WireGuard Fast VPN tunnels

CONCEPTS

Each peer is a public key plus the IPs it may use (AllowedIPs). The same list routes outgoing packets and filters incoming ones.

Packet to 10.0.0.2
↓ AllowedIPs
Peer matching key
↓ encrypt, UDP
Endpoint ip:port

KEYS

`wg genkey > privatekey`
Generate a private key

`wg pubkey < privatekey > publickey`
Derive the public key

BRING UP AND DOWN

`sudo wg-quick up <iface>`
Start tunnel from /etc/wireguard/<iface>.conf

`sudo wg-quick down <iface>`
Stop tunnel and undo routes

`sudo systemctl enable --now wg-quick@<iface>`
Start now and at boot

INSPECT

`sudo wg`
Show all interfaces and peers

`sudo wg show <iface>`
Show one interface

CHANGE LIVE

`sudo wg set <iface> peer <pubkey> allowed-ips <cidr> endpoint <ip:port>`
Add or update a peer

`sudo wg set <iface> peer <pubkey> remove`
Remove a peer

CONFIG FILE

[Interface]

Field	Meaning
`PrivateKey`	Own secret key
`Address`	Tunnel IP, e.g. 10.0.0.1/24
`ListenPort`	UDP port (51820 usual)
`DNS`	Resolver for the tunnel
`MTU`	Override detected MTU
`Table`	off stops route creation
`PostUp / PostDown`	Shell hooks; %i is iface
`SaveConfig`	Rewrite file on down

[Peer]

Field	Meaning
`PublicKey`	Peer's public key
`AllowedIPs`	IPs sent to peer
`Endpoint`	host:port to dial
`PersistentKeepalive`	Secs; 25 if NAT
`PresharedKey`	Optional extra secret

Minimal /etc/wireguard/wg0.conf

```
[Interface]
PrivateKey = <private key>
Address = 10.0.0.2/24

[Peer]
PublicKey = <peer public key>
Endpoint = vpn.example.com:51820
AllowedIPs = 10.0.0.0/24
PersistentKeepalive = 25
```

ROUTING

AllowedIPs choices

Value	Effect
`10.0.0.2/32`	Only that one host
`10.0.0.0/24`	A whole subnet
`0.0.0.0/0, ::/0`	All traffic via peer

0.0.0.0/0 makes wg-quick set up policy routing so the default route goes through the tunnel; the endpoint itself stays reachable.

TROUBLESHOOT

`sudo wg show`
No handshake line means no contact yet

`ping <peer tunnel ip>`
Test the tunnel