OpenVPN
Grundlagen, Server und Clients

PC-Treff-BB Aidlingen

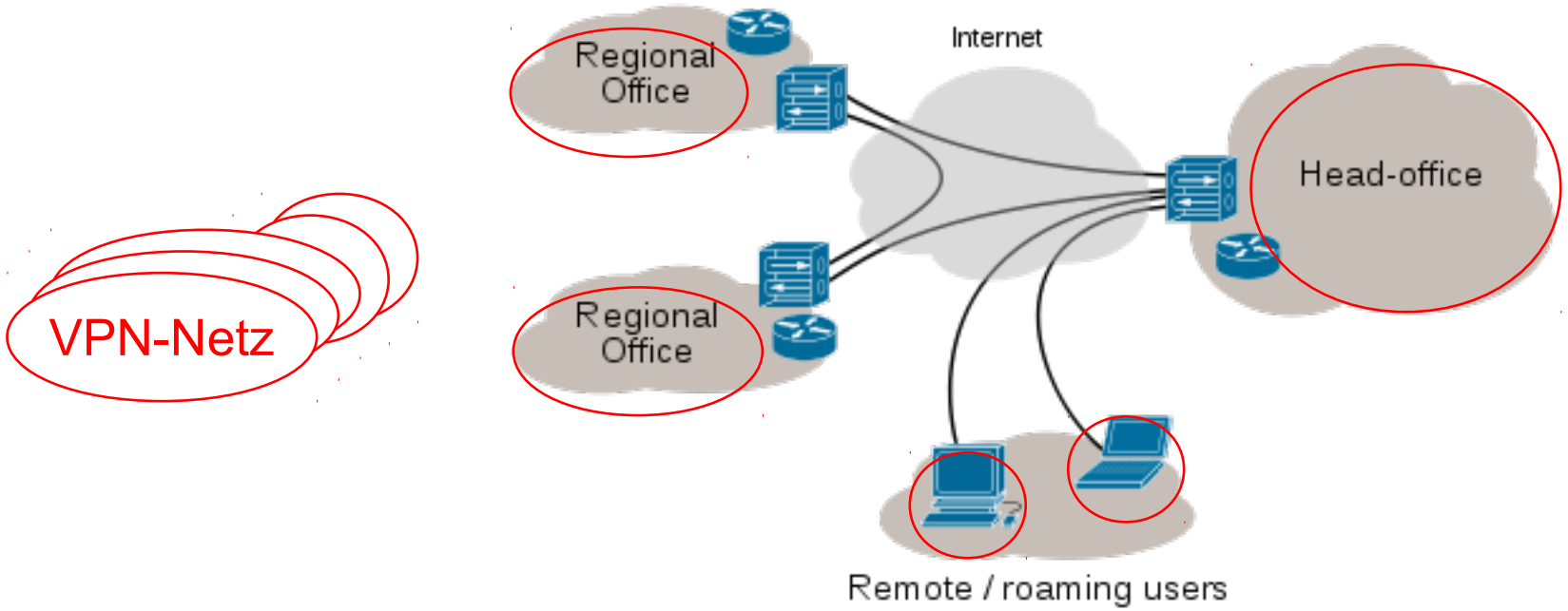
Günter Waller
Roland Egeler

08.09.2018

Agenda

- Virtual Private Networks - Grundlagen
- Die wichtigsten VPN-Verfahren
- OpenVPN Basics
- Server auf OpenWRT
- Server auf pfSense
- Linux Client
- Android Client
- iOS Client
- Windows Client
- Links

Was ist ein VPN?



- Struktur eines VPNs: Unten abgebildet sind Heimarbeitsplätze, die sich per VPN durch das Internet hindurch in den Hauptsitz einer Firma einwählen, wobei der blaue Kasten ein VPN-Gateway ist (bzw. VPN-Server). Darüber hinaus ist der Hauptsitz per VPN auch mit zwei seiner Filialen verbunden, wobei das dazwischen liegende Netz auch hier das Internet ist, das dem VPN als Transportweg dient (**aus Sicht der VPN-Verbindung wird das Internet auf die Funktion eines Verlängerungskabels reduziert**).
- Es gibt also Netz-Netz-Verbindungen wie auch Einzelrechner-Netz-Verbindungen.
- Die Verbindungen über das öffentliche Internet sind verschlüsselt (**Tunnel**).
- Die beteiligten Netze bzw. Einzelrechner befinden sich scheinbar in einem Netz und kommunizieren ohne Kenntnis der (**transparenten**) Tunnel miteinander.

Bildquelle: https://de.wikipedia.org/wiki/Virtual_Private_Network#/media/File:Virtual_Private_Network_overview.svg

Die wichtigsten VPN-Verfahren

- OpenVPN
- PPTP
- IPSec
- SSTP
- SoftEther
- WireGuard
- ...

VPN-Verfahren: OpenVPN

- Open Source
- Benutzt TLS (wie HTTPS und SSH)
- Keine Probleme mit Firewalls
- Kryptoalgorithmen wählbar
- Viel Drittsoftware
- Benutzt Konfigurationsdateien
- Bei Fehlkonfiguration unsicher (PEBKAC)

VPN-Verfahren: PPTP

- Point to Point Tunneling Protocol
- Benutzt mit „GRE“ ein eventuell von der Firewall geblocktes Protokoll
- Von Microsoft initiiert
- Einfach zu konfigurieren, da mitgeliefert
- Schlecht implementiert
- Seit 2012 geknackt
- Nicht zu empfehlen

VPN-Verfahren: IPSEC

- Level-3-Protokoll wie IP
- Eventuell Probleme mit Firewalls
- Sichere Verschlüsselung, aber möglicherweise durch NSA geschwächt
- Braucht weitere Protokolle wie L2TP und IKEv2
- Beide langsam durch doppeltes Kapseln
- Einrichtung eher kompliziert

VPN-Verfahren: SSTP

- Entwickelt von Microsoft
- Ziel: Weniger Support wie bei IPSec
- Benutzt TLS über Port 443 (wie HTTPS)
- Keine Probleme mit Firewalls
- Wenig Unterstützung für andere Betriebssysteme
- Kann nicht von Dritten überprüft werden
- Nicht zu empfehlen

OpenVPN - Basics

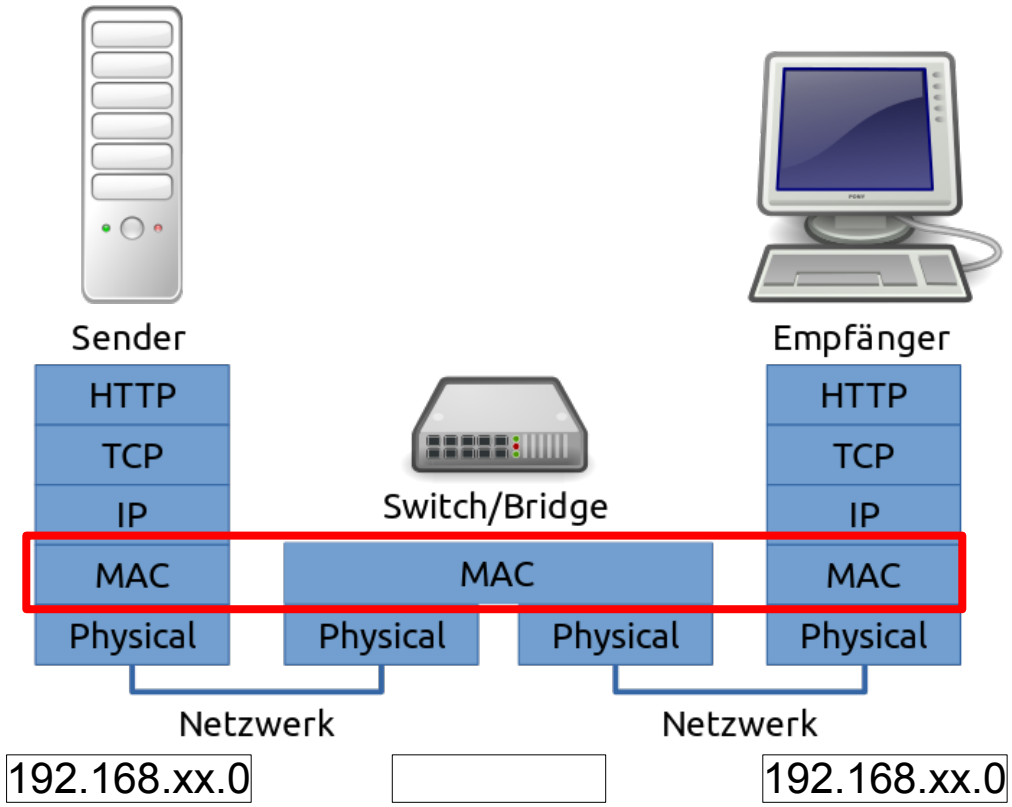
- OpenVPN Quickstart.
- Installing OpenVPN.
- Determining whether to use a routed or bridged VPN.
- Numbering private subnets.
- Setting up your own Certificate Authority (CA) and generating certificates and keys for an OpenVPN server and multiple clients.
- Creating configuration files for server and clients.
- Starting up the VPN and testing for initial connectivity.
- Configuring OpenVPN to run automatically on system startup.
- Controlling a running OpenVPN process.
- Expanding the scope of the VPN to include additional machines on either the client or server subnet.
- Pushing DHCP options to clients.
- Configuring client-specific rules and access policies.
- Using alternative authentication methods.
- How to add dual-factor authentication to an OpenVPN configuration using client-side smart cards.
- Routing all client traffic (including web-traffic) through the VPN.
- Running an OpenVPN server on a dynamic IP address.
- Connecting to an OpenVPN server via an HTTP proxy.
- Connecting to a Samba share over OpenVPN.
- Implementing a load-balancing/failover configuration.
- Hardening OpenVPN Security.
- Revoking Certificates.
- Additional Security Notes.

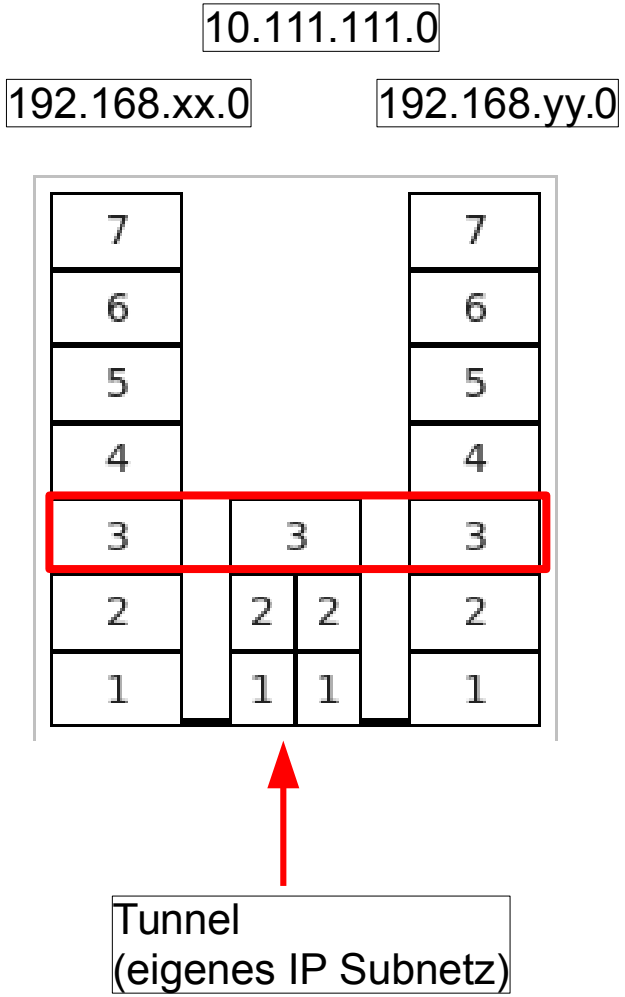
- Aktuelle Version 2.4.6
- Meine: 2.4.4
- Routed vs. Bridged
- IP-Adress-Plan
- **Eigene CA für Zertifikate und Schlüssel**
- Konfigurationsdateien für Server und Clients
- Starten, testen, Verbindungsaufbau
- Überwachen, steuern
- DHCP, Client-spezifische Regeln, Access Policies
- Alternative Authentifizierungsmethoden
- Hardening
- Clients entfernen

PC-Treff-BB Aidlingen

OpenVPN – Grundlagen, Server und Clients

Einschub: Bridging vs. Routing





Bildquelle 1: https://de.wikipedia.org/wiki/Datei:IP-Paket_Switch.svg

Bildquelle 2: https://de.wikipedia.org/wiki/Router

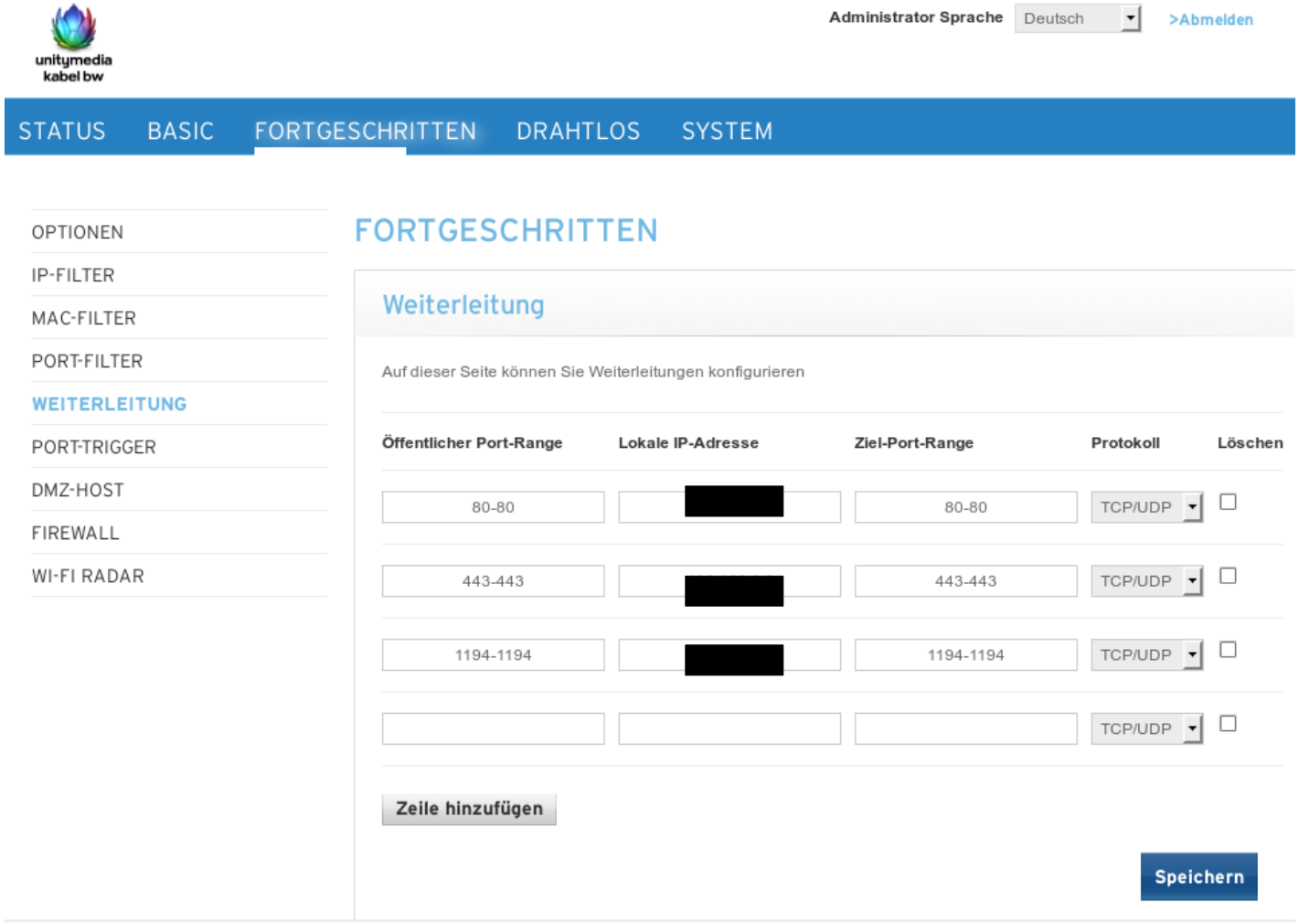
OpenVPN vs. andere VPNs (Selbstzeugnis)

- Open Source (GPL)
- Cross Plattform, Portability
  - Linux, Solaris, OpenBSD, FreeBSD, NetBSD, Mac OS X, QNX, Windows, Android, iOS
  - OpenWRT, Fritz!Box, Dreambox, u.v.m.
- IP4 und IP6 Support
- Modularer Aufbau
- Management Interface (verschiedene GUIs)
- Smartcard Support (unter Windows)
- Robust über unzuverlässige Netze (Tunnel Recovery)
- Performance

https://community.openvpn.net/openvpn/wiki/OverviewOfOpenvpn#WhatdistinguishesOpenVPNfromotherVPNpackages

OpenVPN - Voraussetzungen

- Port Forwarding für den VPN-Standardport 1194 ist Grundvoraussetzung. (Wir zeigen hier alle Ports, auch die für die Weboberfläche des Routers.)



OpenVPN - Voraussetzungen

- Port-Forwarding auf der Fritzbox

Freigaben

Status	Bezeichnung	Protokoll	IP-Adresse im Internet	Port extern vergeben
	SSH-Server	TCP	IPv4	[redacted](22)
	openVPN	UDP	IPv4	1194 (1194)
	HTTP-Server	TCP	IPv4	80 (80)

Neue Freigabe

Firewall-Settings bei OpenWRT (automatisch erzeugt)

Zones

Zone ⇒ Forwardings	Input	Output	Forward	Masquerading	MSS clamping
lan: lan: ⇒ wan vpn_turris	accept	accept	accept	☐	☐
wan: wan: wan6: ⇒ REJECT	reject	accept	reject	☑	☑
guest_turris: guest_turris: *(empty)* ⇒ wan	reject	accept	reject	☐	☐
vpn_turris: vpn_turris: ⇒ lan	accept	accept	reject	☑	☐

Rule

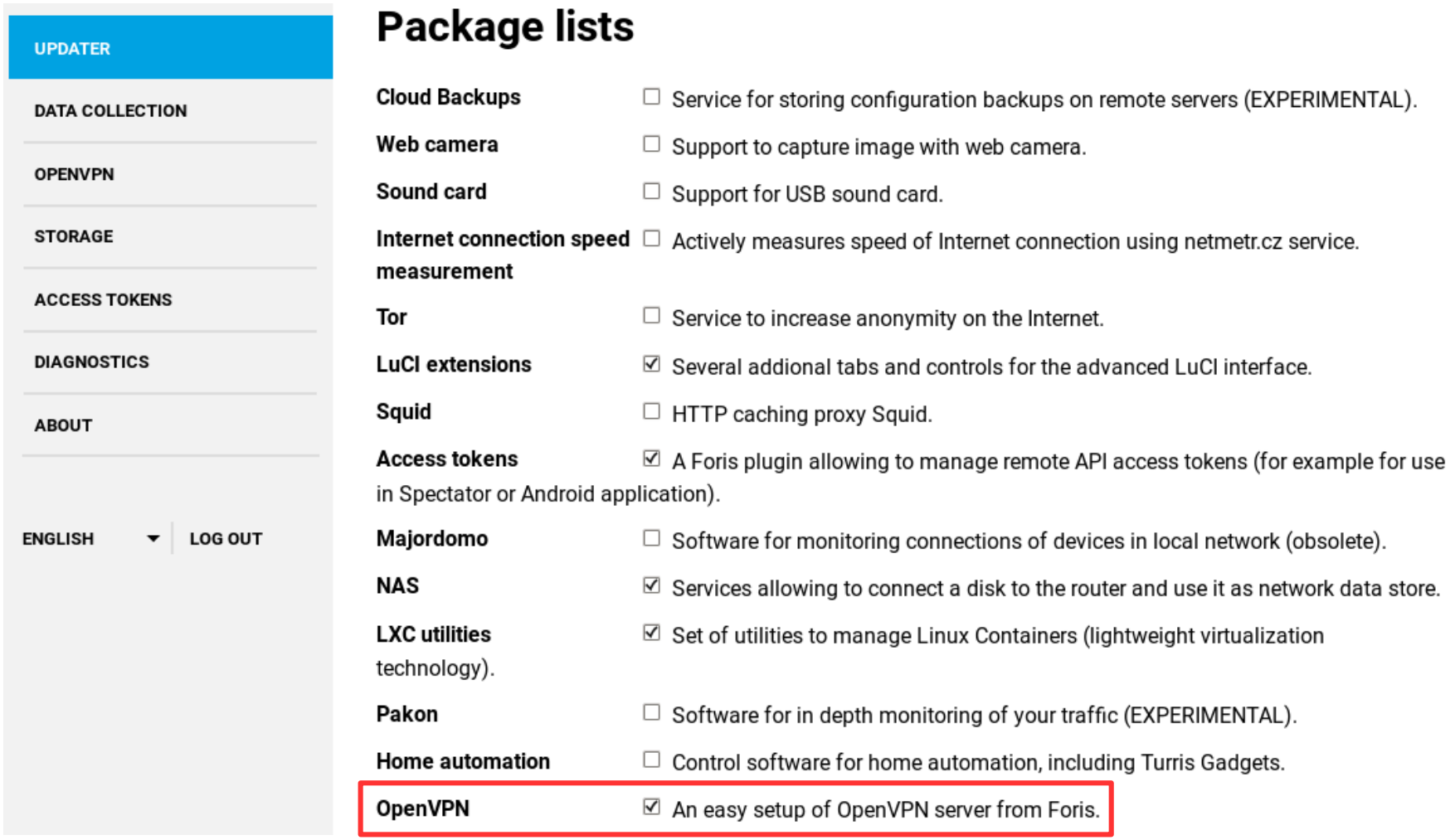
vpn_turris_rule	Any udp From *any host* in *wan* To *any router IP* at port *1194* on *this device*	*Accept input*	☑

OpenVPN - Voraussetzungen

- Für Verbindung aus dem Internet in ein privates Netz ist ein DynDNS-Server hilfreich
- Früher dyndns.org
- Heute no-ip.org u.a.
- Strato bietet diesen Service auch an

Server auf OpenWRT (Turris Omnia Foris) vereinfachte Methode

- Installation nicht über den Paketinstaller sondern über den Updater, dadurch besser integriert mit dem OS
  - Gilt nicht für andere OpenWRT Implementierungen!



Server auf OpenWRT – eigene CA

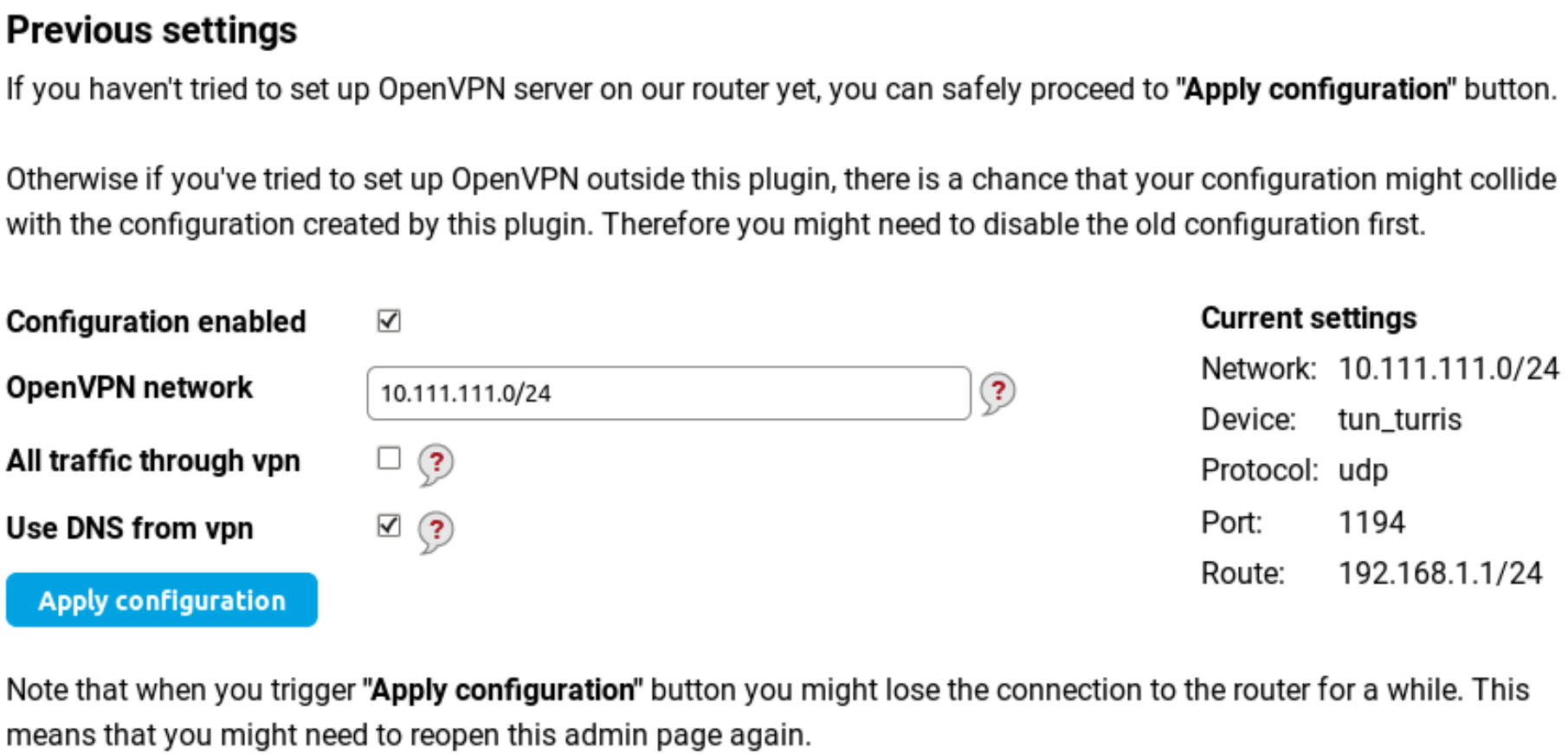
- OpenVPN verwendet zertifikatsbasierte Authentifizierung
- Mein OpenWRT-Router (Turris Omnia) hat eine vereinfachte Konfiguration über die Web-Oberfläche
  - Vereinfachtes Installieren OpenVPN Paket, **kein bridged mode**!
  - **CA generieren** (dauert etwas). Ergebnis im Verzeichnis `/etc/ssl/ca/openvpn`
    - CA: Zertifikat und Private Key
    - Server (01): Zertifikat, Certificate Request, Private Key, Zertifikat im PEM Format

```
-rw-r--r--    1 root     root          6704 Sep  3 00:32 01.crt
-rw-r--r--    1 root     root          1582 Sep  3 00:32 01.csr
-r--------    1 root     root          3268 Sep  3 00:32 01.key
-rw-r--r--    1 root     root          6704 Sep  3 00:32 01.pem
-rw-r--r--    1 root     root          1862 Sep  3 00:32 ca.crt
-r--------    1 root     root          3272 Sep  3 00:32 ca.key
```

Server auf OpenWRT - Konfiguration

Es wird dann auf der gleichen Seite eine Konfiguration vorgeschlagen. Ich habe sie so gelassen und „applied“. Dabei geht es um die Networking-Aspekte.

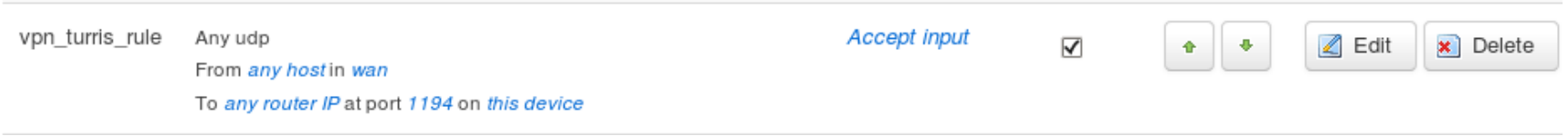
Der VPN-Tunnel bekommt eine eigene Netzadresse: 10.111.111.0/24



Server auf OpenWRT - Networking

Das virtuelle Netz ist also ein Klasse-C Netz 10.111.111.0. Es entsteht dabei ein virtuelles Netzwerk-Device, welches hier **tun_turris** heißt. In der Literatur kommt meistens einfach **tun** vor. Protokoll udp und Port 1194 sind die Defaults.

Das Aktivieren dieser Einstellungen bedeutet gleichzeitig, dass eine Firewall-Regel für diesen Port angelegt und aktiviert wird.



Schließlich gibt es noch eine Route, die auf den Router selbst zeigt.

Die Konfigurationsdatei findet sich nicht, wie man annehmen könnte, in `/etc/openvpn/vpn.conf` (so steht es in manchen Internetseiten), sondern in `/etc/config/openvpn`. Tückisch!

Server auf OpenWRT - Konfigurationsdatei

```
config openvpn 'server_turris'
option enabled '1'
option port '1194'
option proto 'udp'
option dev 'tun_turris'
option mssfix '1300'
option ca '/etc/ssl/ca/openvpn/ca.crt'
option crl_verify '/etc/ssl/ca/openvpn/ca.crl'
option cert '/etc/ssl/ca/openvpn/01.crt'
option key '/etc/ssl/ca/openvpn/01.key'
option dh '/etc/dhparam/dh-default.pem'
option server '10.111.111.0 255.255.255.0'
option ifconfig_pool_persist '/tmp/ipp.txt'
option duplicate_cn '0'
option keepalive '10 120'
option comp_lzo 'yes'
option persist_key '1'
option persist_tun '1'
option status '/tmp/openvpn-status.log'
option verb '3'
option mute '20'
list push 'route 192.168.xx.0 255.255.255.0'
list push 'dhcp-option DNS 192.168.xx.1'
list push 'dhcp-option DOMAIN lan'
```

Die rot markierten Parameter habe ich später eingefügt, weil nicht alles funktioniert hat. Probleme gab es (a) mit DNS (Auflösung der lokalen Namen ging nicht) und (b) mit HTTP (hatte offenbar mit MTU-Size zu tun). Die Lösungen kamen von Google, wurden nicht bis ins letzte Detail analysiert, sondern ausprobiert.

Vorsicht: evtl. Konflikt der Subnetze, wenn bei Server und Client identisch (192.168.xx.0)

Server auf OpenWRT - Zertifikate

In der Konfigurationsdatei finden sich folgende, mit Zertifikaten befassten Zeilen (CA ist hier separate Entity):

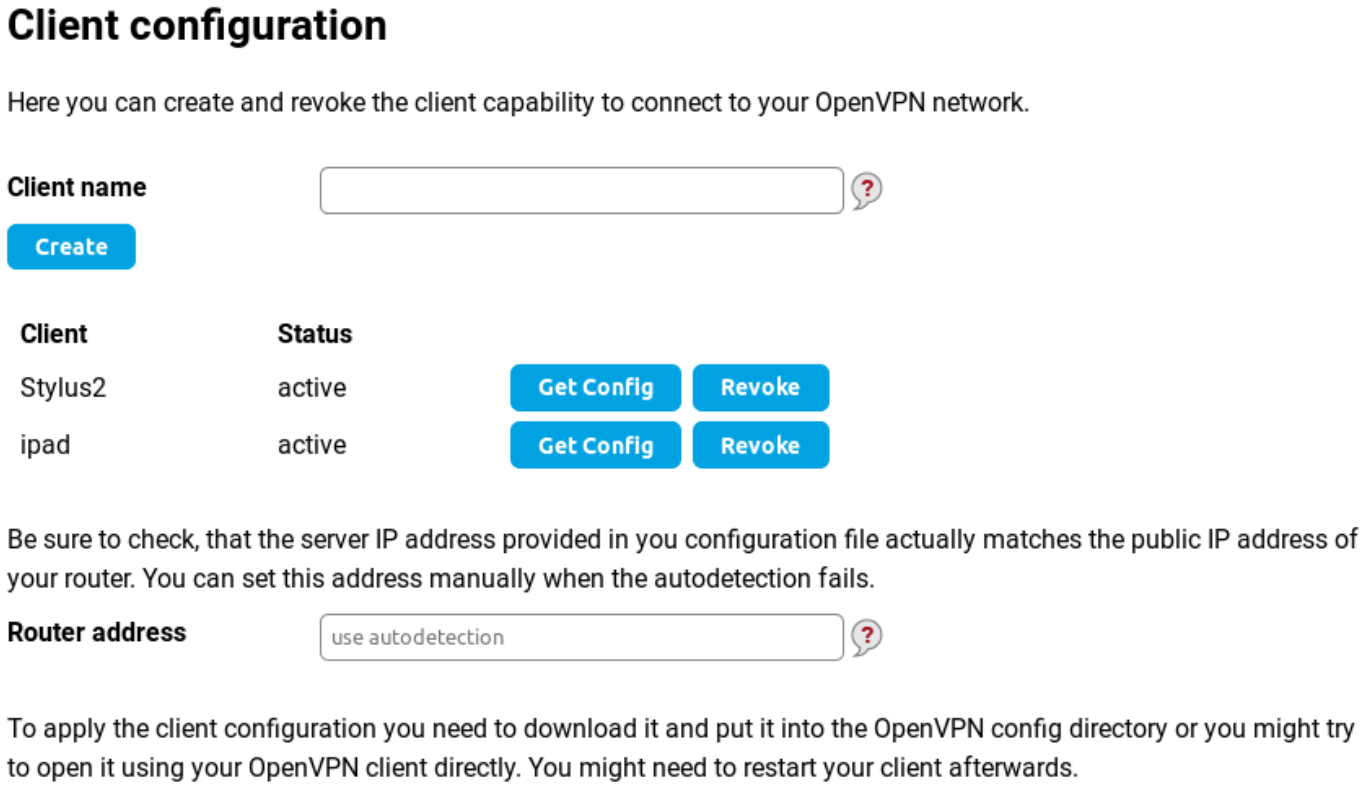
```
option ca '/etc/ssl/ca/openvpn/ca.crt'
option crl_verify '/etc/ssl/ca/openvpn/ca.crl'
option cert '/etc/ssl/ca/openvpn/01.crt'
option key '/etc/ssl/ca/openvpn/01.key'
option dh '/etc/dhparam/dh-default.pem'
```

OpenVPN arbeitet rein auf Zertifikatsbasis. Unter Turris/Foris werden die Clientzertifikate (hier: 02) auf dem Server erzeugt und dann - ggf. mit Nachbearbeitung - auf die Clients gebracht. Dateien pro Client:

```
-rw-r--r--    1 root     root          6911 Sep  3 01:04 02.crt
-rw-r--r--    1 root     root          1582 Sep  3 01:04 02.csr
-r--------    1 root     root          3272 Sep  3 01:04 02.key
-rw-r--r--    1 root     root          6911 Sep  3 01:04 02.pem
```

Server auf OpenWRT – Client-Generierung

Die mit „Get Config“ herunterladbare Datei enthält sowohl Konfigurationsparameter als auch den öffentlichen sowie den privaten Schlüssel. Das ist eher unüblich (unsicher?), erspart aber dem Client, einen privaten Schlüssel selbst erzeugen zu müssen.

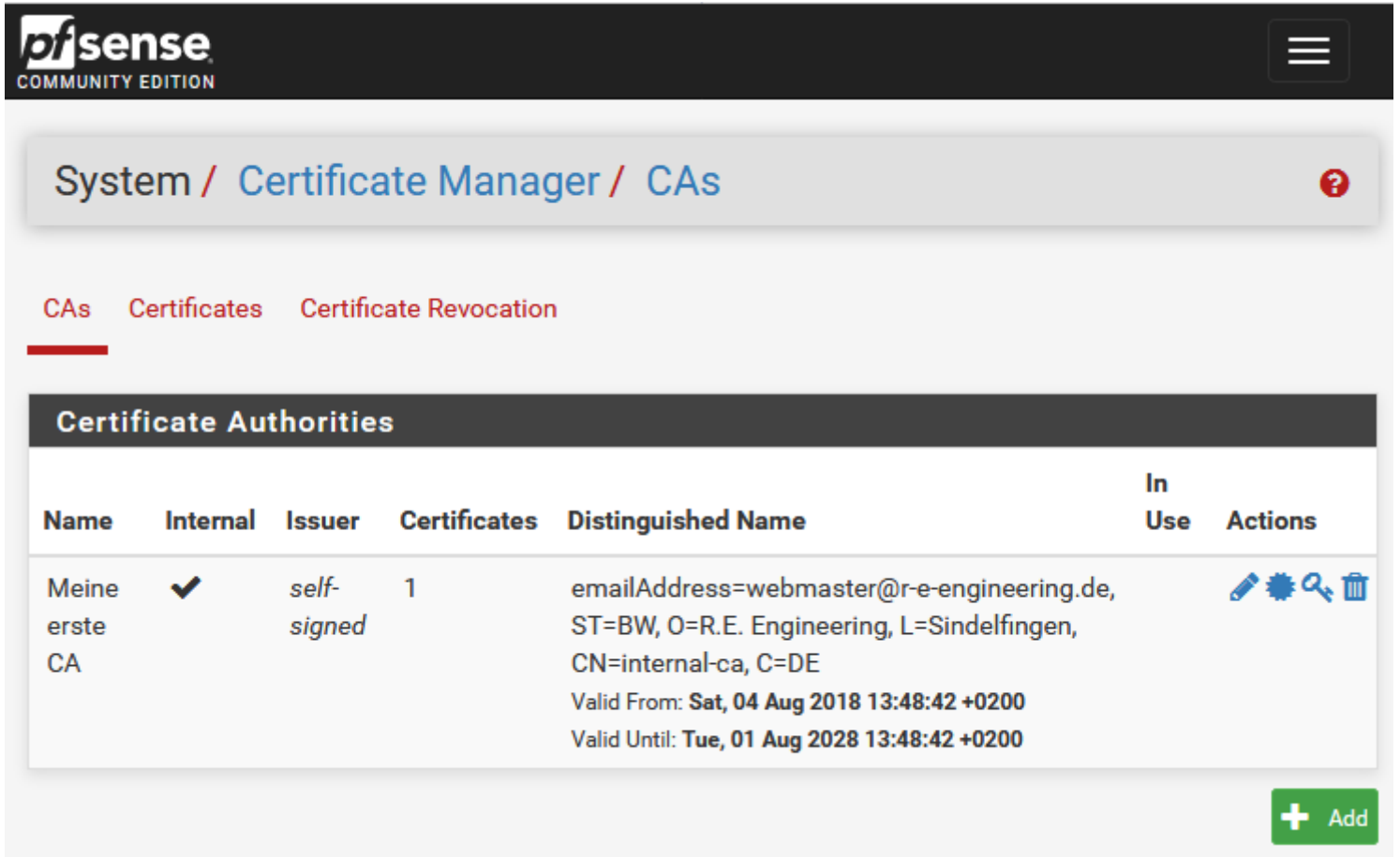


Server auf pfSense

- OpenVPN wird mitgeliefert und aktualisiert
- pfSense hat eigene CA (Certificate Authority)
- Verwaltet eigene Zertifikate und Schlüssel
- Konfiguration über Weboberfläche
- Unterstützt viele Szenarien

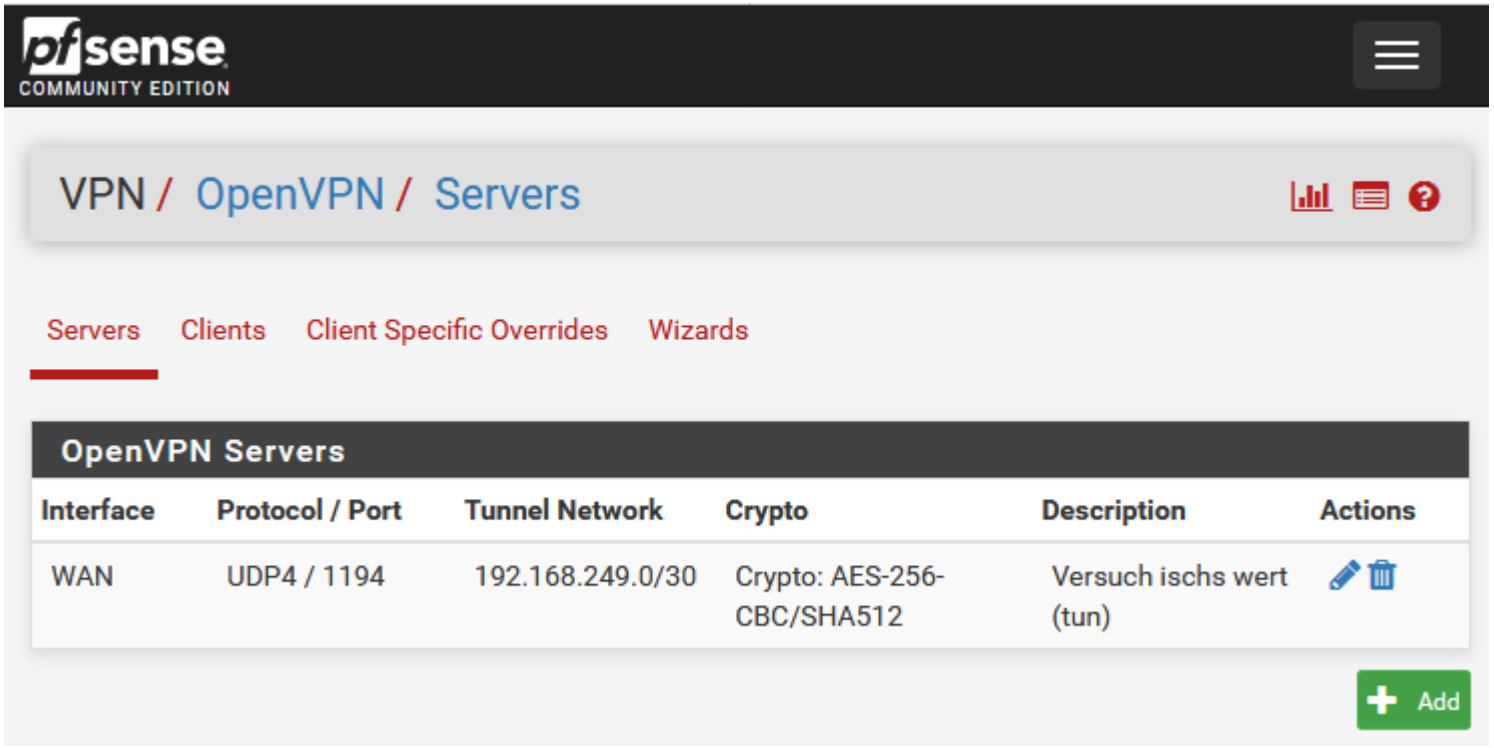
Server auf pfSense

- Certificate Manager



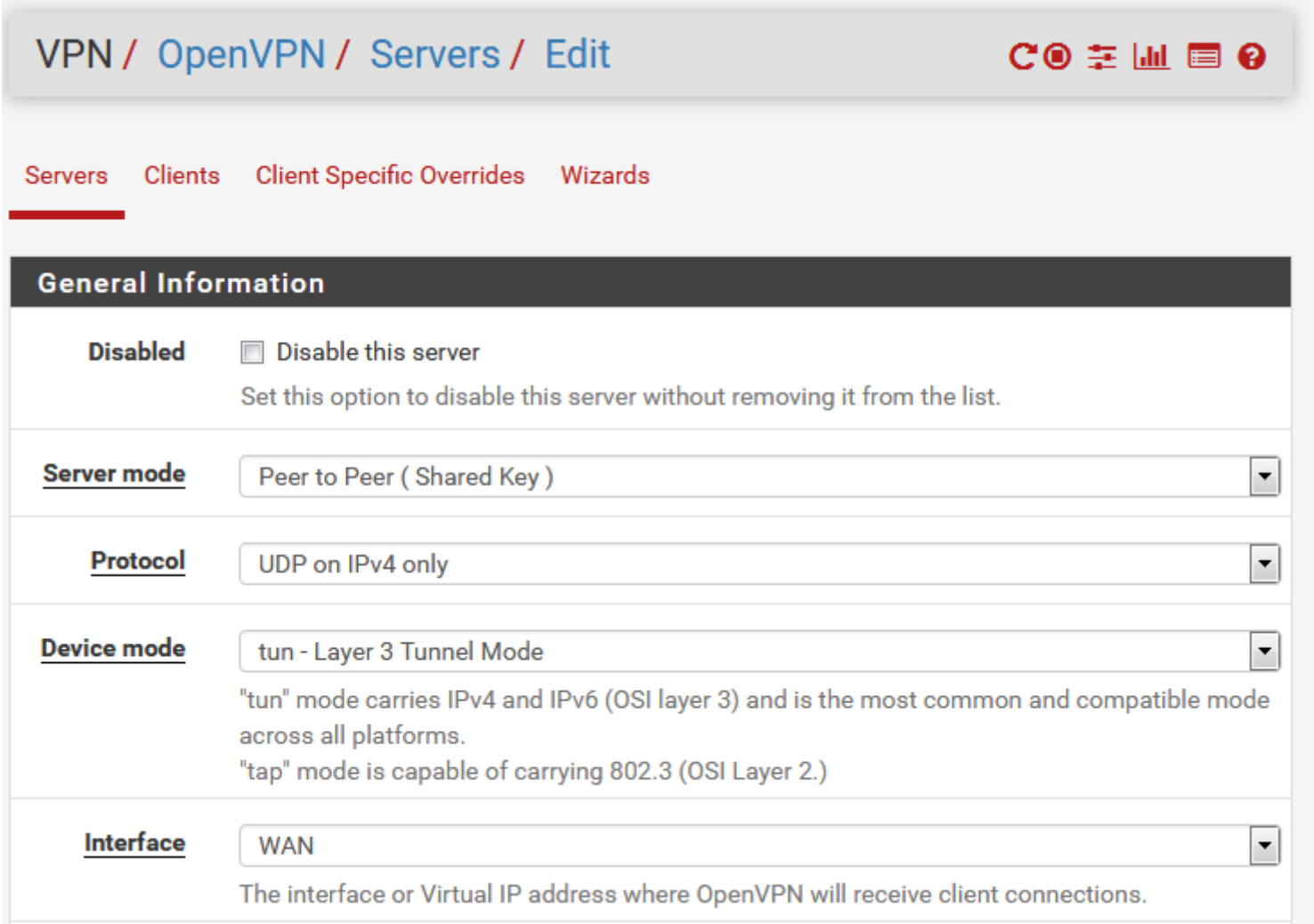
Server auf pfSense

- Konfiguration Überblick



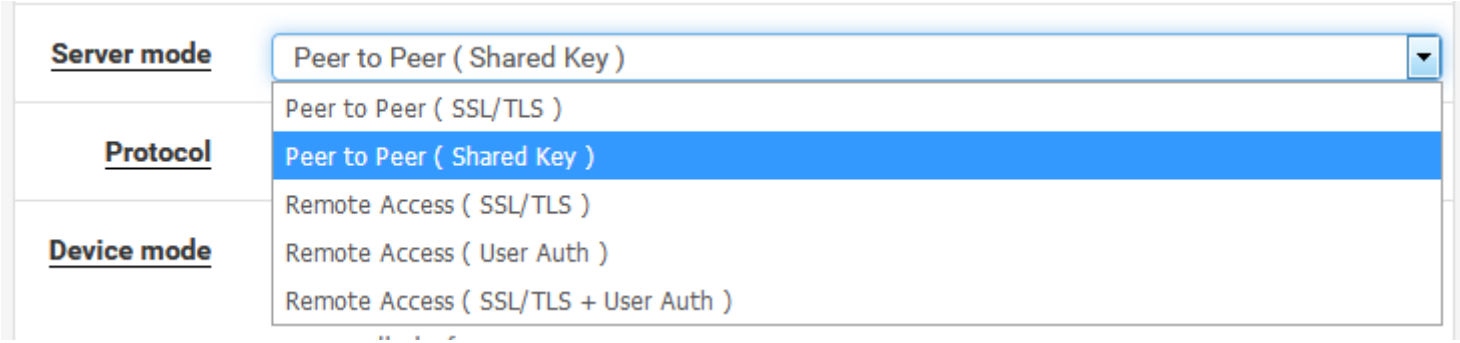
Server auf pfSense

- Konfigurationsdetails (Teile)



Server auf pfSense

- Konfigurationsmöglichkeiten



Server auf pfSense

- Einfachste Konfiguration ist „Shared Key“
- Braucht keine CA
- Einzelnutzerlösung
- Jeder, der den Schlüssel hat, kommt rein
- Schlüssel nur auf kryptografisch gesicherten Medien aufbewahren
- VPN nur im Urlaub auf Fritzbox geöffnet

Android Client (1/2)

- App „**OpenVPN für Android**“ installieren.
- Für Android muss die Datei `turris.conf` nach `turris.ovpn` umbenannt werden. Auf diese Datei muss man gut aufpassen, sie enthält u.a. den Private Key. Durch Download via „Get Config“ Button (im Webinterface des Routers) gelangt sie auf das Handy.
  - In meinem konkreten Fall muss die Zeile
    `remote 192.168.0.2 1194`
    nach
    `remote <Ihr-DynDNS-name> 1194`
    geändert werden.
  - Der Server hält die vom Kabelrouter erhaltene Adresse auf seinem WAN-Interface für seine von außen erreichbare Internetadresse. Wir ersetzen dies durch den dynamischen Hostnamen (hier: von No-IP). **Dies gilt für alle so generierten und heruntergeladenen Client-Konfigurationen** (Linux, iOS).

Android Client (2/2)

- Öffnet man die heruntergeladene Datei mit der App, wird daraus ein VPN-Profil erzeugt. Damit hat die Verbindung auf Anhieb geklappt. Der Client bekam die IP 10.111.111.6, es entsteht offenbar ein Tunnel, an dessen anderem Ende 10.111.111.7 hängt. SSH war damit möglich.
- HTTP ging zunächst nicht, die Lösung der zusätzliche Parameter option mssfix '1300' auf der Serverseite (siehe dort), ohne Änderung am Client.
- Zweites Problem: Es gab keine Namensauflösung für das LAN. Die Lösung brachte ein Traceroute zu einer LAN-Adresse. Es tauchte die Adresse 10.111.111.1 als einziger Hop auf. Die Lösung war ein Eintrag in der Profilkonfiguration unter `IP UND DNS`:
  `DNS-Server 10.111.111.1`

Linux Client

- Mit der Weboberfläche Foris werden eine Reihe von Schritten für jeden Client durchgeführt. Die resultierende Datei muß dann auf den Client gebracht werden. Sie heißt stets `turris.conf`. Unter Linux kann der Name beibehalten werden.
- Jetzt entweder Zertifikate extrahieren und an zentraler Stelle speichern oder die Datei so einbinden:

```
sudo openvpn --config turris.conf
```

- DNS-Auflösung klappt nicht. Lösung: hier
- Man muss das Skript erweitern, damit es dynamisch die `/etc/resolv.conf` anpasst. Dafür braucht man das Paket `resolvconf` (war schon vorhanden) oder `openresolv` (wie es an anderer Stelle gefordert wird). Damit ergänzt man `turris.conf` um folgende 3 Zeilen:

```
script-security 2
up /etc/openvpn/update-resolv-conf
down /etc/openvpn/update-resolv-conf
```

iOS Client

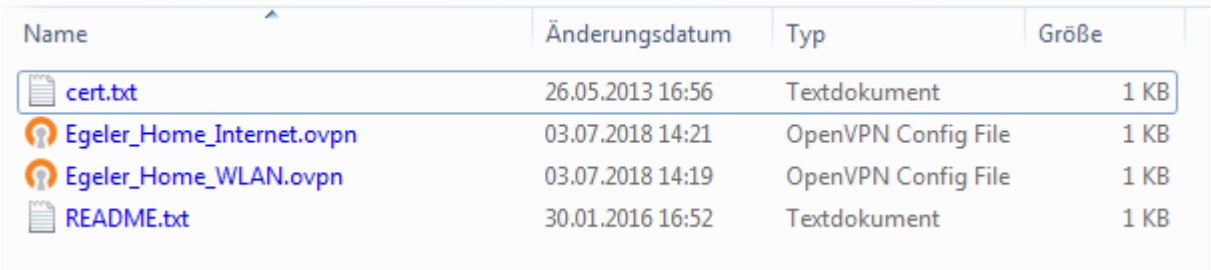
- Die zu installierende App heißt einfach **OpenVPN Connect**. Die in iOS enthaltene VPN-Unterstützung funktionierte nicht.
- Das Übertragen der wie zuvor (diesmal für iPad generierten, modifizierten und umbenannten) Konfigurationsdatei auf das iPad erwies sich als schwierig, weil die Daten unter iOS nicht ohne weiteres zwischen den Apps austauschbar sind. Am Ende wählte ich den nicht empfohlenen Weg über E-Mail.
- Das HTTP-Problem bestand auch hier (und wurde mit der gleichen Maßnahme auf dem Server mitgelöst)
- Das DNS-Problem bestand nicht.

Windows Client

- Bei der Windows-Version von OpenVPN wird mittlerweile ein GUI mitgeliefert
- Braucht Administratorrechte wegen Änderung der Netzwerkkonfiguration
- Kann Verbindung starten und anhalten
- Zeigt Verbindungsprotokoll in eigenem Fenster
- Eigentliche Konfiguration in Dateien im Verzeichnis „...\config“

Windows Client

- Inhalt von Verzeichnis „...\config“



Name	Änderungsdatum	Typ	Größe
cert.txt	26.05.2013 16:56	Textdokument	1 KB
Egeler_Home_Internet.ovpn	03.07.2018 14:21	OpenVPN Config File	1 KB
Egeler_Home_WLAN.ovpn	03.07.2018 14:19	OpenVPN Config File	1 KB
README.txt	30.01.2016 16:52	Textdokument	1 KB

Windows Client

- Inhalt von „.vpn“-Datei (Teile)
  - auth SHA512
  - cipher AES-256-CBC
  - secret cert.txt
  - dev tun
  - remote taking-it-seriously.de
  - ifconfig 192.168.249.2 192.168.249.1
  - ...

Windows Client

- Inhalt von „cert.txt“
  - #
  - # 2048 bit OpenVPN static key
  - #
  - -----BEGIN OpenVPN Static key V1-----
  - 78ec5a56858cf75f7c07644aa5ca95d1
  - ...
  - -----END OpenVPN Static key V1-----

Nützliche Links

- Grundsatzartikel Wikipedia zu Zertifikaten
  - https://de.wikipedia.org/wiki/Digitales_Zertifikat
- Wikipedia-Artikel zu OpenVPN
  - https://de.wikipedia.org/wiki/OpenVPN
- Turris Website: OpenVPN Anleitung für „advanced“ users
  - https://www.turris.cz/doc/en/howto/openvpn
- OpenVPN Website: HOWTO
  - https://openvpn.net/index.php/open-source/documentation/howto.html
- PfSense OpenVPN HOWTO
  - https://www.netgate.com/docs/pfsense/vpn/openvpn/openvpn-remote-access-server.html
- Vergleich VPN-Protokolle
  - https://thebestvpn.com/pptp-l2tp-openvpn-sstp-ikev2-protocols/