How to Set Up WireGuard VPN on Ubuntu 20.04

linuxize.com/post/how-to-set-up-wireguard-vpn-on-ubuntu-20-04

Linuxize October 26, 2020

Published on Oct 26, 2020

-

9 min read

WireGuard is a modern VPN (Virtual Private Network) technology that utilizes state-of-the-art cryptography. Compared to other popular VPN solutions, such as IPsec and OpenVPN , WireGuard is faster, easier to configure, and has a smaller footprint. It is cross-platform and can run almost anywhere, including Linux, Windows, Android, and macOS.

Wireguard is a peer-to-peer VPN; it does not use the client-server model. Depending on its configuration, a peer can act as a traditional server or client. It works by creating a network interface on each peer device that acts as a tunnel. Peers authenticate each other by exchanging and validating public keys, mimicking the SSH model. Public keys are mapped with a list of IP addresses that are allowed in the tunnel. The VPN traffic is encapsulated in UDP.

In this article, we'll discuss how to set up a WireGuard VPN on Ubuntu 20.04 that will act as a VPN server. We'll also show you how to configure WireGuard as a client. The client's traffic will be routed through the Ubuntu 20.04 server.

This setup can be used as a protection against Man in the Middle attacks, surfing the web anonymously, bypassing Geo-restricted content, or allowing your colleagues who work from home to connect to the company network securely.

Prerequisites

To follow this guide, you’ll need an Ubuntu 20.04 server with root or sudo access .

Setting Up the WireGuard Server

We’ll start by installing WireGuard on the Ubuntu machine and set it up to act as a server. We’ll also configure the system to route the clients’ traffic through it.

Install WireGuard on Ubuntu 20.04

WireGuard is available from the default Ubuntu repositories. To install it, run the following commands:

```
sudo apt updatesudo apt install wireguard
```

This will install the WireGuard module and tools.

WireGuard runs as a kernel module.

Configuring WireGuard

The `wg` and `wg-quick` command-line tools allow you to configure and manage the WireGuard interfaces.

Each device in the WireGuard VPN network needs to have a private and public key. Run the following command to generate the key pair:

```
wg genkey | sudo tee /etc/wireguard/privatekey | wg pubkey | sudo tee
/etc/wireguard/publickey
```

The files will be generated in the `/etc/wireguard` directory. You can view the contents of the files with `cat` or `less` . The private key should never be shared with anyone and should always be kept secure.

Wireguard also supports a pre-shared key, which adds an additional layer of symmetric-key cryptography. This key is optional and must be unique for each peer pair.

The next step is to configure the tunnel device that will route the VPN traffic.

The device can be set up either from the command line using the `ip` and `wg` commands, or by creating the configuration file with a text editor.

Create a new file named `wg0.conf` and add the following contents:

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

/etc/wireguard/wg0.conf

```
[Interface]
Address = 10.0.0.1/24
SaveConfig = true
ListenPort = 51820
PrivateKey = SERVER_PRIVATE_KEY
PostUp = iptables -A FORWARD -i %i -j ACCEPT; iptables -t nat -A POSTROUTING -o
ens3 -j MASQUERADE
PostDown = iptables -D FORWARD -i %i -j ACCEPT; iptables -t nat -D POSTROUTING -o
ens3 -j MASQUERADE
```

The interface can be named anything, however it is recommended to use something like `wg0` or `wgvpn0`. The settings in the interface section have the following meaning:

- Address - A comma-separated list of v4 or v6 IP addresses for the `wg0` interface. Use IPs from a range that is reserved for private networks (10.0.0.0/8, 172.16.0.0/12 or 192.168.0.0/16).

- ListenPort - The listening port.

- PrivateKey - A private key generated by the `wg genkey` command. (To see the contents of the file type: `sudo cat /etc/wireguard/privatekey`)

- SaveConfig - When set to true, the current state of the interface is saved to the configuration file when shutdown.

- PostUp - Command or script that is executed before bringing the interface up. In this example, we're using iptables to enable masquerading. This allows traffic to leave the server, giving the VPN clients access to the Internet.

  Make sure to replace `ens3` after `-A POSTROUTING` to match the name of your public network interface. You can easily find the interface with:

  ```
  ip -o -4 route show to default | awk '{print $5}'
  ```

- PostDown - command or script which is executed before bringing the interface down. The iptables rules will be removed once the interface is down.

The `wg0.conf` and `privatekey` files should not be readable to normal users. Use `chmod` to set the permissions to `600`:

```
sudo chmod 600 /etc/wireguard/{privatekey,wg0.conf}
```

Once done, bring the `wg0` interface up using the attributes specified in the configuration file:

```
sudo wg-quick up wg0
```

The command will produce an output similar to the following:

```
[#] ip link add wg0 type wireguard
[#] wg setconf wg0 /dev/fd/63
[#] ip -4 address add 10.0.0.1/24 dev wg0
[#] ip link set mtu 1420 up dev wg0
[#] iptables -A FORWARD -i wg0 -j ACCEPT; iptables -t nat -A POSTROUTING -o ens3 -
j MASQUERADE
```

To check the interface state and configuration, enter:

```
sudo wg show wg0
```

```
interface: wg0
  public key: r3imyh3MCYggaZACmkx+CxlD6uAmICI8pe/PGq8+qCg=
  private key: (hidden)
  listening port: 51820
```

You can also run `ip a show wg0` to verify the interface state:

```
ip a show wg0
```

```
4: wg0: <POINTOPOINT,NOARP,UP,LOWER_UP> mtu 1420 qdisc noqueue state UNKNOWN group
default qlen 1000
    link/none
    inet 10.0.0.1/24 scope global wg0
       valid_lft forever preferred_lft forever
```

WireGuard can also be managed with Systemd.

To bring the WireGuard interface at boot time, run the following command:

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

Server Networking and Firewall Configuration

IP forwarding must be enabled for NAT to work. Open the `/etc/sysctl.conf` file and add or uncomment the following line:

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

/etc/sysctl.conf

```
net.ipv4.ip_forward=1
```

Save the file and apply the change:

```
sudo sysctl -p
```

```
net.ipv4.ip_forward = 1
```

If you are using UFW to manage your firewall you need to open UDP traffic on port `51820`:

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

That's it. The Ubuntu peer that will act as a server has been set up.

Linux and macOS Clients Setup

The installation instructions for all supported platforms are available at https://wireguard.com/install/ . On Linux systems, you can install the package using the distribution package manager and on macOS with `brew`.

Once installed follow the steps below to configure the client device.

The process for setting up a Linux and macOS client is pretty much the same as you did for the server. First generate the public and private keys:

```
wg genkey | sudo tee /etc/wireguard/privatekey | wg pubkey | sudo tee
/etc/wireguard/publickey
```

Create the file `wg0.conf` and add the following contents:

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

/etc/wireguard/wg0.conf

```
[Interface]
PrivateKey = CLIENT_PRIVATE_KEY
Address = 10.0.0.2/24


[Peer]
PublicKey = SERVER_PUBLIC_KEY
Endpoint = SERVER_IP_ADDRESS:51820
AllowedIPs = 0.0.0.0/0
```

The settings in the interface section have the same meaning as when setting up the server:

- Address - A comma-separated list of v4 or v6 IP addresses for the `wg0` interface.
- PrivateKey - To see the contents of the file on the client machine run: `sudo cat /etc/wireguard/privatekey`

The peer section contains the following fields:

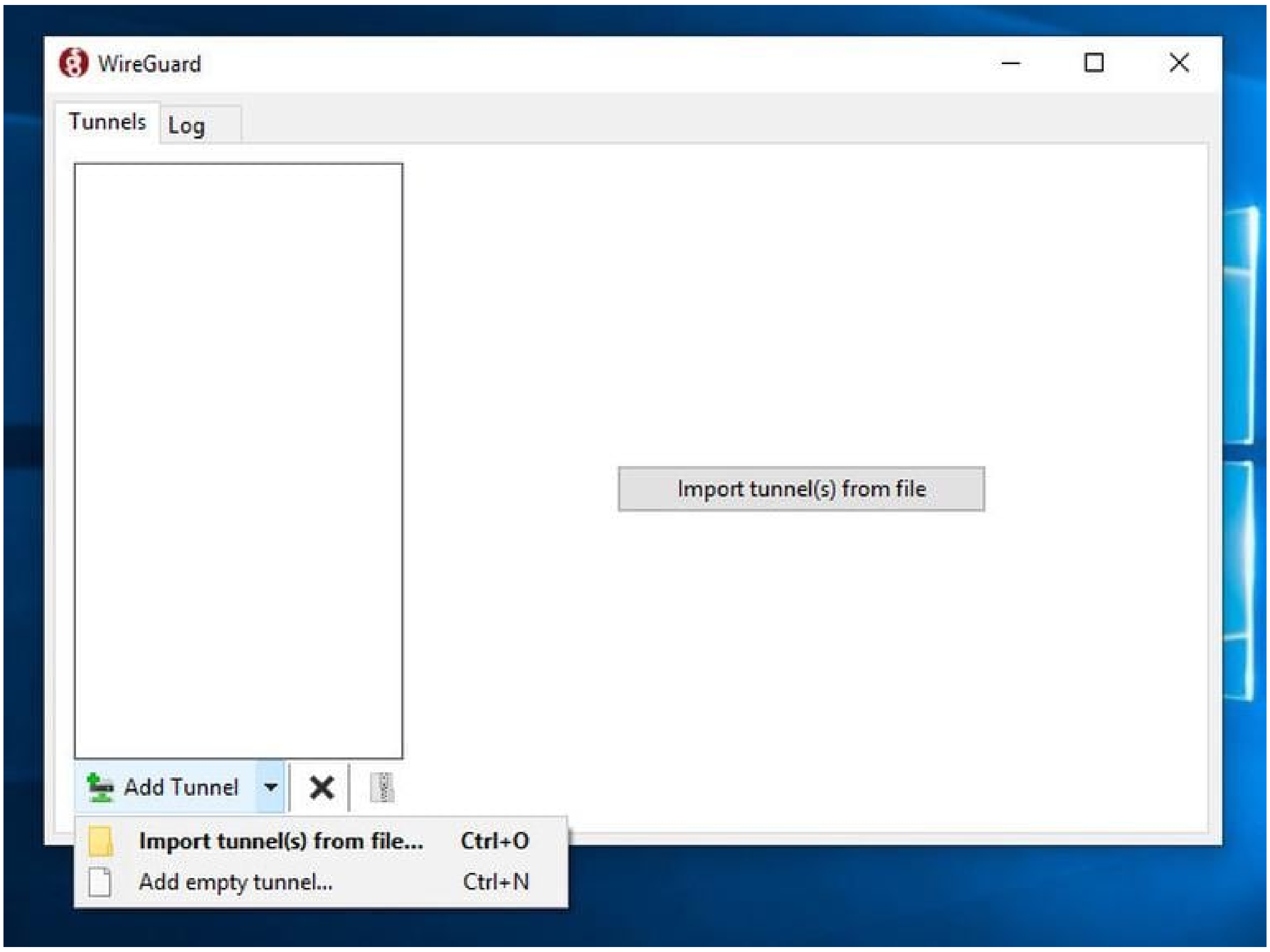
- PublicKey - A public key of the peer you want to connect to. (The contents of the server's `/etc/wireguard/publickey` file).
- Endpoint - An IP or hostname of the peer you want to connect to followed by a colon, and then a port number on which the remote peer listens to.
- AllowedIPs - A comma-separated list of v4 or v6 IP addresses from which incoming traffic for the peer is allowed and to which outgoing traffic for this peer is directed. We're using 0.0.0.0/0 because we are routing the traffic and want the server peer to send packets with any source IP.

If you need to configure additional clients, just repeat the same steps using a different private IP address.

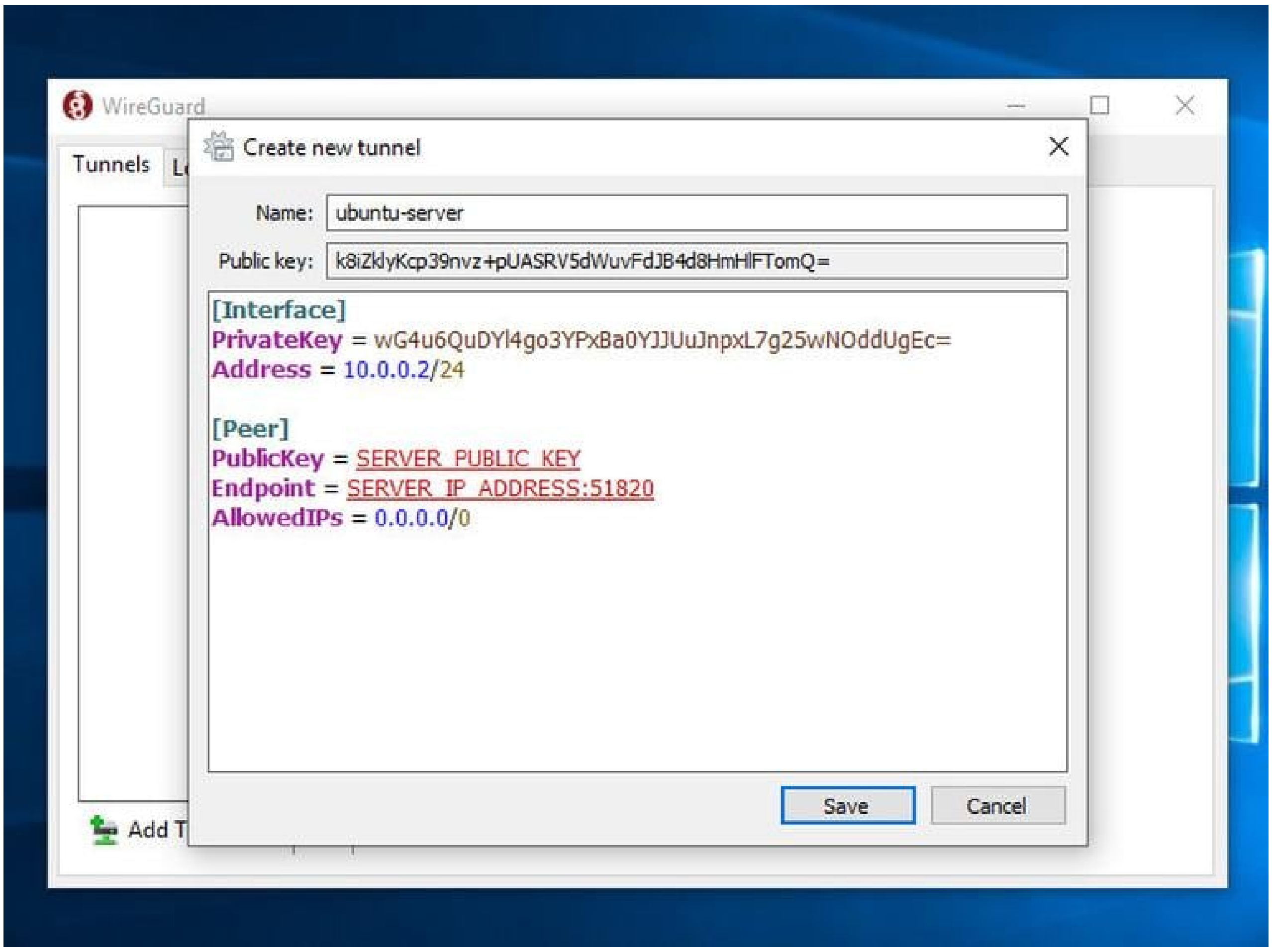
Windows Clients Setup

Download and install the Windows msi package from the WireGuard website .

Once installed, open the WireGuard application and click on “Add Tunnel” -> “Add empty tunnel…” as shown on the image below:


WireGuard
Tunnels
Log
Import tunnel(s) from file
Add Tunnel
Import tunnel(s) from file... Ctrl+O
Add empty tunnel... Ctrl+N

A publickey pair is automatically created and displayed on the screen.



Name: ubuntu-server

Public key: k8iZklyKcp39nvz+pUASRV5dWuvFdJB4d8HmHlFTomQ=

```
[Interface]
PrivateKey = wG4u6QuDYl4go3YPxBa0YJJUuJnpxL7g25wNOddUgEc=
Address = 10.0.0.2/24

[Peer]
PublicKey = SERVER_PUBLIC_KEY
Endpoint = SERVER_IP_ADDRESS:51820
AllowedIPs = 0.0.0.0/0
```

Enter a name for the tunnel and edit the configuration as follows:

```
[Interface]
PrivateKey = CLIENT_PRIVATE_KEY
Address = 10.0.0.2/24


[Peer]
PublicKey = SERVER_PUBLIC_KEY
Endpoint = SERVER_IP_ADDRESS:51820
AllowedIPs = 0.0.0.0/0
```

In the interface section, add a new line to define the client tunnel Address.

In the peer section, add the following fields:

- PublicKey - The public key of the Ubuntu server (`/etc/wireguard/publickey` file).
- Endpoint - The IP address of the Ubuntu server followed by a colon, and WireGuard port (51820).
- AllowedIPs - 0.0.0.0/0

Once done, click on the “Save” button.

Add the Client Peer to the Server

The last step is to add the client's public key and IP address to the server. To do that, run the following command on the Ubuntu server:

```
sudo wg set wg0 peer CLIENT_PUBLIC_KEY allowed-ips 10.0.0.2
```

Make sure to change the `CLIENT_PUBLIC_KEY` with the public key you generated on the client machine (`sudo cat /etc/wireguard/publickey`) and adjust the client IP address if it is different. Windows users can copy the public key from the WireGuard application.

Once done, go back to the client machine and bring up the tunneling interface.

Linux and macOS Clients

Run the following command the bring up the interface:

```
sudo wg-quick up wg0
```

Now you should be connected to the Ubuntu server, and the traffic from your client machine should be routed through it. You can check the connection with:

```
sudo wg

interface: wg0
  public key: gFeK6A16ncnT1FG6fJhOCMPMeY4hZa97cZCNWis7cSo=
  private key: (hidden)
  listening port: 53527
  fwmark: 0xca6c

peer: r3imyh3MCYggaZACmkx+CxlD6uAmICI8pe/PGq8+qCg=
  endpoint: XXX.XXX.XXX.XXX:51820
  allowed ips: 0.0.0.0/0
  latest handshake: 53 seconds ago
  transfer: 3.23 KiB received, 3.50 KiB sent
```

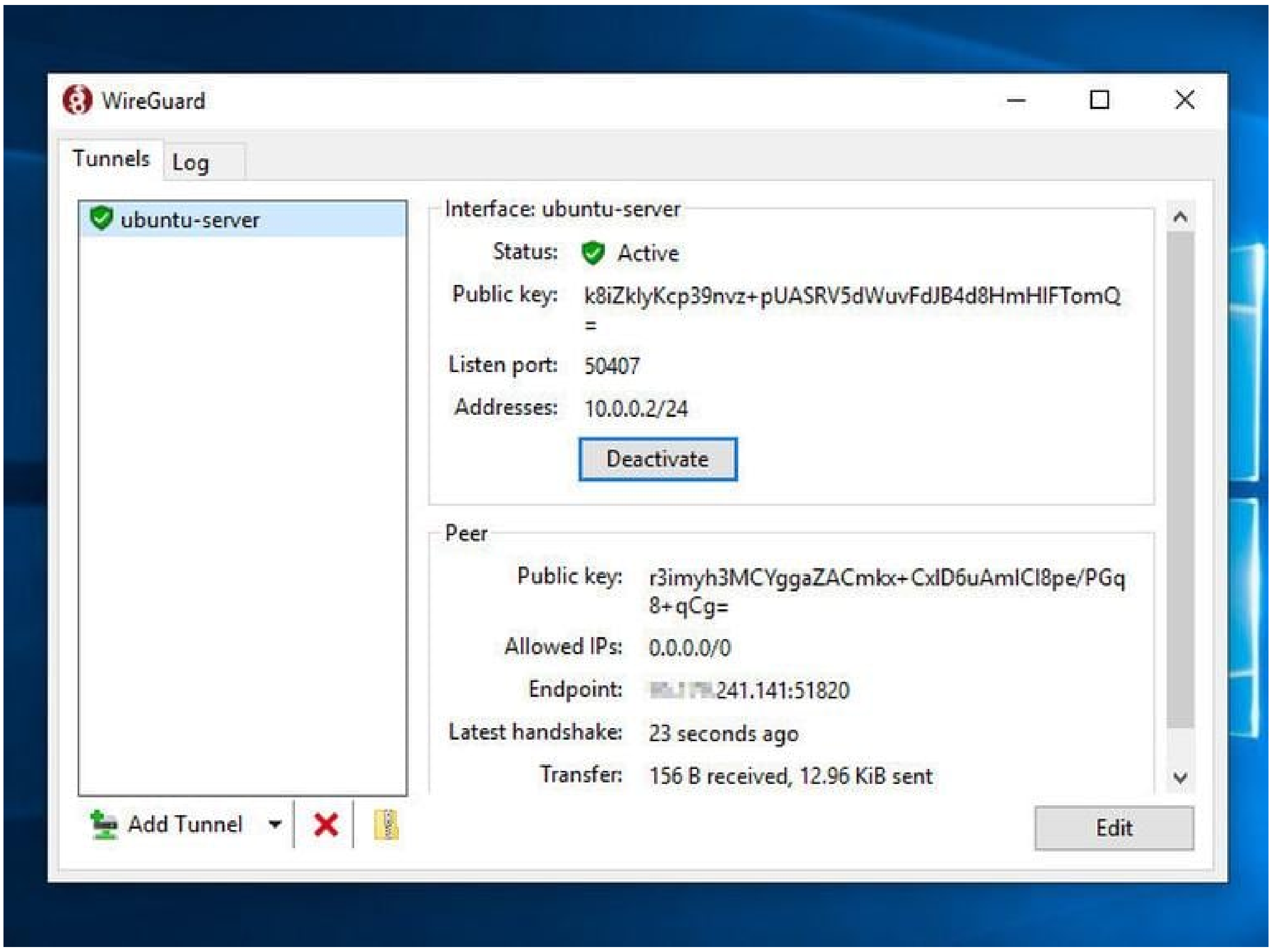
You can also open your browser, type "what is my ip", and you should see your Ubuntu server IP address.

To stop the tunneling, bring down the `wg0` interface:

```
sudo wg-quick down wg0
```

Windows Clients

If you installed WireGuard on Windows, click on the "Activate" button. Once the peers are connected, the tunnel status will change to Active:



Interface: ubuntu-server

```
Status:        Active
Public key:    k8iZklyKcp39nvz+pUASRV5dWuvFdJB4d8HmHIFTomQ
               =
Listen port:   50407
Addresses:     10.0.0.2/24
```

Peer

```
Public key:        r3imyh3MCYggaZACmkx+CxlD6uAmICl8pe/PGq
                   8+qCg=
Allowed IPs:       0.0.0.0/0
Endpoint:          [redacted].241.141:51820
Latest handshake:  23 seconds ago
Transfer:          156 B received, 12.96 KiB sent
```

Conclusion

We have shown you how to install WireGuard on an Ubuntu 20.04 machine and configure it as a VPN server. This setup allows you to surf the web anonymously by keeping your traffic data private.

If you are facing any problems, feel free to leave a comment.