WireGuard VPN

See https://github.com/pirate/wireguard-docs for more!

Table of Contents

1. Why Use WireGuard?

- The setup is easy, with only the keys needing to be shared + one simple conf
- Uses modern and fast cryptography everywhere
- Minimal attack surface (~4,000 lines of kernel code vs 600,000 for OpenVPN)
- High performance (https://www.wireguard.com/performance/)

2. Basic Setup

2.1. Installation

2.1.1. Module

- Shipped by default with Linux 5.6 and up
- `uname -r` to see the kernel version

2.1.2. Tools

- `pacman -S wireguard-tools`

2.2. Key-Pair Generation

- `wg genkey | tee peer.key | wg pubkey > peer.pub`
- `wg genkey` generates a private key

- `wg pubkey`, generates a public key from some private key

2.3. Configuration

`/etc/wireguard/wg0.conf`

```
[Interface]
Address = 10.0.0.1/24, fdc9:281f:04d7:9ee9::1/64
ListenPort = 51820
PrivateKey = PEER_A_PRIVATE_KEY

[Peer]
PublicKey = PEER_B_PUBLIC_KEY
AllowedIPs = 10.0.0.2/32, fdc9:281f:04d7:9ee9::2/128
Endpoint = peer-b.example:51820
PersistentKeepalive = 25
```

2.3.1. Interface

Address

IPv4 & IPv6 (optional) addresses on the VPN subnet

ListenPort

Port to listen for VPN connections on

PrivateKey

The private key unique to this peer

2.3.2. Peer

PublicKey

The public key of the peer being connected to

AllowedIPs

Used for routing & as a firewall (depending on the direction)

Endpoint

The public address of the peer – only needed by one of a pair

PersistentKeepalive

Ping every N seconds to keep NATed connections open

2.4. Starting the Service

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

3. Forwarding + NAT

- PostUp in `wg0.conf` enables forwarding and NAT
- PostDown runs after WireGuard is disabled, reverting the PostUp

3.1. sysctl

- `sysctl net.ipv4.ip_forward=1/0`

3.2. NAT

- `iptables -t nat -A/D POSTROUTING -o eth0 -j MASQUERADE`