WireGuard

WireGuard® is an extremely simple yet fast and modern VPN.

Installation (Raspberry Pi 2/3/4)

> At the time of this writing, the WireGuard module has yet to be included in the Raspberry Pi kernel. The Debian Testing repository must be added to the Raspberry Pi in order to install the necessary tools and the kernel module.

Add the Debian Testing Repository

The following Commands will add the Debian testing repository and set it's priority lower than the Raspberry Pi OS stable repository. This way, all packages will be updated against the stable repos, unless they are not available in which case apt will fall back to check the Debian testing repo.

```
$ echo "deb http://archive.raspbian.org/raspbian testing main" | sudo tee --append
/etc/apt/sources.list.d/testing.list
$ printf 'Package: *\nPin: release a=testing\nPin-Priority: 50\n' | sudo tee --append
/etc/apt/preferences.d/limit-testing
$ sudo apt update
```

Install the WireGuard Package

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

Allow Remote Access to the Local Network

- Configure a static IP address or DHCP reservation on the Raspberry Pi
- Configure a static WAN IP address or DDNS service on the router
- **Port forward UDP port 51820** from the router to the Raspberry Pi

Enable IP Forwarding

On the Raspberry Pi, edit `/etc/sysctl.conf` uncommenting the following line:

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

This will require a reboot to take effect.

Create the the server/client keys

Start by creating a directory in a secure location for the keys. inside this directory run the following command for the "server" keys:

```
$ mkdir wg-configs
$ cd wg-configs
$ wg genkey | tee wg0_privkey | wg pubkey > wg0_pubkey
```

This will create two files, `wg0_privkey` and `wg0_pubkey` that each contain a hashed key.

For each "client" peer, run the same command as above, changing the output filenames to reflect which peer key-pair is being generated:

```
$ wg genkey | tee wg0client1_privkey | wg pubkey > wg0client1_pubkey
$ wg genkey | tee wg0client2_privkey | wg pubkey > wg0client2_pubkey
...
```

Server Configuration

Create the file `/etc/wireguard/wg0.conf` containing the following, replace items indicated by `<...>` with the key hashes found in the files created above.

Additional peers may be added by appending more `[Peer]` blocks.

```
[Interface]
Address = 192.168.2.1/24
PrivateKey = <wg0_privkey>
ListenPort = 51820  # udp
```

```
# allow access to local network from wireguard interface, change eth0 to wlan0 if using wifi
PostUp = iptables -A FORWARD -i %i -j ACCEPT; iptables -A FORWARD -o %i -j ACCEPT; iptables -t nat -A POSTROUTING -o eth0 -j MASQUERADE
PostDown = iptables -D FORWARD -i %i -j ACCEPT; iptables -D FORWARD -o %i -j ACCEPT; iptables -t nat -D POSTROUTING -o eth0 -j MASQUERADE

[Peer]
PublicKey = <wg0client1_pubkey>
AllowedIPs = 192.168.2.2/32

[Peer]
PublicKey = <wg0client2_pubkey>
AllowedIPs = 192.168.2.3/32
```

Activate the wg0 interface with systemd

Enable and start the WireGuard Interface with:

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

Client Configuration

> Note: the filename of the config file determines the WireGuard interface name, for example `wg0client1.conf` creates an interface called `wg0client1`. Interfaces can be named whatever you want, it may be helpful to name them after the network you're connecting to and/or the peer name

Very similar to the "server" configuration above, each client config should contain the following, again replacing the items indicated by `<...>` with the key hashes:

```
[Interface]
Address = 192.168.2.2/24
DNS = 192.168.1.1
PrivateKey = <wg0client1_privkey>

[Peer]
```

```
PublicKey = <wg0_pubkey>
AllowedIPs = 0.0.0.0/0  # routes all traffic
Endpoint = <DNS-resolvable-name>:51820
```

> The endpoint can be either the router's WAN Ip address or the (D)DNS name, but must also contain the port number of the server, i.e. www.franklin57.com:51820.

Create a config file or enter the information above for each "client" device, updating the `Address` and `PrivateKey` to match what is in the "server's" config and the specific client's public key file.

Revision #7
Created 2021-01-11 14:37:44 UTC by uncarvedblock
Updated 2021-01-14 04:28:47 UTC by uncarvedblock