WireGuard - VPN-Server

Quellen:

- Wireguard
- Wireguard unter Debian 11
- Wireguard: Private-Key mit GPG verschlüsselt speichern
- https://oliver-kaestner.de/posts/anleitung-wireguard-vpn-server-einrichten-internetrouting/

Server-Installation

```
sudo apt update
```

```
sudo apt install wireguard
```

Schlüssel generieren

Für den Server und für jeden Client wird je ein privater und ein daran gebundener öffentlicher Schlüssel generiert. Für die Funktionalität dürfen sie weder vertauscht noch verändert werden.
Der jeweils private Schlüssel (Server und Clients) ist sicherheitsrelevant und sollte entsprechend behandelt werden.
Schlüssel bedeutet hier: eine lange Zeichenkette, ähnlich einem Passwort. Die Datei dient als Speicher für diese Zeichenkette.

Die Schlüsseldateien können beliebig benannt werden. Z.B. Username, PC-Name eines Laptops, etc. Eine eindeutige Bezeichnung hilft die Schlüssel zu identifizieren, wenn z.B. Freigaben erteilt oder entzogen werden müssen.
Eine Kopie der Schlüssel/Keys wird in den im folgenden genannten Verzeichnissen abgelegt.

Server-Schlüssel - privat (key)

```
wg genkey | sudo tee /etc/wireguard/server.key
```

Leserechte des privaten Server-Keys auf root begrenzen

```
sudo chmod 0400 /etc/wireguard/server.key
```

Server-Schlüssel öffentlich (pub) aus dem privaten Schlüssel generieren

```
sudo cat /etc/wireguard/server.key | wg pubkey | sudo tee
/etc/wireguard/server.pub
```

Client-Schlüssel - privat (key)
Werden die Client-Schlüssel auf dem Server generiert, dann sollten diese in das Unterverzeichnis Clients gelegt werden. In diesem Fall muss der geheime private Schlüssel zum Client transferiert werden.
Besser ist es daher, die Client-Schlüssel auf dem Client-System zu erstellen und nur den öffentlichen Schlüssel zum Server zu transferieren.

```
sudo mkdir -p /etc/wireguard/clients
```

```
wg genkey | sudo tee /etc/wireguard/clients/client1.key
```

Client-Schlüssel öffentlich (pub) aus dem privaten Schlüssel generieren

```
sudo cat /etc/wireguard/clients/client1.key | wg pubkey | sudo tee
/etc/wireguard/clients/client1.pub
```

Zur Nutzung werden die Schlüssel wie folgt hinterlegt:

auf dem Server in der Server-Konfiguration

- privater Key des Servers
- öffentliche Keys aller Clients

auf jedem Client in der Client-Konfiguration

- privater Key des Clients
- öffentlicher Key des Servers

Server einrichten

Server-Konfiguration erstellen

```
sudo nano /etc/wireguard/wg0.conf
```

```
[Interface]
# Wireguard Server private key - server.key
PrivateKey = <PRIVATKEY-SERVER>
# Wireguard interface will be run at 10.8.0.1
Address = 10.8.0.1/24
# Clients will connect to UDP port 51820
ListenPort = 51820
# Ensure any changes will be saved to the Wireguard config file
# SaveConfig = true
PostUp = iptables -A FORWARD -i wg0 -j ACCEPT; iptables -t nat -A
POSTROUTING -o eth0 -j MASQUERADE
PostDown = iptables -D FORWARD -i wg0 -j ACCEPT; iptables -t nat -D
POSTROUTING -o eth0 -j MASQUERADE

[Peer]
```

```
# Client public key - client1.pub
PublicKey = <PUBLIC-KEY-CLIENT1>
# clients' VPN IP addresses you allow to connect
AllowedIPs = 10.8.0.2/32

[Peer]
# weitere Clients - Laptop13.pup
PublicKey = <PUBLIC-KEY-Laptop13>
AllowedIPs = 10.8.0.3/32
```

Muss ein einzelner Peer gelöscht werden:

```
wg set <interface> peer <key> remove
```

<interface> z.B. wg0

„SaveConfig = true“ hat Probleme verursacht, beim Hinzufügen weiterer Clients. Daher wurde diese Zeile später gelöscht.

Werden neue Clients hinzugefügt, muss der Dienst neu gestartet werden:

```
sudo systemctl reload wg-quick@wg0
```

ACHTUNG: Mit der folgenden Variante sperrt man sich nach der ersten Zeile aus, wenn man es aus der Ferne versucht.
sudo wg-quick down wg0 sudo wg-quick up wg0

Tunnel, Port-Forwarding

Um den gesamten Traffic über den Tunnel zu leiten (erforderlich für den aktiven Client für Internetdaten, Zugriff auf Remote-Desktops, etc.) ist Port-Forwarding auf dem Server erforderlich.

Debian

```
sudo nano /etc/sysctl.conf
```

Einfügen in die Datei

```
# Port Forwarding for IPv4
net.ipv4.ip_forward=1
# Port forwarding for IPv6
net.ipv6.conf.all.forwarding=1
```

Änderungen übernehmen

```
sudo sysctl -p
```

Ubuntu

Ist die Netzverwaltung auf dem Server mit Netplan realisiert, siehe:
Tunnel mit Netplan.
Netplan, Github
Netplan, Github wireguard.yaml

Firewall

siehe Firewall

```
sudo ufw allow OpenSSH
```

Tunnelendpunkt erreichbar machen

```
sudo ufw allow 51820/udp
```

Daten aus dem Tunnel annehmen

```
sudo ufw allow in on wg0
```

Weiterleiten von Internetverkehr

```
sudo ufw route allow in on wg0
```

```
sudo ufw route allow out on wg0
```

```
sudo ufw reload
```

Wireguard-Server starten

```
sudo systemctl start wg-quick@wg0.service
```

enable = in Zukunft automatisch mit dem System starten

```
sudo systemctl enable wg-quick@wg0.service
```

```
sudo systemctl status wg-quick@wg0.service
```

Schnittstelle wg0 überprüfen

```
ip a show wg0
```

Verbindungsinformationen anzeigen

```
sudo wg
```

Linux Client einrichten

```
sudo apt install wireguard-tools
```

Schlüssel generieren
siehe Schlüssel generieren

Konfigurationsdatei erstellen.
Diese Datei kann auch für den Import mit einem entsprechenden Client-Programm genutzt werden - Windows oder Linux.

ACHTUNG: In dieser Datei ist der private Schlüssel enthalten, sie ist daher sicherheitsrelevant und sollte entsprechend behandelt werden.

```
sudo nano /etc/wireguard/client1.conf
```

```
[Interface]
# Define the IP address for the client - must be matched with wg0 on
Wireguard Server
Address = 10.8.0.2/32
# specific DNS Server
DNS = 192.168.178.1
# Private key for the client
PrivateKey = <PRIVATKEY-CLIENT1>

[Peer]
# Public key of the Wireguard server - server.pub
PublicKey = <PUBLICKEY-SERVER>
# Allow all traffic to be routed via Wireguard VPN
AllowedIPs = 0.0.0.0/0
# Public IP address of the Wireguard Server
Endpoint = <IP_SERVER>:51820
# Sending Keepalive every 25 sec
PersistentKeepalive = 25
```

Unter DNS ist die IP-Adresse für den Internetzugang auf dem Server einzutragen. Da sämtliche Internetdaten durch den Tunnel geleitet werden, benötigt der Client die Route auf dem Server, wenn er parallel eine Internetverbindung nutzen möchte[1)].
Ist die DNS-Adresse des Servers nicht bekannt, kann diese wie folgt abgerufen werden:

```
cat /etc/resolv.conf
```

Ist der Server (nur) über eine **IPV6**-Verbindung erreichbar > diese als <IP_SERVER> eintragen. Allerdings muss sie in eckige Klammern gesetzt werden, andernfalls kollidieren die Doppelpunkte bei der Interpretation der Konfiguration. Beispiel: [2001:0db8:85a3:08d3:1319:8a2e:0370:7344]:51820
Siehe auch Andys Block

Wireguard starten

Hier wird die Verwaltung über die Konsole beschrieben. Wird das Linux-System über eine GUI betrieben, ist es auch möglich, die o.a. client1.conf-Datei über die Netzwerk-Verwaltung zu importieren.

```
sudo wg-quick up client1
```

Evtl. installieren bei Fehlermeldung beim Start VPN

```
sudo apt install openresolv
```

Schnittstelle prüfen

```
ip a show client1
```

Verbindung anzeigen » einmal auf dem Client und einmal auf dem Server

```
sudo wg show
```

Server-Zugriff vom Client testen

```
ping -c5 10.8.0.1
```

```
ping -c5 1.1.1.1
```

```
ping -c5 duckduckgo.com
```

Wireguard beenden (Server und Client)

```
sudo wg-quick down client1
```

Windows-Client einrichten

1. Installationsdatei herunterladen
2. Installieren
3. Konfigurationsdatei über das installierte Programm importieren, siehe Linux Client einrichten einrichten
4. Verbindung aktivieren/deaktivieren

[1)]

Webseiten aufrufen, E-Mail abrufen, etc.