TP-Link TL-MR3020

OpenWrt

HOWTOs and Documentations

Author:
Patrick Kowalik

30. Juli 2012

Inhaltsverzeichnis

1 Installation von OpenWrt

Neu erhaltenen TP-Link auspacken und per Mini-USB an die Stromversorgung anschließen. Schalter sollte beim ersten mal auf AP gestellt werden, so dass der TP-Link als AccesPoint ausgibt und man sich über die Netzwerkverbindungen mit Ihm verbinden kann. Sobald man dies Erfolgreich getan hat, kann man im Webbrowser folgende IP:192.168.0.254 aufrufen um auf die Webpage von dem TP-Link zu gelangen. Bevor wir jetzt OpenWrt über die Webpage vom TP-Link installieren muss die richtige Trunk Version für den TL-MR3020 auf folgender Seite: Trunk-Firmware heruntergeladen werden kann. Dieses runtergelade *.bin File kann dann über die Webpage unter Firmware Upgrade auf den TP-Link gespielt werden. Nach Erfolgreichem Upgrade auf OpenWrt kann man den TP-Link ans eigene LAN hängen und mit Telnet (Windows) oder minicom (Linux) auf ihn connecten. Sobald man Zugriff auf dem TP-Link hat sollte man zuerst das Passwort ändern, was man mit folgenden Befehl machen kann.

```
root@openwrt# passwd
```

Listing 1: Passwort ändern

Daraufhin sollte man auch noch die Netzwerkkonfiguration für LAN anpassen, so dass der TP-Link beim anschließen ans eigene LAN per DHCP eine eigene IPv4 Adresse bekommt. Dazu muss man unter /etc/config/ die Datei network mit dem vi-Editor öffnen :

```
vi /etc/config/network
```

Listing 2: Öffnen der Netzwerkkonfiguration

Hat man diese Datei jetzt mit dem vi-Editor geöffnet, so muss man dort erst in den Einfüge Modus wechseln, indem man die Taste i auf der Tastatur betätigt. Danach ändert man den folgenden Wert:

```
static 'proto'  => static 'dhcp'
```

Listing 3: Ändern der Netzwerkkonfiguration

Nach der Änderung mit der Escape Taste aus dem EInfüge Modus gehen und die Datei mit :wq abspeichern! Jetzt kann der TP-Link neugestartet werden und bekommt in jedem LAN per DHCP eine IPv4 Adresse zugewiesen.

Zum Abschluss noch einige wichtige vi-Kommandos:

```
i   in den Einfüge Modus wechseln
I   zu Begin der Zeile etwas einfügen
Escape aus dem aktuellen Modus raus
:q! Datei schließen ohne zu speichern
:wq Datei speichern und schließen
```

Listing 4: Wichitge Kommands für den vi Editor

1.1 TP-Link wiederherstellen

Um einen TP-Link wiederherzustellen benötigt man einmal das Backup, dass man vom Auslieferungszustand gemacht hat. Dieses befindet sich auf dem Test-Recher auf dem Desktop im Ordner TP-Link und heißt Backup.bin Wenn man dieses Backup über einen Windows Rechner auf den auf den TP-Link kopieren möchte, dann sollte man am besten WinSCP benutzen und unter Linux scp. Das Backup sollte in das Temp Verzeichnis auf den TP-Link kopiert werden. Dafür wird jedoch bisschen Platz auf dem TP-Link benötigt. Folgender SCP-Command:

```
scp -r Backup.bin root@192.168.103.100:/tmp
```

Nachdem man nun das Backup Erfolgreich ins Temp-Verzeichnis auf den TP-Link kopiert hat, muss man auf dem TP-Link folgenden Befehl ausführen. Jedoch sollte man diesen Befehl im Failsafemode ausführen, indem man folgendermaßen gelangt:

- TP-Link seriell an einen Linux PC anschließen
- mit minicom auf den TP-Link connecten
- TP an machen
- irgendwann steht auf der Konsole das man mit [F] + [ENTER] in den Failsafe Mode gelangt!

Nachdem man nu im Failsafemode ist kann man folgenden Befehl eingeben: mtd -r write Backup.bin firmware

Eine andere Möglichkeit ist es direkt auf dem TP-Link mit wget den Firmware Status zurückzusetzen. Folgendermaßen geht es:

```
cd /tmp
wget http://downloads.openwrt.org/snapshots/trunk/
   ar71xx/openwrt-ar71xx-generic-tl-mr3020-v1-
   squashfs-sysupgrade.bin
sysupgrade -n /tmp/openwrt-ar71xx-generic-tl-mr3020-
   v1-squashfs-sysupgrade.bin
```

Folgende OpenWrt-Version runterladen: http://downloads.openwrt.org/snapshots/trunk/ar71xx/openwrt-ar71xx-generic-tl-mr3020-v1-squashfs-factory.bin
Danach ist der TL-MR3020 im Original OpenWrt Zustand!

Hier kann man sich die richtige Url dafür raussuchen:
http://downloads.openwrt.org/snapshots/trunk/ar71xx/

2 Crosscompiler einrichten

In diesem Kapitel wird ein Crosscompiler auf einem Linux Rechner eingerichtet um die die Datei tunslip.c für den TP-Link TL-MR3020 zu compolieren. Folgendes Linux Betriebssystem wurde verwendet (**Ubuntu 12.04 TLS**).

1. **Installation von Build Tools**
   Zu aller erst muss man sich die Build Tools mit folgenden Befehl installieren:

```
sudo apt-get install subversion build-essential
```

Listing 5: Build-Tools installieren

2. **OpenWrt herunterladen**
   Daraufhin sollte man sich in einem Öffentlichen Verzeichnis einen neuen Ordner erstellen, indem man dann das trunk OpenWrt herunterlädt. Danach sollte man überprüfen und wenn nicht vorhanden **svn** auf dem Linux System installieren. In diesem Beispiel erstellen wir einen Ordner und laden die aktuelle trunk OpenWrt Firmware herunter:

```
mkdir ~/openwrt
cd openwrt
svn co svn://svn.openwrt.org/openwrt/trunk/
```

Listing 6: trunk OpenWrt

3. **Feeds updaten und installieren**
   Nachdem herunterladen von OpenWrt kann man noch die aktuellen Feeds updaten und installieren (jedoch nicht zwingend nötig)

```
./scripts/feeds update -a
./scripts/feeds install -a
```

Listing 7: Feeds installieren

4. **Package erstellen**
   Jetzt kann man ein belibiegs Paket erstellen. Folgende Anleitung ist für tunslip6. Zuerst legt man sich im Ordner trunk/package/ eine Ordner names tunslip6 an und in diesem Ordner legt man sich dann einen weiteren Ordner names /src an. Daraufhin sollte man folgende Verzeichnis Struktur erhalten:
   /trunk/package/tunslip6/
   /trunk/package/tunslip6/src/
   Nun kopiert man die tunslip6.c Datei in den /src Ordner (liegt auch als Anhang bei) und erstellt dafür noch ein passendes Makefile was folgendermaßen aussieht:

```
tunslip6: tunslip6.o
	$(CC) $(LDFLAGS) tunslip6.o -o tunslip6
tunslip6.o: tunslip6.c
	$(CC) $(CFLAGS) -c tunslip6.c

# remove object files and executable when user
   executes "make clean"
clean:
	rm *.o tunslip6
```

Listing 8: Package erstellen

Nun kann man überprüfen ob das Makefile auch funktioniert indem man

```
make
```

in die Console einigbt. Daraufhin sollte eine ausführbare tunslip6 Datei entstehen und eine *.o Datei. Diese kann/sollte man danach wieder mit dem Befehl

```
make clean
```

löschen.
Nun haben wir folgende Verzeichnisstruktur mit folgenden Dateien:
/trunk/package/tunslip6/src/tunslip6.c
/trunk/package/tunslip6/src/Makefile

Nun geht man einen Verzeichnis zurück, so dass man im tunslip6 Verzeichnis ist und erstellt folgendes Makefile:

```
include $(TOPDIR)/rules.mk

# Name and release number of this package
PKG_NAME:=tunslip6
PKG_RELEASE:=1

# This specifies the directory where we're
   going to build the program.
# The root build directory, $(BUILD_DIR), is by
    default the build_mipsel
# directory in your OpenWrt SDK directory
PKG_BUILD_DIR := $(BUILD_DIR)/$(PKG_NAME)

include $(INCLUDE_DIR)/package.mk

# Specify package information for this program.
# The variables defined here should be self
   explanatory.
define Package/tunslip6
       SECTION:=utils
       CATEGORY:=Utilities
endef

# Specify what needs to be done to prepare for
   building the package.
# In our case, we need to copy the source files
    to the build directory.
# This is NOT the default. The default uses the
    PKG_SOURCE_URL and the
# PKG_SOURCE which is not defined here to
   download the source from the web.
# In order to just build a simple program that
   we have just written, it is
# much easier to do it this way.
define Build/Prepare
       mkdir -p $(PKG_BUILD_DIR)
       $(CP) ./src/* $(PKG_BUILD_DIR)/
endef
```

```
# We do not need to define Build/Configure or
   Build/Compile directives
# The defaults are appropriate for compiling a
   simple program such as this one

# Specify where and how to install the program.
    Since we only have one file,
# the helloworld executable, install it by
   copying it to the /bin directory on
# the router. The $(1) variable represents the
   root directory on the router running
# OpenWrt. The $(INSTALL_DIR) variable contains
    a command to prepare the install
# directory if it does not already exist.
   Likewise $(INSTALL_BIN) contains the
# command to copy the binary file from its
   current location (in our case the build
# directory) to the install directory.
define Package/tunslip6/install
       $(INSTALL_DIR) $(1)/bin
       $(INSTALL_BIN) $(PKG_BUILD_DIR)/
          helloworld $(1)/bin/
endef

# This line executes the necessary commands to
   compile our program.
# The above define directives specify all the
   information needed, but this
# line calls BuildPackage which in turn
   actually uses this information to
# build a package.
$(eval $(call BuildPackage,tunslip6))
```

Listing 9: Makefile

Sollte ein Fehler beim compilieren Auftreten, wo signalisiert wird, dass das Makefile nicht in Ordnung ist, so muss man ins obige Makefile wechseln und drauf achten, das alle Zeilen von define Package/tunslip6 mit der Leertaste eingerückt sind und das alle Zeilen von define Build/Prepare und define Package/tunslip6/install mit einen Tab eingerückt sind.

Jetzt sollten wir folgende Verzeichnisstruktur mit folgenden Dateien beinhalten:
/trunk/package/tunslip6/Makefile
/trunk/package/tunslip6/src/Makefile
/trunk/package/tunslip6/src/tunslip6.c
Jetzt müssen wir ins root verzeichnis wechseln also in /trunk. Dort führen wir jetzt folgenden Befehl aus:

```
make menuconfig
```

In diesem Menu müssen wir zuerst den richtigen Prozessor für den TP-Link TL-MR3020 einstellen, welches der AR9XXX ist und danach muss man im Ordner ipv6, dass zu installierende tunslip6 Packet auswählen. Als letztes sollte man dann die Konfiguration noch speichern. Als letztes muss man dann im /trunk Verzeichniss den folgenden Befehl ausführen:

```
make V=99
```

Nachdem dies Erfolgreich gemacht wurde, findet man im Verzeichnis /trunk/bin/ar71xxx/package/ eine compilierte tunslip6 Datei mit der *.ipk Endung. Diese muss man dann auf den TP-Link TL-MR3020 kopieren. Man kann die compilierte tunslip6 Datei z.B. mit dem Programm scp auf dem TP-Link TL-MR3020 kopieren. (TP-Link-MR3020 bekommt im 103 Netz folgende Ipv4: 192.168.103.100) Folgendes scp Kommando ausführen:

```
scp tunslip6_1_ar71xx.ipk root@192
   .168.103.100:./tmp
```

3 WLAN-Konfiguration

Um den TP-Link so zu konfigurieren, damit er über WLAN sich mit einem Ipv4-Netz verbindet muss folgender durchgeführt werden. Zuerst muss man sich eine Datei names „wpa_supplicant.conf“ im /etc/ Verzeichnis des Tp-Link erstellen. Diese Datei sollte folgendermaßen aussehen:

```
ctrl_interface=/var/run/wpa_supplicant
ap_scan=1

network={
        ssid="SSID-NAME-DES-NETZWERKS"
        scan_ssid=1
        proto=WPA
        key_mgmt=WPA-PSK
        psk=PSK-SCHLÜSSEL
}
```

Nachdem diese Datei erstellt wurde ist es nötig sich für das Funk-Netzwerk einen PSK Schlüssel zu erstellen. Dies ist jedoch auf dem TP-Link nicht möglich, da dort nicht der „wpa_supplicant“ installiert ist. Daher kann man dies auch ganz einfach auf einem beliebigen Linux Rechner machen. Dazu öffnet man die Konsole und gibt folgenden Befehl ein:

```
#wpa_passphraase <SSID-NAME> <WIFI-SCHLÜSSEL>
```

Beispiel:

```
#wpa_passphraase TEST hallodasisteinTest
```

Als Ausgabe sollte man nun folgendes auf der Konsole erhalten:

```
        network={
                        ssid=\TEST\
                        #psk=\hallodasisteinTest\
                        psk=0234b34438ff120383......
                }
```

Diesen nun erstellten PSK-Schlüssel muss man ich die zuvor erstellte „wpa_supplicant.conf“ Datei kopieren. Als letztes muss man noch zwei Dateien im Verzeichnis /etc/config editieren. Zuerst muss man die /etc/config/wireless Datei editieren. Sie muss nun folgendermaßen aussehen:

```
config wifi-device radio0
        option type    mac80211
        option channel 1
        option macaddr 90:f6:52:81:ec:6a
        option hwmode  11ng
        option htmode  HT20
        list ht_capab  SHORT-GI-20
        list ht_capab  SHORT-GI-40
        list ht_capab  TX-STBC
        list ht_capab  RX-STBC1
        list ht_capab  DSSS_CCK-40
        # REMOVE THIS LINE TO ENABLE WIFI:
        option disabled 0

config wifi-iface
        option device  radio0
        option network wan
        option mode    sta
        option ssid    DelaroGmbH
        option encryption psk2
        option key noaccessin2012.
```

Als letztes müssen wir im selben Verzeichnis noch die Network-Konfigurationsdatei editieren „network“. Sie muss nun folgendermaßen aussehen:

```
config interface 'wan'
#       option ifname 'radio0'
#       option type bridge
        option proto static
        option proto 'dhcp'
```

Es ist wichitg hier die Option „ifname“ zu löschen oder auszukommentieren, das es ein BUG im TP-Link ist und er es nicht schaffen würde sich eine Ipv4-Adresse über Wifi zu beziehen. Nachdem diese zwei Dateien nun erfolgreich editiert wurden kann man den

TP-Link wieder mit dem Befehl:

```
reboot
```

neustarten und man sollte nach dem Neustart erkennen, dass das Wifi-Interface hochgefahren wird und das es eine Ipv4-Adresse vom DHCP-Server bezieht.

4 Skripte

4.1 USB-Skripte (RPL)

Damit beim pluggen eines RPL-Border-Routers (Econotag) in den TP-Link die kompilierte „tunslip6“ Datei automatisch ausgeführt wird ist es notwendig zwei USB-Skripte zu erstellen und im Ordner /etc/hotplug.d/usb/ abzulegen. Es müssen komischerweise zwei Skripte erstellt werden, damit das erste Skript das usb0-Interface hochfährt und tunslip6 ausführt und das zweite Skript dem usb0-Interface die Subnetz-Ipv6-Adresse gibt. Zuerst erstellen wir in dem Ordner /etc/hotplug.d/usb/ eine Datei names, „20-myaction“. Die Zahl zu Begin der Datei gibt an in welcher Reihenfolge die Skripte gestartet werden. Diese Datei sollte folgenden Inhalt besitzen:

```
#!/bin/sh
f[ "$ACTION" == "add" -a "$NODE" == "
   ttyUSB1" ]; then
    tunslip6 -s /dev/ttyUSB1 fd25:
       a491:2:3b1::1/64 &
fi
```

Diese Datei ist auch eine ausführbare BASH-Datei, die tunslip mit dem dazugehörigen Subnetz versucht zu öffnen. Die zweite Datei, die erstellt werden muss hat den Name „30-tunslipaction“ und hat den folgenden Inhalt:

```
#!/bin/sh
if[ "$ACTION" == "add" ]; then
    tunslip6 -s /dev/ttyUSB1 fd25:
       a491:2:3b1::1/64
    ifconfig tun0 add fd25:a491:2:3b1
       ::1/64
fi
```

Diese Datei ist wiederum auch eine ausführbare BASH-Datei, die zusätzlich dem usb-Interface sein dazugehöriges Subnetz hinzufügt.

5 USB

5.1 USB Treiber Installation

- **Module für USB 1.1**
  Für USB 1.1 werden die UHCI und OHCI Treiber benötigt, die wie folgt installiert werden können:

```
opkg update
opkg install kmod-usb-uhci
```

  Die OHCI Treiber werden wie folgt installaiert:

```
opkg update
opkg install kmod-usb-ohci
```

- **Module für USB 2.0**
  Für USB 2.0 können zusätzlich zu den UHCI und OHCI Treiber die USB2 Treiber installiert werden, was wie folgt geht:

```
opkg update
opkg install kmod-usb2
```

5.2 USB Stick (Speicher) mounten:

- USB Stick in USB Port stecken
- TL-MR3020 seriel oder übers Netzwerk mit einem PC (Linux) verbinden.
- Nachdem der TL-MR3020 hochgefahren ist, kann man mit folgendem Befehl den USB Stick in ein Verzeichnis mounten:

```
mount /dev/sda1 /mnt/shares
```

- Jetzt kann man in das Verzeichnis */mnt/shares* wechseln, wo sich alle Daten vom USB Stick befinden.

6 OpenVPN installieren

7 TP-Link mit Jackdaw

Um den TP-Link mit einem Jackdaw in Betrieb zu nehmen ist es nicht nötig die selben Pakete wie für die RPL – Konfiguration zu installieren. Der Jackdaw benötigt andere Pakete, die folgenden:

```
kmod-usb-acm
kmod-usb-core
kmod-usb-net
kmod-usb-net-cdc-ether
kmod-usb-net-rndis
```

Nach Installation dieser Pakete ist es möglich den Jackdaw ähnlich wie bei den DreamPlugs zu konfigurieren. Sobald der Jackdaw in USB gepluggt wird funktioniert er.

8 Humotion TL-MR3020 Konfiguration

Im folgenden Abschnitt werden alle Schritte beschrieben um den TP-Link TL-MR3020 für das Nutzen von Humotion zu konfigurieren. Vorausgesetzt wird, dass OpenWrt auf dem TL-MR3020 installiert wurde. Daraufhin sollte man die Netzwerkverbindung des TL-MR3020 prüfen, da für die folgenden Schritte der Zugriff zum Internet benötigt wird. Dies kann man schnell testen, indem man folgenden Befehl ausführt:

```
ping www.google.de
```

Wenn man von der Internetseite eine Antwort bekommt, kann man sicher sein das man Zugriff aufs Internet hat, ansonsten sollte man nochmals die Netzwerkverbindungen des TL-MR3020 prüfen, was wie folgt funktioniert:

- Ins Verzeichnis */etc/config* wechseln
- Im Verzeichnis die Datei ***network*** mit dem *vi-Editor* öffnen
- In der Datei den Parameter ***proto*** vom *eth0 Interface* von ***static*** auf dhcp setzen, so dass der TL-MR3020 über einen DHCP Server IPv4 Adressen zugewiesen bekommt.
- Datei speichern und TL-MR3020 neustarten

Hat man nun Zugriff aufs Internet kann man mit den herunterladen und installieren der folgenden Pakete beginnen:

- **Slip Treiber**
  Slip Treiber werden benötigt um den Tunnel mit Tunslip6 zu öffnen. Werden wie folgt installiert:

```
opkg update
opkg install kmod-slip
```

- **Tunnel Treiber**
  Als nächstes müssen Treiber zum Öffnen eines Tunnels in OpenWrt installiert werden, was wie folgt geht:

```
opkg update
opkg install kmod-tun
```

- **FTDI Treiber**
  Daraufhin müssen Treiber für den FTDI Chip installiert werden. Durch die Installation der Treiber werden Geräte wie der Econotag, die einen FTDI Chip besitzen im Verzeichnis /dev/ aufgelistet und können verwendet werden. Installation wie folgt:

```
opkg update
opkg install kmod-usb-serial-ftdi
```

- **Speicherplatz prüfen**
  Da der TL-MR3020 nur 4MB Speicher besitzt, welcher sehr schnell voll wird, sollte man zwischen durch oder vor dem herunterladen von Paketen prüfen ob genügend Speicher frei ist. Dies kann man mit dem folgenden Befehl machen:

```
df -h
```

- **IPv6-Paket installieren**

```
opkg update
opkg install kmod-ipv6
```

  Diese Paket muss installiert werden, damit IPv6-Adressen vergeben werden können und IPv6-Routing aktiviert wird.

- **OpenVPN**
  Zu guter letzt muss auch noch das OpenVPN Paket heruntergeladen werden, damit man vom TL-MR3020 erfolgreich einen Tunnel zum SSLServer aufbauen kann. Herunterladen wie folgt:

```
opkg update
opkg install openvpn
```

  Nach Erfolgreichen Installation des OpenVPN Pakets kann man sich an die Konfiguration für OpenVPN machen.

  Konfiguration:

  Nach der Installation von OpenVPN muss zuerst aktiviert werden, damit es automatisch beim booten des TL-MR3020 gestartet wird. Dieses wird folgendermaßen im folgenden Ordner durchgeführt **/etc/init.d**

In diesem Ordner muss nun OpenVPN aktiviert werden:

```
./openvpn enable
```

Jetzt wird OpenVPN bei jedem Starten automatisch ausgeführt. Als nächstes müssen OpenVPN-Skripte konfiguriert werden und erstellte Zertifikate auf den Tl-MR3020 kopiert werden. Diese Sachen müssen im Verzeichnis **/etc/openvpn/** erstellt oder konfiguriert werden.

Zertifikate:

Zertifikate müssen auf dem OpenSSL-Server erstellt werden. Dazu wechselt man ins folgende Verzeichnis **/usr/share/doc/openvpn/examples/easy-rsa/2.0**. In diesem Verzeichnis kann nun mit dem folgenden Befehl ein neues Zertifikat mit dazugehörigen Schlüssel erstellt werden:

```
./build-... server/client
```

Nachdem Erstellen des Zertifikats und dem dazugehörigen Schlüssel, muss das Erstellte Zertifikat mit der Endung **.crt** und der dazugehörige Schlüssel mit der Endung **.key** ins **/etc/openvpn/** Verzeichnis auf den TL-MR3020 kopiert werden.

OpenVPN-Konfiguration:

Anschließend müssen noch zwei Konfigurationsdateien für den Verbindungsaufbau erstellt werden. Die erste Datei hat den Namen **client.ovpn** und hat folgenden Inhalt:

```
client
dev tap
float

remote 89.244.151.55 49638
tls-remote HFSSERVER

persist-key
persist-tun

#Skript laden
```

```
script-security 3
up /etc/openvpn/ipv6_con.up

ca /etc/openvpn/ca.crt
cert /etc/openvpn/tplink1.crt
key /etc/openvpn/tplink1.key

nobind
ping 30
verb 3
```

Die zweite Datei hat den Namen **ipv6_con.up** und beinhaltet die IPv6-Routen für den Host. Diese Datei hat den folgenden Inhalt:

```
#!/bin/sh
ifconfig $dev up
ifconfig $dev add fd25:a491:2:3a1::100/64
route add -A inet6 fd23:a491:2:3a1::/64 dev
   $dev
route add -A inet6 fd00:feed:cafe:101::/64 gw
   fd23:a491:2:3a1::20 dev $dev
```

Da es eine ausführbare Datei ist, muss nach dem Erstellen der Datei Ihr Rechte zugewiesen werden. Man kann einer Datei Rechte mit folgenden Befehl zuweisen:

```
chmod u+x ipv6\_con.up
```

In dieser Datei wird die „ipv6_con.up“ Datei aufgerufen, daher musste man Sie vorher ausführbar machen. Ist ist auch möglich die zwei Dateien sich ins Verzeichnis zu kopieren und nur die folgenden Zeilen in den jeweiligen Dateien anzupassen: „ipv6_con.up“: (dritte Zeile ändern)

```
ifconfig $dev add fd25:a491:2:3a1::100/64
```

In der dritten Zeile muss das neue Subnetz für den neuen TP-Link geändert werden. „client.ovpn“:

```
        cert /etc/openvpn/tplink1.crt
       key /etc/openvpn/tplink1.key
```

Hier müssen die zwei Name der Zertifikate geändert werden. Anschließend muss die Konfigurationsdatei für OpenVPN angepasst werden, die sich im Ordner /etc/config/ befindet. Die Datei hat den Dateinamen: openvpn und muss folgendermaßen geändert werden:

```
package openvpn

config openvpn ptp_config
      option enable 1
      option config /etc/openvpn/client.ovpn
```

Nach erfolgreichen Erstellen dieser beiden Konfigurationsdateien für OpenVPN kann nun der Tunnel zum SSL-Server aufgebaut werden. Dies kann einmal mit dem folgenden Befehl im /etc/openvpn/ Verzeichnis ausgeführt werden:

```
openvpn --config client.ovpn
```

oder der TP-Link wird neugestartet, sodass OpenVPN automatisch beim booten startet.

9 Ablaufplan - neuer TP-Link

Flussdiagramm erläutert darstellen