

# How to Install Wireguard VPN on Debian 12

Learn how to install and configure Wireguard VPN on Debian 12 with our step-by-step guide. Secure your connection with this modern, efficient VPN solution.

---

# Contents

01	Introduction	3
02	Prerequisites	3
03	Install WireGuard	3
04	Configure WireGuard	4
05	Manage WireGuard VPN System Processes	9
06	Set Up Firewall Rules	10
07	Connect Client Devices to the WireGuard VPN Server	12
08	Conclusion	14

# Introduction

WireGuard is a modern, efficient, and secure VPN protocol that overcomes the complexities and performance issues related to old VPN solutions. WireGuard enables every peer to function as a server and a client by allowing each peer to have unique keys and configurations. This architecture facilitates direct, encrypted communication between peers without relying on a centralized server.

This article shows you how to install WireGuard on Debian 12.

## Prerequisites

Before you begin:

- [Deploy two Debian 12](#) instances on Vultr to use as the VPN server and client respectively.
- Access the WireGuard server [using SSH](#) as a non-root user with sudo privileges.
- [Update the server](#).

## Install WireGuard

WireGuard is available in the default repositories on Debian 12. Follow the steps below to install WireGuard using the Advanced Package Tool (APT).

1. Update the server's package index.

CONSOLE

```
$ sudo apt update
```

2. Install WireGuard.

## CONSOLE

```
$ sudo apt install wireguard -y
```

3. View the WireGuard version on your server.

## CONSOLE

```
$ wg --version
```

Output:

```
wireguard-tools v1.0.20210914 - https://git.zx2c4.com/wireguard-tools/
```

## Configure WireGuard

WireGuard uses key pairs to create secure and encrypted connections between peers. Each WireGuard client or server uses a unique private key to generate a public key. A private key signs outgoing traffic and decrypts incoming traffic, while a public key encrypts traffic from other peers when sending traffic to the server. Follow the steps below to create a new WireGuard interface and generate the server key pair.

1. Generate a new private key for the WireGuard server and store it in the `/etc/wireguard/` directory.

## CONSOLE

```
$ wg genkey | sudo tee /etc/wireguard/private.key
```

Output:

```
+E0z2nY0ezr5oHWASuMJJEGXTouBrdyhK7DKxyE8iEQ=
```

2. Change the private key file permissions to give the directory owner read privileges.

CONSOLE

```
$ sudo chmod go= /etc/wireguard/private.key
```

3. Use the private key to generate a new public key.

CONSOLE

```
$ sudo cat /etc/wireguard/private.key | wg pubkey | sudo tee /etc/wireguard/public.key
```

Output:

```
01fiUPB73tNIJGPgsjuA613N3VBuA/KuQp8xPjwJwnU=
```

4. Create a new WireGuard interface configuration file `wg0.conf` under the `/etc/wireguard/` directory.

CONSOLE

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

5. Add the following configurations to the file. Replace `+E0z2nY0ezr5oHWASuMJJEGXTouBrdyhK7DKxyE8iEQ=` with the private key you generated earlier.

INI

```
[Interface]  
PrivateKey = +E0z2nY0ezr5oHWASuMJJEGXTouBrdyhK7DKxyE8iEQ=  
Address = 10.0.0.1/24  
ListenPort = 51820
```

Save and close the file.

The above configuration creates a new `wg0` WireGuard interface and `10.0.0.1` as the tunnel address. Within the configuration:

- `[Interface]`: Creates a new WireGuard interface.
- `PrivateKey`: Sets the WireGuard server's private key.
- `Address`: Sets the WireGuard interface's IP address when communicating with connected peers. The `10.0.0.0/24` network address enables a subnet that supports up to 256 addresses, while `10.0.0.1` is the WireGuard interface address.
- `ListenPort`: Specifies the UDP port the WireGuard interface should use to listen for incoming VPN connections. The value `51820` is the default port number for WireGuard interfaces.

#### 6. Generate a new WireGuard client private key.

CONSOLE

```
$ wg genkey | tee client1_private.key
```

Output:

```
gHVZgu6TsxQj+SRQl90K8gIJ8w71h4v7mdUUixb/7nw=
```

#### 7. Generate a new public key using the WireGuard client's private key.

CONSOLE

```
$ cat client1_private.key | wg pubkey | tee client1_public.key
```

Output:

```
n2nSd/o0VlukX4wx2y4p5IPF3XETA/3g0jI8zSupFlg=
```

#### 8. Open the WireGuard server `wg0` interface configuration file.

CONSOLE

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

9. Add the following WireGuard client configurations at the end of the file. Replace `n2nSd/o0VlukX4wx2y4p5IPF3XETA/3g0jI8zSUpFlg=` with the actual client's public key you generated earlier

```
INI
```

```
[Peer]
```

```
PublicKey = n2nSd/o0VlukX4wx2y4p5IPF3XETA/3g0jI8zSUpFlg=  
AllowedIPs = 10.0.0.2/32
```

Save and close the file.

The above WireGuard configuration directives enable a new WireGuard client to connect to the interface with the tunnel IP address `10.0.0.2`.

10. View your WireGuard server's public IP address.

```
CONSOLE
```

```
$ ifconfig
```

Output:

```
enp1s0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST> mtu 1500  
    inet 192.0.2.4 netmask 255.255.254.0 broadcast 192.0.2.255  
    inet6 fe80::5400:5ff:fe09:669f prefixlen 64 scopeid 0x20<link>  
    ether 56:00:05:09:66:9f txqueuelen 1000 (Ethernet)  
    RX packets 1066 bytes 223709 (218.4 KiB)  
    RX errors 0 dropped 0 overruns 0 frame 0 TX packets 842  
    bytes 146160 (142.7 KiB)  
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0  
  
lo: flags=73<UP,LOOPBACK,RUNNING> mtu 65536  
    inet 127.0.0.1 netmask 255.0.0.0  
    inet6 ::1 prefixlen 128 scopeid 0x10<host>  
    loop txqueuelen 1000 (Local Loopback)  
    RX packets 0 bytes 0 (0.0 B)  
    RX errors 0 dropped 0 overruns 0 frame 0
```

```
TX packets 0 bytes 0 (0.0 B)
TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0
```

The WireGuard server uses the public IP address `192.0.2.4` based on the above `enp1s0` interface information output.

#### 11. Create a new WireGuard client configuration file.

CONSOLE

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

#### 12. Add the following configurations to the file. Replace

`gHVZgu6TsxQj+SRQl90K8gIJ8w71h4v7mdUUIxb/7nw=` with your actual client private key and `01fiUPB73tNIJGPgsjuA613N3VBuA/KuQp8xPjwJwnU=` with your WireGuard server's public key values you generated earlier.

INI

```
[Interface]
PrivateKey = gHVZgu6TsxQj+SRQl90K8gIJ8w71h4v7mdUUIxb/7nw=
Address = 10.0.0.2/32

[Peer]
PublicKey = 01fiUPB73tNIJGPgsjuA613N3VBuA/KuQp8xPjwJwnU=
Endpoint = 192.0.2.4:51820
AllowedIPs = 10.0.0.0/24
```

Save and close the file.

The above client configuration creates a new tunnel connection using `192.0.2.4` as the WireGuard server's public IP address. Within the configuration:

- `[Interface]`: Creates a new WireGuard client interface.
- `PrivateKey`: Sets the WireGuard private key when encrypting traffic.
- `Address`: Sets the WireGuard client's IP address when connecting to the VPN tunnel.
- `[Peer]`: Enables the WireGuard server remote peer configuration when establishing tunnel connections.

- `PublicKey`: Sets the WireGuard server's public key.
- `Endpoint`: Specifies the target WireGuard server public IP address and port.
- `AllowedIPs`: Specifies the client IP network connections to allow through the VPN tunnel. When set to `0.0.0.0/0`, the WireGuard client forwards all network traffic to the VPN tunnel.

## Manage WireGuard VPN System Processes

WireGuard uses the `wg-quick` utility to manage all tunnel interfaces on a server. Follow the steps below to manage the `wg0` WireGuard interface and enable tunnel connections to the server.

1. Enable the WireGuard interface to automatically start at boot.

### CONSOLE

```
$ sudo systemctl enable wg-quick@wg0
```

Output:

```
Created symlink /etc/systemd/system/multi-user.target.wants/wg-quick@wg0.service  
→ /lib/systemd/system/wg-quick@.service.
```

2. Start the WireGuard interface.

### CONSOLE

```
$ sudo systemctl start wg-quick@wg0
```

3. View the WireGuard interface status and verify that it's active.

### CONSOLE

```
$ sudo systemctl status wg-quick@wg0
```

Output:

```
● wg-quick@wg0.service - WireGuard via wg-quick(8) for wg0
  Loaded: loaded (/lib/systemd/system/wg-quick@.service; enabled; preset:
  enabled)
  Active: active (exited) since Sun 2024-07-28 20:08:52 UTC; 12s ago
  Docs: man:wg-quick(8)
        man:wg(8)
        https://www.wireguard.com/
        https://www.wireguard.com/quickstart/
        https://git.zx2c4.com/wireguard-tools/about/src/man/wg-quick.8
        https://git.zx2c4.com/wireguard-tools/about/src/man/wg.8
  Process: 1318 ExecStart=/usr/bin/wg-quick up wg0 (code=exited, status=0/SUCCESS)
 Main PID: 1318 (code=exited, status=0/SUCCESS)
  CPU: 18ms

Jul 28 20:08:52 vultr systemd[1]: Starting wg-quick@wg0.service - WireGuard via
wg-quick(8) for wg0...
Jul 28 20:08:52 vultr wg-quick[1318]: [#] ip link add wg0 type wireguard
Jul 28 20:08:52 vultr wg-quick[1318]: [#] wg setconf wg0 /dev/fd/63
Jul 28 20:08:52 vultr wg-quick[1318]: [#] ip -4 address add 10.0.0.1/24 dev wg0
Jul 28 20:08:52 vultr wg-quick[1318]: [#] ip link set mtu 1420 up dev wg0
Jul 28 20:08:52 vultr systemd[1]: Finished wg-quick@wg0.service - WireGuard via
wg-quick(8) for wg0.
```

#### 4. Stop the WireGuard interface.

##### CONSOLE

```
$ sudo systemctl stop wg-quick@wg0
```

#### 5. Restart the WireGuard interface.

##### CONSOLE

```
$ sudo systemctl restart wg-quick@wg0
```

## Set Up Firewall Rules

WireGuard listens for incoming tunnel connection requests on the default UDP port `51820` specified in your interface configuration. Follow the steps below to

configure the Uncomplicated Firewall (UFW) active on Vultr Debian servers by default to allow network connections to the WireGuard interface.

1. View the UFW status and verify that it's active.

CONSOLE

```
$ sudo ufw status
```

Output:

```
Status: active
.....
```

2. Allow WireGuard UDP port `51820` through the firewall.

CONSOLE

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

Output:

```
Rule added
Rule added (v6)
```

3. Reload UFW to apply changes.

CONSOLE

```
$ sudo ufw reload
```

Output:

```
Firewall reloaded
```

4. View the UFW status and verify that port `51820` is allowed through the firewall configuration.

CONSOLE

```
$ sudo ufw status
```

Your output should be similar to the one below:

```
Status: active
```

To	Action	From
--	-----	----
22/tcp	ALLOW	Anywhere
51820/udp	ALLOW	Anywhere
22/tcp (v6)	ALLOW	Anywhere (v6)
51820/udp (v6)	ALLOW	Anywhere (v6)

## Connect Client Devices to the WireGuard VPN Server

A WireGuard VPN server accepts network connections from all clients with valid keypairs to connect to the tunnel interface. Follow the steps below to access your Debian 12 WireGuard client instance and connect to the WireGuard server.

1. Access your Debian 12 WireGuard client instance using SSH.

CONSOLE

```
$ ssh user@SERVER-IP
```

2. Update the server's package information index.

CONSOLE

```
$ sudo apt update
```

3. Install WireGuard.

## CONSOLE

```
$ sudo apt install wireguard
```

- Download the client configuration from your WireGuard server using a secure file transfer protocol such as SCP and save it as `wg0.conf`.

## CONSOLE

```
$ sudo scp user@192.0.2.4:/etc/wireguard/client1.conf /etc/wireguard/wg0.conf
```

- Start the `wg0` VPN interface using your client configuration file.

## CONSOLE

```
$ sudo wg-quick up /etc/wireguard/wg0.conf
```

## Output:

```
[#] ip link add wg0 type wireguard
[#] wg setconf wg0 /dev/fd/63
[#] ip -4 address add 10.0.0.2/32 dev wg0
[#] ip link set mtu 1420 up dev wg0
[#] ip -4 route add 10.0.0.0/24 dev wg0
```

- View the WireGuard tunnel status and verify that the client connects to the WireGuard server.

## CONSOLE

```
$ sudo wg
```

Your output should be similar to the one below.

```
interface: wg0
  public key: n2nSd/o0VlukX4wx2y4p5IPF3XETA/3g0jI8zSupFlg=
  private key: (hidden)
```

```
listening port: 35047

peer: 01fiUPB73tNIJGPgsjuA613N3VBuA/KuQp8xPjwJwnU=
  endpoint: 192.0.2.4:51820
  allowed ips: 10.0.0.0/24
  latest handshake: 59 seconds ago
  transfer: 4.09 KiB received, 7.53 KiB sent
```

7. Test the connection to your WireGuard server's tunnel address `10.0.0.1` using the Ping utility.

#### CONSOLE

```
$ ping -c 4 10.0.0.1
```

Your output should be similar to the one below when the connection is successful.

```
PING 10.0.0.1 (10.0.0.1) 56(84) bytes of data:
64 bytes from 10.0.0.1: icmp_seq=1 ttl=64 time=0.651 ms
64 bytes from 10.0.0.1: icmp_seq=2 ttl=64 time=0.626 ms
64 bytes from 10.0.0.1: icmp_seq=3 ttl=64 time=0.575 ms
64 bytes from 10.0.0.1: icmp_seq=4 ttl=64 time=0.587 ms

--- 10.0.0.1 ping statistics ---
4 packets transmitted, 4 received, 0% packet loss, time 3080ms
rtt min/avg/max/mdev = 0.575/0.609/0.651/0.030 ms
```

## Conclusion

You have installed WireGuard on a Debian 12 and created tunnel interfaces to enable network connections between peers. You can create multiple WireGuard interfaces with unique ports and connect multiple clients using valid key pairs on the server.



VULTR

