"Anonym" surfen mit der opnSense und Cyberghost

Vorwort

Viele Leute haben sicherlich ein Abo bei einen VPN-Anbieter, beispielhaft hier Cyberghost. Normalerweise läuft es so, dass man sich eine App auf Smartphone oder PC installiert, die jeglichen Verkehr durch das VPN an einen anderen Punkt (ein anderes Land) ins Internet einleitet. Das hat nun im Heimnetz mehrere Nachteile:

- Die Anzahl der Geräte ist beschränkt, je nach Abo
- Installation der Software auf jeden Client
- Keine Möglichkeit der Installation auf diversen Endgeräten wie Smart-TV, Drucker, Hausautomation, ...
- Keine Möglichkeit, das VPN zu „erzwingen“

Gesucht wird also eine Möglichkeit, direkt am Router zu steuern, welcher Verkehr über das VPN geleitet wird. Dazu gibt es diverse Router, die die Möglichkeit bieten, per VPN-Client eine Verbindung aufzubauen. Wer nun eine opnSense in Verwendung hat, hat alle Mittel, die er dazu benötigt „on Board“.

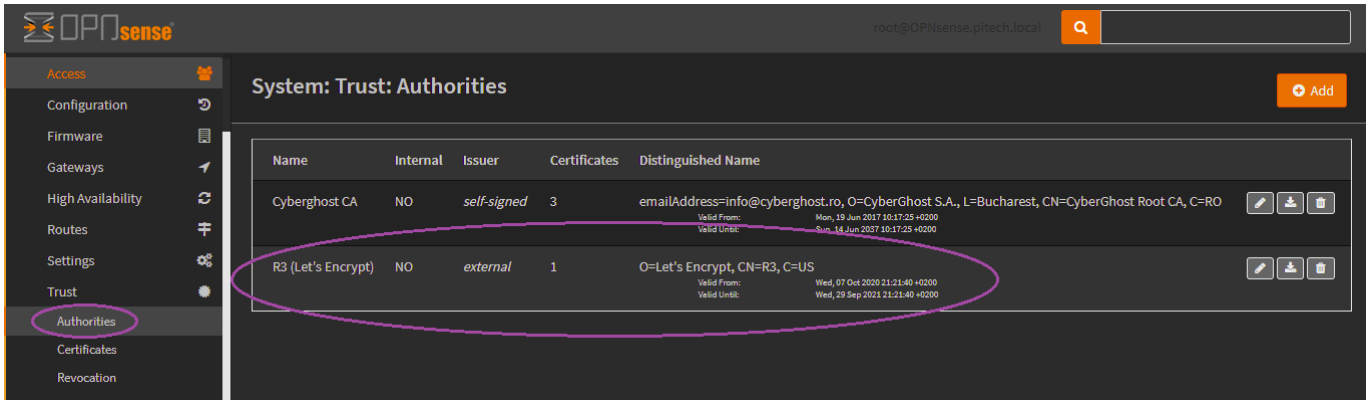
Ich beziehe mich in der folgenden Anleitung auf die jetzt gerade aktuelle opnSense-Version 21.x, in den Screenshots verwende ich aus Darstellungesgründen das dunkle Theme „rebellion“ und Sprache „en“.

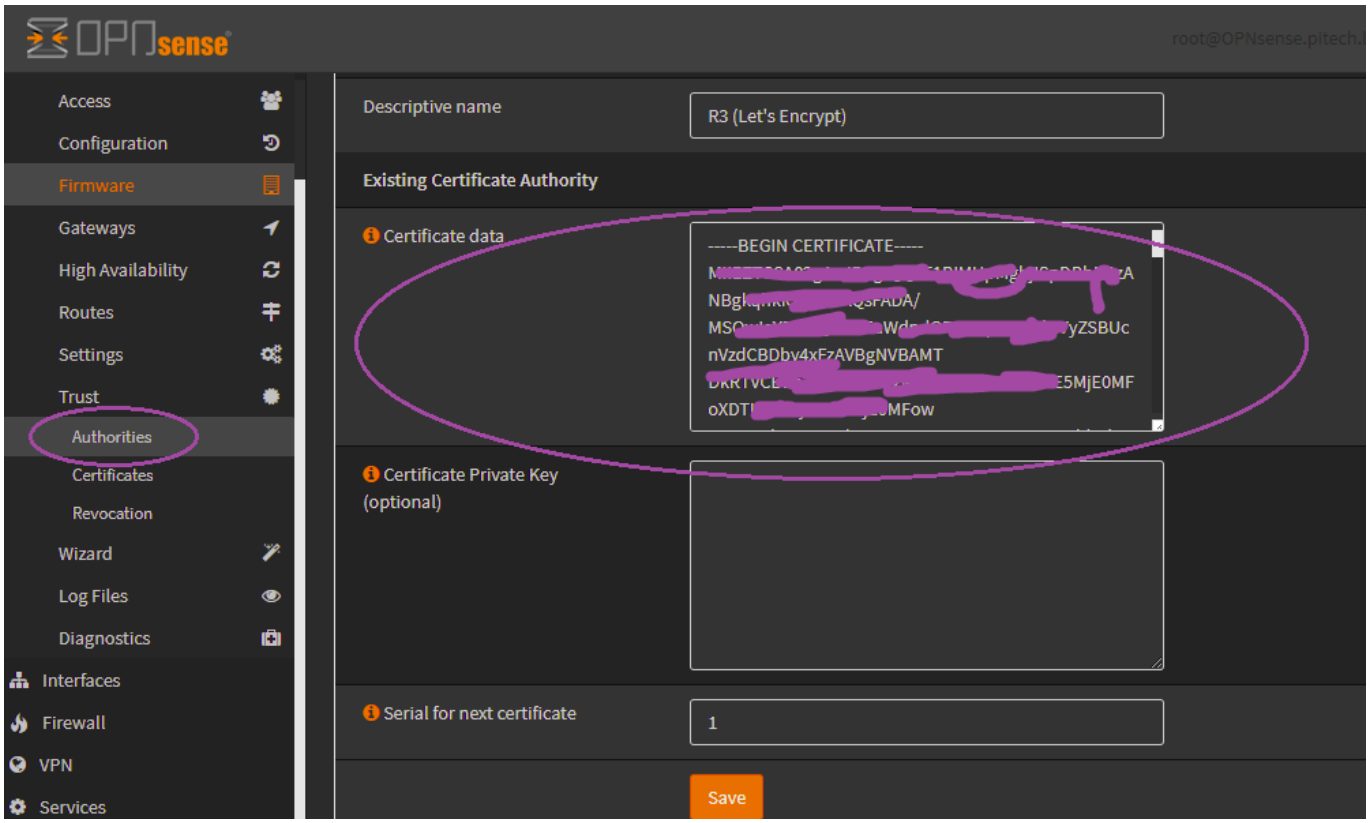
Einrichtung

Nach dem Einrichten (OpenVPN) des gewünschten Landes in seinem Cyberghost-Account lädt man die Einstellungen runter (Zertifikate, Kennwort, Servername, ...). Man kann auch gleich mehrere Tunnel anlegen, um später umzuschalten, eine Redundanz zu haben, oder verschiedene Geräte über verschiedene Wege zu tunneln.

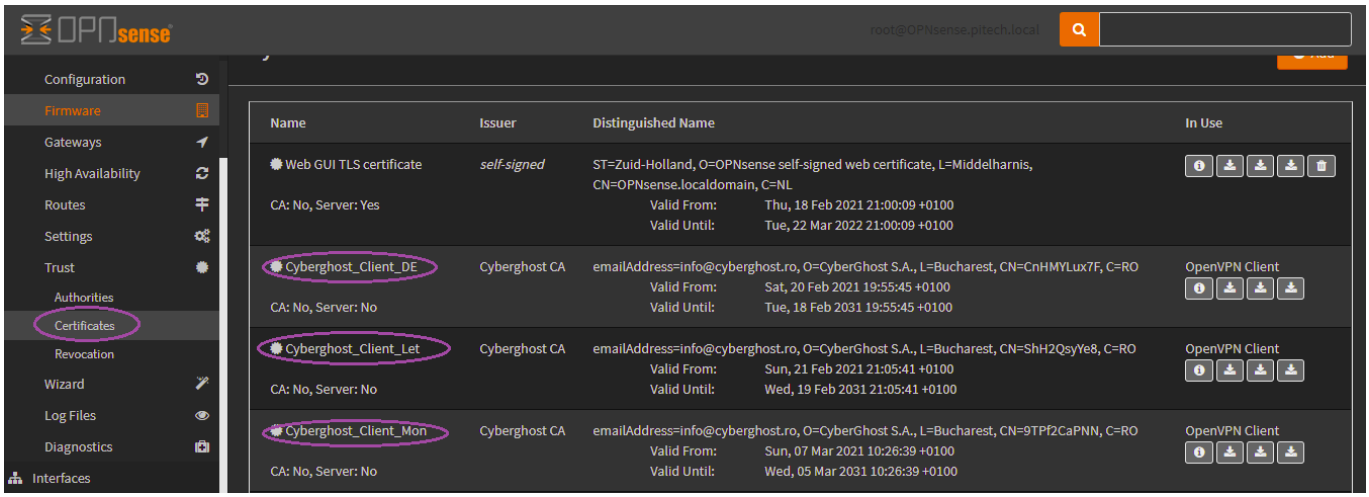
1. Zertifikate

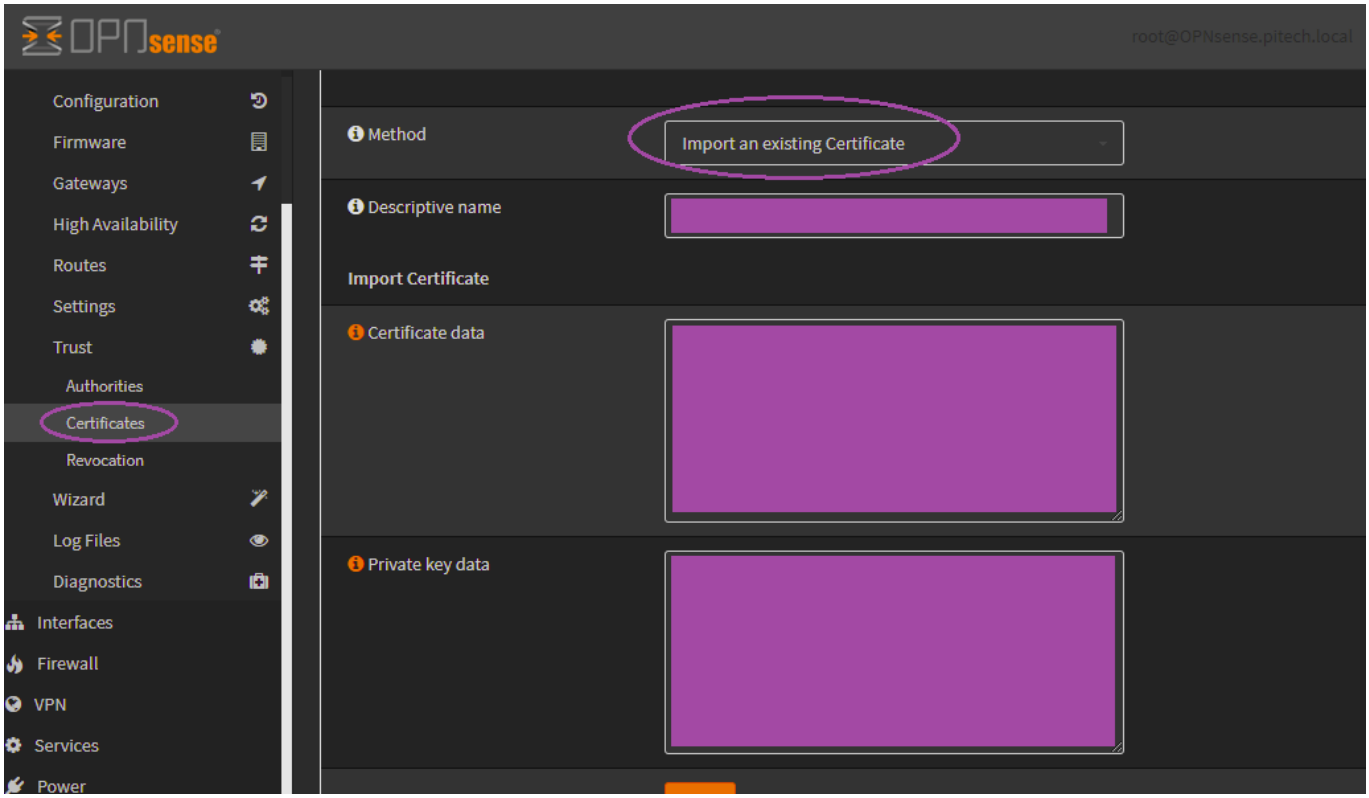
Als erster Schritt wird das Cyberghost-Zertifikat eingespielt, dazu einfach unter Trust/Authorities auf Add klicken und den Inhalt der ca.crt per Copy&Paste ins Feld Certificate data kopieren... fertig. Das ist nur einmal nötig, auch bei mehreren Tunneln, klar, die ca.crt ist immer das selbe Zertifikat, man sollte nur auf das Ablaufdatum achten, dann einfach eine neue bei Cyberghost runterladen und drüberspielen.





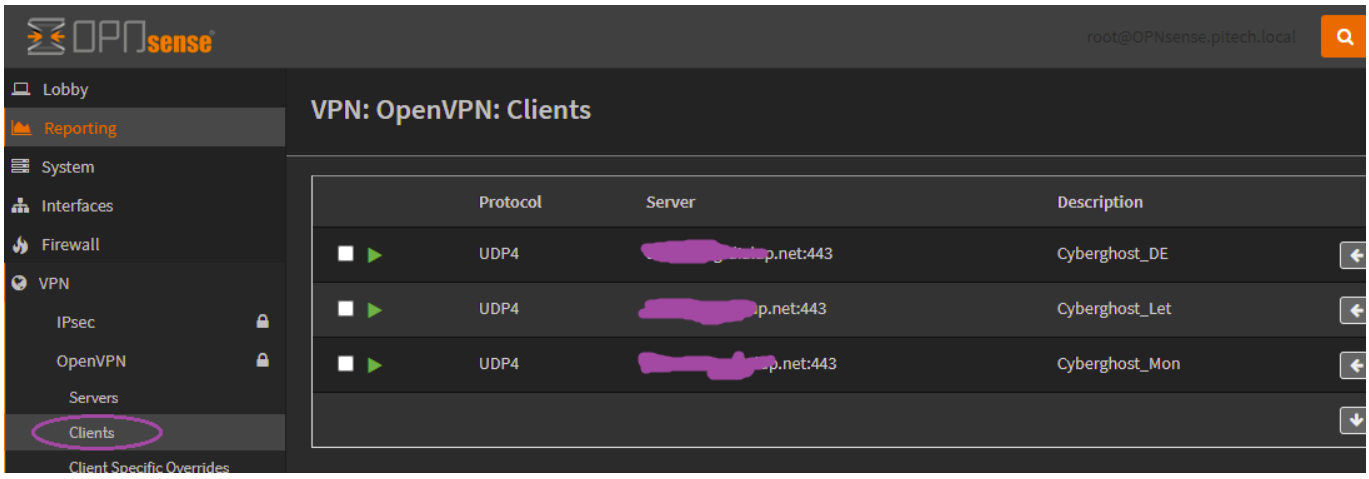
Nun folgt das Anlegen der Zertifikate für die Tunnel. In der Konfigurationsdatei von Cyberghost findet man eine client.crt und eine client.key. Unter Trust/Certificates legt man nun ein neues Zertifikat an und kopiert den Inhalt oben genannter Dateien in die entsprechenden Felder. Das kann man auch gleich mehrfach machen, falls man mehrere Tunnel anlegt. Im Screenshot hier habe ich beispielhaft mal 3 Zertifikate (Deutschland, Lettland, Monaco) angelegt.

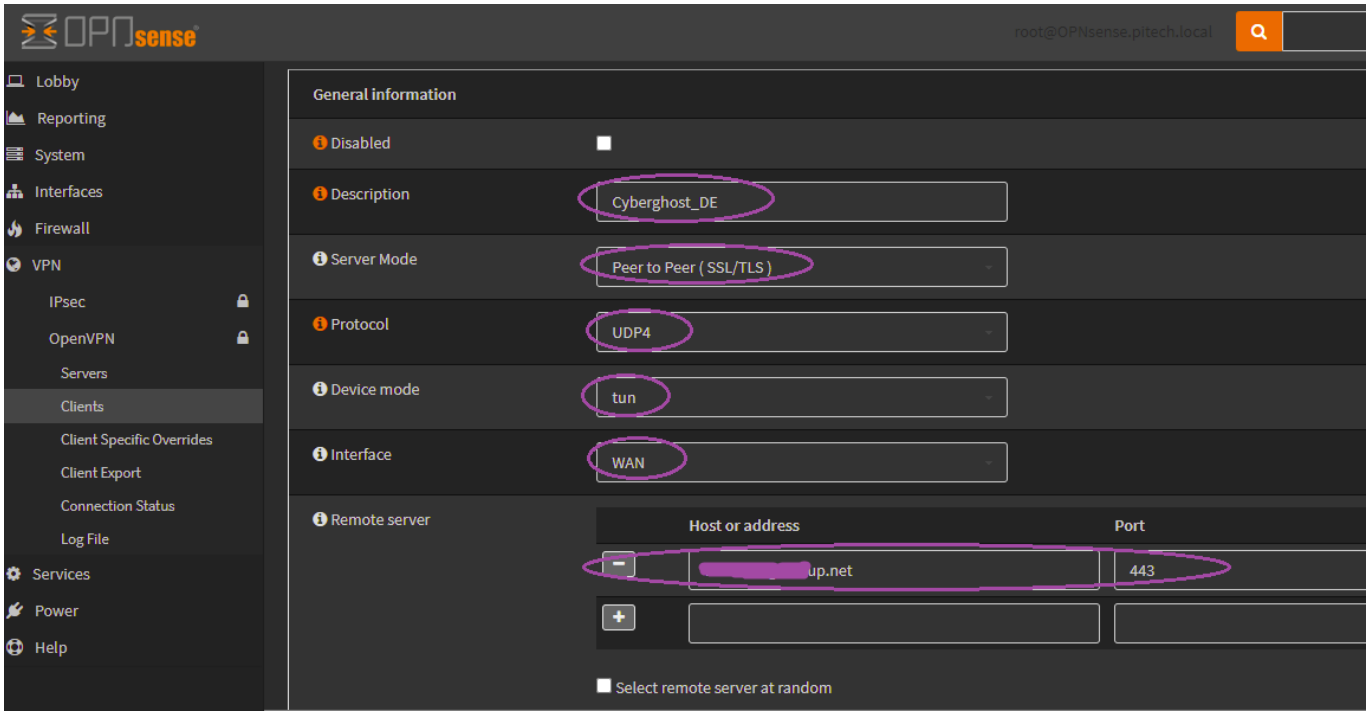


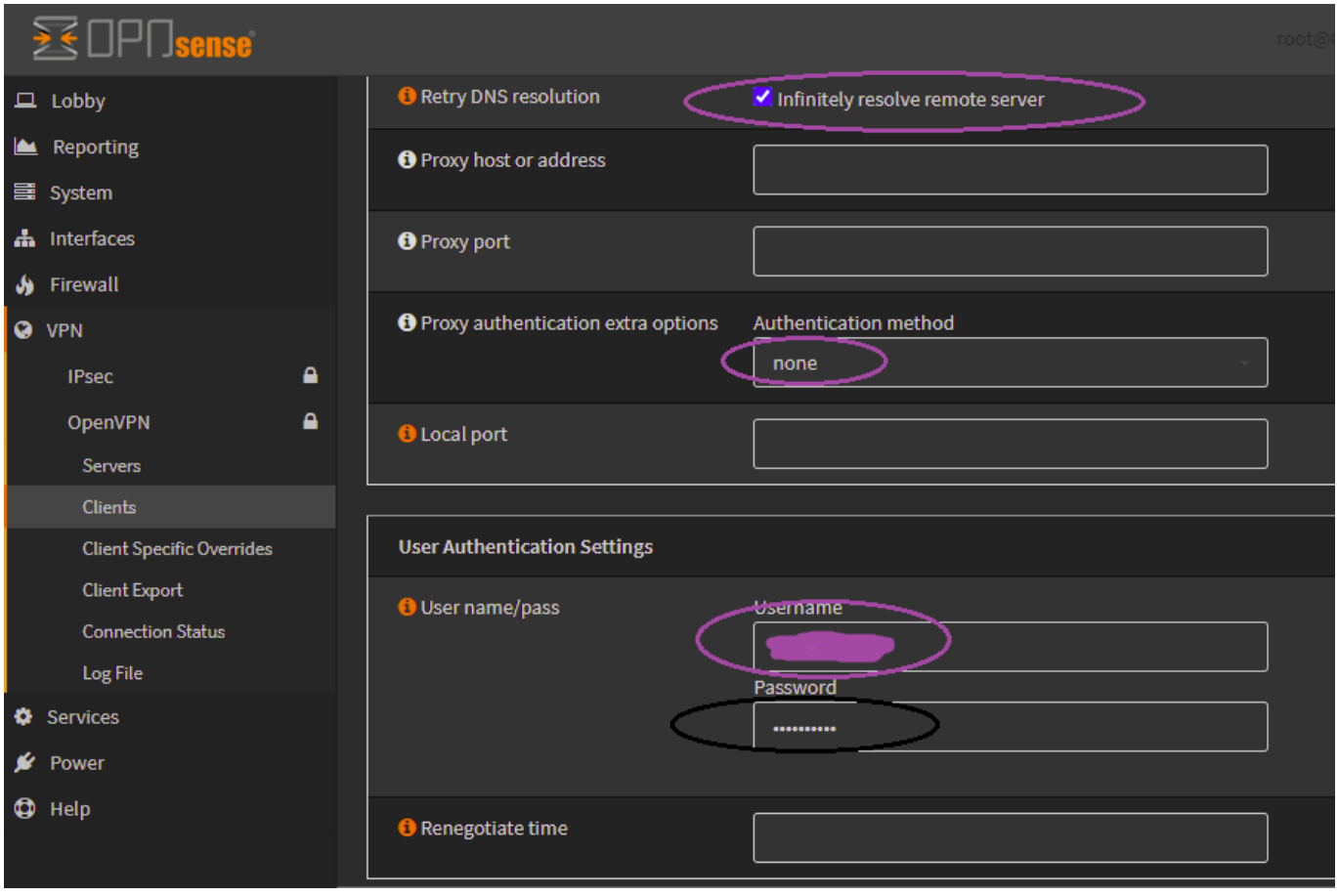


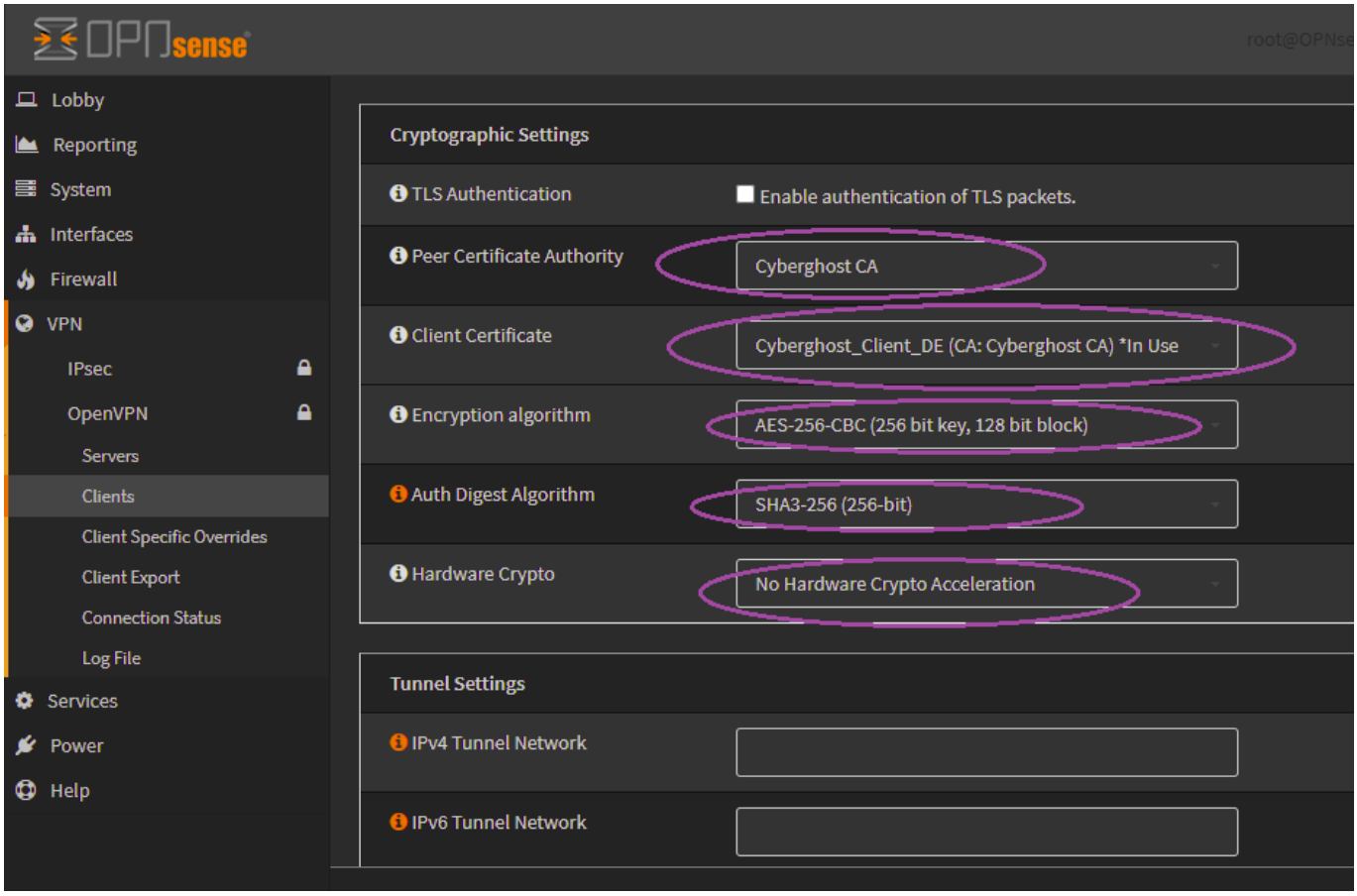
2. VPN

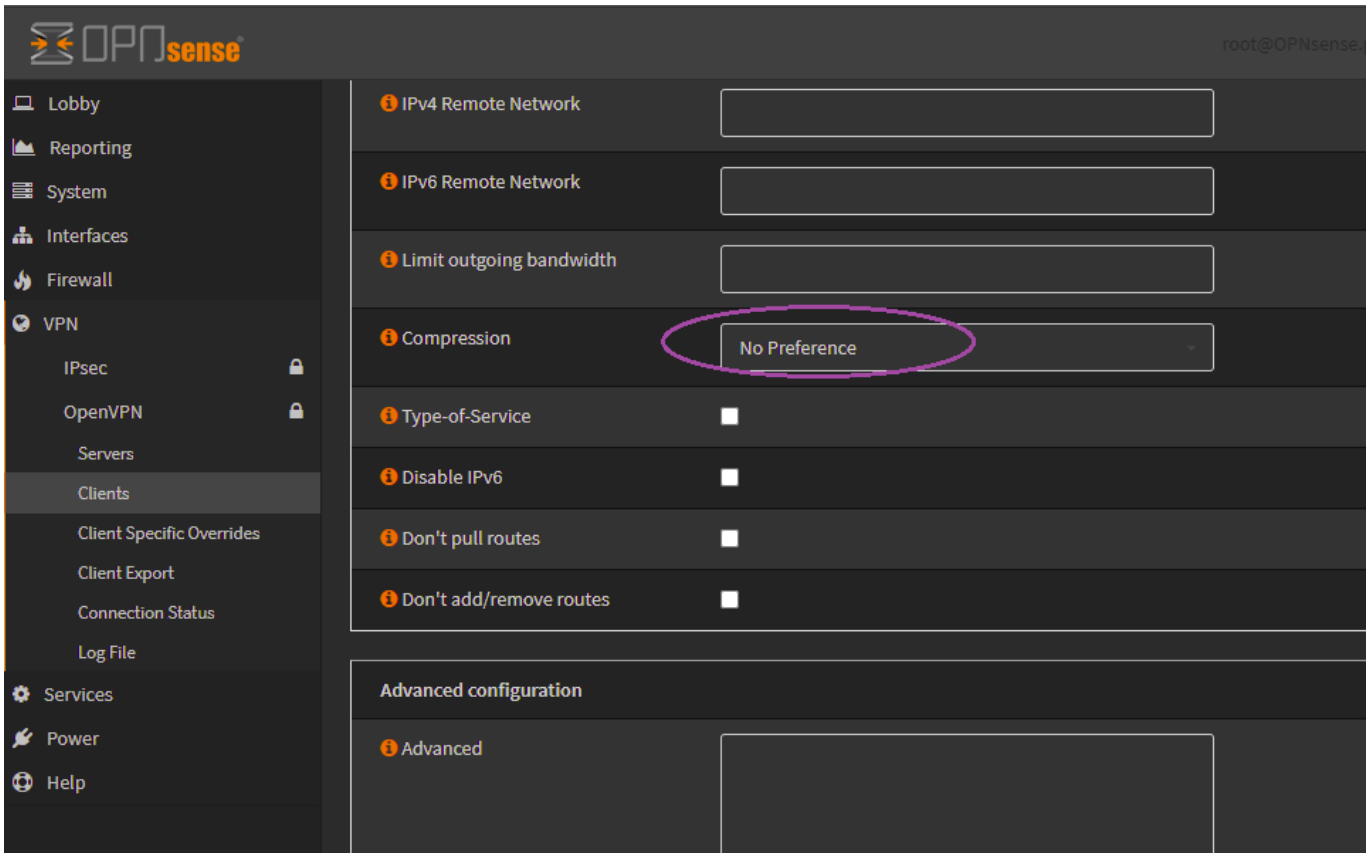
Nun folgt das Anlegen des Tunnels (oder der Tunnel). Dazu geht man zum Menüpunkt VPN/OpenVPN/Clients und legt hier einen neuen Tunnel an. Man übernimmt einfach die Werte aus der openvpn.ovpn-Datei von Cyberghost, bzw. die online im Cyberghost angezeigten Parameter. Im folgenden fünf Screenshots der etwas längeren Eingabemaske für den Tunnel, alles, was ausgefüllt werden muss, ist lila markiert.




OPNsense
root@OPNsense.pitech.local
Lobby
Reporting
System
Interfaces
Firewall
VPN
IPsec
OpenVPN
Servers
Clients
Client Specific Overrides
Client Export
Connection Status
Log File
Services
Power
Help
General information
Disabled
Description
Cyberghost_DE
Server Mode
Peer to Peer (SSL/TLS)
Protocol
UDP4
Device mode
tun
Interface
WAN
Remote server
Host or address
Port
up.net
443
Select remote server at random


OPNsense
Lobby
Reporting
System
Interfaces
Firewall
VPN
IPsec
OpenVPN
Servers
Clients
Client Specific Overrides
Client Export
Connection Status
Log File
Services
Power
Help
Retry DNS resolution
Infinitely resolve remote server
Proxy host or address
Proxy port
Proxy authentication extra options
Authentication method
none
Local port
User Authentication Settings
User name/pass
Username
Password
Renegotiate time

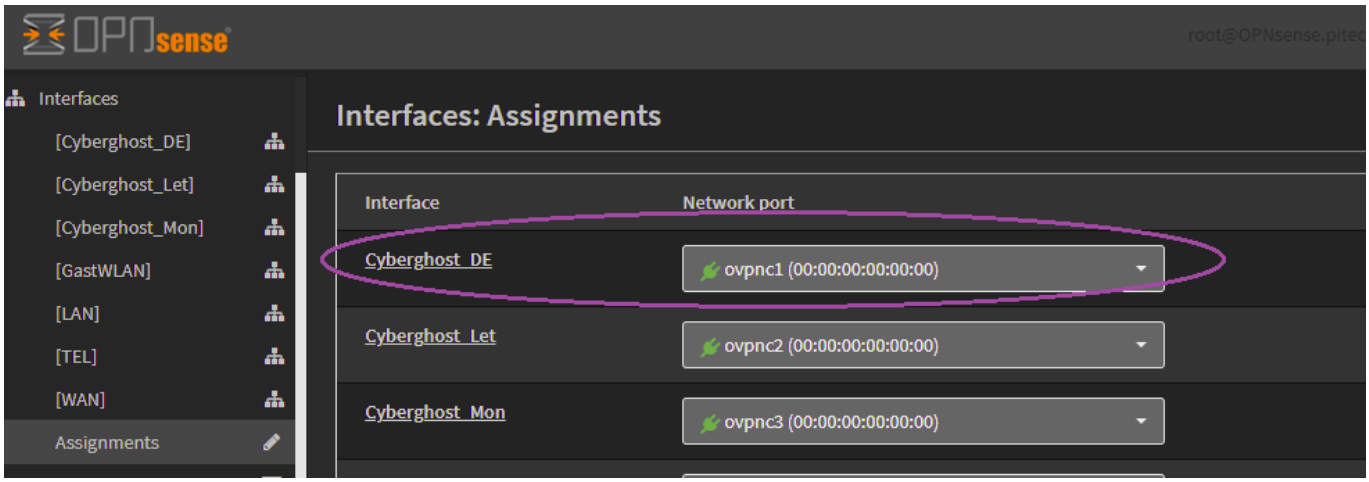


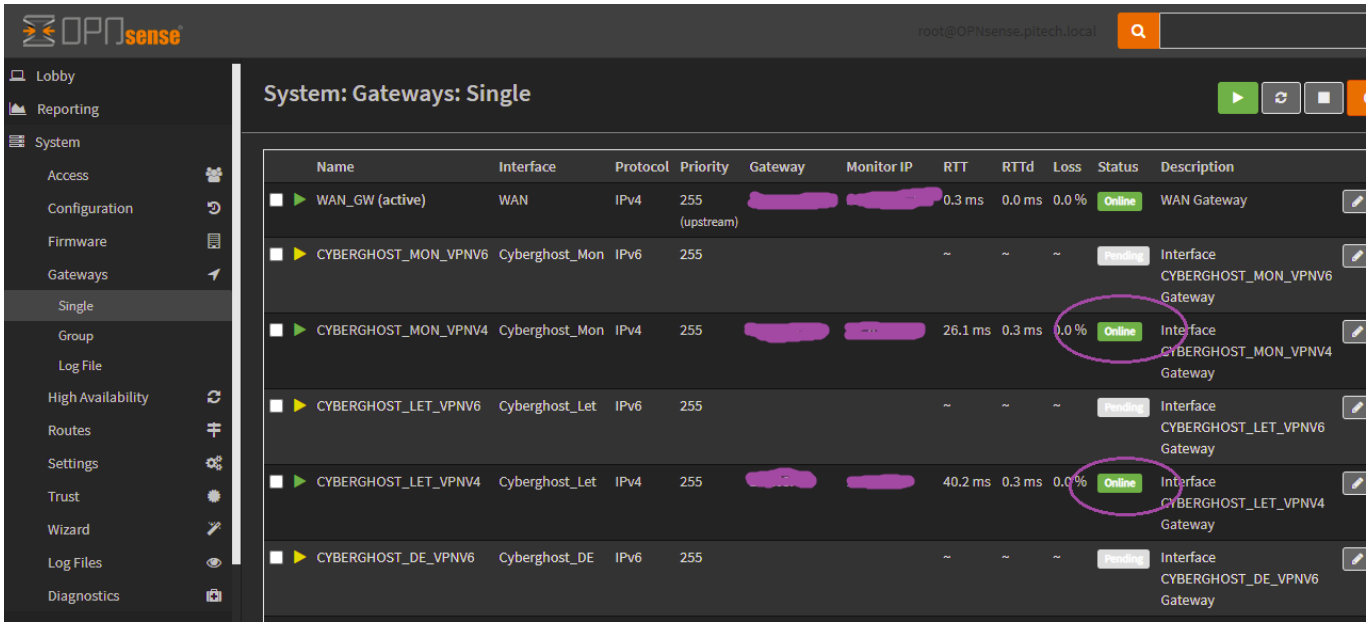


3. Schnittstelle erzeugen

Um nun mit dem Tunnel arbeiten zu können, benötigen wir dessen (dynamisches) Gateway. Hierzu

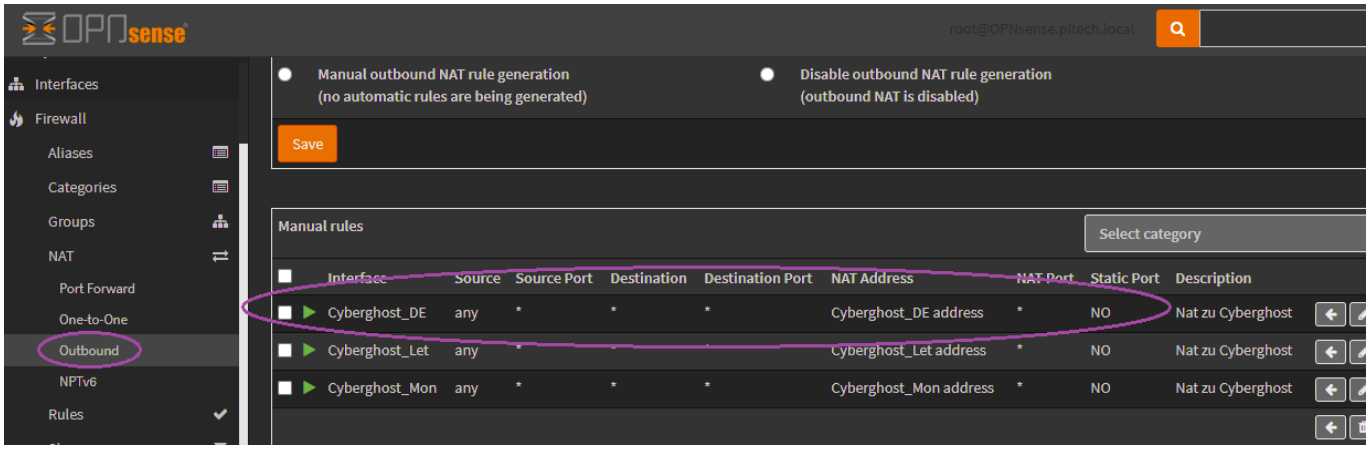
muss eine imaginäre Schnittstelle erzeugt werden. Wir gehen zum Menüpunkt Interfaces/Assignments. Als Name hab ich hier beispielhaft wieder Cyberghost_DE vergeben, für den deutschen Tunnel. Wie man hier sieht, werden die Netzwerk-Ports der Tunnel einfach durchnummeriert vergeben, hier eben ovpnc1. Wenn man auf einmal mehrere Tunnel anlegt, muss man sich halt im Schritt zuvor die Portnamen merken, oder man legt einen Tunnel komplett an, bevor man den nächsten erzeugt. Nach der Zuweisung sieht man sich die Eigenschaften von „Cyberghost_DE“ nochmal an. Das Interface muss aktiviert werden, bei IP wird nichts eingetragen. Durch Aktivieren dieser „imaginären“ Schnittstelle ohne IP passiert nun folgendes: Unter System/Gateways/Single erkennt man, dass automatisch das Gateway mit der gerade aktuellen (dynamischen) Gateway-IP angelegt wurde, hier sieht man auch schon mal, ob der Tunnel „steht“ (Das sieht man auch unter VPN/OpenVPN/Connection Status oder im Dashboard). Mit diesem Gateway können wir im nächsten Schritt arbeiten, um die Regeln für die Benutzung dieses Tunnels zu bestimmen.





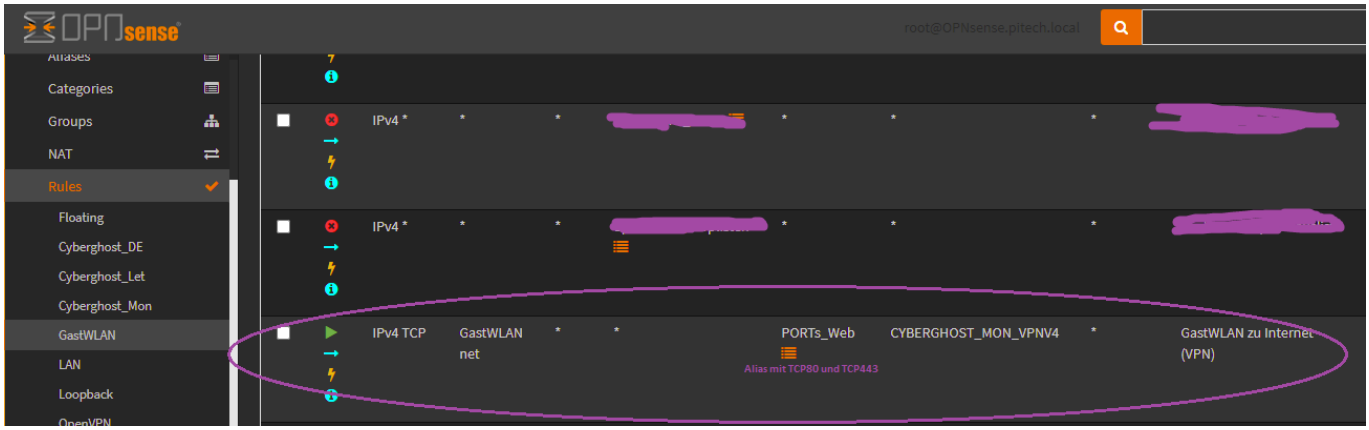
4. NAT -Regeln

Für jeden Tunnel brauchen wir eine Regel, welche die Pakete auf die Gateway-Adresse des Tunnels umsetzt. Hierzu unter Firewall/NAT/Outbound eine Regel erzeugen.



5. Firewallregel

Hier benutzt man eine Technik, die sich allgemein „Policy Route“ nennt. Man erstellt ganz gewohnt eine Allow-Regel im entsprechenden Netz, im unteren Teil der Maske kann man ein Gateway wählen, welches benutzt wird, wenn die Regel „zieht“. Das Beispiel unten zeigt: *Leite alle (*) mit Ports 80+443 durch den Tunnel Monaco.* Der Phantasie sind nun keine Grenzen gesetzt, Man erzeugt sich einen Alias mit 5 Geräten, welche nach Lettland sollen, dann einfach vorher die selbe Regel setzen: *Leite die 5 Geräte zum Tunnel Lettland*... alle anderen kommen dann nach Monaco.

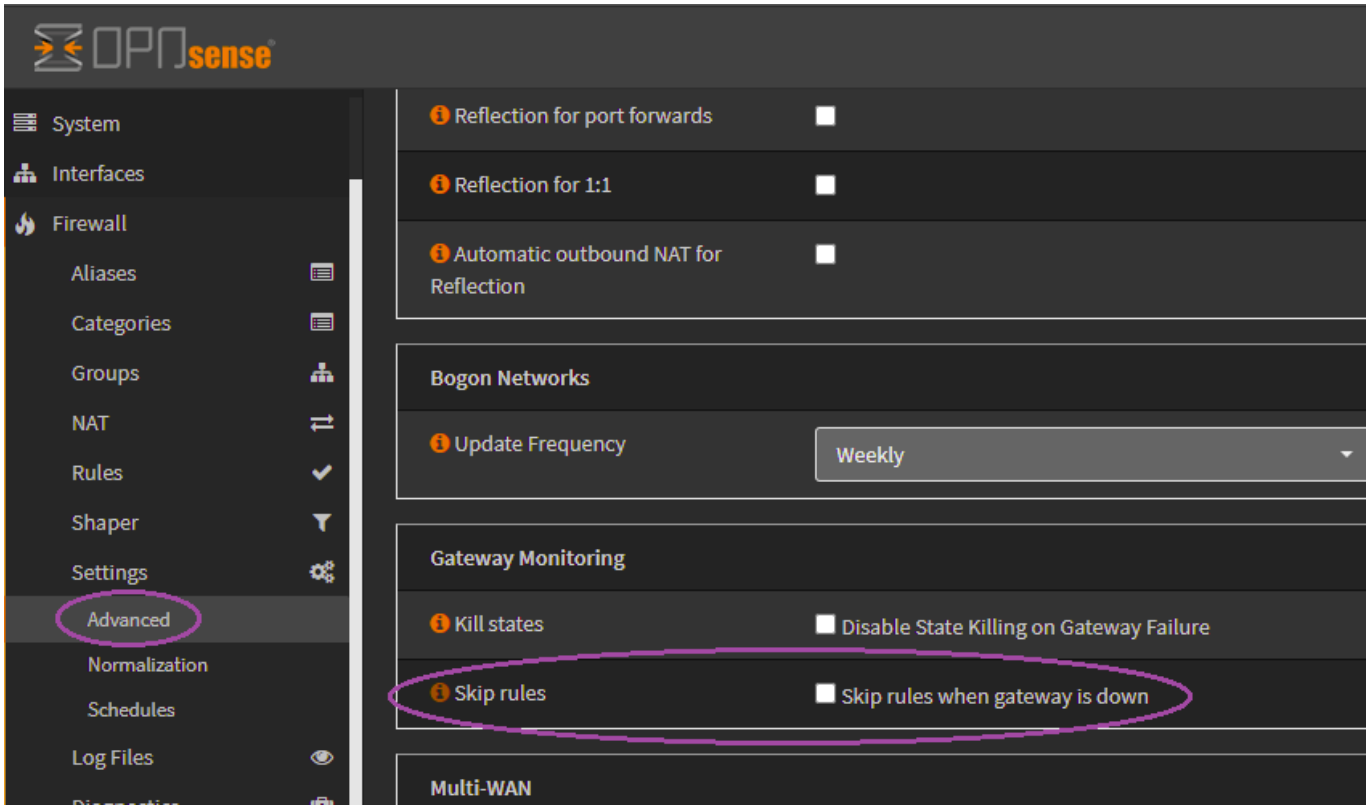


Man kann natürlich auch alle Ports in den Tunnel lassen, nicht nur 80 und 443!

6. Gedanken zur "Gateway-Down"-Problematik

So, zum jetzigen Zeitpunkt funktioniert alles, außer einer Kleinigkeit, die wir nicht bedacht haben. Die opnSense reagiert wie viele Firewalls: Wenn ein Gateway nicht „Online“ ist, dann wird der ganze Verkehr ans Standardgateway (WAN) geleitet. Falls man also dieses Verhalten so möchte, ist hier Schluss: Wenn VPN down, dann surfen alle „ungeschützt“ weiter.

Wenn man jedoch möchte: Falls VPN down, dann kein Internet... Ist noch etwas nötig, um dieses Verhalten herbeizuführen. Die Quick&Dirty Lösung wäre, unter Firewall/Advanced den Haken zu setzen, Regeln zu ignorieren, deren Gateway down ist.

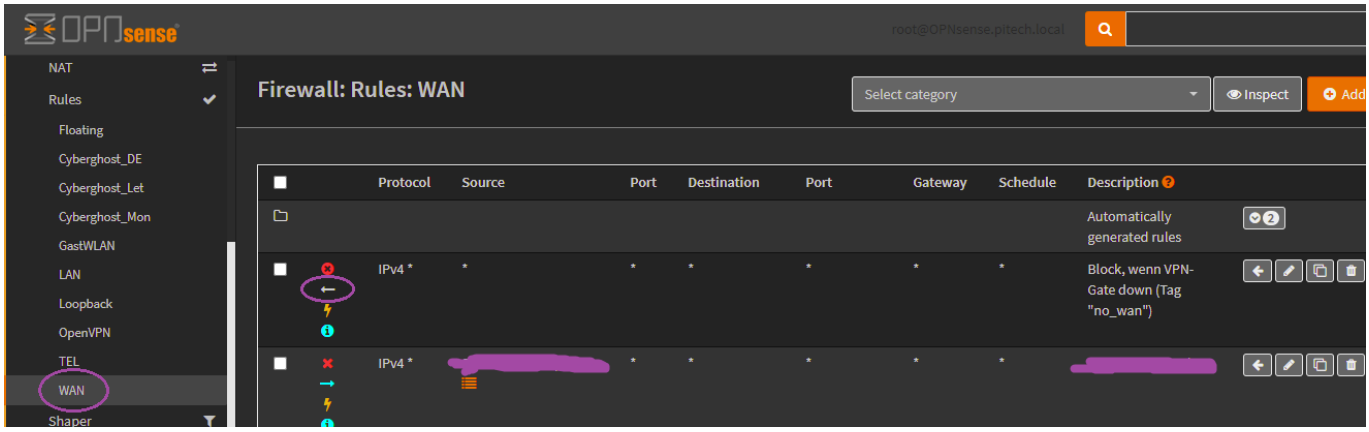


Das wirkt aber global für alle Regeln und ist manchmal unflexibel! Man muss halt im Hinterkopf haben, dass es so ist.

Lösung 1 - Der GatewayDown-Haken bleibt aus

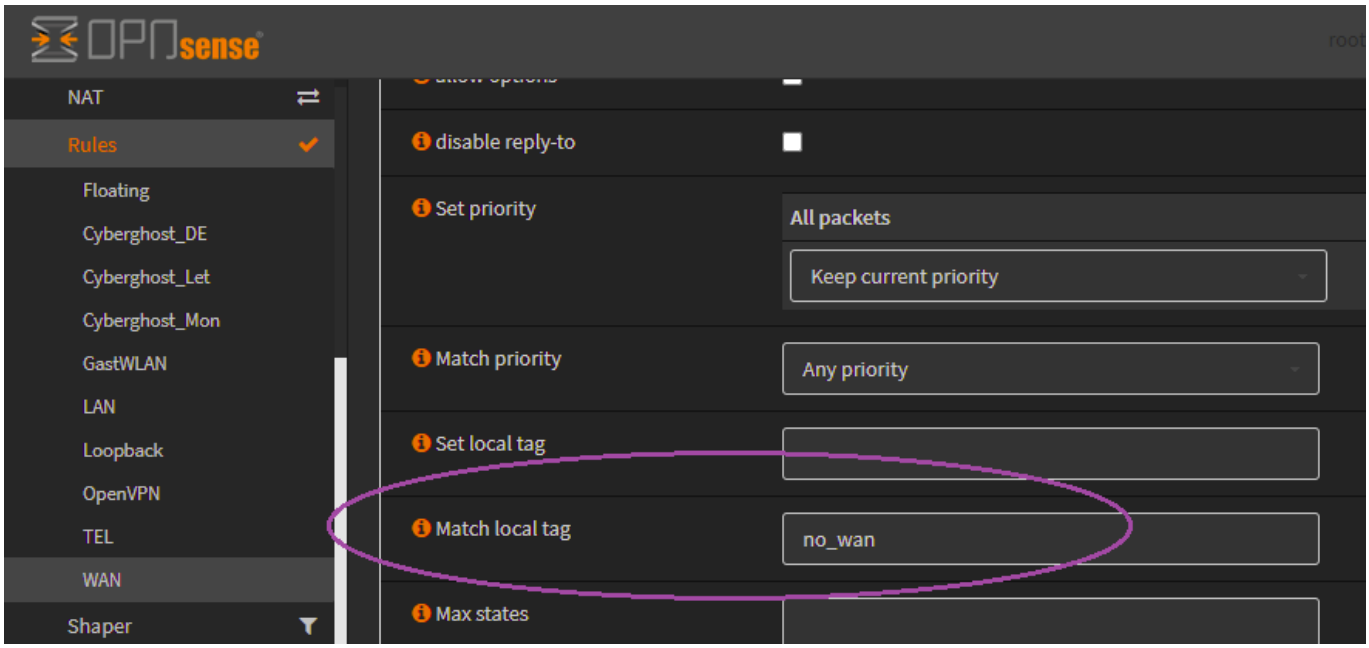
Ein Paket läuft in die Firewall rein und wird da verarbeitet bei den Regeln der Netzwerkkarte. Um bei meiner Beispielkonfiguration zu bleiben, kommen als erstes die Regeln GastWLAN IN. Da haben wir ja festgelegt, dass der Verkehr zum Tunnel gehen soll. Anschließend werden aber noch die Regeln WAN OUT abgearbeitet (Falls der Verkehr eben grad mal nicht durch den Tunnel gehen kann). ...und hier setzen wir an!

Jede Regel kann dem durchlaufenden Paket ein „TAG“ mitgeben, das wir frei verwenden können innerhalb der Firewall. Wir erzeugen eine Regel unter „WAN“, und zwar diesmal (wichtig) als OUT und als ERSTE Regel: Jeglicher Verkehr der über WAN raus geht und als Tag „no_wan“ hat wird rejected!

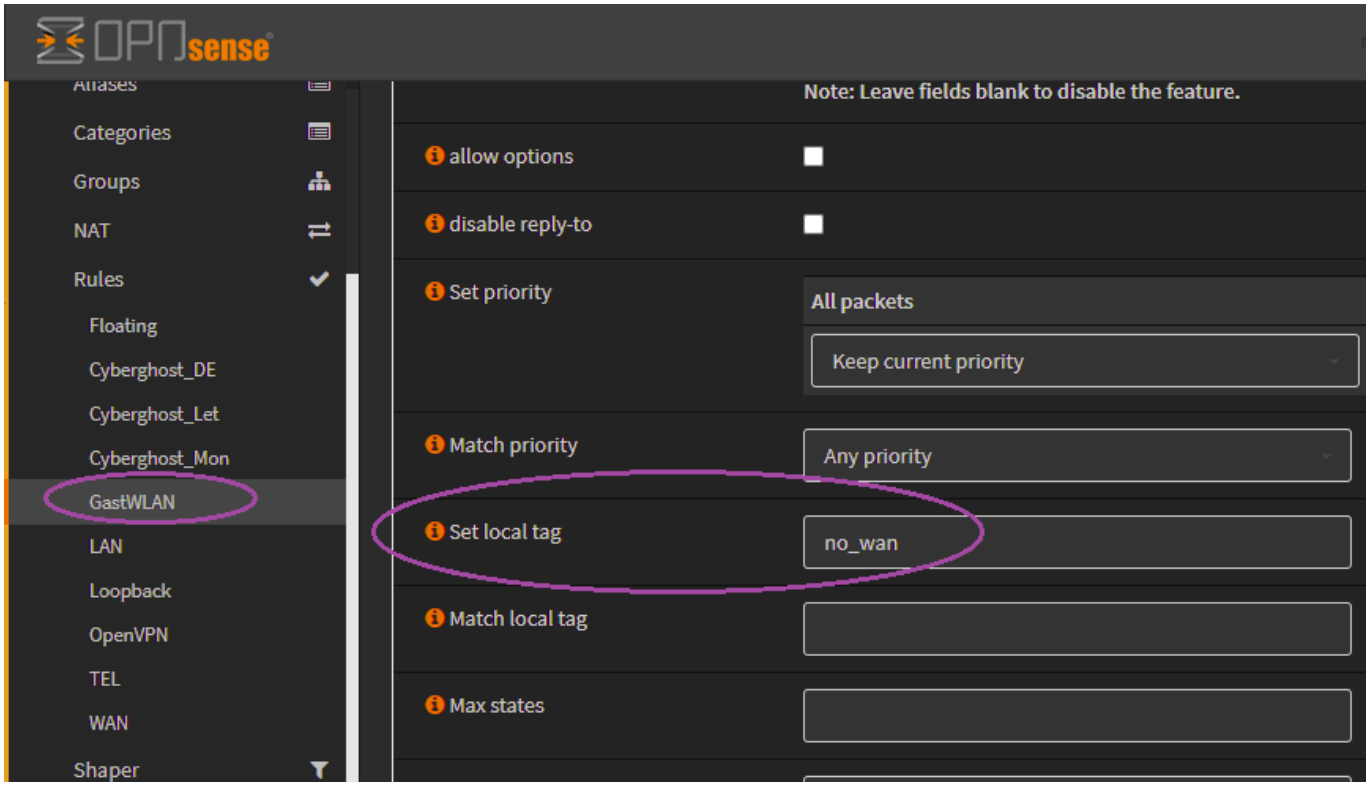


Anmerkung: Man sieht in obigen Screenshot, dass die erste Regel „rejected“, die zweite „blockt“. Ich habe es mir angewohnt, alles was nach aussen verboten wird zu rejecten, alles was reinkommt zu

blocken. Von aussen schalte ich einfach auf Stur, aber von innen ist eine Rückmeldung nett, dann läuft z.B. der Browser nicht in einen ewig langen Timeout, sondern weiß gleich, was Sache ist...



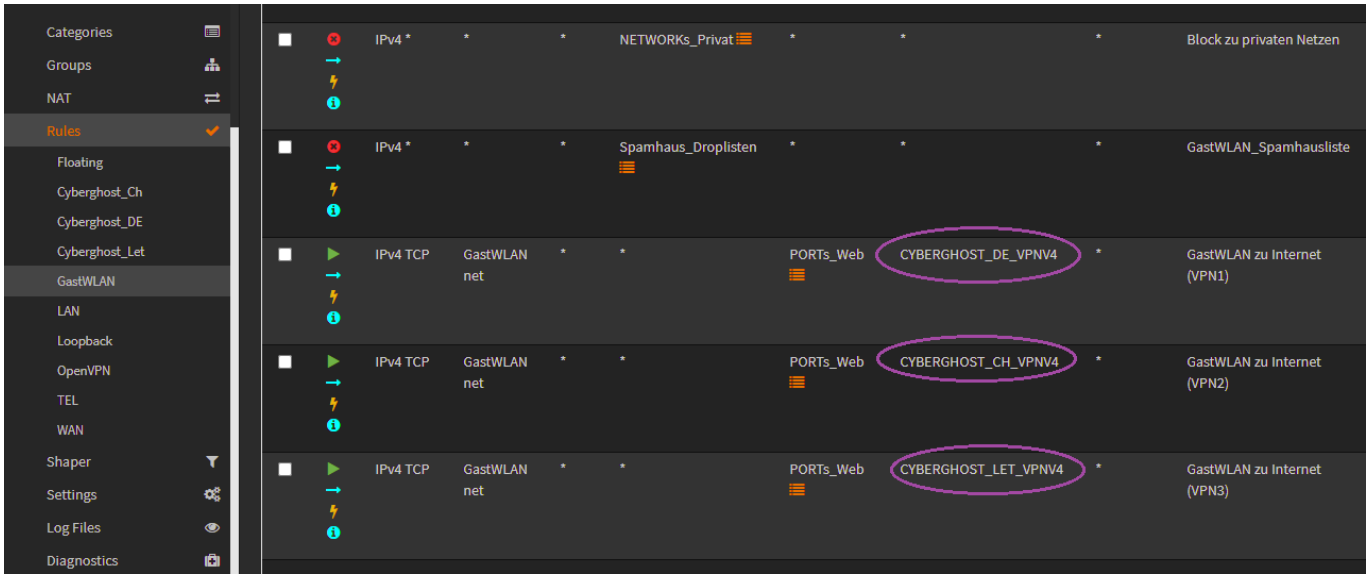
Diese Regel allein bewirkt jetzt gar nichts und kann einfach für immer so bleiben. Wir wissen nur: Wenn wir in irgendein Paket in das Tag-Feld „no_wan“ reinschreiben, wird dieses Paket niemals die Firewall Richtung WAN verlassen. Also können wir je nach Belieben die Tunnelregeln, bei der wir ans VPN-Gateway umleiten, um das „no_wan“-Tag erweitern, wenn gewünscht.



Die Rechner, welche in der Tunnel-Regel umgeleitet werden surfen über das VPN. Wenn der Tunnel down ist, wird rejected.

Lösung 2 - Der GatewayDown-Haken wird gesetzt

Wenn man das Verhalten der Firewall so ändert, dass eine Regel komplett ignoriert wird, wenn das Gateway down ist, kann man mehrere Tunnel bequem redundant nacheinander schalten:



Erklärung: Wenn der erste Tunnel (hier im Beispiel DE) down ist, wird die Regel ja diesmal ignoriert, also wird alles über „CH“ ins Internet geleitet, ist auch dieser down, geht alles über den Tunnel „LET“. Wenn alle drei Tunnel down sind, gibt's kein Internet. Möchte man es so gestalten, dass bei Ausfall aller drei VPN's der direkte Zugriff ins Internet erfolgt, fügt man danach einfach noch eine Regel ein, die den Verkehr ans Standardgateway erlaubt.