Networking

Wireguard VPN

Was ist Wireguard?

Wireguard ist ein moderner VPN Tunnel.

In meinem Setup:

```
Internet → Hetzner (46.54.2.140)
             ↓
      Wireguard-Tunnel (verschlüsselt)
             ↓
         Homeserver (10.100.0.2)
            ↓
         Services (Docker)
```

Vorteile:

- Verschlüsselt
- Schnell
- Einfach
- Stabil

Installation auf Hetzner

Auf Hetzner-Server:

```
  ssh hetzner

# Wireguard installieren
  apt install -y wireguard

# Keys generieren
  cd /etc/wireguard
  umask 077
  wg genkey | tee hetzner-private.key | wg pubkey > hetzner-public.key

# Keys anzeigen und notieren!
  cat hetzner-private.key
  cat hetzner-public.key
```

Config erstellen

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

Inhalt:

```
[Interface]
Address = 10.100.0.1/24
ListenPort = 51820
PrivateKey = <HETZNER_PRIVATE_KEY>

PostUp = sysctl -w net.ipv4.ip_forward=1
PostDown = sysctl -w net.ipv4.ip_forward=0

[Peer]
PublicKey = <HOMESERVER_PUBLIC_KEY>
AllowedIPs = 10.100.0.2/32
PersistentKeepalive = 25
```

Aktivieren

```
echo "net.ipv4.ip_forward=1" >> /etc/sysctl.conf
sysctl -p

systemctl enable wg-quick@wg0
systemctl start wg-quick@wg0
systemctl status wg-quick@wg0
```

Installation auf Homeserver

Auf Homeserver:

```
# Wireguard installieren
  sudo apt update
  sudo apt install -y wireguard

# Keys generieren
  sudo -i
  cd /etc/wireguard
  umask 077
  wg genkey | tee homeserver-private.key | wg pubkey > homeserver-public.key

# Keys anzeigen und notieren!
  cat homeserver-private.key
  cat homeserver-public.key
```

Config erstellen:

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

Inhalt:

```
[Interface]
Address = 10.100.0.2/24
PrivateKey = <HOMESERVER_PRIVATE_KEY>

[Peer]
PublicKey = <HETZNER_PUBLIC_KEY>
Endpoint = 46.224.8.110:51820
AllowedIPs = 10.100.0.1/32
PersistentKeepalive = 25
```

Akivieren:

```
sudo systemctl enable wg-quick@wg0
sudo systemctl start wg-quick@wg0
sudo systemctl status wg-quick@wg0
```

Tunnel setzen

Von Hetzner zum Homeserver:

```
# Auf Hetzner
  ping -c 4 10.100.0.2
```

Von Homeserver auf Hetzner:

```
# Auf Homeserver
  ping -c 4 10.100.0.1
```

-> Beide sollten Antworten

Status prüfen

`wg show`

Troubleshooting

```
# Wireguard neu starten
  sudo systemctl restart wg-quick@wg0
# Logs ansehen
  journalctl -u wg-quick@wg0 -f
```

Caddy Reverse Proxy

Was ist Caddy?

Caddy ist ein moderner Webserver und Reverse Proxy.

Features:

- automatische SSL Zertifikate (Let's Encrypt)
- Einfache Konfiguration
- HTTP/2 & HTTP/3 Support
- Reverse Proxy funktionalität

In meinem Setup

- Caddy auf Hetzner: Öffentlicher Eingang, SSL-Termination
- Caddy auf Homeserver: Lokales Routing zu Services

Caddy Reverse Proxy

Caddy auf Hetzner

Installation:

```
ssh hetzner

apt install -y debian-keyring debian-archive-keyring apt-transport-https curl
curl -1sLf 'https://dl.cloudsmith.io/public/caddy/stable/gpg.key' | gpg --dearmor -o
/usr/share/keyrings/caddy-stable-archive-keyring.gpg
curl -1sLf 'https://dl.cloudsmith.io/public/caddy/stable/debian.deb.txt' | tee
/etc/apt/sources.list.d/caddy-stable.list
apt update
apt install -y caddy
```

Caddyfile:

```
nano /etc/caddy/Caddyfile
```

Inhalt:

```
enode.eu {
    basicauth {
        denode $2a$14$<BCRYPT_HASH>
    }

    root * /var/www/denode
    file_server
    encode gzip
}

nextcloud.denode.eu {
    encode gzip
    reverse_proxy 10.100.0.2:80 {
        header_up Host {host}
        header_up X-Real-IP {remote_host}
        header_up X-Forwarded-For {remote_host}
        header_up X-Forwarded-Proto {scheme}
```

```
    }
}

notes.denode.eu {
    basicauth {
        denode $2a$14$<BCRYPT_HASH>
    }

    encode gzip
    reverse_proxy 10.100.0.2:4567 {
        header_up Host {host}
        header_up X-Real-IP {remote_host}
        header_up X-Forwarded-For {remote_host}
        header_up X-Forwarded-Proto {scheme}
    }
}

overleaf.denode.eu {
    basicauth {
        denode $2a$14$<BCRYPT_HASH>
    }

    encode gzip
    reverse_proxy 10.100.0.2:4568 {
        header_up Host {host}
        header_up X-Real-IP {remote_host}
        header_up X-Forwarded-For {remote_host}
        header_up X-Forwarded-Proto {scheme}
    }
}
```

Passwort hashen

```
caddy hash-password
```

Caddy reload

```
systemctl reload caddy
systemctl status caddy
```

Caddy Reverse Proxy

Caddy auf Homeserver

Caddyfile

```
sudo nano /etc/caddy/Caddyfile
```

Inhalt:

```
enode.eu {
    basicauth {
        denode $2a$14$<BCRYPT_HASH>
    }

    root * /var/www/denode
    file_server
    encode gzip
}

nextcloud.denode.eu {
    encode gzip
    reverse_proxy 10.100.0.2:80 {
        header_up Host {host}
        header_up X-Real-IP {remote_host}
        header_up X-Forwarded-For {remote_host}
        header_up X-Forwarded-Proto {scheme}
    }
}

notes.denode.eu {
    basicauth {
        denode $2a$14$<BCRYPT_HASH>
    }

    encode gzip
    reverse_proxy 10.100.0.2:4567 {
        header_up Host {host}
        header_up X-Real-IP {remote_host}
```

```
        header_up X-Forwarded-For {remote_host}
        header_up X-Forwarded-Proto {scheme}
    }
}

overleaf.denode.eu {
    basicauth {
        denode $2a$14$<BCRYPT_HASH>
    }

    encode gzip
    reverse_proxy 10.100.0.2:4568 {
        header_up Host {host}
        header_up X-Real-IP {remote_host}
        header_up X-Forwarded-For {remote_host}
        header_up X-Forwarded-Proto {scheme}
    }
}
```

Caddy reload

```
sudo systemctl reload caddy
sudo systemctl status caddy
```

Caddy Reverse Proxy

Wichtige Caddy-Befehle

Config testen

```
caddy validate --config /etc/caddy/Caddyfile
```

Caddy neuladen

```
sytemctl reload caddy
```

Status prüfen

```
systemctl status caddy
```

Logs ansehen

```
journalctl -u caddy -f
```

Config Pfad:

- Hetzner: `/etc/caddy/Caddyfile`
- Homeserver: `/etc/caddy/Caddyfile`