

# WireGuard VPN Setup

WireGuard is an extremely simple yet fast and modern VPN that utilizes state-of-the-art cryptography. It aims to be faster, simpler, leaner, and more useful than IPSec, while avoiding the massive headache. It intends to be considerably more performant than OpenVPN.



WireGuard is designed as a general purpose VPN for running on embedded interfaces and super computers alike, fit for many different circumstances. Initially released for the Linux kernel, it is now cross-platform and widely deployable.

## Install VPN-Server on CentOS 7.x

For the **Debian** installation Tutorial klick [here](#)

Ausgangslage:

- **LAN Network**=192.168.1.0/24
- **VPN Network**=192.168.100.0/24
- **VPN Port**=53666/UDP

### For CentOS 7 ONLY:

```
# yum install epel-release
# curl -Lo /etc/yum.repos.d/wireguard.repo
https://copr.fedorainfracloud.org/coprs/jdoss/wireguard/repo/epel-7/jdoss-wireguard-epel-7.repo
# yum update

# yum install wireguard-dkms wireguard-tools
```

### For CentOS 8 ONLY:

```
# yum install epel-release
# yum config-manager --set-enabled PowerTools
```

```
# yum copr enable jdoss/wireguard
# yum install wireguard-dkms wireguard-tools
```

**continue here:**

```
# mkdir /etc/wireguard && cd /etc/wireguard/
# umask 077
# wg genkey > wg0.conf
```

```
# vim /etc/wireguard/wg0.conf
```

```
[Interface]
Address = 192.168.100.1/24
SaveConfig = true
PostUp = iptables -I FORWARD -i wg0 -j ACCEPT; iptables -I FORWARD -o wg0 -j ACCEPT
PostDown = firewall-cmd --reload
ListenPort = 53666
PrivateKey = INVH3hPTDtaQVB7TkGy/qLMeEgbiiUjV2PbPF0B4+ns=
```

```
# firewall-cmd --zone=public --add-port=53666/udp --permanent
# firewall-cmd --reload
# systemctl net.ipv4.ip_forward=1
```

```
# vim /etc/sysctl.d/99-sysctl.conf
```

```
# sysctl settings are defined through files in
# /usr/lib/sysctl.d/, /run/sysctl.d/, and /etc/sysctl.d/.
#
# Vendors settings live in /usr/lib/sysctl.d/.
# To override a whole file, create a new file with the same in
# /etc/sysctl.d/ and put new settings there. To override
# only specific settings, add a file with a lexically later
# name in /etc/sysctl.d/ and put new settings there.
#
# For more information, see sysctl.conf(5) and sysctl.d(5).
net.ipv4.ip_forward=1
```

```
# systemctl -p
# systemctl start wg-quick@wg0.service
```

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

Setup POSTROUTING, do this ONLY if you don't want to setup routing!

```
# firewall-cmd --permanent --direct --add-rule ipv4 nat POSTROUTING 0 -s  
192.168.100.0/24 ! -d 192.168.100.0/24 -j SNAT --to 192.168.1.8  
# firewall-cmd --reload
```

```
# wg
```

```
interface: wg0  
public key: g5C+DlBfxAzk+QHU6wSDC9PGKoSHTf5j9NC9fBQcrks=  
private key: (hidden)  
listening port: 53666
```

## Setup Router Settings

### Fritzbox - Port Forwarding Konfigurieren

The screenshot shows the Fritz!Box 5490 web interface. The left sidebar contains navigation options: Übersicht, Internet, Filter, Freigaben (highlighted), MyFRITZ!-Konto, Fiber-Informationen, Telefonie, Heimnetz, WLAN, DECT, Diagnose, System, and Assistenten. The main content area is titled 'Freigaben für Gerät' and shows the configuration for a device named 'reverse-proxy-v2'. The device's IPv4 address is 192.168.1.2 and its MAC address is 00:0D:B9:4E:3A:18. There is a checkbox for 'Selbstständige Portfreigaben für dieses Gerät erlauben.' which is unchecked. Below this, there are 'IPv4-Einstellungen' with a checkbox for 'Dieses Gerät komplett für den Internetzugriff über IPv4 freigeben (Exposed Host)' which is also unchecked. The 'Freigaben' section contains a table with the following data:

Status	Bezeichnung	Protokoll	IP-Adresse im Internet	Port extern vergeben		
●	HTTP-Server	TCP	83.150.6.68	80		
●	HTTPS-Server	TCP	83.150.6.68	443		
●	WireGuard	UDP	83.150.6.68	53666		

At the bottom right of the table, there is a button labeled 'Neue Freigabe'. At the bottom of the configuration area, there are 'OK' and 'Abbrechen' buttons. Red circles with numbers 1, 2, 3, and 4 are overlaid on the image to indicate specific steps: 1 points to the device name dropdown, 2 points to the 'Neue Freigabe' button, 3 points to the 'Freigaben' section header, and 4 points to the 'OK' button.

## Vorgehen:

Unter: "Internet" → "Freigaben" → "Freigaben für Gerät hinzufügen"

1. Auswählen des Gerätes auf welchem der VPN-Server installiert wurde
2. Neue Freigabe
3. Neuer Service - WireGuard Port 3x hinterlegen UDP und Service hinzufügen
4. Bestätigen und speichern

## Fritzbox - Routing Konfigurieren

**Nur falls kein POSTROUTING als Interface Forwarding eingesetzt wird. (So wie in diesem Tutorial)**

Statische IPv4-Routing-Tabelle

Wenn Ihr Netzwerk aus mehreren Subnetzen besteht, die nicht direkt mit der FRITZ!Box verbunden sind, können Sie für diese statische IPv4-Routen in der FRITZ!Box einrichten.

**Achtung!**  
Änderungen auf dieser Seite können dazu führen, dass die FRITZ!Box nicht mehr erreichbar ist. Beachten Sie unbedingt die Hilfe, bevor Sie Änderungen vornehmen.

Aktiv	Netzwerk	Subnetzmaske	Gateway	
<input checked="" type="checkbox"/>	172.168.0.0	255.255.255.0	192.168.1.2	<input type="text"/> <input type="text"/>
<input checked="" type="checkbox"/>	192.168.100.0	255.255.255.0	192.168.1.2	<input type="text"/> <input type="text"/>

1 Neue IPv4-Route

3 OK Abbrechen

## Vorgehen:

Unter: "Heimnetzwerk" → "Netzwerk" → "Statische IPv4 Routing-Tabelle bearbeiten"

1. Neue IPv4-Route
2. Erstellen von LAN Routing in VPN Netzwerk (192.168.100.X) - Der Gateway ist hierbei die LAN IP des Servers auf welchem WireGuard installiert ist.
3. Bestätigen und speichern

## Connect Android Smartphone with VPN

1. Install [WireGuard App](#) on Smartphone:
2. Create new WireGuard Tunnel:
3. Generate Private-Key and Public-Key for Smartphone:
4. Add static IP Adress and DNS to Smartphone VPN connection:
5. Stop Server and add Client VPN Peer (Client-Public-Key) to Server:

- Edit VPN configuration-file on Server:

```
# systemctl stop wg-quick@wg0.service
# vim /etc/wireguard/wg0.conf
```

```
[Interface]
Address = 192.168.100.1/24
SaveConfig = true
PostUp = iptables -I FORWARD -i wg0 -j ACCEPT; iptables -I FORWARD
-o wg0 -j ACCEPT
PostDown = firewall-cmd --reload
ListenPort = 53666
PrivateKey = INVH3hPTDtaQVB7TkGy/qLMeEgbiiUjV2PbPF0B4+ns=

[Peer]
PublicKey = 9RaYFNNWSk/l6uU3so44XqXErW5en2q74BsSayyEB1A=
AllowedIPs = 192.168.100.10/32
```

- Restart VPN connection daemon:

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

- Print WireGuard information:

```
# wg
```

```
interface: wg0
  public key: g5C+d1BfxAzk+QHU6wSDC9PGKoSHTf5j9NC9fBQcrks=
  private key: (hidden)
  listening port: 53666

peer: 9RaYFNNWSk/l6uU3so44XqXErW5en2q74BsSayyEB1A=
  allowed ips: 192.168.100.10/32
```

## 6. Add Server VPN Peer (Server-Public-Key) on Client:

## 7. Connect & Test:

# Connect Windows PC with VPN

## 1. Install [WireGuard App](#) on Computer:

TO DO

1. **Create new WireGuard Tunnel:**
2. **Generate Private-Key and Public-Key for Smartphone:**
3. **Add static IP Adress and DNS to Smartphone VPN connection:**

#### 4. Stop Server and add Client VPN Peer (Client-Public-Key) to Server:

- Edit VPN configuration-file:

```
# systemctl stop wg-quick@wg0.service
# vim /etc/wireguard/wg0.conf
```

```
[Interface]
Address = 192.168.100.1/24
SaveConfig = true
PostUp = iptables -I FORWARD -i wg0 -j ACCEPT; iptables -I FORWARD
-o wg0 -j ACCEPT
PostDown = firewall-cmd --reload
ListenPort = 53666
PrivateKey = INVH3hPTDtaQVB7TkGy/qLMeEgbiiUjV2PbPF0B4+ns=

[Peer]
PublicKey = 9RaYFNNWSk/l6uU3so44XqXErW5en2q74BsSayyEB1A=
AllowedIPs = 192.168.100.10/32
```

- Restart VPN connection daemon:

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

- Print WireGuard information:

```
# wg
```

```
interface: wg0
  public key: g5C+d1BfxAzk+QHU6wSDC9PGKoSHTf5j9NC9fBQcrks=
  private key: (hidden)
  listening port: 53666

peer: 9RaYFNNWSk/l6uU3so44XqXErW5en2q74BsSayyEB1A=
  allowed ips: 192.168.100.10/32
```

#### 5. Add Server VPN Peer (Server-Public-Key) on Client:

#### 6. Connect & Test:

---

## Weiteres

- [Use Raspberry Pi as WiFi AP and route traffic through Wireguard \(port 53\)](#)

Last update: **2020/01/31 22:52**