SoftEther VPN server on Turris Omnia with l2tp/IPsec

This document shows you how to install a SoftEther VPN server on your Turris Omnia router and get it up and running. I will use a lxc container for the vpn server.

preliminaries

The configuration I am using consists of a Turris Omnia router with a 250GB mSATA SSD installed in it. For the installation of the mSATA I followed the instructions as shown in How to install an mSATA SSD into your Omnia. I created 2 partitions on the mSATA disk, both with ext4 as file system. The first partition, /dev/SDA1, will contain the lxc containers that I create and has mountpoint /srv. The second partition, /dev/SDA2, is being used for NAS and has mount point /mnt/sda2.

Since my computer is a Mac and I will frequently use Terminal to connect to the lxc container, it is important to disable (uncheck) **Set locale environment variables on startup** in preferences of Terminal (tab *Profiles/Advanced*) to avoid error messages when installing packages.

step 1: create a container

Use the manual How to work with LXC containers to create a new lxc container. Choose Ubuntu Xenial (16.04 lts) as template for the container. Choose also a name for the container - I named mine vpn_ubuntu_xenial as it is going to be my vpn server - and click the create button and finally save and apply.

Don’t forget to edit the file /etc/config/lxc-auto to enable automatic startup of the container at boot time (see How to work with LXC containers).

After the creation was successful, open a terminal window and ssh to the router at its ip-address:

```
ssh root@<your router’s ip-address>
```

Your router will ask for a password. It is the same password you use to connect to the LuCI interface of your router. After the login was successful the router replies with:

```
BusyBox v1.23.2 (2016-12-05 17:54:40 CET) built-in shell (ash)

  _______  _    _  _____   _____   _____   _____
 |__   __|| |  | ||  __ \ |  __ \ |_   _| / ____|
    | |   | |  | || |__) || |__) |  | |  | (___
    | |   | |  | ||  _  / |  _  /   | |   \___ \
    | |   | |__| || | \ \ | | \ \  _| |_  ____) |
    |_|    \____/ |_|  \_\|_|  \_\|_____||_____/

root@turris:~#
```

The prompt `root@turris:~#` means that we are in the command shell of the router.

We will now be able to connect to the lxc container we just created.

However first we want to collect some information on the new container, so we enter the command (replace <name of your lxc container> by the name you gave to your container):

```
root@turris:~# lxc-info -n <name of your lxc container>
```

As reply we get something like:

```
Name:           the name of your lxc container
State:          RUNNING
PID:            25486
IP:             ip4 address assigned to your container   <--- write down
IP:             ip6 address assigned to your container
CPU use:        1.72 seconds
Memory use:     11.56 MiB
Link:           veth5FJVAI
 TX bytes:      2.60 KiB
 RX bytes:      712.00 KiB
 Total bytes:   714.60 KiB
```

Write down the ip4 address your router has assigned to your container, we will need it later on.

step 2: connect to the container

In order to connect to the container we give the command (replace <name of your lxc container> by the name you gave to your container):

```
root@turris:~# lxc-attach -n <name of your lxc container>
```

The prompt changes to `root@LXC_NAME:~#` , to indicate that we are now in the command shell of the lxc container. The first thing to do is to set a password for the root account of the container:

```
root@LXC_NAME:~# passwd
```

Store it in your password manager or another secure place and don’t forget it! You will need it further on.

Next we set the time zone:

```
root@LXC_NAME:~# dpkg-reconfigure tzdata
```

Since our container actually is a ubuntu 16.04 computer, let’s check if there are updates:

```
root@LXC_NAME:~# apt update
```

and let us install them:

```
root@LXC_NAME:~# apt upgrade
```

As a further step we want to have automatic security updates installed:

```
root@LXC_NAME:~# apt install unattended-upgrades
```

Our container is a rather ‘bare’ ubuntu computer, so we will first ‘dress’ it with some handy tools and one essential package before we can install SoftEther VPN.

Step 3: install Nano, OpenSSH, Vsftp and Build Essential

Nano is a userfriendly editor that comes in handy when editing configuration files:

```
root@LXC_NAME:~# apt install nano
```

Installing OpenSSH enables us to use SFTP connections and SSH shell:

```
root@LXC_NAME:~# apt install openssh-server
```

Before using OpenSSH we need to configure it. First make a backup copy of the config file and make that copy read-only. Then edit the config file:

```
root@LXC_NAME:~# cp /etc/ssh/sshd_config /etc/ssh/sshd_config.original

root@LXC_NAME:~# chmod a-w /etc/ssh/sshd_config.original

root@LXC_NAME:~# nano /etc/ssh/sshd_config
```

Make the following changes in sshd_config:

```
LoginGraceTime 30
PermitRootLogin yes
PermitEmptyPasswords no
StrictModes yes
AllowUsers root
```

Save the edited config file and restart the SSH service:

```
root@LXC_NAME:~# service ssh restart
```

Similarly we install and adapt VSFTP:

```
root@LXC_NAME:~# apt install vsftpd
```

Edit the config file:

```
root@LXC_NAME:~# nano /etc/vsftpd.conf
```

and change the setting so we can write:

```
write_enable=YES
```

and restart VSFTP:

```
root@LXC_NAME:~# service vsftpd restart
```

Finally we have to install the build essential package in order to be able to build (compile) SoftEther VPN in our container:

```
root@LXC_NAME:~# apt-get install build-essential
```

We are now ready to download and install SoftEther VPN. Leave the root shell of the container by typing `exit` and pressing ENTER, and the root shell of the router by once again typing `exit` and pressing ENTER.

step 4: download SoftEther VPNSERVER and transfer it to your container

Open the browser of your computer and go to the website of SoftEther VPN: http://www.softether-download.com/en.aspx?product=softether

Select:

- Software: SoftEther VPN(Freeware)
- Component: SoftEther VPN Server
- Platform: Linux
- CPU: ARM EABI (32bit)

and download the software. Once the software has been downloaded you have to transfer it to your container.

Recall the ip address that your router assigned to your container and that you wrote down in step 1, and open the FTP client of your computer to make an SFTP connection to it. Login as `root` with the password you created in step 2.

Transfer the downloaded SoftEther VPN file to your lxc container. The file will be stored in the home directory of user root.

After the transfer to the container was successful, you can close your FTP client and again log on your container.

step 5: compile SoftEther VPNSERVER

Open a terminal window and SSH to your lxc container (replace <ipaddress of your container> by the ip4-address you wrote down in step 1 and just used in step 4):

```
ssh root@<ipaddress of your container>
```

By giving the command `ls` you should be able to see that the SoftEther VPNSERVER installation file is there, so let’s extract it:

```
root@LXC_NAME:~# tar zxvf softether-vpnserver-v4.22-9634-beta-2016.11.27-linux-arm_eabi-32bit.tar.gz
```

It will extract to a folder `/vpnserver` in the home folder of root.

Change to that folder:

```
root@LXC_NAME:~# cd vpnserver
```

and compile it by giving the command `make`:

```
root@LXC_NAME:~/vpnserver# make
```

During the make process you will be asked some questions: do you want to read the license agreement (answer yes), do you understand the license agreement (answer yes), do you agree the license agreement (answer yes). The proces continues to make the necessary files and runs several checks.

The output you are going to see on your screen during the make process is roughly as follows:

```
SoftEther VPN Server (Ver 4.22, Build 9634, ARM EABI) for Linux Install Utility
Copyright (c) SoftEther Project at University of Tsukuba, Japan. All Rights Reserved.
.
.

Did you agree the License Agreement ?
.
.

make[1]: Entering directory '/root/vpnserver'
Preparing SoftEther VPN Server...
ranlib lib/libcharset.a
ranlib lib/libcrypto.a
ranlib lib/libedit.a
ranlib lib/libiconv.a
ranlib lib/libncurses.a
ranlib lib/libssl.a
ranlib lib/libz.a
ranlib code/vpnserver.a
gcc code/vpnserver.a -O2 -fsigned-char -lm -ldl -lrt -Wl,--no-warn-mismatch -lpthread -L./ lib/libssl.a lib/
libcrypto.a lib/libiconv.a lib/libcharset.a lib/libedit.a lib/libncurses.a lib/libz.a -o vpnserver
ranlib code/vpncmd.a
gcc code/vpncmd.a -O2 -fsigned-char -lm -ldl -lrt -Wl,--no-warn-mismatch -lpthread -L./ lib/libssl.a lib/libcrypto.a
lib/libiconv.a lib/libcharset.a lib/libedit.a lib/libncurses.a lib/libz.a -o vpncmd

./vpncmd /tool /cmd:Check
vpncmd command - SoftEther VPN Command Line Management Utility
SoftEther VPN Command Line Management Utility (vpncmd command)
Version 4.22 Build 9634   (English)
Compiled 2016/11/27 15:23:56 by yagi at pc30
Copyright (c) SoftEther VPN Project. All Rights Reserved.

VPN Tools has been launched. By inputting HELP, you can view a list of the commands that can be used.

VPN Tools>Check
Check command - Check whether SoftEther VPN Operation is Possible
---------------------------------------------------
SoftEther VPN Operation Environment Check Tool

Copyright (c) SoftEther VPN Project.
All Rights Reserved.

If this operation environment check tool is run on a system and that system passes, it is most likely that SoftEther
VPN software can operate on that system. This check may take a while. Please wait...

Checking 'Kernel System'...
              Pass
Checking 'Memory Operation System'...
```

```
              Pass
Checking 'ANSI / Unicode string processing system'...
              Pass
Checking 'File system'...
              Pass
Checking 'Thread processing system'...
              Pass
Checking 'Network system'...
              Pass

All checks passed. It is most likely that SoftEther VPN Server / Bridge can operate normally on this system.

The command completed successfully.


--------------------------------------------------------------------
The preparation of SoftEther VPN Server is completed !


*** How to switch the display language of the SoftEther VPN Server Service ***
SoftEther VPN Server supports the following languages:
  - Japanese
  - English
  - Simplified Chinese

You can choose your prefered language of SoftEther VPN Server at any time.
To switch the current language, open and edit the 'lang.config' file.


*** How to start the SoftEther VPN Server Service ***

Please execute './vpnserver start' to run the SoftEther VPN Server Background Service.

And please execute './vpncmd' to run the SoftEther VPN Command-Line Utility to configure SoftEther VPN Server.

Of course, you can use the VPN Server Manager GUI Application for Windows / Mac OS X on the other Windows / Mac OS X
computers in order to configure the SoftEther VPN Server remotely.
.
.
*** For Mac OS X users ***
In April 2016 we released the SoftEther VPN Server Manager for Mac OS X.
You can download it from the http://www.softether-download.com/ web site.
VPN Server Manager for Mac OS X works perfectly as same as the traditional Windows versions. It helps you to
completely and easily manage the VPN server services running in remote hosts.

--------------------------------------------------------------------

make[1]: Leaving directory '/root/vpnserver'
```

The VPNSERVER program has now been created and resides in the folder `/root/vpnserver`. Before starting VPNSERVER, we will move it to a more suitable place and set proper permissions.

step 6: move VPNSERVER to `/usr/local` and set proper permissions

Use the following command to move the vpnserver directory to `/usr/local/`.

```
root@LXC_NAME:~/vpnserver# cd ..
```

```
root@LXC_NAME:~# mv vpnserver /usr/local
```

Verify that it was successful:

```
root@LXC_NAME:~# ls -l /usr/local/vpnserver/
```

```
total 8800
-rwxrwxrwx 1 root root    2784 Nov 27 08:07 Authors.txt
```

```
drwx------ 2 root root    4096 Dec 15 21:13 chain_certs
drwxrwxrwx 2 root root    4096 Dec 15 21:13 code
-rwxrwxrwx 1 root root 1296225 Nov 27 08:07 hamcore.se2
-rw------- 1 root root     867 Dec 15 21:13 lang.config
drwxrwxrwx 2 root root    4096 Dec 15 21:12 lib
-rwxrwxrwx 1 root root    2859 Nov 27 08:07 Makefile
-rwxrwxrwx 1 root root   30801 Nov 27 08:07 ReadMeFirst_Important_Notices_cn.txt
-rwxrwxrwx 1 root root   36296 Nov 27 08:07 ReadMeFirst_Important_Notices_en.txt
-rwxrwxrwx 1 root root   50695 Nov 27 08:07 ReadMeFirst_Important_Notices_ja.txt
-rwxrwxrwx 1 root root   58932 Nov 27 08:07 ReadMeFirst_License.txt
-rwxr-xr-x 1 root root 3751120 Dec 15 21:13 vpncmd
-rwxr-xr-x 1 root root 3751196 Dec 15 21:13 vpnserver
```

If the user does not have root permissions, the files in the vpnserver directory cannot be read, so change and protect the permissions.

Go to `/usr/local/vpnserver/` and change the permissions:

```
root@LXC_NAME:~# cd /usr/local/vpnserver/
root@LXC_NAME:/usr/local/vpnserver# chmod 600 *
root@LXC_NAME:/usr/local/vpnserver# chmod 700 vpncmd
root@LXC_NAME:/usr/local/vpnserver# chmod 700 vpnserver
```

Again verify that it has been successful:

```
root@LXC_NAME:/usr/local/vpnserver# ls -l
```

```
total 8800
-rw------- 1 root root    2784 Nov 27 08:07 Authors.txt
drw------- 2 root root    4096 Dec 15 21:13 chain_certs
drw------- 2 root root    4096 Dec 15 21:13 code
-rw------- 1 root root 1296225 Nov 27 08:07 hamcore.se2
-rw------- 1 root root     867 Dec 15 21:13 lang.config
drw------- 2 root root    4096 Dec 15 21:12 lib
-rw------- 1 root root    2859 Nov 27 08:07 Makefile
-rw------- 1 root root   30801 Nov 27 08:07 ReadMeFirst_Important_Notices_cn.txt
-rw------- 1 root root   36296 Nov 27 08:07 ReadMeFirst_Important_Notices_en.txt
-rw------- 1 root root   50695 Nov 27 08:07 ReadMeFirst_Important_Notices_ja.txt
-rw------- 1 root root   58932 Nov 27 08:07 ReadMeFirst_License.txt
-rwx------ 1 root root 3751120 Dec 15 21:13 vpncmd
-rwx------ 1 root root 3751196 Dec 15 21:13 vpnserver
```

This completes the changing of the location of the vpnserver program and the setting of proper permissions.

It is recommended to perform a final check to see whether VPNSERVER can operate properly before starting VPNSERVER.

You can use the `check` command on the vpncmd command line management utility to automatically check whether the system has sufficient functions to operate VPNSERVER. (For details, please refer to 6.6 VPN Tools Command Reference.)

First, start `vpncmd` by typing the command `./vpncmd`. Next, select option `3. Use of VPN Tools (certificate creation or communication speed measurement)` and execute the `check` command.

```
root@LXC_NAME:/usr/local/vpnserver# ./vpncmd
```

```
vpncmd command - SoftEther VPN Command Line Management Utility
SoftEther VPN Command Line Management Utility (vpncmd command)
Version 4.22 Build 9634   (English)
Compiled 2016/11/27 15:23:56 by yagi at pc30
Copyright (c) SoftEther VPN Project. All Rights Reserved.

By using vpncmd program, the following can be achieved.

1. Management of VPN Server or VPN Bridge
2. Management of VPN Client
3. Use of VPN Tools (certificate creation and Network Traffic Speed Test Tool)
```

```
Select 1, 2 or 3: 3
```

```
VPN Tools has been launched. By inputting HELP, you can view a list of the commands that can be used.

VPN Tools>check
Check command - Check whether SoftEther VPN Operation is Possible
---------------------------------------------------
SoftEther VPN Operation Environment Check Tool

Copyright (c) SoftEther VPN Project.
All Rights Reserved.

If this operation environment check tool is run on a system and that system passes, it is
most likely that SoftEther VPN software can operate on that system. This check may take a
while. Please wait...

 Checking 'Kernel System'...
Pass
 Checking 'Memory Operation System'...
Pass
 Checking 'ANSI / Unicode string processing system'...
Pass
 Checking 'File system'...
Pass
 Checking 'Thread processing system'...
Pass
 Checking 'Network system'...
Pass

All checks passed. It is most likely that SoftEther VPN Server / Bridge can operate normally
on this system.

The command completed successfully.
```

Type `exit` to leave the command line management utility and return to the shell prompt:

```
VPN Tools>exit
```

Before actually starting VPNSERVER we will configure the system to operate VPNSERVER as a service. This will be done in the next step.

step 7: configure the system to operate the VPNSERVER program as a service and start the program

You can configure your system to operate the vpnserver program as a service mode program by registering the `/usr/local/vpnserver/vpnserver` program as a daemon process that continues to run in the background while Linux is starting.

To register VPNSERVER to Linux as a daemon process, create a startup script, as shown below, with the name `/etc/init.d/vpnserver`.

=======script text below this line==============================

```
#!/bin/sh
#
### BEGIN INIT INFO
# Provides:          vpnserver
# Required-Start:    $remote_fs $syslog
# Required-Stop:     $remote_fs $syslog
# Default-Start:     2 3 4 5
# Default-Stop:      0 1 6
# Short-Description: Start daemon at boot time
# Description:       Enable service provided by daemon.
### END INIT INFO
#
# chkconfig: 2345 99 01
# description: SoftEther VPN Server
DAEMON=/usr/local/vpnserver/vpnserver
LOCK=/var/lock/subsys/vpnserver
test -x $DAEMON || exit 0
case "$1" in
start)
$DAEMON start
touch $LOCK
;;
stop)
$DAEMON stop
rm $LOCK
;;
restart)
$DAEMON stop
sleep 3
$DAEMON start
;;
*)
echo "Usage: $0 {start|stop|restart}"
exit 1
esac
exit 0
```

=======script text above this line==============================

You can use a text editor like nano or the cat command to write the above script to `/etc/init.d/vpnserver` as a text file. To use the cat command to create the script, press Ctrl + D after the line break in the final line:

```
root@LXC_NAME:/usr/local/vpnserver# cat > /etc/init.d/vpnserver
```

After creating the `/etc/init.d/vpnserver` startup script, change the permissions for this script so that the script cannot be rewritten by a user without permissions.

```
root@LXC_NAME:/usr/local/vpnserver# chmod 755 /etc/init.d/vpnserver
```

To verify whether the startup script starts, we first have to install SYSV-RC-CONF:

```
root@LXC_NAME:/usr/local/vpnserver# sudo apt-get install sysv-rc-conf
```

After the installation was successful, give the command:

```
root@LXC_NAME:/usr/local/vpnserver# sysv-rc-conf --list vpnserver
```

The output has to be:

```
vpnserver
```

We can now give the command to start the vpnserver:

```
root@LXC_NAME:/usr/local/vpnserver# /etc/init.d/vpnserver start
```

If successful, you will see:

```
The SoftEther VPN Server service has been started.
```

step 8: configure VPNSERVER

Next we have to configure VPNSERVER. We want to be able to connect to the server using l2tp/ipsec.

First invoke the SoftEther VPN Command Line Management Utility, `vpncmd`, by typing `./vpncmd` in the shell prompt:

```
root@LXC_NAME:/usr/local/vpnserver# ./vpncmd
```

```
vpncmd command - SoftEther VPN Command Line Management Utility
SoftEther VPN Command Line Management Utility (vpncmd command)
```

```
Version 4.22 Build 9634   (English)
Compiled 2016/11/27 15:23:56 by yagi at pc30
Copyright (c) SoftEther VPN Project. All Rights Reserved.

By using vpncmd program, the following can be achieved.

1. Management of VPN Server or VPN Bridge
2. Management of VPN Client
3. Use of VPN Tools (certificate creation and Network Traffic Speed Test Tool)

Select 1, 2 or 3: 1            (we specify option 1, to manage the vpn server)

Specify the host name or IP address of the computer that the destination VPN Server or VPN
Bridge is operating on.
By specifying according to the format 'host name:port number', you can also specify the port
number.
(When the port number is unspecified, 443 is used.)
If nothing is input and the Enter key is pressed, the connection will be made to the port
number 8888 of localhost (this computer).
Hostname of IP Address of Destination:        leave empty - we specify nothing


If connecting to the server by Virtual Hub Admin Mode, please input the Virtual Hub name.
If connecting by server admin mode, please press Enter without inputting anything.
Specify Virtual Hub Name:                     leave empty - we specify nothing
Connection has been established with VPN Server "localhost" (port 443).

You have administrator privileges for the entire VPN Server.
```

As the next step we set VPN Server Administrator Password:

```
VPN Server> ServerPasswordSet
ServerPasswordSet command - Set VPN Server Administrator Password
Please enter the password. To cancel press the Ctrl+D key.

Password: vpn_server_administrator_password        choose a password
Confirm input: vpn_server_administrator_password   and repeat


The command completed successfully.
```

Next we create a new virtual HUB:

```
VPN Server> HubCreate name_of_my_new_HUB           choose a name for your HUB

HubCreate command - Create New Virtual Hub
Please enter the password. To cancel press the Ctrl+D key.
```

And set a password for this HUB:

```
Password: password_for_my_new_HUB                  choose a password
Confirm input: password_for_my_new_HUB             and repeat

The command completed successfully.
```

Now we will enter the new HUB

```
VPN Server> HUB name_of_my_new_HUB
Hub command - Select Virtual Hub to Manage
The Virtual Hub "name_of_my_new_HUB" has been selected.
The command completed successfully.

VPN Server/name_of_my_new_HUB>
```

Create a user for the new HUB:

```
VPN Server/name_of_my_new_HUB> UserCreate name_of_user_for_my_new_HUB
UserCreate command - Create User
```

```
Assigned Group Name:                              you can leave this empty

User Full Name:                                   enter a full name for the user

User Description:                                 enter a description or leave empty

The command completed successfully.

VPN Server/name_of_my_new_HUB>
```

And set a password for this user:

```
VPN Server/name_of_my_new_HUB> UserPasswordSet name_of_user_for_my_new_HUB

UserPasswordSet command - Set Password Authentication for User Auth Type and
Set Password
Please enter the password. To cancel press the Ctrl+D key.

Password: password_of_the_user_for_my_new_HUB          choose a password
Confirm input: password_of_the_user_for_my_new_HUB     and repeat


The command completed successfully.
```

Next we enable l2tp/ipsec:

```
VPN Server/name_of_my_new_HUB> IPsecEnable

IPsecEnable command - Enable or Disable IPsec VPN Server Function

Enable L2TP over IPsec Server Function (yes / no): yes

Enable Raw L2TP Server Function (yes / no): no

Enable EtherIP / L2TPv3 over IPsec Server Function (yes / no): no

Pre Shared Key for IPsec (Recommended: 9 letters at maximum): pre-shared_key

Default Virtual HUB in a case of omitting the HUB on the Username:
name_of_my_new_HUB

The command completed successfully.
```

Finally we enable SecureNAT:

```
VPN Server/name_of_my_new_HUB> SecureNatEnable
SecureNatEnable command - Enable the Virtual NAT and DHCP
Server Function (SecureNat Function)
The command completed successfully.
```

To summarize, we have:

set a VPN Server Administrator Password	: vpn_server_administrator_password
created a HUB	: name_of_my_new_HUB
set a password for the new HUB	: password_for_my_new_HUB
defined a user of the HUB	: name_of_user_for_my_new_HUB

set a password for the user of the HUB : password_of_the_user_for_my_new_HUB

set a pre-shared key for l2tp/ipsec : pre-shared_key

Let’s inspect the status of our vpn server:

```
VPN Server/name_of_my_new_HUB> ServerStatusGet

ServerStatusGet command - Get Current Server Status
Item                                          |Value
----------------------------------------------+-------------------------
Server Type                                   |Standalone Server
Number of Active Sockets                      |43
Number of Virtual Hubs                        |2
Number of Sessions                            |0
Number of MAC Address Tables                  |1
Number of IP Address Tables                   |1
Number of Users                               |1
Number of Groups                              |0
Using Client Connection Licenses (This Server)|0
Using Bridge Connection Licenses (This Server)|0
Outgoing Unicast Packets                      |231 packets
Outgoing Unicast Total Size                   |9,702 bytes
Outgoing Broadcast Packets                    |0 packets
Outgoing Broadcast Total Size                 |0 bytes
Incoming Unicast Packets                      |231 packets
Incoming Unicast Total Size                   |9,702 bytes
Incoming Broadcast Packets                    |464 packets
Incoming Broadcast Total Size                 |28,304 bytes
Server Started at                             |2016-12-16 (Fri) 14:34:26
Current Time                                  |2016-12-16 15:14:33.238
64 bit High-Precision Logical System Clock    |2407123
The command completed successfully.
```

We see there are 2 virtual hubs. One is the HUB we just created, the other is the DEFAULT HUB. Let’s do some housekeeping and delete the DEFAULT HUB since we don’t need it.

We leave our HUB:

```
VPN Server/name_of_my_new_HUB>Hub

Hub command - Select Virtual Hub to Manage
The Virtual Hub selection has been unselected.
The command completed successfully.
```

and delete the DEFAULT HUB:

```
VPN Server>HubDelete DEFAULT
HubDelete command - Delete Virtual Hub
The command completed successfully.
```

This completes the configuration of vpnserver for the use of l2tp/ipsec. There are still two steps to go:

- setting port forwarding in the router for the proper ports

- configuring the vpn-settings of our clients

step 9: give container static lease and configure portforwarding in the router

In Turris Omnia open the LuCI interface and go to *Network/DHCP and DNS* and add a static lease for your vpn container so it will always have the same ip address on your LAN.

After this go to *Network/Firewall* in LuCI , open the tab *Port Forwards* and add two new port forwards:

name	port	external	internal	ip address
SoftEtherVPNudp500	ip4 udp port 500	wan anywhere	lan ip address of vpn container	
SoftEtherVPNudp4500	ip4 udp port 4500	wan anywhere	lan ip address of vpn container	

Click Save and Apply when finished.

step 10: configure your vpn clients

In step 8 we:

created a HUB	: name_of_my_new_HUB
defined a user of the HUB	: name_of_user_for_my_new_HUB
set a password for the user of the HUB	: password_of_the_user_for_my_new_HUB
set a pre-shared key for l2tp/ipsec	: pre-shared_key

On your computer/phone/tablet enter the following settings for the configuration of the vpn connection profile:

Connection type	: l2tp
Server address	: the external ip-address or (D)DNS-name of your router
Account name	: name_of_user_for_my_new_HUB@name_of_my_new_HUB
User authentication - password	: password_of_the_user_for_my_new_HUB
Shared secret	: pre-shared_key

That’s all.