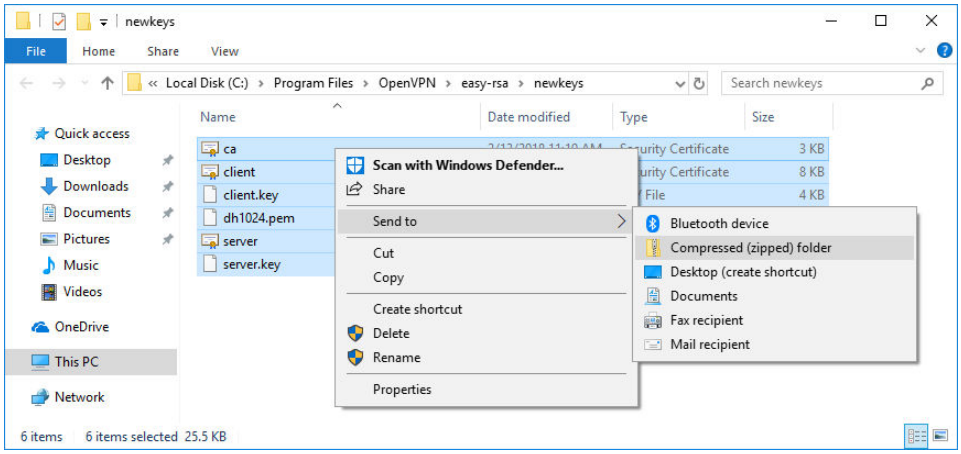
   IMPORTANT: The commands below will rename dh4096.pem to dh1024.pem, because on the unit I have, that's what the OpenVPN configuration files expect. Later, during Step 5, you'll list the files that are actually on your own unit. If you don't see dh1024.pem on your own device, then STOP. Unfortunately I can't predict all the variations that might exist in the field.

2. Transferring all of the new files to the router one by one is going to be a chore. Instead, we can create an archive containing all the files and perform only a single transfer. If we structure the archive carefully we can also make it easier to deploy.

Enter these commands to make a new folder and move into it the files we need to transfer, renaming them in the process:

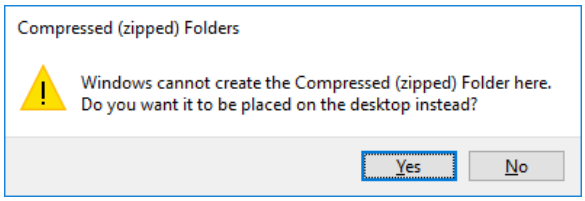
```
md newkeys
copy /Y keys\ca.crt newkeys
copy /Y keys\client.crt newkeys
copy /Y keys\client.key newkeys
copy /Y keys\server.crt newkeys
copy /Y keys\server.key newkeys
copy /Y keys\dh4096.pem newkeys\dh1024.pem
start newkeys
```

Now, using Explorer, make a selection of all the files in the "newkeys" folder then right-click to open the context menu, and choose "Send to" > "Compressed (zipped) folder".



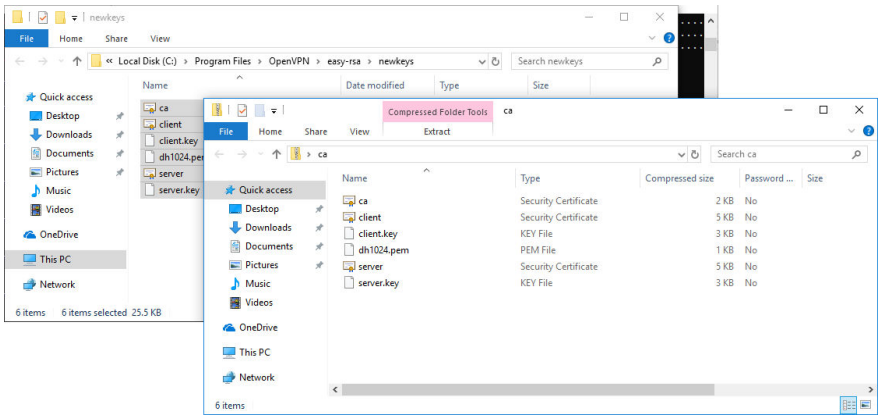
Using this exact technique should produce a standard format .zip file that contains all of the files at the root level, i.e. when you open the .zip file you don't have to click through any other folders before seeing the files. This is very important to ensure the files deploy correctly later.

Windows is likely to complain that it can't make the .zip file in the same folder, and ask if you want to create it on the Desktop instead.



Answer "Yes", then go to your Desktop and make sure you can locate the output file. Windows usually names it after the first file in the selection.

Check the .zip file contents match the "newkeys" folder:



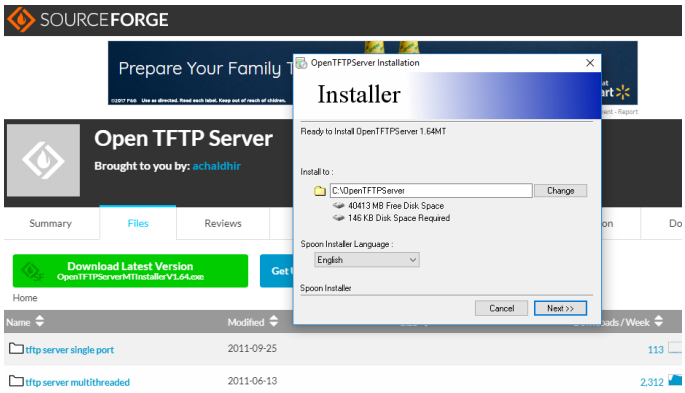
Step 4 - Prepare to transfer files using TFTP

In Step 3 you generated the new VPN keys and certificates for your router and archived them into a .zip file. The next step is to transfer that .zip file to the router. To do that, we'll have to use some of the tools available on the router, and that means our options are limited.

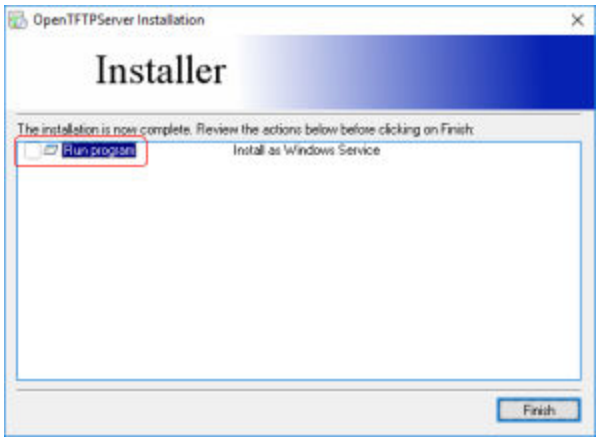
One tool I found to be available on the router was TFTP, which is a very simple file transfer protocol.

4.a: Install a TFTP server

You will need to install a TFTP server for the router to connect to. I chose to use Open TFTP Server, which you can download here:
https://sourceforge.net/projects/tftp-server/



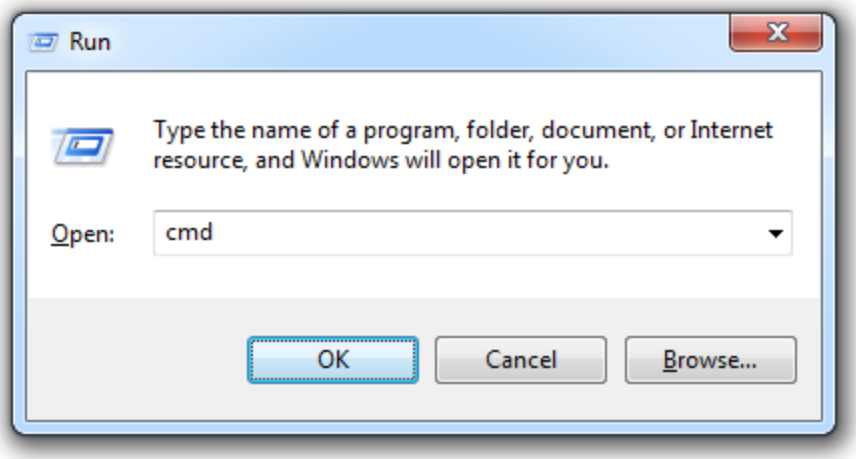
I recommend that at the end of installation you uncheck the box to "Install as a Windows Service", because you almost certainly do not want a TFTP server running at all times.



4.b: Create a TFTP transfer folder

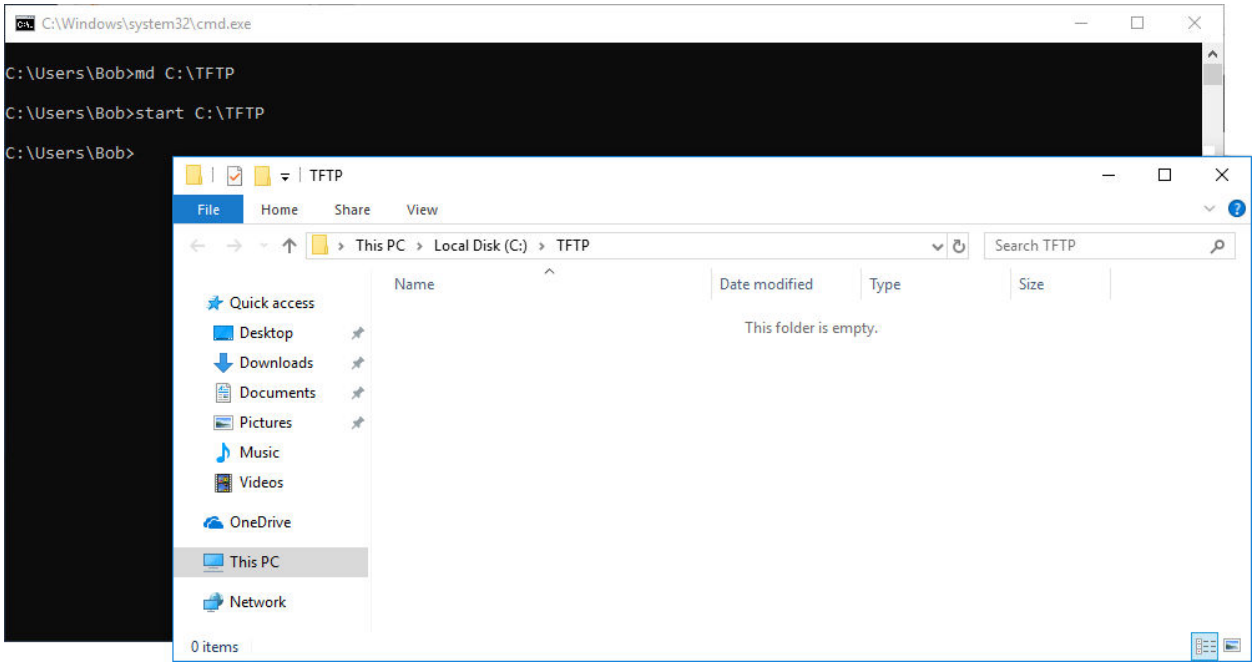
We need to make a folder to transfer files to/from.

Start a new command prompt by hitting Win+R and entering “cmd”:

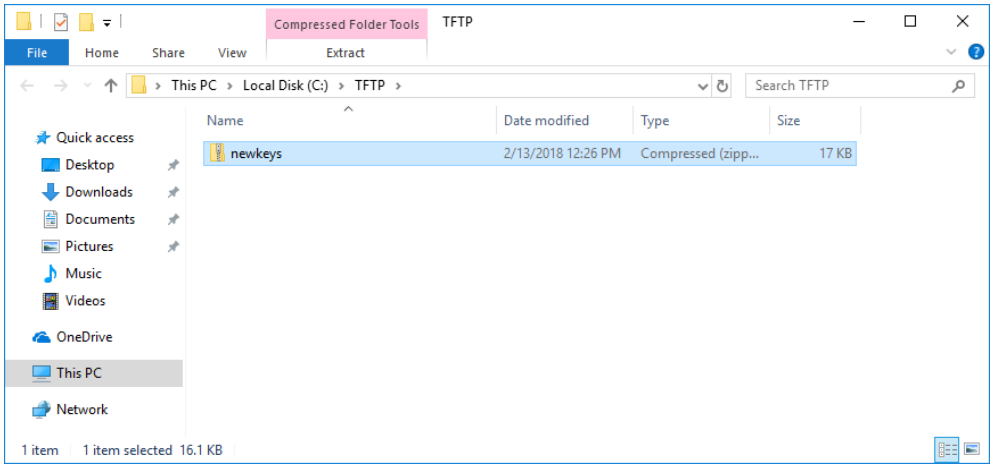


Now enter:

```
md C:\TFTP
start C:\TFTP
```



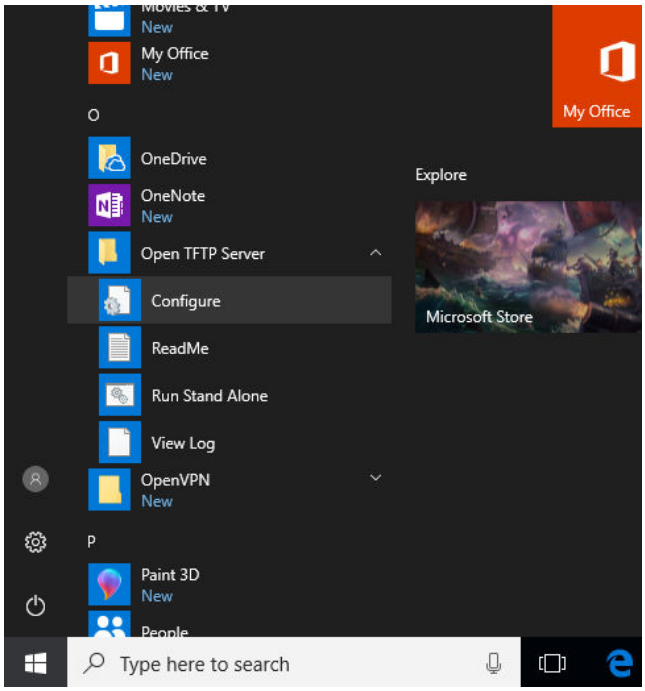
Now find the .zip file you created during Step 3 and copy it into this new “TFTP” folder. Rename the file to “newkeys” (or “newkeys.zip” if you have Windows set to show known file extensions):



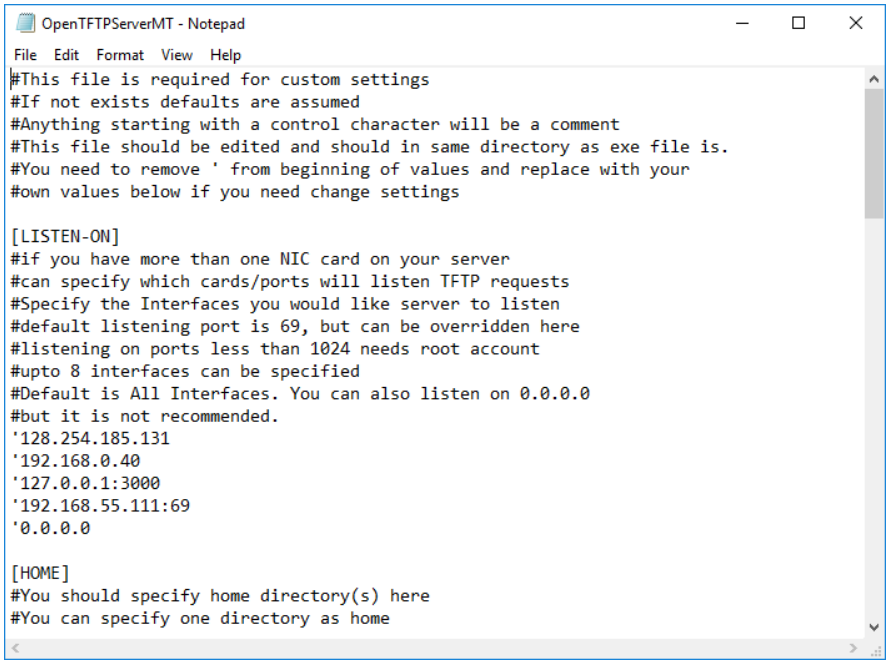
4.c: Configure the TFTP server

Before you can transfer any files using the TFTP server, you must alter its configuration file. The default settings don’t have the permissions we need to transfer files to the router.

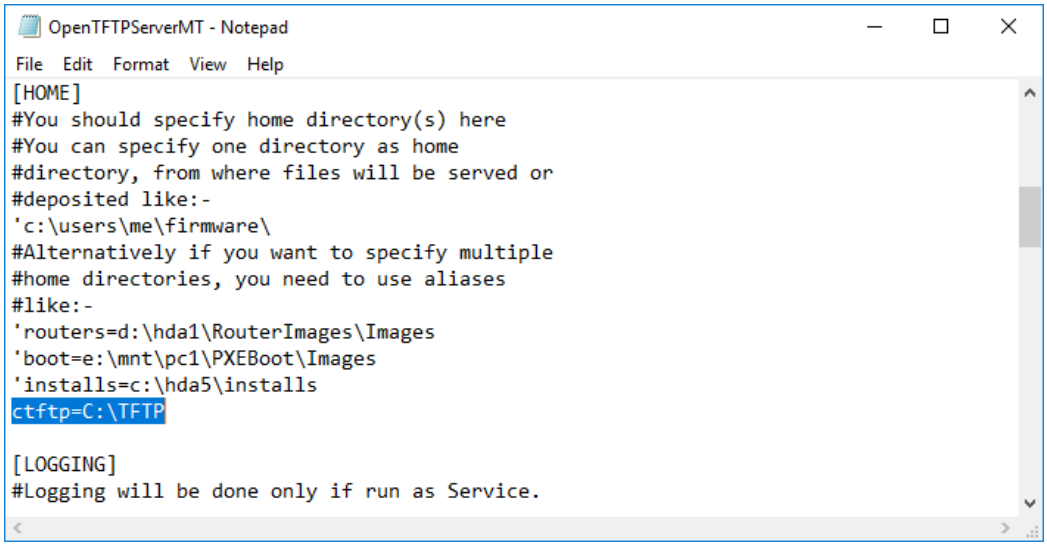
Open the “Configure” link from the “Open TFTP Server” folder on your Start menu.



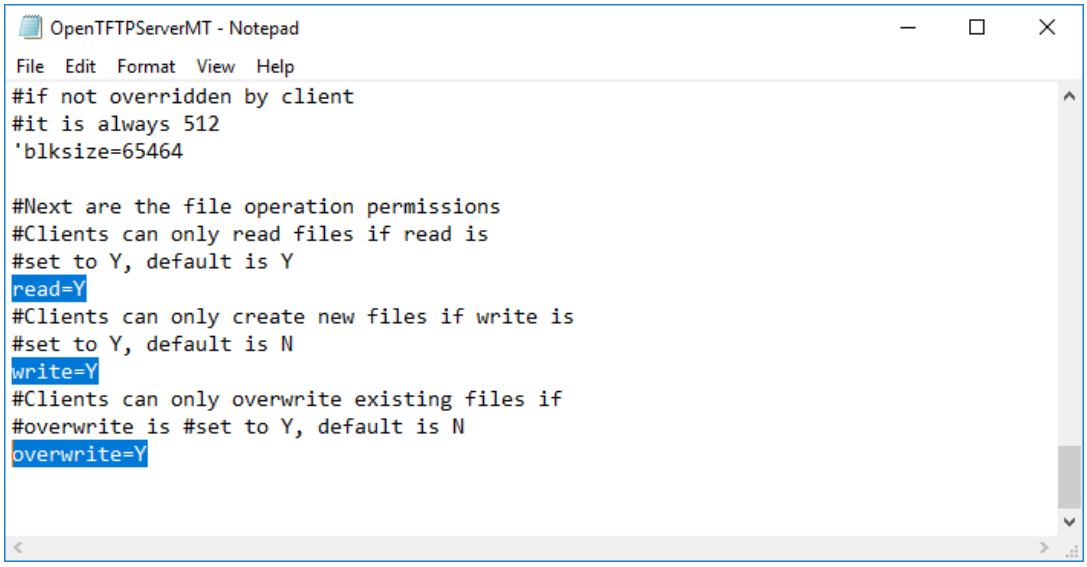
The stock configuration file should open in Notepad (or a suitable plain text editor):



In the [HOME] section, add an entry that maps a new home directory called “ctftp” to the folder we created, C:\TFTP:



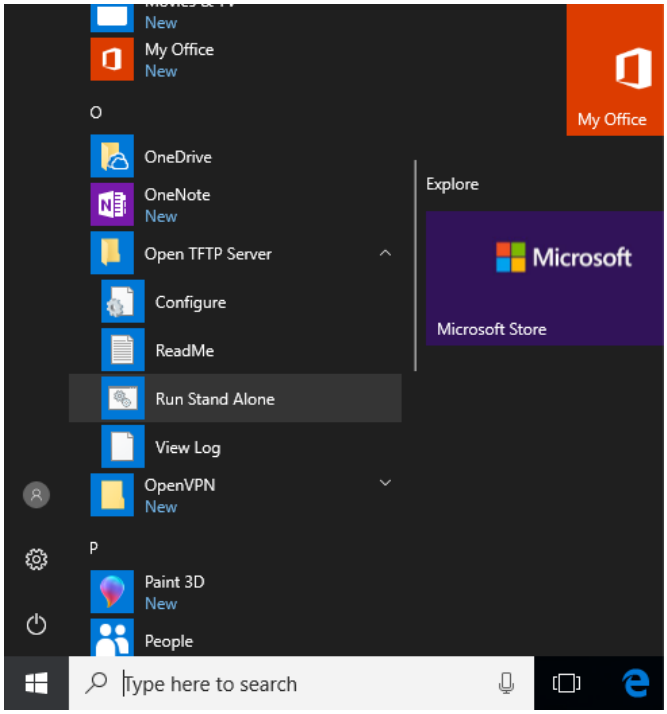
Scroll down to the end of the configuration file and turn on the settings for “read”, “write”, and “overwrite” by removing the apostrophe from the start of the line, and setting them to “Y”:



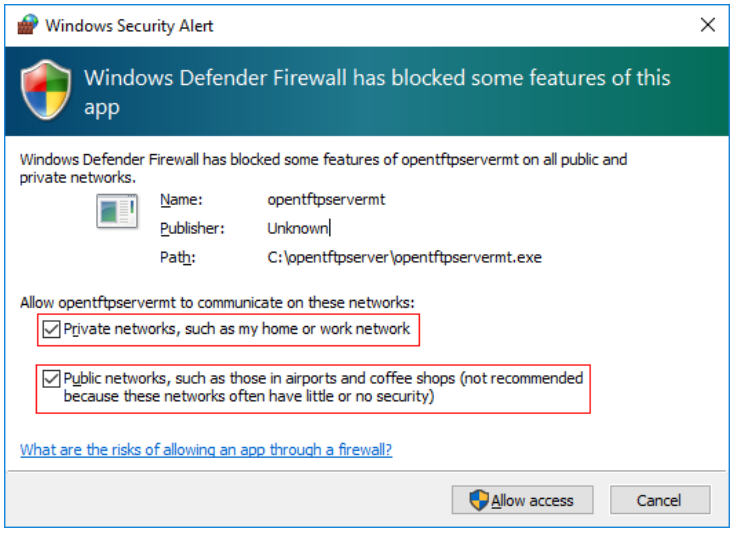
Save the configuration file.

4.d. Start the TFTP server

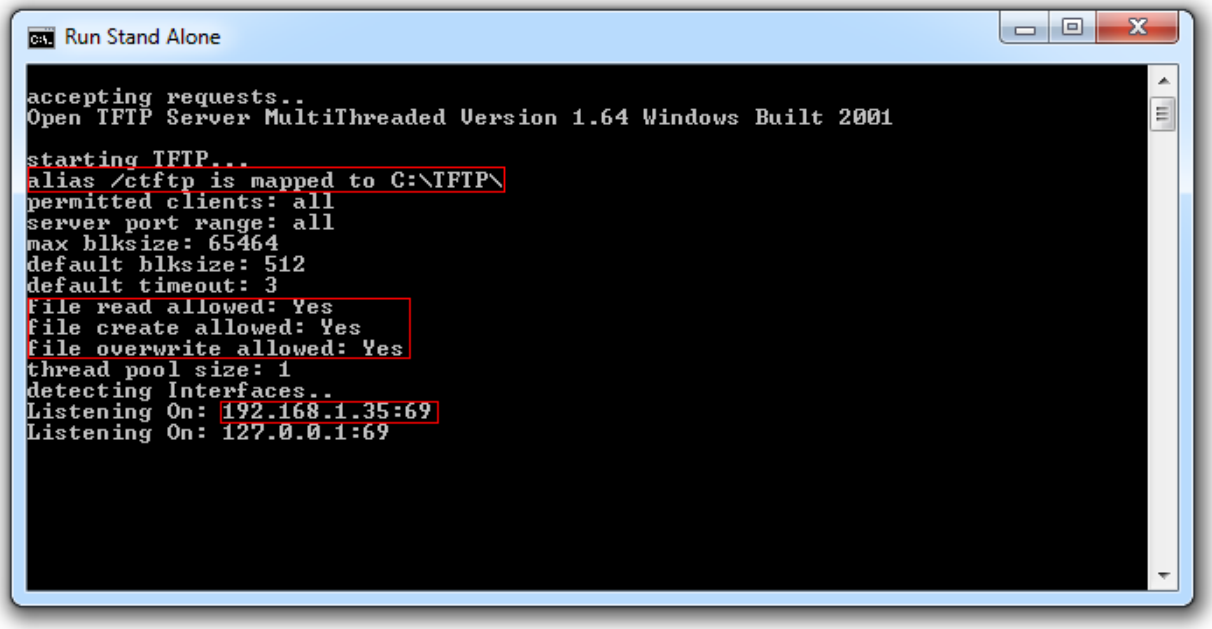
Launch the TFTP server by choosing “Run Stand Alone” from the Start menu.



You will need to approve the server to receive connections through Windows Firewall. If you aren’t sure whether Windows is treating your home network as a public or private one then checking both boxes is the easiest way to make the connection work.



The TFTP client should start, and confirm the settings you applied:



Note that it also lists your LAN IP address, which will normally be in the range 192.168.1.xxx. Make a note of this address. Ignore the ":69", which is the default port for TFTP. For example, as pictured, my system is on address 192.168.1.35. This is the address we will later need to give to the tftp client that runs on the router.

If you receive warnings that the port could not be bound, you might have accidentally installed the TFTP server as a service, and it might already be running. In this case I would recommend uninstalling the service and working through Step 4 again, being careful not to install as a service this time around.

Step 5 - Backup your current VPN keys

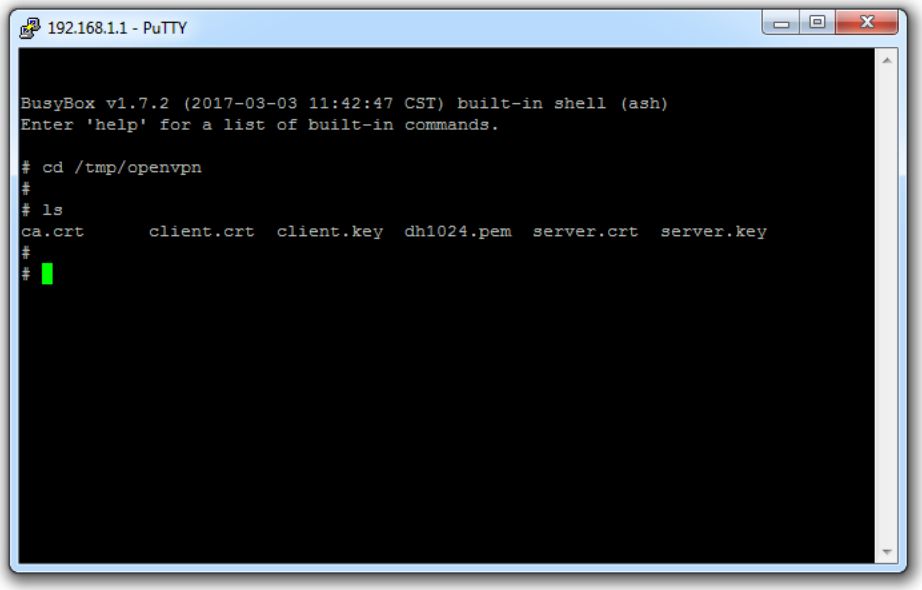
Now we have our new keys, and a server for transferring files, we're almost ready to go. Before we do, let's make sure we have a backup of our current VPN keys. You'll need the IP address of your system running the TFTP server from Step 4, and you'll need the telnet session to your router from Step 2.

WARNING: You will be entering commands that modify the filesystem of your router. Don't mess around.

5.a: Change directory to find the existing VPN keys

At the terminal for the router, enter:

```
cd /tmp/openvpn
ls
```



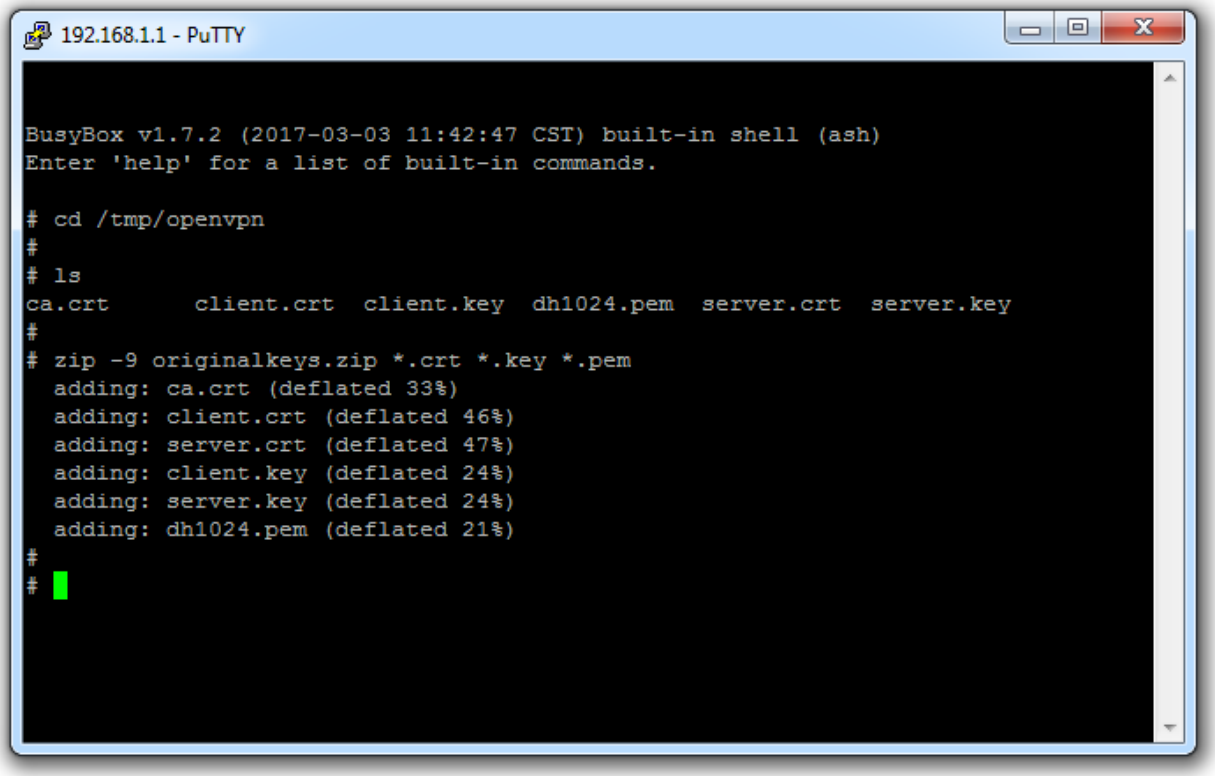
You should see the existing keys listed.

IMPORTANT: During Step 3.e you renamed one of the files for the new keys from "dh4096.pem" to "dh1024.pem". You should see "dh1024.pem" in the file listing created by the "ls" command on your device. If you don't see that exact filename then the files you prepared don't match your device. In this case STOP and do not continue trying to replace the keys.

5.b: Backup the original keys

Zip up the original keys by typing:

```
zip -9 originalkeys.zip *.crt *.key *.pem
```



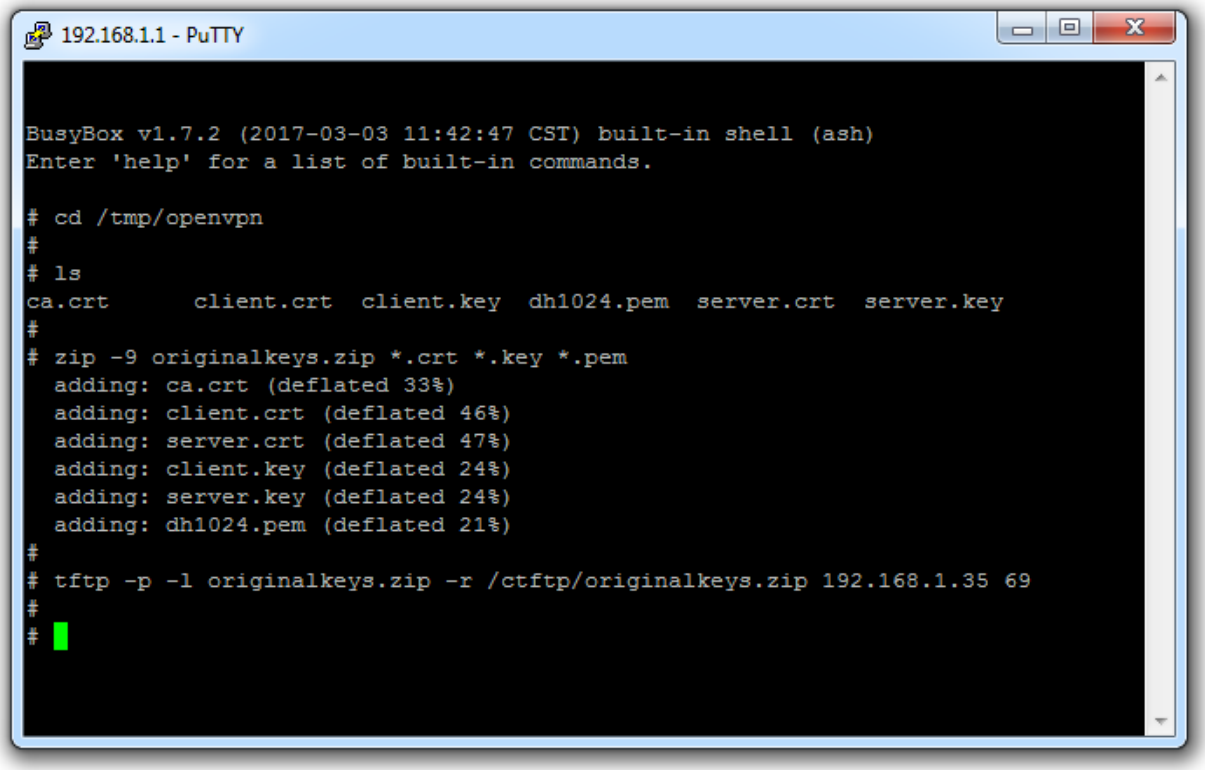
Confirm that all six files are added without error.

Now, with the TFTP server running, transfer the zip file to your server. Remember that this should result in the file being written to the C:\TFTP folder we made earlier. The command will be:

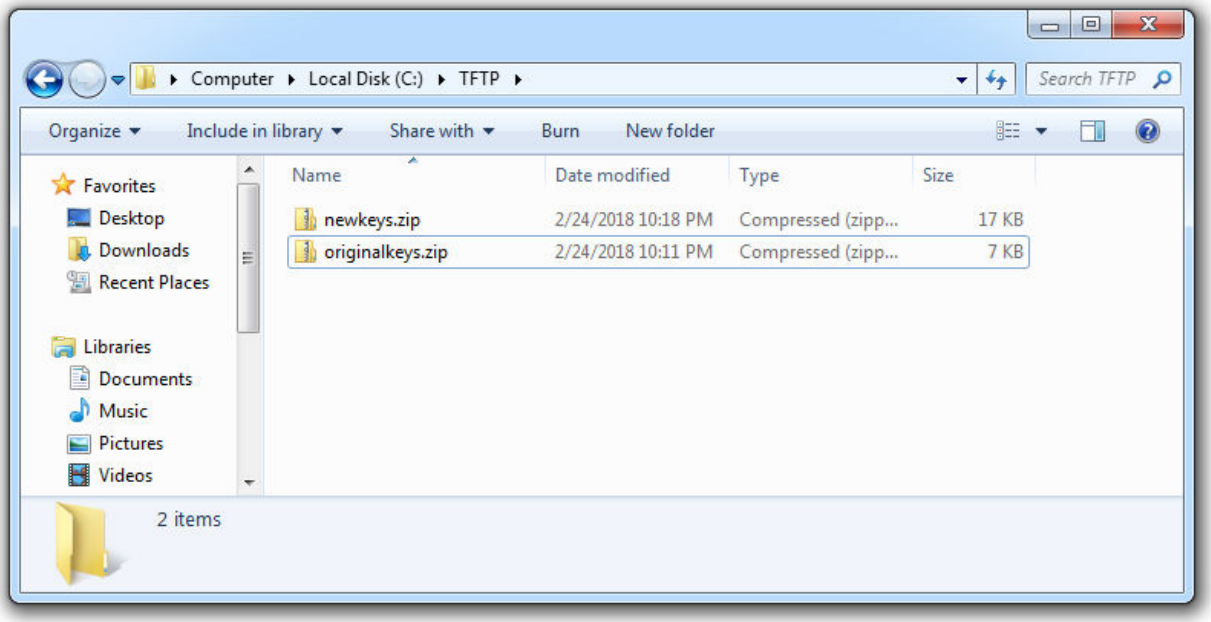
```
tftp -p -l originalkeys.zip -r /ctftp/originalkeys.zip <youriphere> 69
```

e.g. on my system, whose IP is 192.168.1.35, I would enter:

```
tftp -p -l originalkeys.zip -r /ctftp/originalkeys.zip 192.168.1.35 69
```



Check that the file transferred as expected:

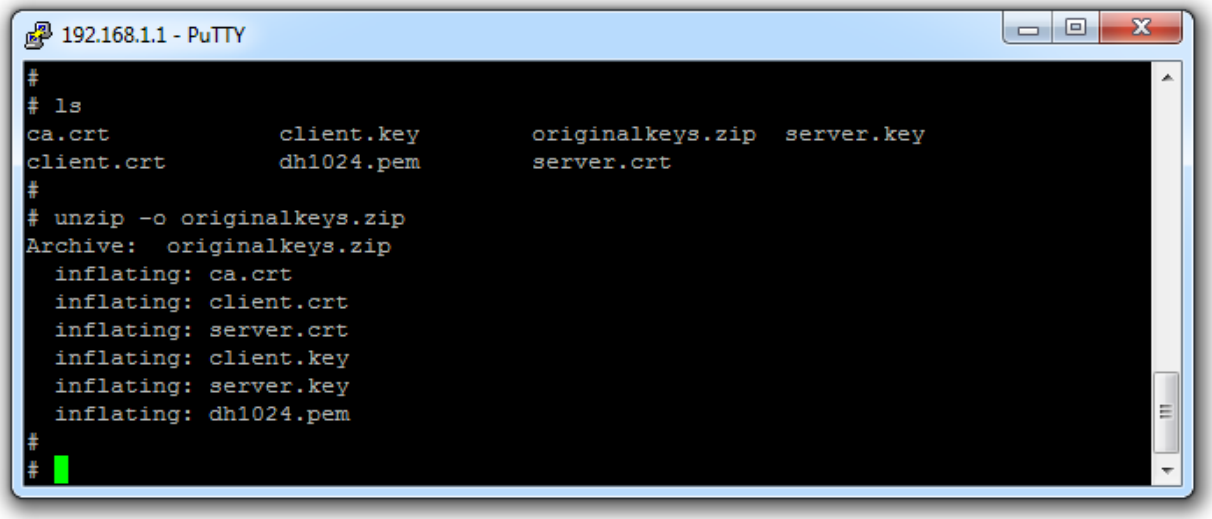


5.c: Test restoring the original keys

The originalkeys.zip file can remain on the router. This makes it easy to restore the original keys if necessary.

Try restoring them with:

```
cd /tmp/openvpn
unzip -o originalkeys.zip
```



Step 6 - Deploy your new VPN keys

The final step is to deploy the new keys to the router.

You'll need the IP address of your system running the TFTP server from Step 4, and you'll need the Telnet session to your router from Step 2.

WARNING:

- You will be entering commands that modify the filesystem of your router. Don't mess around.
- You should have already backed up your original keys during Step 5, and you should know how to restore those keys if needed. **Do not proceed with Step 6 if you did not successfully complete Step 5.**

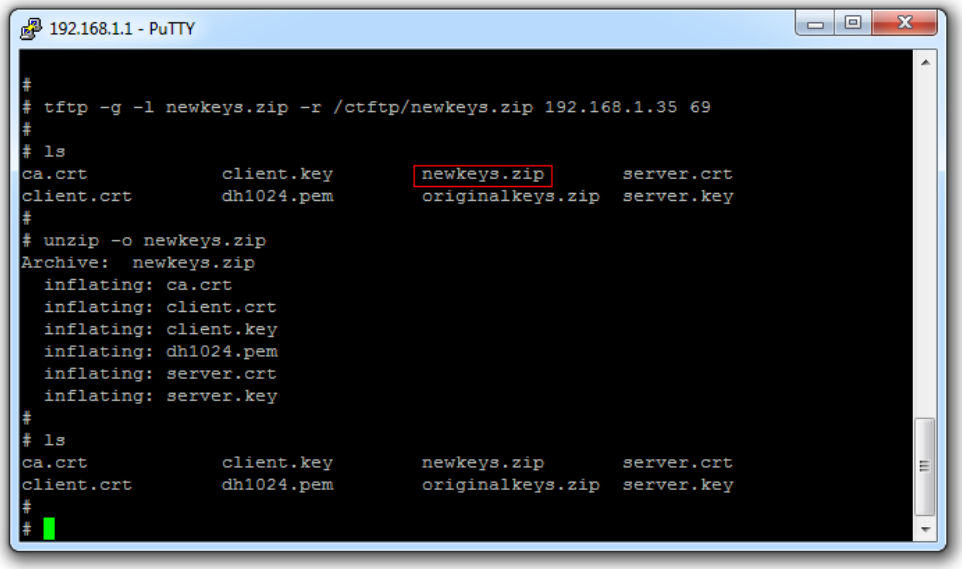
6.a: Transfer the new keys to the router

Use the tftp command to get the new keys from the TFTP server. Assuming that you named the zip file with your new keys "newkeys.zip", the command will be:

`tftp -g -l newkeys.zip -r /ctftp/newkeys.zip <youriphere> 69`

e.g. on my system, whose IP is 192.168.1.35, I would enter:

`tftp -g -l newkeys.zip -r /ctftp/newkeys.zip 192.168.1.35 69`



Then check the zip file was received by typing:

```
ls
```

Unzip the contents to deploy your new keys in place of the old ones:

```
unzip -o newkeys.zip
```

Step 7 - Finish up

7.a: Reboot the router

Only after everything looks okay, reboot the router through the web admin console. This should disable the telnet interface and cause the router to start the VPN service using your new keys.

7.b: Confirm clients with the old keys can no longer connect

It's a good idea to verify that clients with the old keys can no longer connect. This is important if you are replacing the keys because you suspect the older keys were either weak or compromised.

7.c: Set up the new keys for clients

Download the new keys on client devices through the router's web console, exactly as you did the first time.

7.d: Clean up your TFTP transfer folder

During Step 4 you created a TFTP transfer folder, `C:\TFTP`, and copied your new keys into it. During Step 5 you also copied a backup of your old keys to this folder.

I recommend moving these files from the TFTP transfer folder to a different long-term storage location so that there is no risk that your keys remain accessible to any other tftp clients. You could also uninstall the TFTP server if you no longer need it.