Wireguard Project

Calvin Wasilevich

Setup DigitalOcean Droplet

The first step in setting up a remote VPN with Wireguard is starting a remote server to connect to. This is done with DigitalOcean, a cloud infrastructure provider.

After setting up the account, navigate to "Create a Droplet". "Droplet" is the term used by DigitalOcean to describe their virtual machines. Spin up a droplet with the following specifications:

- The closest region to you
- Ubuntu 24.04 (LTS) x64
- Basic shared CPU
- Regular CPU, $6/mo option is sufficient

Then, configure an ssh key pair to allow a secure ssh connection into the vm.

- Use the command `ssh-keygen` to generate a public/private key pair
    - Follow all instructions given by this command
- Upload the public key to the VM before starting it

SSH into the VM by using `ssh -i <path/to/private/key> root@<VM ipv4 address>`

Install Docker

- Update packages with `apt update` and `apt upgrade`
- Follow the instructions on docker's website to install docker engine
    - Uninstall all unofficial docker installs with

```
sudo apt remove $(dpkg --get-selections docker.io docker-compose docker-compose-v2 docker-doc podman-docker containerd runc | cut -f1)
```

    - Setup docker's apt repository

```
# Add Docker's official GPG key:
sudo apt update
sudo apt install ca-certificates curl
sudo install -m 0755 -d /etc/apt/keyrings
sudo curl -fsSL https://download.docker.com/linux/ubuntu/gpg -o /etc/apt/keyrings/docker.asc
sudo chmod a+r /etc/apt/keyrings/docker.asc

# Add the repository to Apt sources:
```

```
sudo tee /etc/apt/sources.list.d/docker.sources <<EOF
Types: deb
URIs: https://download.docker.com/linux/ubuntu
Suites: $(. /etc/os-release && echo "${UBUNTU_CODENAME:-$VERSION_CODENAME}")
Components: stable
Signed-By: /etc/apt/keyrings/docker.asc
EOF

sudo apt update
```

- Install the latest version of docker with

```
sudo apt install docker-ce docker-ce-cli containerd.io docker-buildx-plugin
docker-compose-plugin
```

- Verify docker install with

```
sudo docker run hello-world
```

```
root@wireguard:~# docker run hello-world

Hello from Docker!
This message shows that your installation appears to be working correctly.

To generate this message, Docker took the following steps:
 1. The Docker client contacted the Docker daemon.
 2. The Docker daemon pulled the "hello-world" image from the Docker Hub.
    (amd64)
 3. The Docker daemon created a new container from that image which runs the
    executable that produces the output you are currently reading.
 4. The Docker daemon streamed that output to the Docker client, which sent it
    to your terminal.

To try something more ambitious, you can run an Ubuntu container with:
 $ docker run -it ubuntu bash

Share images, automate workflows, and more with a free Docker ID:
 https://hub.docker.com/

For more examples and ideas, visit:
 https://docs.docker.com/get-started/
```

Create Wireguard Image with Docker

- Create the following directory structure with `mkdir -p wireguard/config`

```
wireguard/
└── config
```

- Make the file `wireguard/compose.yml` and paste the following code:

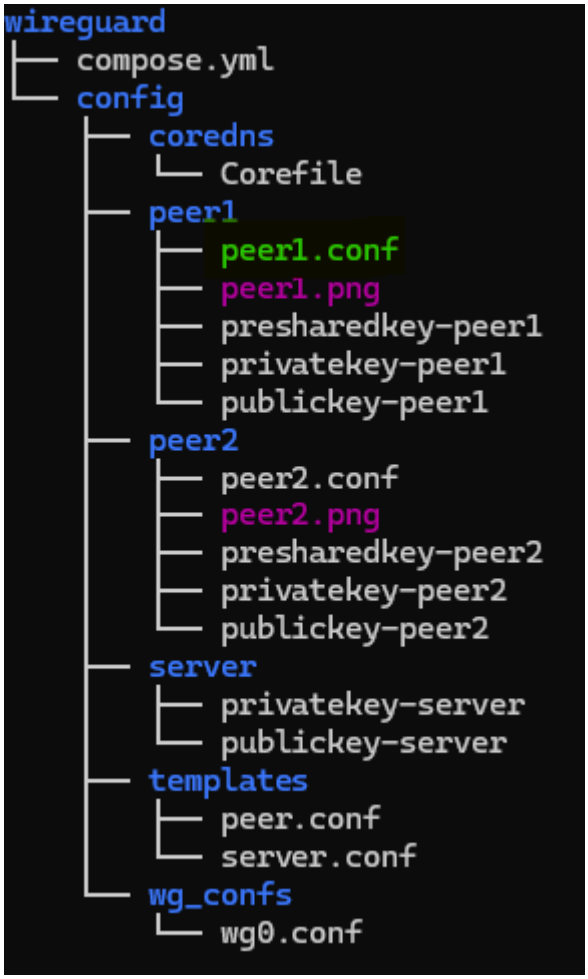
```
version: '3.8'
services:
  wireguard:
    container_name: wireguard
    image: linuxserver/wireguard
    environment:
      - PUID=1000
      - PGID=1000
      - TZ= # set this line with the time zone
      - SERVERURL= # set this line with the server url
      - SERVERPORT=51820
      - PEERS=2 # set this line with the number of peers
      - PEERDNS=auto
      - INTERNAL_SUBNET=10.0.0.0
    ports:
      - 51820:51820/udp
    volumes:
      - type: bind
        source: ./config/
        target: /config/
      - type: bind
        source: /lib/modules
        target: /lib/modules
    restart: always
    cap_add:
      - NET_ADMIN
      - SYS_MODULE
    sysctls:
      - net.ipv4.conf.all.src_valid_mark=1
```

- Modify the commented areas:
  - `TZ` is the timezone; consult this page for the proper format
  - `SERVERURL` is the ip address of the cloud VM
  - `PEERS` is the original number of clients to make configuration files for. For this project, we only need two devices, so two is set for peers.
- Run `sudo docker compose up -d` while in the `wireguard` directory to start the image
  - The `-d` indicates to run in detached mode

Connecting to Wireguard

Windows Laptop

- Download and install the proper wireguard client from the wireguard website
- Get the peer configuration file from wireguard
    - In the `wireguard` directory in the VM, locate the `config/peer1` directory
    - In this directory, copy the `peer1.conf` file (highlighted in green) to the windows machine

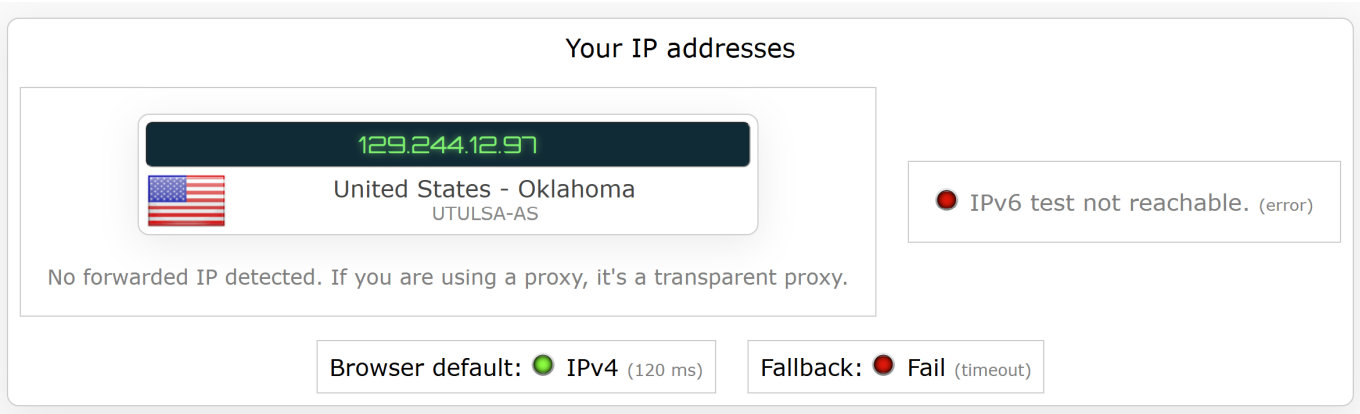


    - Use the `scp` command to use ssh to tranfer the file

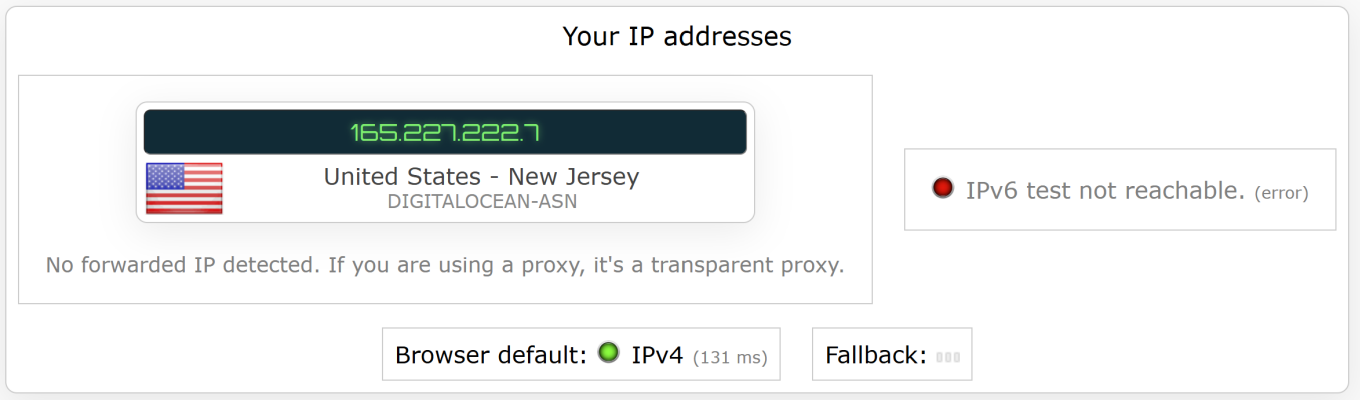
```
scp -i <path/to/private/key> root@<VM ipv4 address>:
</path/to/config/file> .
```

- From the Wireguard client, add a tunnel from configuration file with the newly copied file
- Click activate

Before Connection:



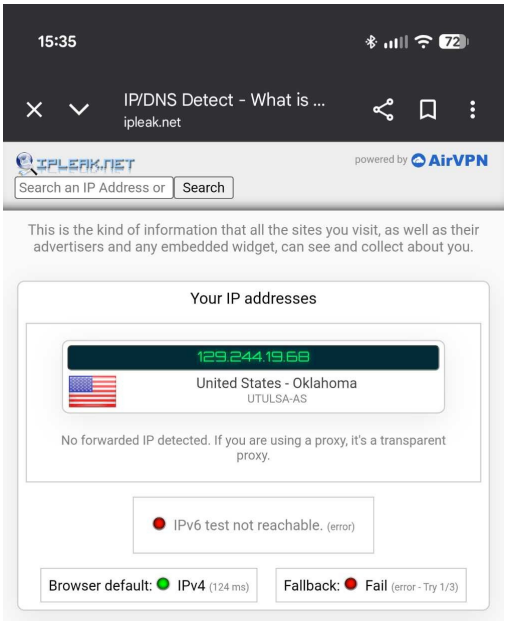
After Connection:



Android Device

- Download and install the wireguard client app from the Google Play Store
- Get the peer configuration file from wireguard
  - In the `wireguard` directory in the VM, locate the `config/peer2` directory
  - In the this directory, copy the `peer2.png` file
    - This is a QR code for easy tunnel setup
  - Use the `scp` command as in the windows setup
- From the Wireguard app, select the plus and then select the option to scan from QR code
- Scan the QR code downloaded from the server
- Activate the VPN connection

Before Connection:



After Connection: