# OpenVPN

Mit OpenVPN von Überall aus ins eigene/heimische Netz. Hier als Beispiel mit FLI4L als Router.

## Quellen:

- EasyRSA3-OpenVPN-Howto
- How To Set Up and Configure an OpenVPN Server on Ubuntu 20.04

## Grundlagen

### Dateien des Server

für den OpenVPN Server werden folgende Dateien unterhalb von '/etc/openvpn/' benötigt:

- `ca.crt` - das sog. Wurzel-Zertifikat aka. Root CA certificate (**C**ertificate **A**uthority)
- `dh2048.pem` - Diffie Hellman Parameter
- `server.crt` - Server Zertifikat
- `server.key` - Server Schlüssel
- `server.conf` - enthält die (Basis-) Konfiguration zum Betrieb des OpenVPN Servers
- `tls-auth.key` - TLS Schlüssel

#### ''server.conf''

Die Datei 'server.conf' enthält alle notwendigen Konfigurationsparameter zum Betrieb des Servers. Wesentlich sind hier z.B. die Pfandangaben und Dateinamen der o.a. Dateien.

server.conf

```
port 9711
proto udp
dev tun
ca /etc/openvpn/ca.crt
cert /etc/openvpn/server.crt
key /etc/openvpn/server.key
dh /etc/openvpn/dh2048.pem
server 192.168.200.0 255.255.255.0
ifconfig-pool-persist ipp.txt
push "route 192.168.200.0 255.255.255.0"
push "redirect-gateway def1 bypass-dhcp"
push "dhcp-option DOMAIN wg"
push "dhcp-option DNS 192.168.100.1"
client-to-client
keepalive 10 120
```

```
tls-auth /etc/openvpn/tls-auth.key 0
cipher AES-256-CBC
#comp-lzo
max-clients 5
persist-key
persist-tun
status openvpn-status.log
verb 2
ping-timer-rem
verb 2
resolv-retry infinite
writepid /var/run/openvpn/server/pid
persist-local-ip
mlock
reneg-sec 3600
status /var/run/openvpn/server/status 15
status-version 1
management 127.0.0.1 0
management-log-cache 100
management-writeport /var/run/openvpn/server/mport
script-security 2
setenv ovpn_ipv6 no
fast-io
down-pre
float
mssfix 1450
tun-mtu 1450
mtu-disc yes
```

## Dateien des Client

- `client.ovpn` - Konfigurationsdatei je Client, enthält alle notwendigen Parameter

Die Konfigurationsdatei `client.ovpn` besteht, als reine Textdatei, aus folgenden Blöcken:

- OpenVPN Client Konfigurationsparameter
- `ca.crt` - Root CA Zertifikat
- `client1.crt` - Client1 Zertifikat
- `client1.key` - Client1 Schlüssel
- `tls-auth.key` - TLS Schlüssel

template_client_config.ovpn

```
client
;dev tap
dev tun
;dev-node MyTap
;proto tcp
proto udp
```

```
remote <FQDN> 9711 udp4
;remote-random
resolv-retry infinite
nobind
;user nobody
;group nogroup
persist-key
persist-tun
;http-proxy-retry # retry on connection failures
;http-proxy [proxy server] [proxy port #]
;mute-replay-warnings
#ca ca.crt       --> see below
#cert client.crt --> see below
#key client.key  --> see below
remote-cert-tls server
tls-auth ta.key 1
cipher AES-256-CBC
data-ciphers AES-256-CBC
#comp-lzo
verb 3
;mute 20
mssfix 1450
tun-mtu 1450
ifconfig 192.168.200.3 192.168.200.1
route 192.168.200.0 255.255.255.0

<ca>
# INSERT "ca.crt" HERE
</ca>

<cert>
# INSERT "client.crt" HERE
</cert>

<key>
# INSERT "client.key" HERE
</key>
key-direction 1

<tls-auth>
# INSERT "tls-auth.key" HERE
</tls-auth>
```

# Erstellung der Konfiguration

## Server

Zertifikate und Schlüssel für den **Server** erzeugen:
Vorbereitung:

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

Arbeitsverzeichnis für die Erstellung der Schlüssel und Zertifikate erstellen:

```
make-cadir My_Certificate_Authority && cd My_Certificate_Authority
```

Der Befehl `make-cadir` erzeugt einen Sym-Link auf `/usr/share/easy-rsa/easyrsa` im Verzeichnis `My_Certificate_Authority`

Parameter des Zertifikatsausstellers bearbeiten:

```
mcedit vars
...
set_var EASYRSA_REQ_COUNTRY "DE"
set_var EASYRSA_REQ_PROVINCE    "NIEDERSACHSEN"
set_var EASYRSA_REQ_CITY    "Hildesheim"
set_var EASYRSA_REQ_ORG         "Familie von Thuelen"
set_var EASYRSA_REQ_EMAIL   "Christoph@von-Thuelen.de"
set_var EASYRSA_REQ_OU      "Familie"
...
set_var EASYRSA_CA_EXPIRE   7300
...
set_var EASYRSA_CERT_EXPIRE 5400
...
#EOF
```

PKI Initialisieren (Unterverzeichnisse werden erstellt):

```
./easyrsa init-pki
```

„.rnd“-File generieren:

```
openssl rand -writerand ./pki/.rnd
```

CA Zertifikat erstellen:

```
./easyrsa build-ca nopass
# ohne "nopass" --> Password: ZFxhJYWCGj2QUhMQ
# mit "nopass" --> keine weitere Passwortabfrage
# ... Common Name (eg: your user, host, or server name) [Easy-RSA CA]:
<empty>
# --> pki/private/ca.key
# --> pki/ca.crt wird  generiert
```

Server Schlüssel erstellen:

```
./easyrsa gen-req server nopass
# --> pki/reqs/server.req
# --> pki/private/server.key
```

Server Zertifikat erstellen:

```
./easyrsa sign-req server server
# --> pki/issued/server.crt
```

Diffie-Hellman Parameter erstellen:

```
./easyrsa gen-dh
# --> pki/dh.pem
cp pki/dh.pem pki/dh2048.pem
```

TLS-Auth Schlüssel erstellen:

```
openvpn --genkey --secret tls-auth.key
# --> tls-auth.key
```

## Client

Zertifikate und Schlüssel für den **Client** erzeugen:

```
sudo su
make-cadir OpenVPN_Clients && cd OpenVPN_Clients
./easyrsa init-pki
```

Privaten Client Schlüssel (**.key**) und Anforderung (**.req**) erstellen:

```
./easyrsa gen-req client1 nopass
# --> pki/reqs/client1.req
# --> pki/private/client1.key
```

Client Zertifikat (.crt**) aus der Anforderung (**.req**)** auf dem „Server“** erstellen:

```
# Wechseln in die "Server"-CA:
cd ../My_Certificate_Authority
./easyrsa import-req ../OpenVPN_Clients/pki/reqs/client1.req client1
./easyrsa sign-req client client1
# --> pki/issued/client1.crt
```

Anschließend das Client Zertifikat `client1.crt` auf den Client zurück kopieren!

OpenVPN Client Konfigurationsdatei „*.ovpn“ erstellen:

```
cd ..
```

```
mkdir client1
cp client_template.ovpn client1/
cp My_Certificate_Authority/pki/ca.crt client1/
cp My_Certificate_Authority/tls-auth.key client1/
cp My_Certificate_Authority/pki/issued/client1.crt client1/
cp OpenVPN_Clients/pki/private/client1.key client1/
./make_openvpn_client_config.sh client1
```

make_openvpn_client_config.sh

```
#!/bin/bash
# First argument: Client identifier, e.g. Smartphone_ User_1
OUTPUTDIR="$1"
CONFIGNAME=$OUTPUTDIR
BASE_CONFIG="client_template.ovpn"
OUTPUT=$CONFIG_NAME.ovpn

#function() check_dir
## Parameter: $CONFIG_NAME
#{
#if [ ! -d $OUTPUTDIR ]; then
#  mkdir $OUTPUTDIR
#else
#  echo "Directory \"$OUTPUTDIR\" already exsists!"
#  #read -p "Continue? (Y/N): " confirm && [[ $confirm == [yY] ||
$confirm == [yY][eE][sS] ]] || exit 1
#fi
#}

check_tools()
# Check, if necessary tooles are available:
{
if [ -e /usr/share/easy-rsa/easyrsa ]
then
  :
else
  echo "Tool: \"easyrsa\" not installed! --> EXIT"
  exit 1
fi
}

# MAIN
check_tools

if [ -z "${CONFIGNAME}" ]; then
  echo "\"${CONFIGNAME}\" is empty!"
  echo "Please specify client config name. --> EXIT!"
  exit 1
else
  if [ ! -d $OUTPUTDIR ]; then
     echo "ERROR: Folder for $CONFIG_NAME not found --> exit!"
```

```
    exit 1
  else
    echo -n "Generating ${CONFIGNAME} ... "
    cat ${BASE_CONFIG} \
      <(echo -e '<ca>') \
      $CONFIGNAME/ca.crt \
      <(echo -e '</ca>\n<cert>') \
      $CONFIGNAME/$CONFIGNAME.crt \
      <(echo -e '</cert>\n<key>') \
      $CONFIGNAME/$CONFIGNAME.key \
      <(echo -e '</key>\n<tls-auth>') \
      $CONFIGNAME/tls-auth.key \
      <(echo -e '</tls-auth>') \
      > ${CONFIGNAME}/${CONFIGNAME}.ovpn
   echo "done!"
  fi
fi
```