

QuantenVPN

- Das Konzept
- Die Netze
- Basis für Client Config
- DNS
- Verbindungs Check

Das Konzept

Multihop

Mullvad bietet ein Multihop Feature, bei welchem der Traffic durch einen weiteren Entry Server geleitet wird. Dabei besteht nur eine VPN Verbindung, nämlich mit dem Exit Server, während der Entry lediglich den Traffic weiterleitet. Das funktioniert, da jeder Mullvad Server einen eindeutigen "Multihop Port" hat, welcher auf allen Servern gleich ist. So hat bspw. der Server `de-fra-wg-001` den Multihop Port `3053`. Konfiguriert man im Client nun den VPN Endpoint `se-sto-wg-005:3053`, weiß der Server `se-sto-wg-005` dass der Traffic zu `de-fra-wg-001` geleitet werden soll.

Wir packen auf dieses Setup noch einen drauf und fügen unseren eigenen Server *Sentinel* als "vor-Entry" ein. Dadurch ergibt sich folgendes Bild:

```
Client --1--> Sentinel --2--> Mullvad Entry --2--> Mullvad Exit --> Internet
```

Mullvad sieht somit zu keinem Zeitpunkt die wahre Client IP und der Mullvad Exit zu keinem Zeitpunkt beides: Traffic und Sentinel IP.

Eigenes Netz vs Weiterleiten

Wir hätten ebenfalls den Traffic einfach weiterleiten können, ähnlich wie es Mullvad implementiert. Das hätte jedoch die folgenden Nachteile gehabt:

- Änderungen am Setup (bspw. Mullvad Server wechsel) hätten im Client **und** Sentinel Anpassungen benötigt. Denn der Client benötigt den richtigen (Exit Server) Public Key und Sentinel die richtige IP:Port des Entry Servers.
- 1 Client = 1 Mullvad Slot

Ein eigenes Wireguard Netz zwischen Client und Sentinel bietet folgende Vorteile:

- Einfaches Wechseln der Mullvad Server
 - Keine Client Anpassung nötig
 - Kann im Hintergrund passieren ohne dass der Client etwas davon mitbekommt
- Eigener DNS Server im VPN Netz
 - Dieser ist optional. In der Client Config kann ein eigener oder auch der Mullvad DNS Server eingetragen werden.
- N Clients auf 1 Mullvad Slot

Die Netze

Es stehen die folgenden Netze zur Verfügung.

Netz	UDP	TCP	Subnet	Mullvad Hops	Server Wechsel
AdaNet	60524	8443	10.128.0.0/24	0	-
CoddNet	53978	8080	10.192.0.0/24	1 (Nur DE)	1x/h
HuffNet	57937	80	10.224.0.0/24	2	1x/Tag
HopperNet	48074	443	10.240.0.0/24	2	Min. 1x/h
BadenTunnel	37264		10.248.0.0/24	1	Privat

AdaNet hat **kein Mullvad**, sondern einen direkt Exit bei Sentinel!

Wofür welches Netz

Netz	Use Case	Beispiel
AdaNet	Öffentliches WLAN absichern	Online Banking
CoddNet	Hohe Privacy für normale Tätigkeiten	Allgemeines online browsen
HuffNet	Höhere Privacy für zeitlich längere Tasks	Torrent
HopperNet	Höchste Privacy für maximale Verschleierung	Because we can.

Das AdaNet und CoddNet nur für "regulären" Traffic verwenden!

Kommunikation der verbunden Clients untereinander ist blockiert

Basis für Client Config

Es müssen alle Felder der Art `<Text>` geändert werden. Die Zeichen `<` und `>` sind zu entfernen.

Was wird benötigt

Allgemeine Informationen (erhaltet ihr von uns):

- Sentinel IP
- Sentinel Wireguard Public Key
- Secret für udp2raw

An euch angepasst:

- Private Key
 - Option 1: Ihr generiert eure Keys (Private/Public) und gebt uns den Public Key.
 - Zum Erstellen von einem Keypair im Terminal: `wg genkey | tee private | wg pubkey`
`> public`
 - Option 2: Wir generieren die Keys und geben euch den Private Key
- Von uns euch zugewiesene Client IP (für jedes Netz eine)

Die MTU muss für udp2raw herunter gesetzt werden. In den unteren Configs auf jeden Fall die MTU lassen und nicht verändern! Das gilt auch für Verbindungen ohne udp2raw

Wireguard Nativ (UDP)

```
# Client configuration
[Interface]
Address = <zugewiesene Client IP>
DNS = 10.128.0.3
PrivateKey = <Private Key>
MTU=1342

# Server configuration
[Peer]
```

```
PublicKey = <Sentinel Public Key>
AllowedIPs = 0.0.0.0/0
Endpoint = <Sentinel IP>:<UDP Port>
```

Kill Switch

Der folgende von Mullvad erstellte Code implementiert einen Kill Switch, welcher verhindert das Traffic am VPN Tunnel vorbei geht. Insbesondere relevant für IPv6! Er kann **auf Linux** der Config hinzugefügt.

```
PostUp = iptables -I OUTPUT ! -o %i -m mark ! --mark $(wg show %i fwmark) -m addrtype ! --dst-type LOCAL -j
REJECT && ip6tables -I OUTPUT ! -o %i -m mark ! --mark $(wg show %i fwmark) -m addrtype ! --dst-type LOCAL -j
REJECT
PreDown = iptables -D OUTPUT ! -o %i -m mark ! --mark $(wg show %i fwmark) -m addrtype ! --dst-type LOCAL -j
REJECT && ip6tables -D OUTPUT ! -o %i -m mark ! --mark $(wg show %i fwmark) -m addrtype ! --dst-type LOCAL
-j REJECT
```

Getunnelt mit TCP

Use case: Öffentliche Netzwerke wie bspw. an einer Universität oder in der Bahn blockieren häufig UDP Verkehr. Hierbei wird udp2raw verwendet, um einen verschlüsselten Fake TCP Tunnel zwischen dem Client und Server zu erstellen. Über diesen Tunnel wird dann die eigentliche VPN Verbindung aufgebaut. Wir verwenden gängige Ports, wie bspw. 443/HTTPS, um Blockierungen in Firewalls zu umgehen.

Erfordert udp2raw auf dem Client Gerät.

Anleitung für Android Geräte mit root, bzw. Workaround für ohne root.

```
# Client configuration
[Interface]
Address = <zugewiesene Client IP>
DNS = 10.128.0.3
PrivateKey = <Private Key>
MTU=1342

# Verhindert dass udp2raw traffic auch über den VPN Tunnel gesendet wird
```

```
PostUp = ip rule add to <Sentinel IP> lookup main
PostDown = ip rule del to <Sentinel IP> lookup main
```

```
# Start/Stop von udp2raw
```

```
PreUp = udp2raw -c -l 127.0.0.1:50001 -r <Sentinel IP>:<TCP Port> -k "<udp2raw Secret>" -a
>/var/log/udp2raw.log 2>&1 &
PostDown = killall udp2raw || true
```

```
# Server configuration
```

```
[Peer]
```

```
PublicKey = <Sentinel Public Key>
```

```
AllowedIPs = 0.0.0.0/0
```

```
Endpoint = 127.0.0.1:50001
```

Kill Switch

Der folgende von Mullvad erstellte Code implementiert einen Kill Switch, welcher verhindert das Traffic nicht durch den VPN Tunnel geht. Insbesondere relevant für IPv6! Wir haben ihn etwas erweitert um Traffic zu Sentinel zuzulassen, da sonst die udp2raw Verbindung bricht. Er kann **auf Linux** der Config hinzugefügt.

```
PostUp = iptables -I OUTPUT ! -d <Sentinel IP> ! -o %i -m mark ! --mark $(wg show %i fwmark) -m addrtype ! --dst-type LOCAL -j REJECT && ip6tables -I OUTPUT ! -o %i -m mark ! --mark $(wg show %i fwmark) -m addrtype ! --dst-type LOCAL -j REJECT
PreDown = iptables -D OUTPUT ! -d <Sentinel IP> ! -o %i -m mark ! --mark $(wg show %i fwmark) -m addrtype ! --dst-type LOCAL -j REJECT && ip6tables -D OUTPUT ! -o %i -m mark ! --mark $(wg show %i fwmark) -m addrtype ! --dst-type LOCAL -j REJECT
```

Zwischen Netzen wechseln

Zum Wechseln der Netze genügt es in der Config die **zugewiesene Client IP**, bzw. die Netzadresse und den **Port** zu ändern. Daten der Netze können der Tabelle entnommen werden.

Hinweise zu den Keys

Alle Netze verwenden dieselben Keys (Wireguard und udp2raw). Auch euer Public Key wird in allen Netzen hinterlegt. Das bedeutet dass zu Verschiedenen Netzen Verbindungen gleichzeitig genutzt werden können. Wer mehrere Connections möchte, dem empfehlen wir allerdings einfach mehrere Keys zu hinterlegen. Somit bleibt nämlich die Möglichkeit eines einfachen Netzwechsels (durch ändern von zwei Parametern) erhalten. Die Idee ist also so viel Flexibilität wie möglich mit so wenig Informationen (Bsp. Keys) wie nötig.

DNS

Unser streng gefilterter DNS (AdGuard) ist erreichbar unter `10.128.0.3`.

- Mit den Upstream DNS Servern wird ausschließlich über DoH oder DoT kommuniziert.
- Query Logs sind deaktiviert.
- Zum Ändern des DNS die Zeile `DNS = 10.128.0.3` in der Client Config anpassen.
 - z.B.: `#` davor setzen (auskommentieren) zum deaktivieren
 - Ändern der IP für eigenen DNS oder einen DNS von Mullvad
 - Mullvad hat für verschiedene Block-Level dedizierte DNS Server. Siehe Liste.

Der DNS ist aus allen drei Netzen erreichbar! Die IP muss **nicht** angepasst werden!

Blocklisten

In keiner bestimmten Reihenfolge. Mehrere Listen mit gleicher Basis-Liste möglich.

Wer mehr Struktur in die Listen bringen möchte ist herzlich dazu eingeladen.

- AdGuard DNS filter
- AdAway Default Blocklist
- NoCoin Filter List
- Scam Blocklist by DurableNapkin
- Spam404
- Big List of hacked Malware Websites
- Peter Lowe's List
- WindowsSpyBlocker
- sysctl.org
- Simple Tracking
- Simple Ad
- Notrack-malware
- AmazonFireTV
- notserious

- [Phishing-Angriffe](#)
- [spam.mails](#)
- [Win10 Telemetry](#)
- [easylist](#)
- [Samsung](#)
- [malware](#)
- [fake science](#)
- [MS Office Telemetry](#)
- [Dandelion Sprout's Anti-Malware](#)
- [Easyprivacy](#)
- [Progent-Ads](#)
- [Notrack-Blocklist](#)
- [First Party Trackers](#)
- [Multi Party Trackers](#)
- [Ads and Tracking extended](#)
- [Android Tracking](#)
- [SmartTV](#)
- [1Hosts \(Lite\)](#)
- [1Hosts \(mini\)](#)
- [Dan Pollock's List](#)
- [HaGeZi Personal Black & White](#)
- [The No Tracking Blocklist](#)
- [OISD Blocklist Basic](#)
- [OISD Blocklist Full](#)
- [Steven Black's List](#)
- [Dandelion Sprout's Game Console Ads](#)
- [Perflyst and Dandelion Sprout's Smart-TV Blocklist](#)
- [Phishing URL Blocklist \(PhishTank and OpenPhish\)](#)
- [Dandelion Sprout's Anti-Malware List](#)
- [Stalkerware Indicators List](#)
- [The Big List of Hacked Malware Web Sites](#)
- [Malicious URL Blocklist \(URLHaus\)](#)
- [DDG Android](#)
- [HaGeZi's Threat Intelligence Feeds](#)

- [Phishing Army](#)
- [ShadowWhisperer's Malware List](#)
- [Peter Lowe's Blocklist](#)
- [HaGeZi's Pro Blocklist](#)
- [HaGeZi's The World's Most Abused TLDs](#)
- [HaGeZi's Pro+ + Blocklist](#)
- [HaGeZi's Ultimate Blocklist](#)
- [DDG Windows](#)
- [DDG iOS](#)
- [Alexa Telemetry](#)
- [Apple Telemetry](#)
- [Huawei Telemetry](#)
- [Sonos Telemetry](#)
- [Windows Telemetry](#)
- [Xiaomi Telemetry](#)
- [Mullvad Tracker List](#)
- [frellwits swedish](#)
- [AdGuard DNS](#)
- [Urlhaus \(Malware\)](#)

Verbindungs Check

- Mullvad Connection check