WIREGUARD

FAST, MODERN, SECURE VPN TUNNEL

Wireguard

Self-Hosted VPN

Prerequisites

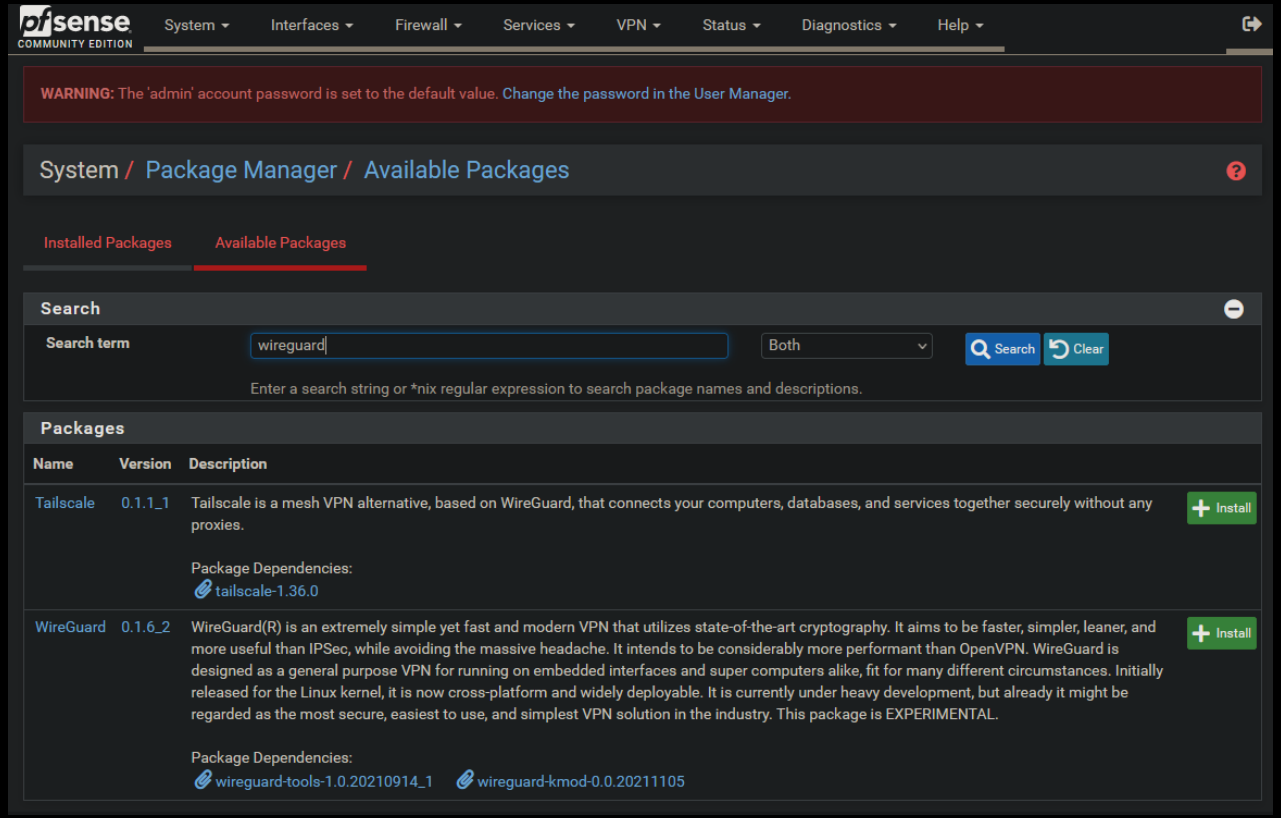
- Android phone with Wireguard app
- iOS phone with Wireguard app
- PFsense Firewall/Router
- OpenWRT router
- Travel router (OpenWRT)
- Virtual Private Server running Debian/Ubuntu based OS

Things to Note

- This guide will only show how to setup a connection from a single peer to a server. In most cases the configurations shown are okay. However, not all security practices are described or setup. You will have to make updates based on your current environment.

PfSense Setup

Install Wireguard Package

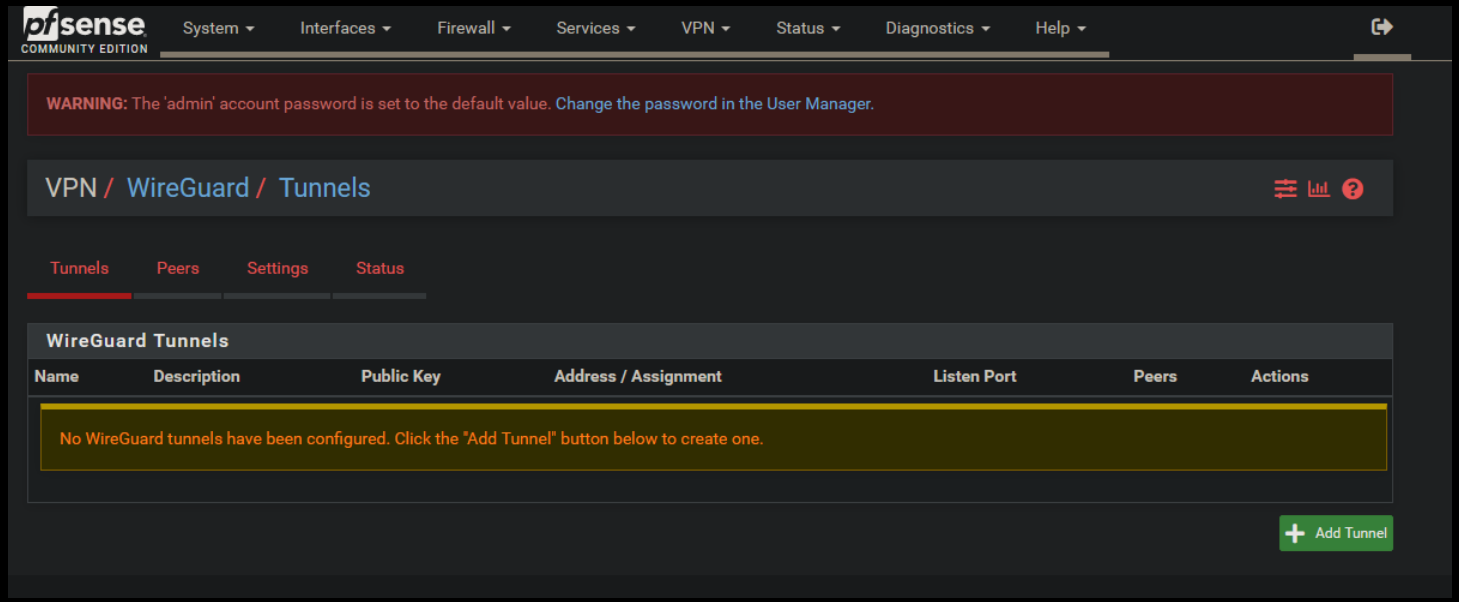


Go to System → Package Manager → Available Packages.

Search for “Wireguard”

At the time of making this, version 0.1.6_2 is the most up to date.

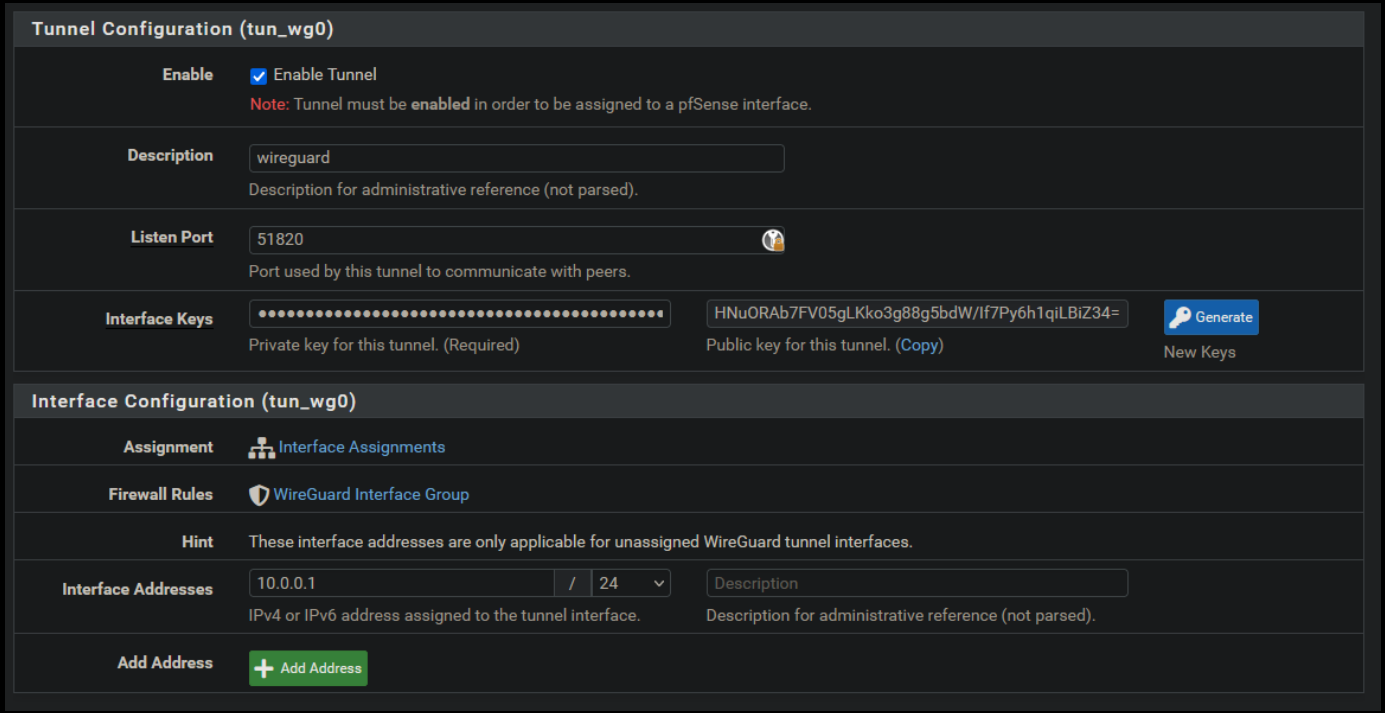
Wireguard Config



From the menu at the top, select VPN → Wireguard.

Select [+ Add Tunnel]

Tunnel Config



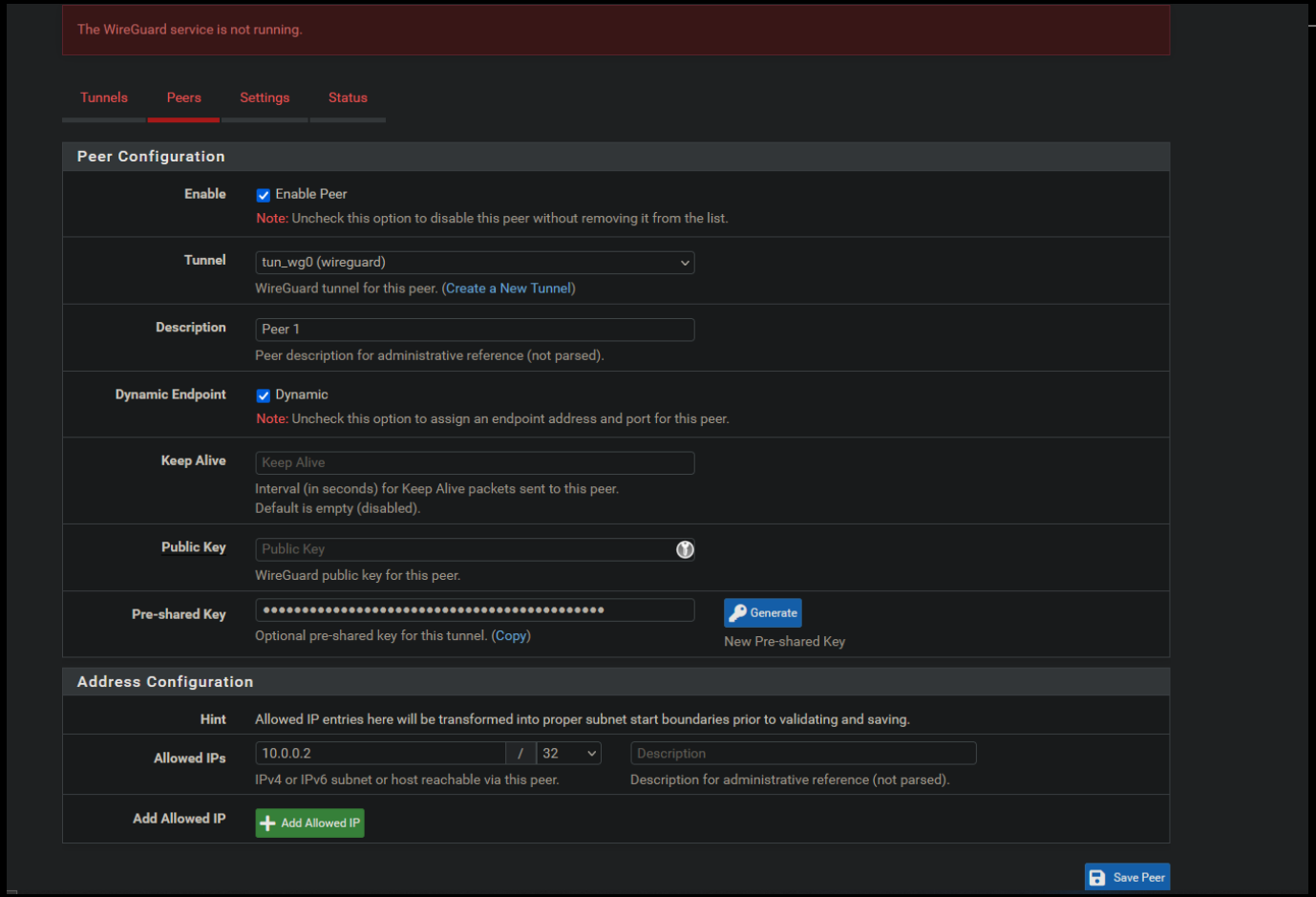
Once finished, click save. Then go to Peers.

1. Enable tunnel
2. Set a description for the tunnel
3. Set the port to anything. (Default is 51820)
4. Click “Generate” to create your public and private keys.
5. Create interface addresses.

Interface addresses can be any internal IP address range. Here I chose the 10.0.0.1/24 subnet.

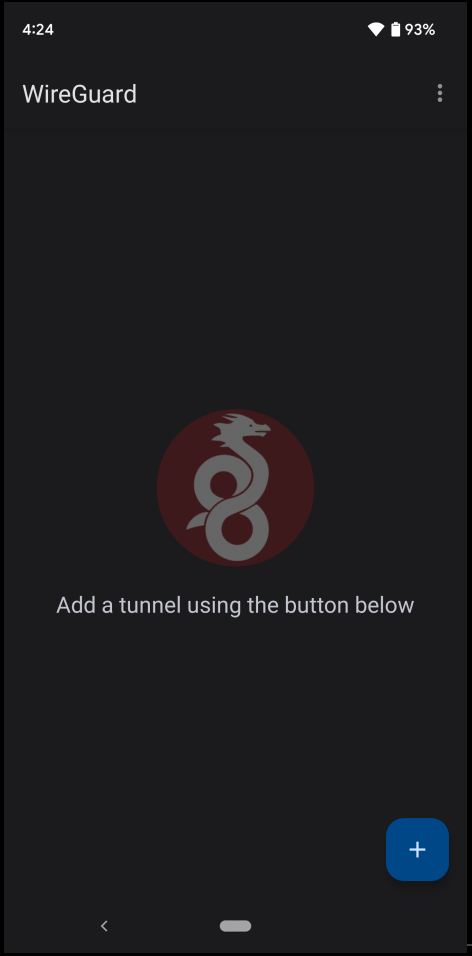
Meaning I can use 10.0.0.2-10.0.0.254
See RCF1918.

Peer Config

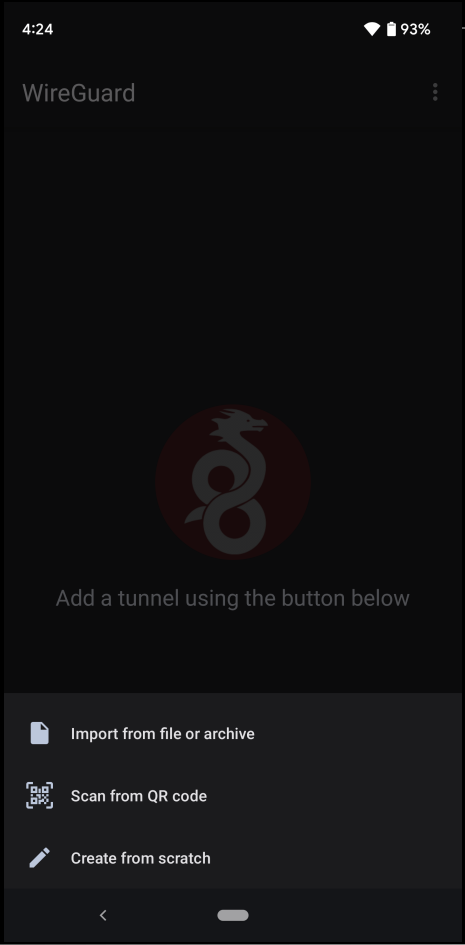


1. Enable peer.
2. Select the tunnel you created (in this case tun_wg0).
3. Set a Description.
4. Keep the endpoint dynamic.
5. Keep Alive: leave blank.
6. Public Key* this will be generate from your app.
7. Pre-Shared Key (optional but can provide quantum protection) Click [Generate] to create one.
8. Allowed IP's: Set this to an address that has not been used in your subnet yet.
Ex: 10.0.0.2/32 (means only 10.0.0.2)

App Config: Android

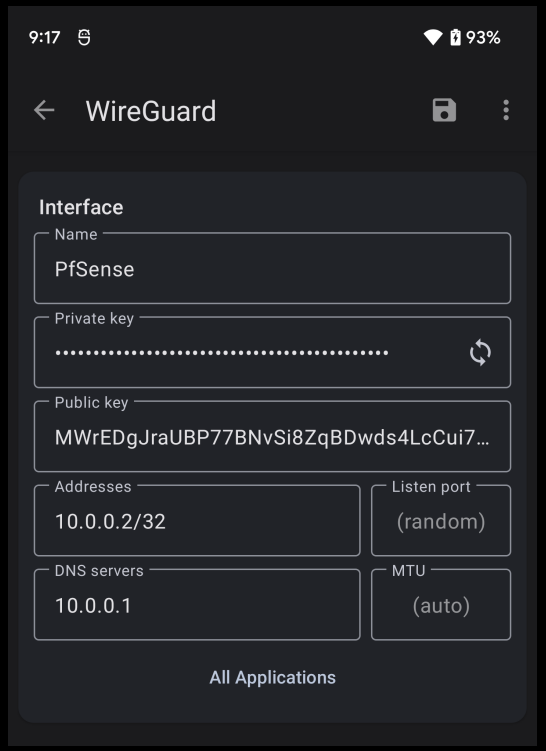


Click the + button



Select create from scratch

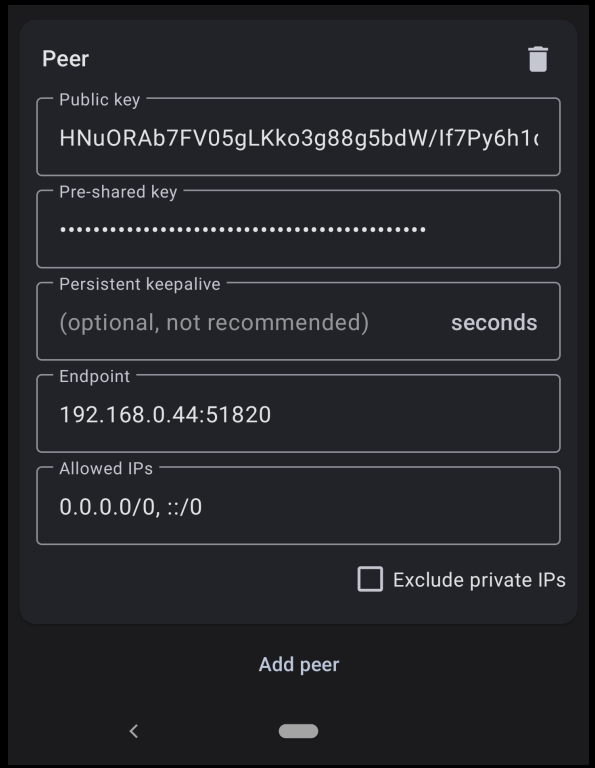
App Config: Android



1. Interface Name: Pfsense or any name you prefer
2 Click the refresh icon in the Private Key section to generate a public and private key.
3. The addresses should be the address you gave to the peer. In this case it is 10.0.0.2/32.
4. DNS Servers: You can set the DNS server to the upstream DNS on your network (usually 192.168.0.1 or 10.0.0.1) if you created a DNS for the tun0 interface, or leave this blank.
5. MTU leave empty.
6. Copy the Public Key and paste it into the Public Key section for the Peer on the server. See Slide 7, #6.

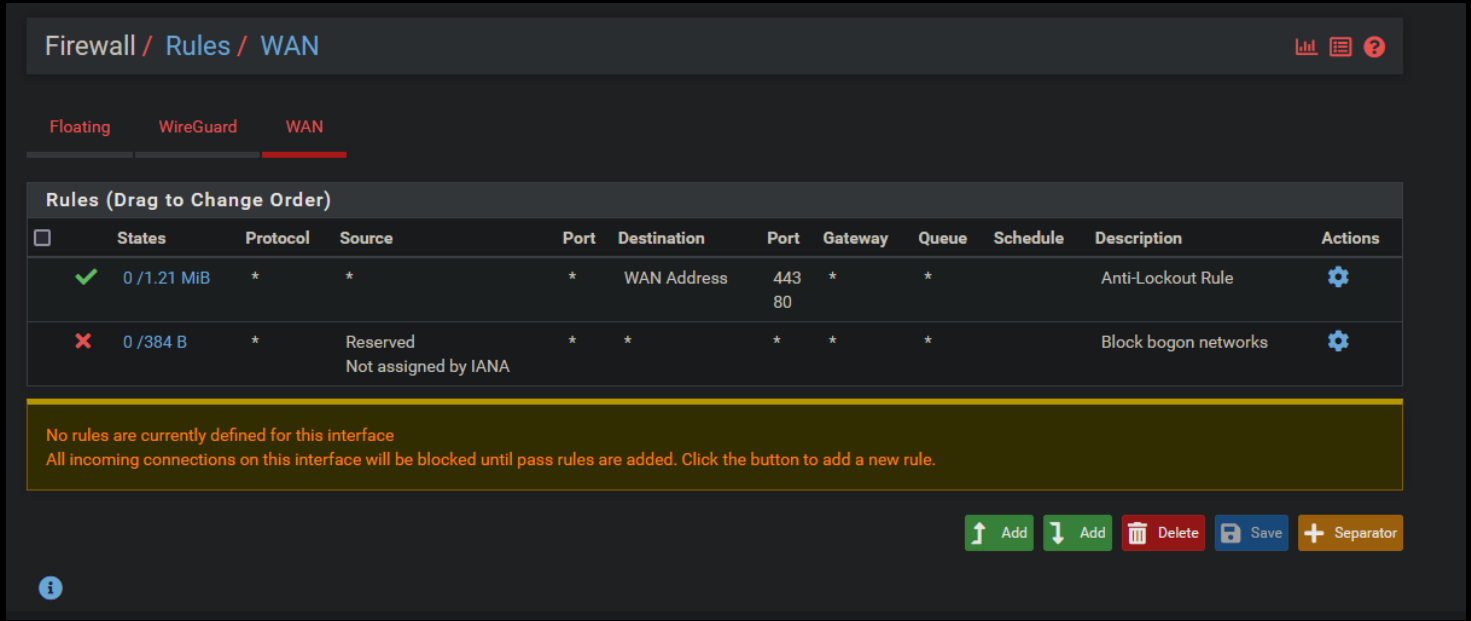
Then select “Add Peer” at the bottom

App Config: Android



1. Public Key: Copy the public from the server. (Slide 6 #4)
2. Pre-shared key: Copy the pre-shared key from the server. (Slide 7 #7)
3. Persistent keepalive: Leave Blank
4. Enpoint: This should be the public IP of your server. In my case my PFsense box is a VM on my local net so the IP is 192.168.0.44. Also :51820 is the port specification we configured.
5. Allowed IPs: 0.0.0.0/0 means any IP address on IPv4 and ::/0 means any IP address on IPv6. Setting it up this way means we created a Full-Tunnel. All traffic on the device will be routed through the wireguard VPN.

Firewall Config



Select Firewall → Rules from the main menu.

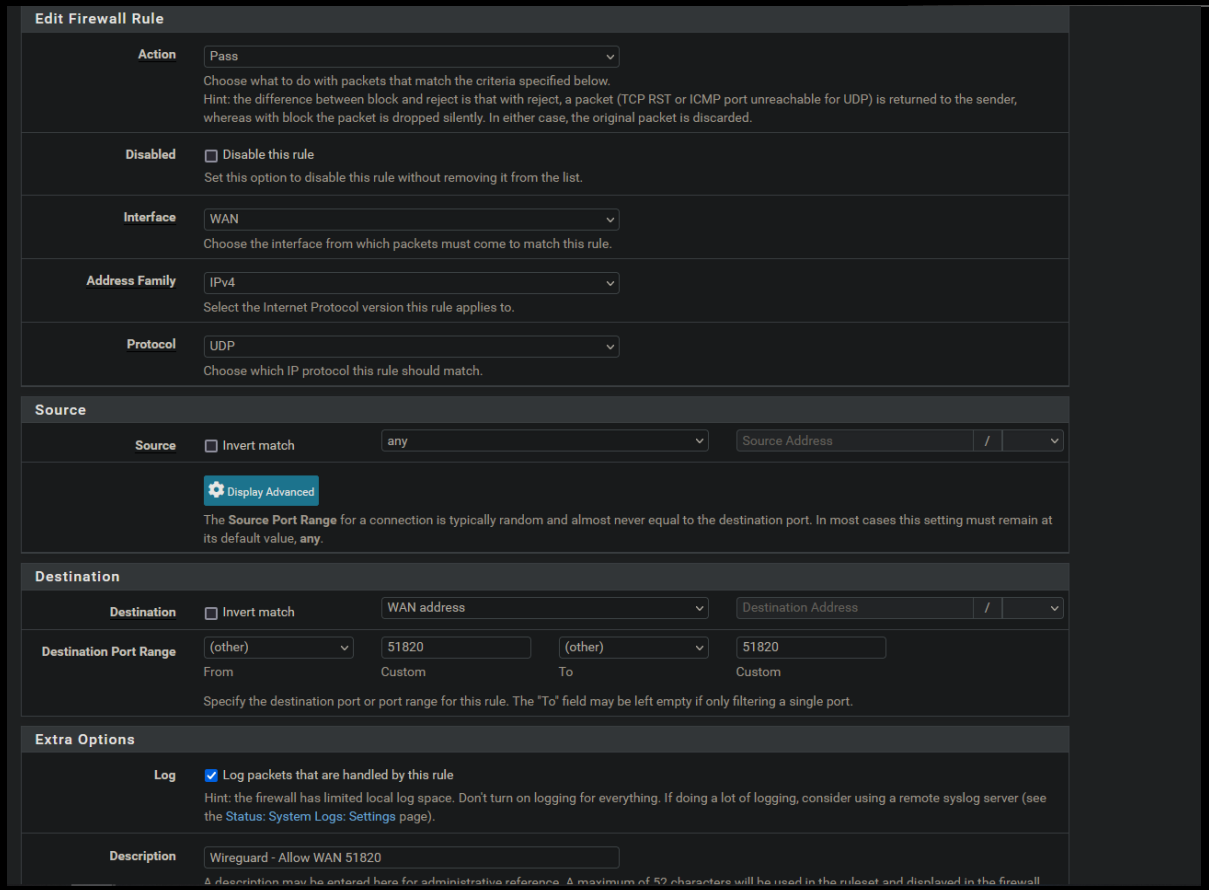
By default your WAN should only have 1 rule.

Block bogon networks.

Mine has an additional rule since I do not have a LAN address on this VM. Ignore this rule.

Select “Add \/”

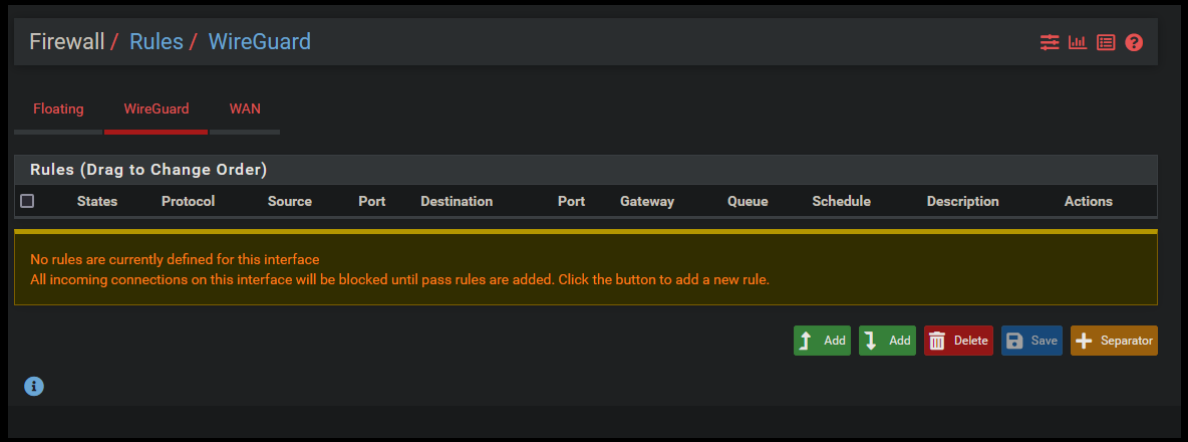
Firewall Config



1. Action: Pass
2. Interface: WAN
3. Address Family: IPv4
4. Protocol: UDP
5. Source: any
6. Destination: WAN address
7. Destination Port Range: Custom: 51820 to Custom: 51820
8: Log Packets: Optional (can create a lot of spam)
9. Description: Wireguard – Allow WAN 51820

Then click Save. You may have to click appy changes after you save.

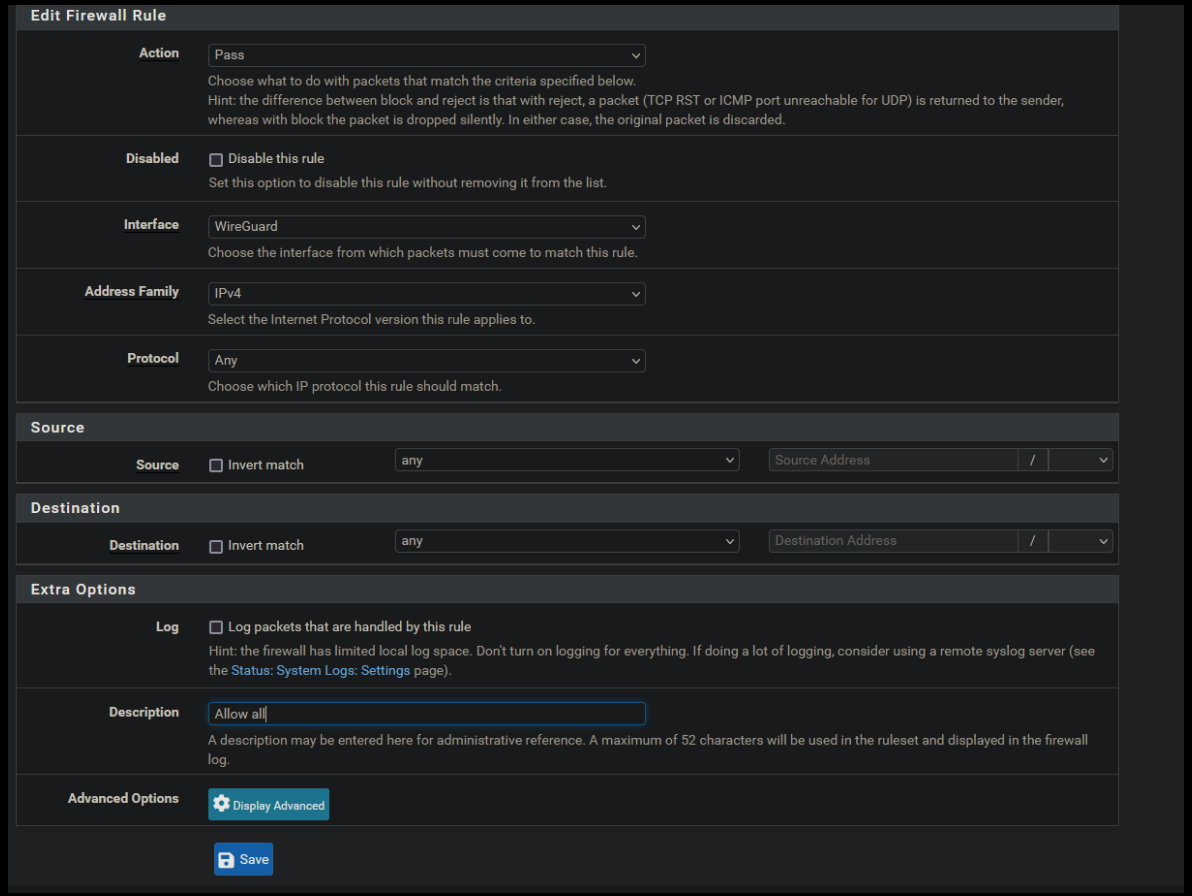
Wireguard Firewall



Select Add to create a new rule

By default there are no rules to allow Wireguard traffic anywhere. We need to create 1 rule to allow all traffic. But be warned, this tutorial does not cover how to properly segment your network and add additional security to your wireguard interface. VLANs, proper firewall rules, and network segmentation is important.

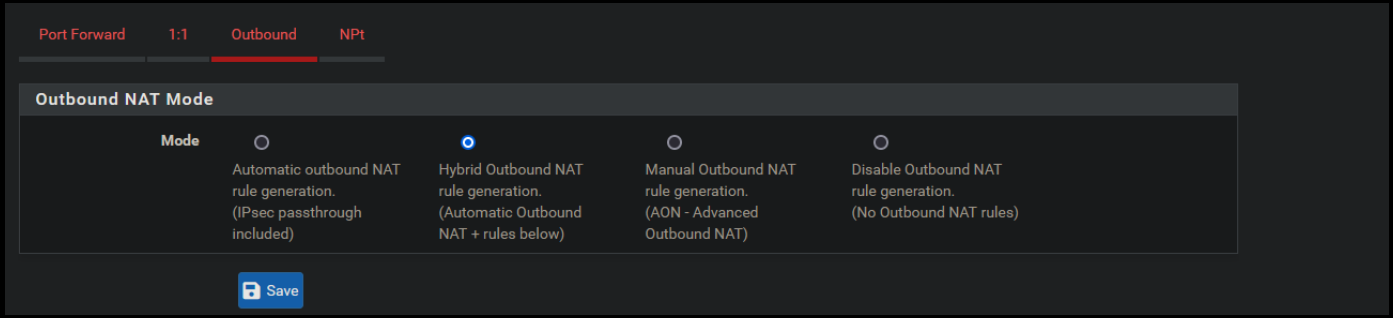
Wireguard Firewall Rules



1. Action: Pass
2. Interface: Wireguard
3. Address Family: IPv4
4. Protocol: Any
5. Source: Any
6. Destination: Any
7. Log: Optional
8. Description: Allow all
9. Save

Apply Changes after saving

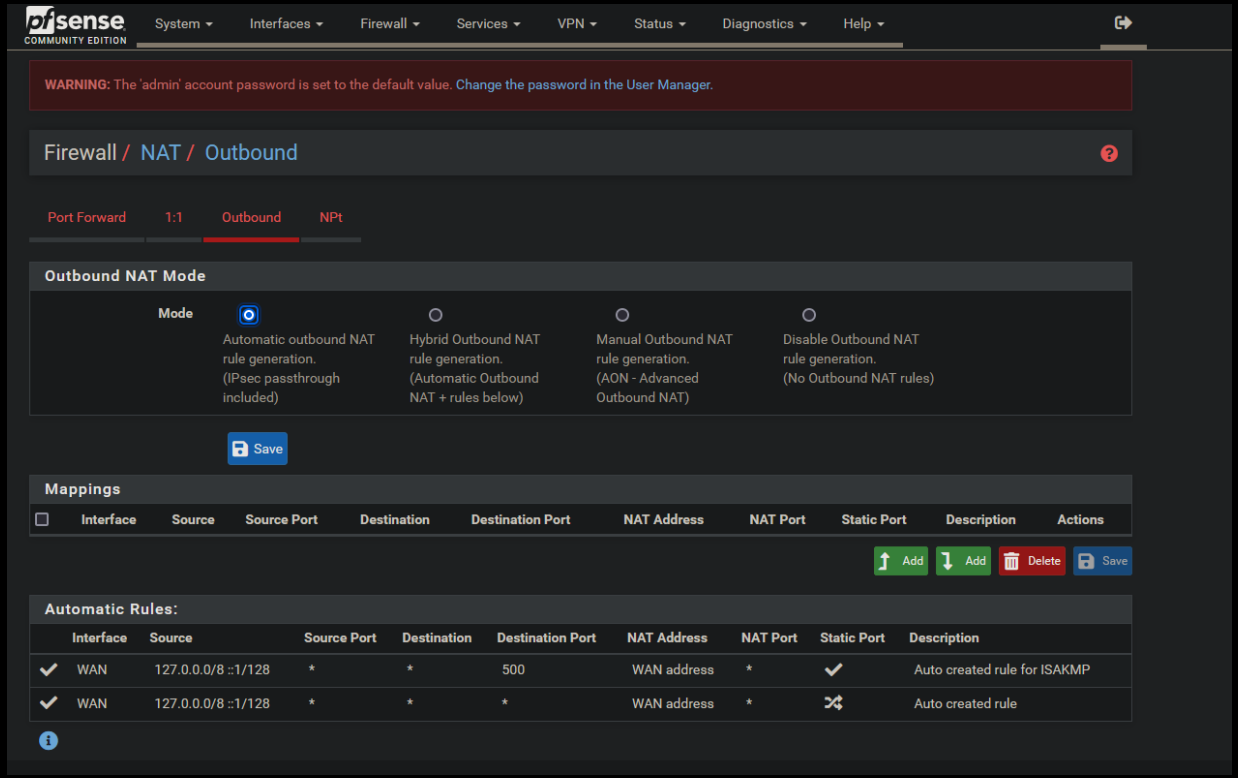
Firewall NAT Rules



Select Firewall → NAT and then Outbound to arrive at this page.

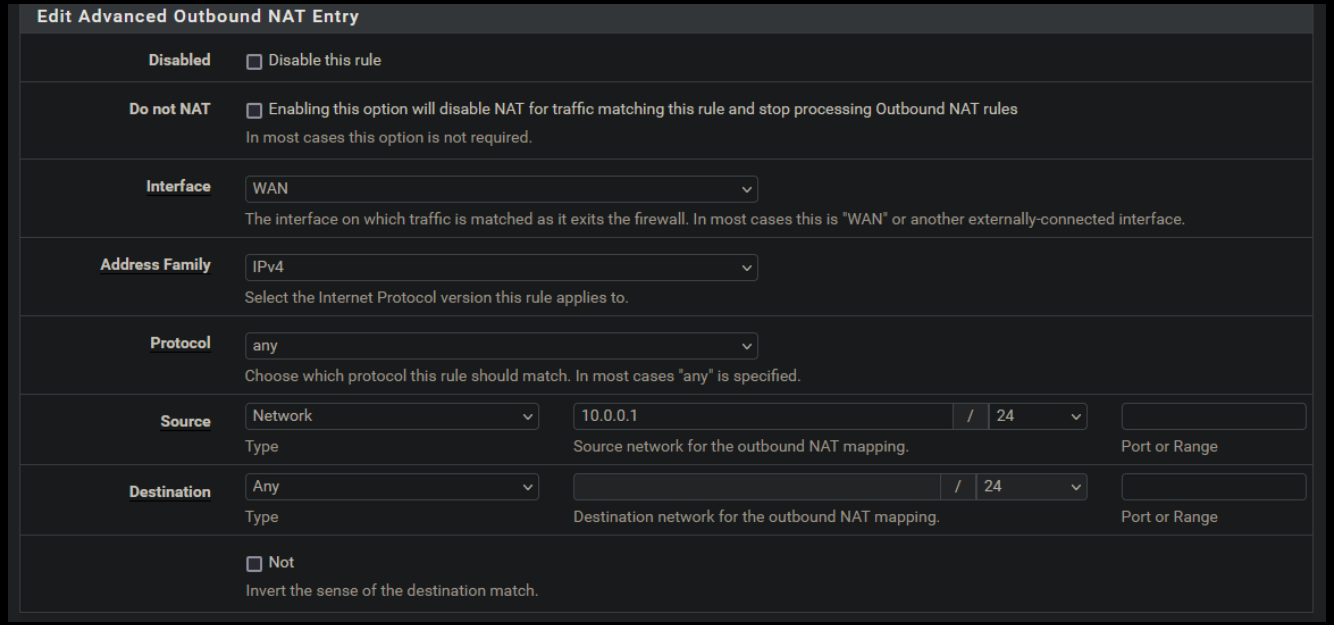
In order to route Wireguard traffic back out the WAN we need to configure a few things in the NAT settings.

Set Outbound NAT Mode to Hybrid.



Select “Add /\” under Mappings to create a new NAT Outbound Mapping.

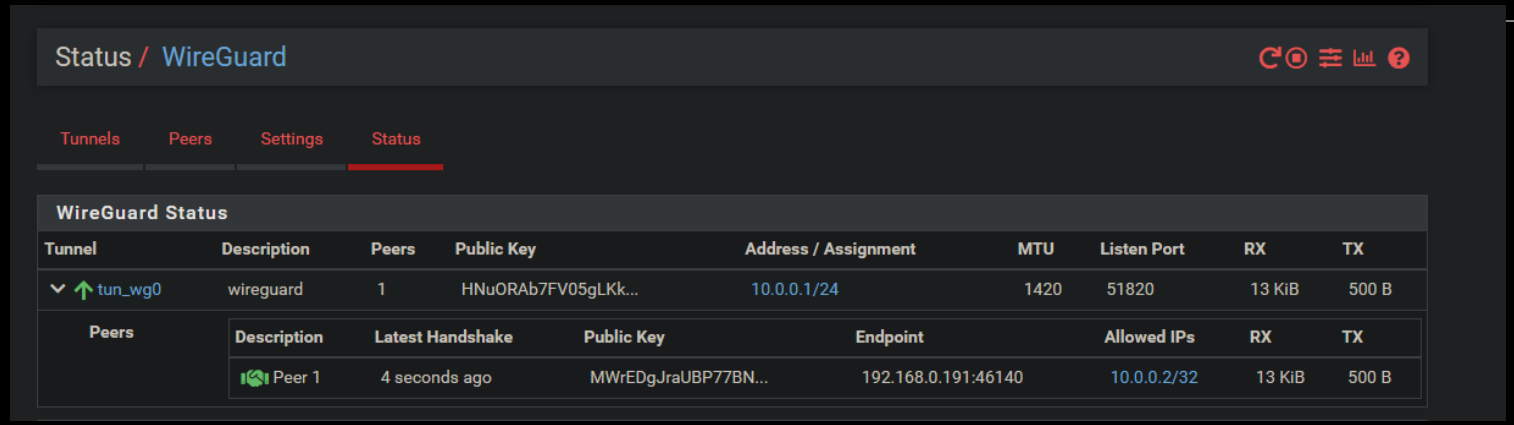
Firewall NAT Outbound



1. Interface: WAN
2. Address Family: IPv4
3. Protocol: any
4. Source: Network – 10.0.0.1/24

Then click save. You may need to apply the changes after clicking save.

Verify



After saving and reloading the firewall go back to VPN → Wireguard → Status. Click the down arrow on the tun_wg0 tunnel. You should see that the peer you created has successfully been created.

OpenWRT Setup

Update and Install Wireguard

```
root@OpenWrt:~# opkg update
Downloading https://downloads.openwrt.org/releases/22.03.4/targets/x86/64/packages/Packages.gz
Updated list of available packages in /var/opkg-lists/openwrt_core
Downloading https://downloads.openwrt.org/releases/22.03.4/targets/x86/64/packages/Packages.sig
Signature check passed.
Downloading https://downloads.openwrt.org/releases/22.03.4/packages/x86_64/base/Packages.gz
Updated list of available packages in /var/opkg-lists/openwrt_base
Downloading https://downloads.openwrt.org/releases/22.03.4/packages/x86_64/base/Packages.sig
Signature check passed.
Downloading https://downloads.openwrt.org/releases/22.03.4/packages/x86_64/luci/Packages.gz
Updated list of available packages in /var/opkg-lists/openwrt_luci
Downloading https://downloads.openwrt.org/releases/22.03.4/packages/x86_64/luci/Packages.sig
Signature check passed.
Downloading https://downloads.openwrt.org/releases/22.03.4/packages/x86_64/packages/Packages.gz
Updated list of available packages in /var/opkg-lists/openwrt_packages
Downloading https://downloads.openwrt.org/releases/22.03.4/packages/x86_64/packages/Packages.sig
Signature check passed.
Downloading https://downloads.openwrt.org/releases/22.03.4/packages/x86_64/routing/Packages.gz
Updated list of available packages in /var/opkg-lists/openwrt_routing
Downloading https://downloads.openwrt.org/releases/22.03.4/packages/x86_64/routing/Packages.sig
Signature check passed.
Downloading https://downloads.openwrt.org/releases/22.03.4/packages/x86_64/telephony/Packages.gz
Updated list of available packages in /var/opkg-lists/openwrt_telephony
Downloading https://downloads.openwrt.org/releases/22.03.4/packages/x86_64/telephony/Packages.sig
Signature check passed.
root@OpenWrt:~#
```

```
root@OpenWrt:~# opkg install wireguard-tools luci-app-wireguard
Package wireguard-tools (1.0.20210424-3) installed in root is up to date.
Installing luci-app-wireguard (git-23.010.72712-6d712c3) to root...
Downloading https://downloads.openwrt.org/releases/22.03.4/packages/x86_64/luci/luci-app-wireguard_git-23.010.72712-6d71
2c3_all.ipk
Installing libuci-lua (2021-10-22-f04f49f0-6) to root...
Downloading https://downloads.openwrt.org/releases/22.03.4/packages/x86_64/base/libuci-lua_2021-10-22-f04f49f0-6_x86_64.
ipk
Installing luci-proto-wireguard (git-23.093.40597-10a1042) to root...
Downloading https://downloads.openwrt.org/releases/22.03.4/packages/x86_64/luci/luci-proto-wireguard_git-23.093.40597-10
a1042_all.ipk
Configuring libuci-lua.
Configuring luci-proto-wireguard.
Configuring luci-app-wireguard.
root@OpenWrt:~#
```

1. Update the repository list using opkg update
2. Install wireguard-tools
3. Install luci-app wireguard

opkg install wireguard-tools luci-app-wireguard

Setup VPN Config

```
root@OpenWrt:~# WG_IF="wg0"
root@OpenWrt:~# WG_PORT="51820"
root@OpenWrt:~# WG_ADDR="10.0.0.1/24"
root@OpenWrt:~#
```

1. Create interface named wg0
2. Set port to 51820
3. Set IP range to 10.0.0.1/24

Key Generation

```
root@OpenWrt:~# umask u=rw,g=,o=
root@OpenWrt:~# wg genkey | tee wgserver.key | wg pubkey > wgserver.pub
root@OpenWrt:~# wg genpsk > wg.psk
root@OpenWrt:~# WG_KEY="$(cat wgserver.key)"
root@OpenWrt:~# WG_PSK="$(cat wg.psk)"
root@OpenWrt:~# WG_PUB="$(cat wgserver.pub)"
root@OpenWrt:~#
```

1. Set the proper permissions for the keys. This will only allow the root user to access the keys
2. Using the wg genkey command generate the server private key, and public key
3. A pre-shared key is also generated using genpsk
4. Set the WG_KEY, WG_PSK, & WG_PUB values to the key values.

Firewall Config

```
root@OpenWrt:~# uci rename firewall.@zone[0]="lan"
root@OpenWrt:~# uci rename firewall.@zone[1]="wan"
root@OpenWrt:~# uci rename firewall.@forwarding[0]="lan_wan"
root@OpenWrt:~# uci del_list firewall.lan.network="${WG_IF}"
root@OpenWrt:~# uci add_list firewall.lan.network="${WG_IF}"
root@OpenWrt:~# uci -q delete firewall.wg
root@OpenWrt:~# uci set firewall.wg="rule"
root@OpenWrt:~# uci set firewall.wg.name="Allow-WireGuard"
root@OpenWrt:~# uci set firewall.wg.src="wan"
root@OpenWrt:~# uci set firewall.wg.dest_port="${WG_PORT}"
root@OpenWrt:~# uci set firewall.wg.proto="udp"
root@OpenWrt:~# uci set firewall.wg.target="ACCEPT"
root@OpenWrt:~# uci commit firewall
root@OpenWrt:~# /etc/init.d/firewall restart
root@OpenWrt:~#
```

1. Set zone 0 to “lan”
2. Set zone 1 to “wan”
3. Set forwarding to lan_wan
4. Delete any existing rules for WG_IF
5. Recreate rule for WG_IF
6. Delete firewall wg
7. Create firewall rule for wg
8. Set the rule name to “Allow-Wireguard”
9. Set Source WAN
10. Set destination port WG_PORT (51820)
11. Set protocol UDP
12. Set target ACCEPT
13. Commit changes
14. Restart firewall

Network Setup

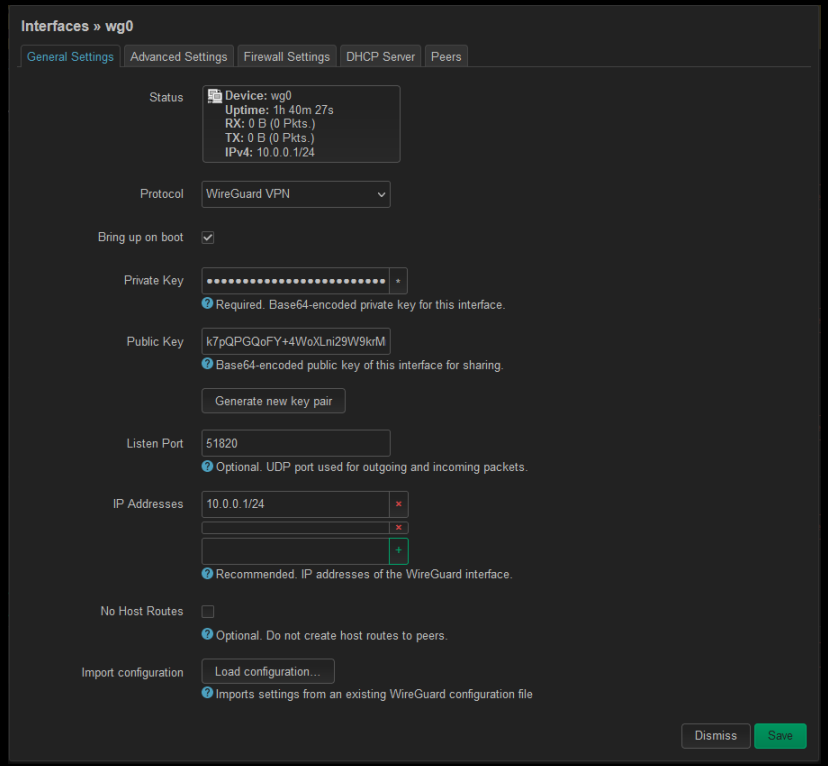
```
root@OpenWrt:~# uci -q delete network.${WG_IF}
root@OpenWrt:~# uci set network.${WG_IF}="interface"
root@OpenWrt:~# uci set network.${WG_IF}.proto="wireguard"
root@OpenWrt:~# uci set network.${WG_IF}.private_key="${WG_KEY}"
root@OpenWrt:~# uci set network.${WG_IF}.listen_port="${WG_PORT}"
root@OpenWrt:~# uci add_list network.${WG_IF}.addresses="${WG_ADDR}"
root@OpenWrt:~# uci add_list network.${WG_IF}.addresses="${WG_ADDR6}"
root@OpenWrt:~#
```

1. Delete any existing networks
2. Setup a new interface wg0
3. Set the protocol to wireguard
4. Assign the private key
5. Assign the listen port
6. Set the interface IP Address range for IPv4
7. Set the interface IP address range for IPv6
9. run uci commit to commit the changes
10. run /etc/init.d/network restart to apply the changes

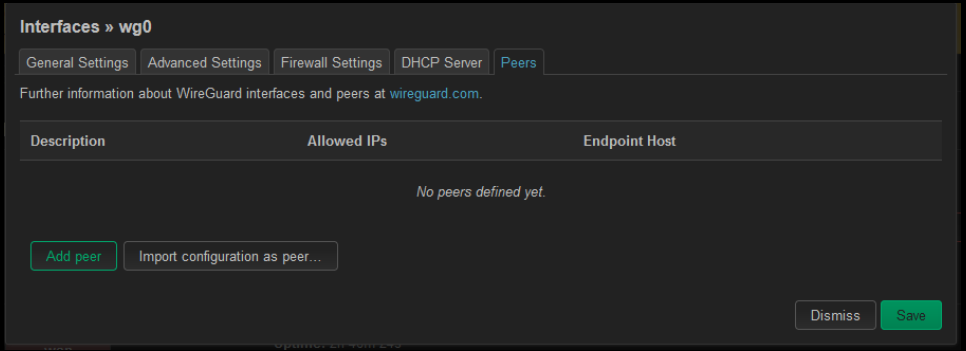
Adding a Peer

- There are 2 ways to go about this in OpenWRT
- Option 1: You can configure the peer on the peer itself and share the keys, as we did in the previous examples.
- Option 2: You can generate all the keys on the server, show a QR code and then the peer can scan that to get all the information that’s required. You must then remove the private key from the server so that the peer is the only one with knowledge of its private key.

Option 1: Manual Config

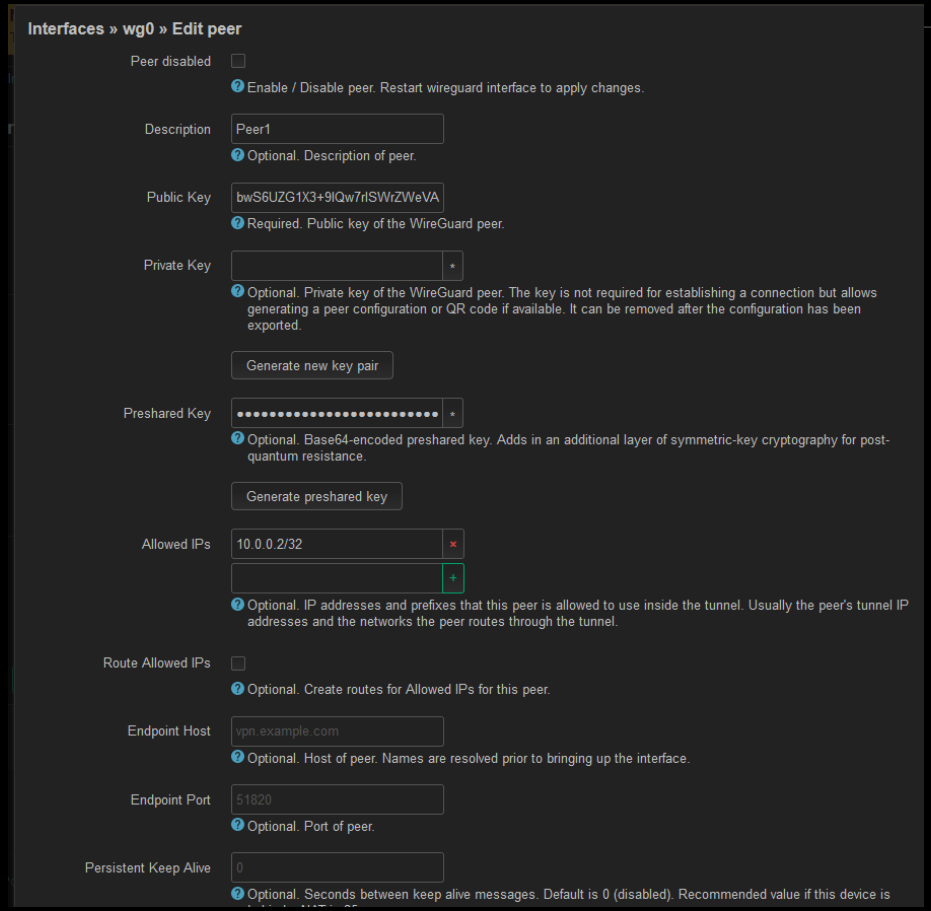


Then go to Network → Interfaces → wg0 → edit → Peers



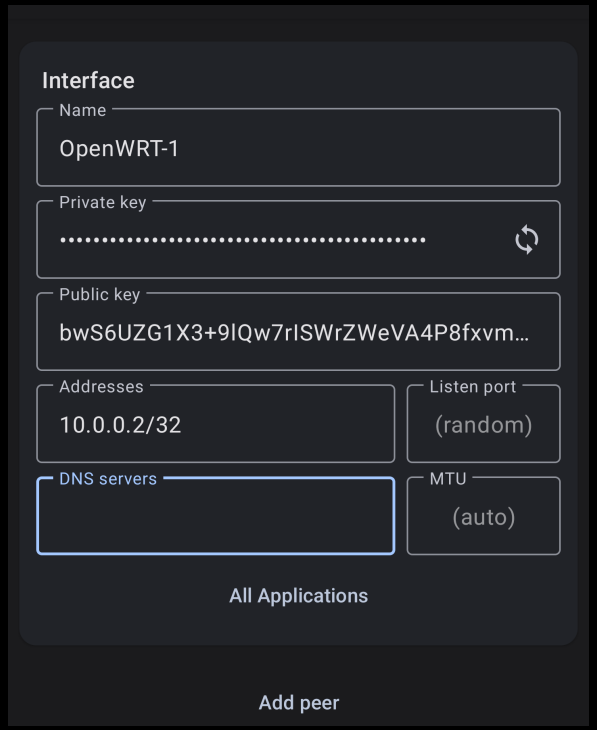
Copy the Public Key from the interface here and send it to the peer.

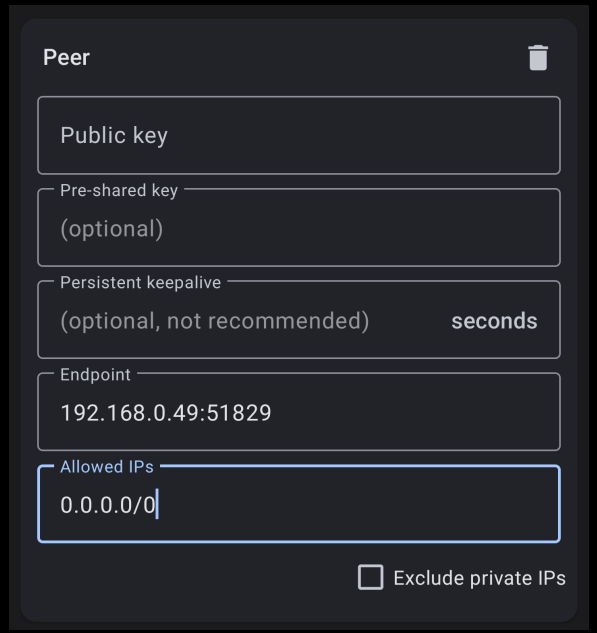
Option 1: Manual Peer Config



1. Set Peer Name
2. Copy Public Key from Public Key generated in the phone app.
3. Click Generate preshared key and send this to the Peer.
4. Allowed IPs: Set this to a unique value within the server subnet range.
5. Save

Option 1: Android

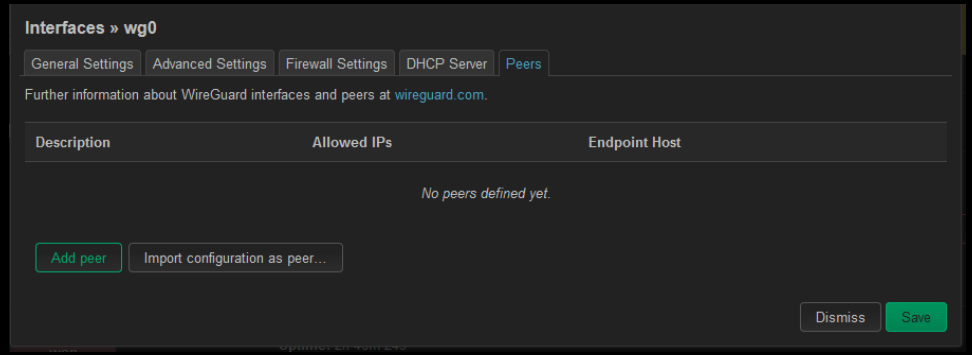




1. Name: OpenWRT
2. Click the refresh Icon to generate the public/private key pair. Send the public key to the server so it can be inserted into the Peer config. See Slide 27 #2
3. Addresses: Set this to the same value populated in the peer config on the server. See Slide 27 #4
4. Click Add Peer
5. Copy the server public key and paste it here
6. Pre-Shared Key. Copy the generated pre-shared key from the peer config on the server. Slide 27 #3
7. Endpoint: public IP of server and the port specified by the server: 51820 is the default.
8. Allowed IPs: 0.0.0.0/0 → All IP's

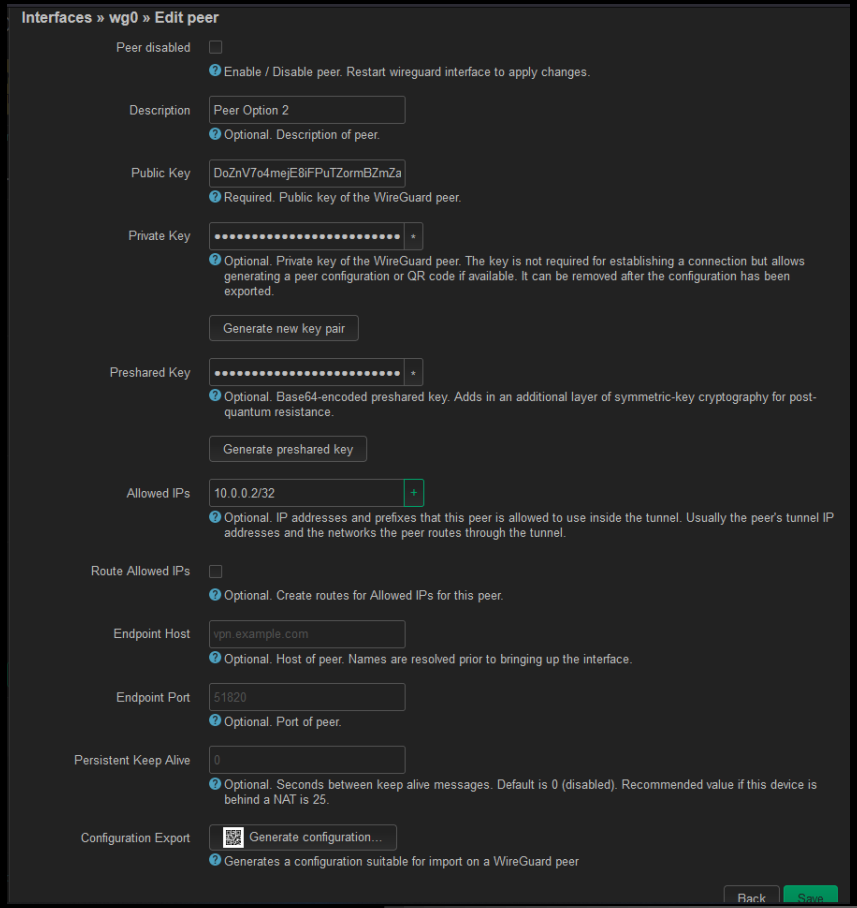
Option 2: Pre-Generated QR

- You will need to install one extra package in order to see the qr code.
- opkg install qrencode



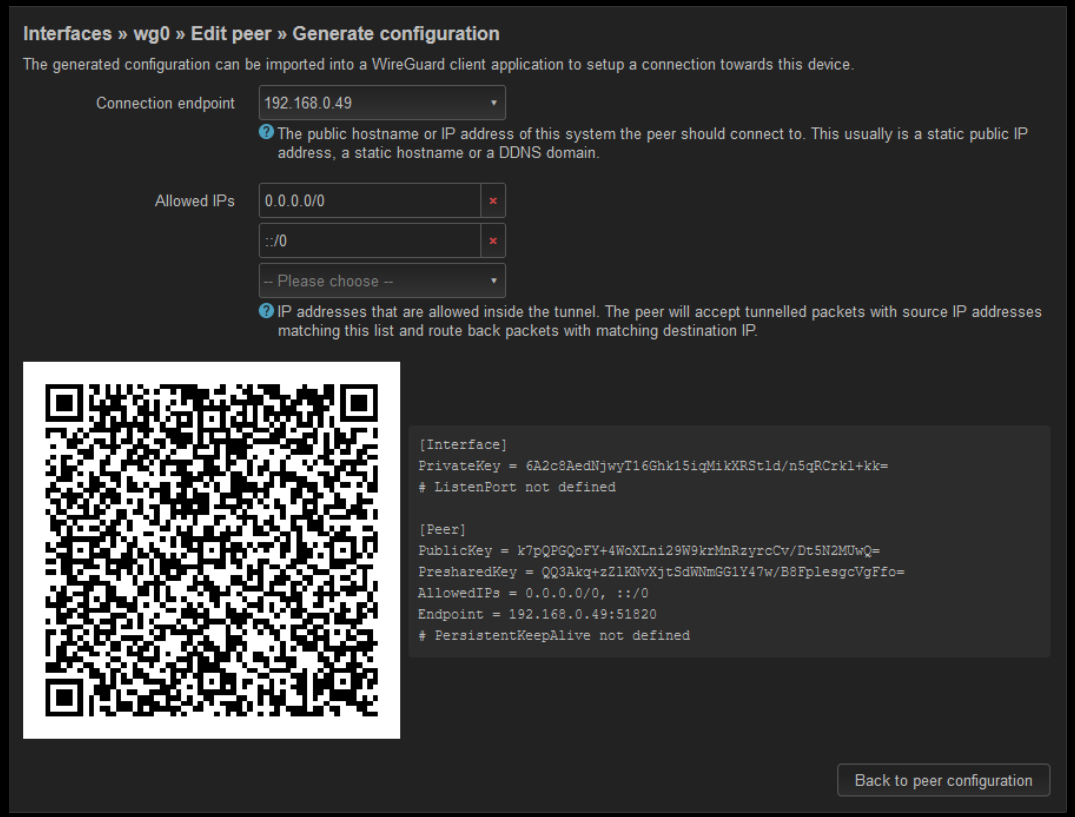
After the package is installed go to Network → Interfaces → wg0 → edit → Peers

Option 2: Pre-Generated QR



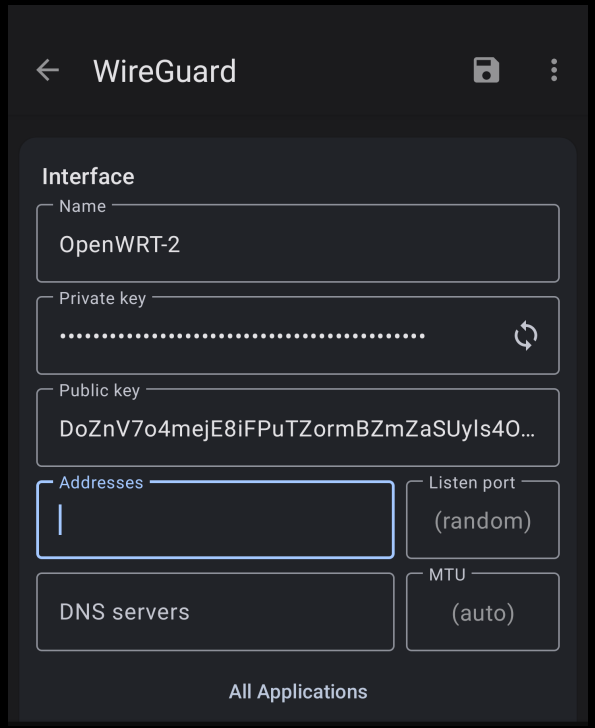
1. Set the description of the peer
2. Click “Generate new key pair” to create both the private and public key.
3. Click “Generate preshared key”
4. Allowed IPs: 10.0.0.2/32 → this should be unique per peer. Make sure you press the + to apply the IP
5. Click Generate Configuration

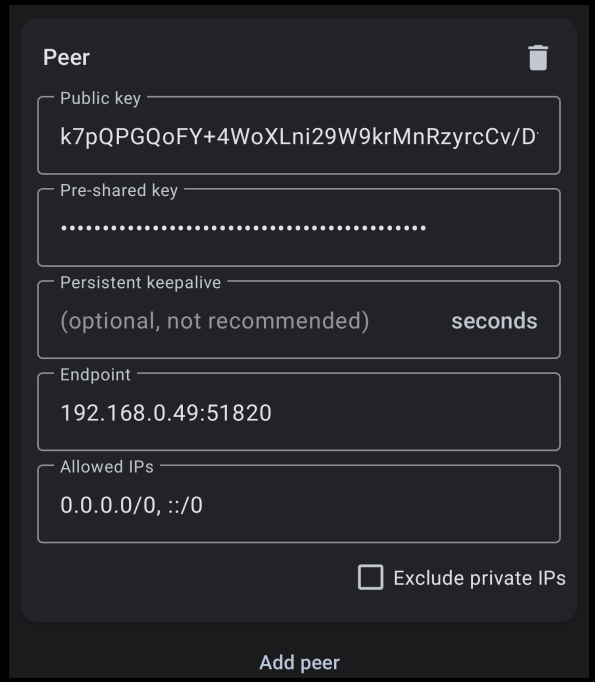
Option 2: Pre-Generated QR



1. All the values here should be pre-populated.
2. You can scan the QR code in your wireguard app to get the config.
3. After you've successfully received the configs you will need to delete the private key from the previous screen.

Android Config





1. After scanning the QR code you should see a config similar to this.
2. You may have to add the Addresses here. It should be an IP within the subnet range specified in the server. Ex: 10.0.0.2/32. (It should be unique per peer)

VPS Setup: Ubuntu 22.04LTS

Update and Upgrade

- sudo apt update && sudo apt upgrade -y
- sudo apt install wireguard

- First update and upgrade your system
- Then install wireguard

Wireguard Key Generation

- umask 077
- wg genkey | tee privatekey | wg pubkey > publickey
- ls

- This will generate base64-encoded public and private keys using the wg utility.
- You should now see 2 keys in your current directory. publickey & privatekey

Enable Forwarding

- sudo nano /etc/sysctl.conf
- net.ipv4.ip_forward=1
- sudo sysctl -p

- Open and edit the sysctl.conf
- Uncomment the line (remove the #)
- Verify net.ipv4.ip_forward=1 is printed

Wireguard Config

- sudo mv publickey /etc/wireguard
- sudo mv privatekey /etc/wireguard
- cd /etc/wireguard
- touch wg0.conf
- sudo wg setconf wg0 wg0.conf

- First move the public and private keys to the /etc/wireguard directory. You may need to become root.
- Navigate to the /etc/wireguard directory
- Create a file named wg0.conf
- Then using the wg utility set the wg config to the file wg0.conf

Wireguard Config Continued

- sudo cat /etc/wireguard/private key
- sudo nano /etc/wireguard/wg0.conf

- Copy the value printed to the screen
- Then open the wg0.conf file.

wg0.conf

```
[Interface]
PrivateKey =
ListenPort = 51820
SaveConfig = true

[Peer]
PublicKey =
AllowedIps = 10.0.0.2/32
```

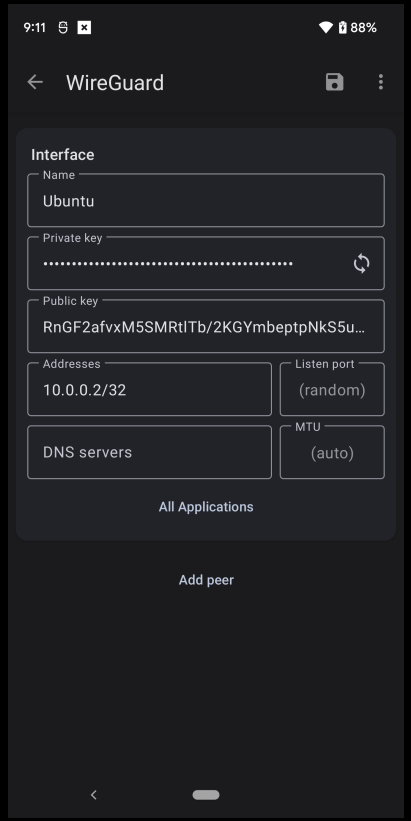
- Paste the private key from the previous screen in the PrivateKey section.
- In your phone application generate a public/private key pair and copy the public key to place here.
- For each new Peer(phone) copy the config here. Change the PublicKey and update the AllowedIps to a different IP in the subnet range. Interface should not change

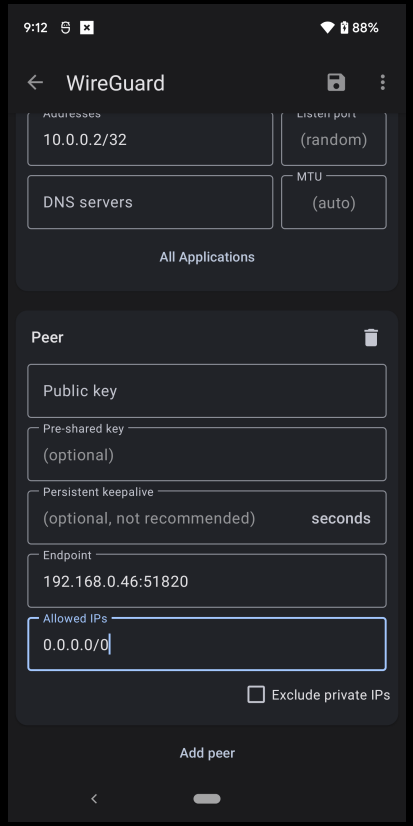
wg0.conf continued

```
PostUp = ufw route allow in on wg0 out on eth0
PostUp = iptables -t nat -I POSTROUTING -o eth0 -j MASQUERADE
PostUp = ip6tables -t nat -I POSTROUTING -o eth0 -j MASQUERADE
PreDown = ufw route delete allow in on wg0 out on eth0
PreDown = iptables -t nat -D POSTROUTING -o eth0 -j MASQUERADE
PreDown = ip6tables -t nat -D POSTROUTING -o eth0 -j MASQUERADE
```

- In the wg0.conf you will need to add these rules just above the [Peer] section. This will allow traffic through wg0 out your eth0 port. NOTE* your main interface may be named something else.
  Run “ip route list default” to find out the name of your interface and replace “eth0” with that main interface name.
- Once you are finished type ctrl + x then y and enter.

Phone Config





1. Name Ubuntu
2. Generate the private/public keys with the refresh icon.
3. Addresses: Set to unique address within the subnet generated on the server. In this case its 10.0.0.2.
4. Click Add Peer
5. The copy the Public Key from the server and paste it here.
6. Endpoint will be the public IP of the server. In this case the VM is on my local network with the IP of 192.168.0.46. :51820 is the port specified.
7. Allowed IPS: 0.0.0.0/0 Allow all network traffic

Firewall

- sudo ufw allow 51820/udp
- sudo ufw allow OpenSSH
- sudo ufw enable

- Using ufw we allow udp traffic in on port 51820 for Wireguard.
- Also we keep port 22 open for SSH connections (more configuration is required).
- Then we enable the firewall.

Enable Wireguard

- sudo wg-quick up wg0
- sudo wg-quick down wg0
- sudo systemctl enable wg-quick@wg0.service

- Bring up wireguard interface
- Bring down wireguard interface
- Enable wireguard to start at boot

OpenWRT Travel Router (RPi)

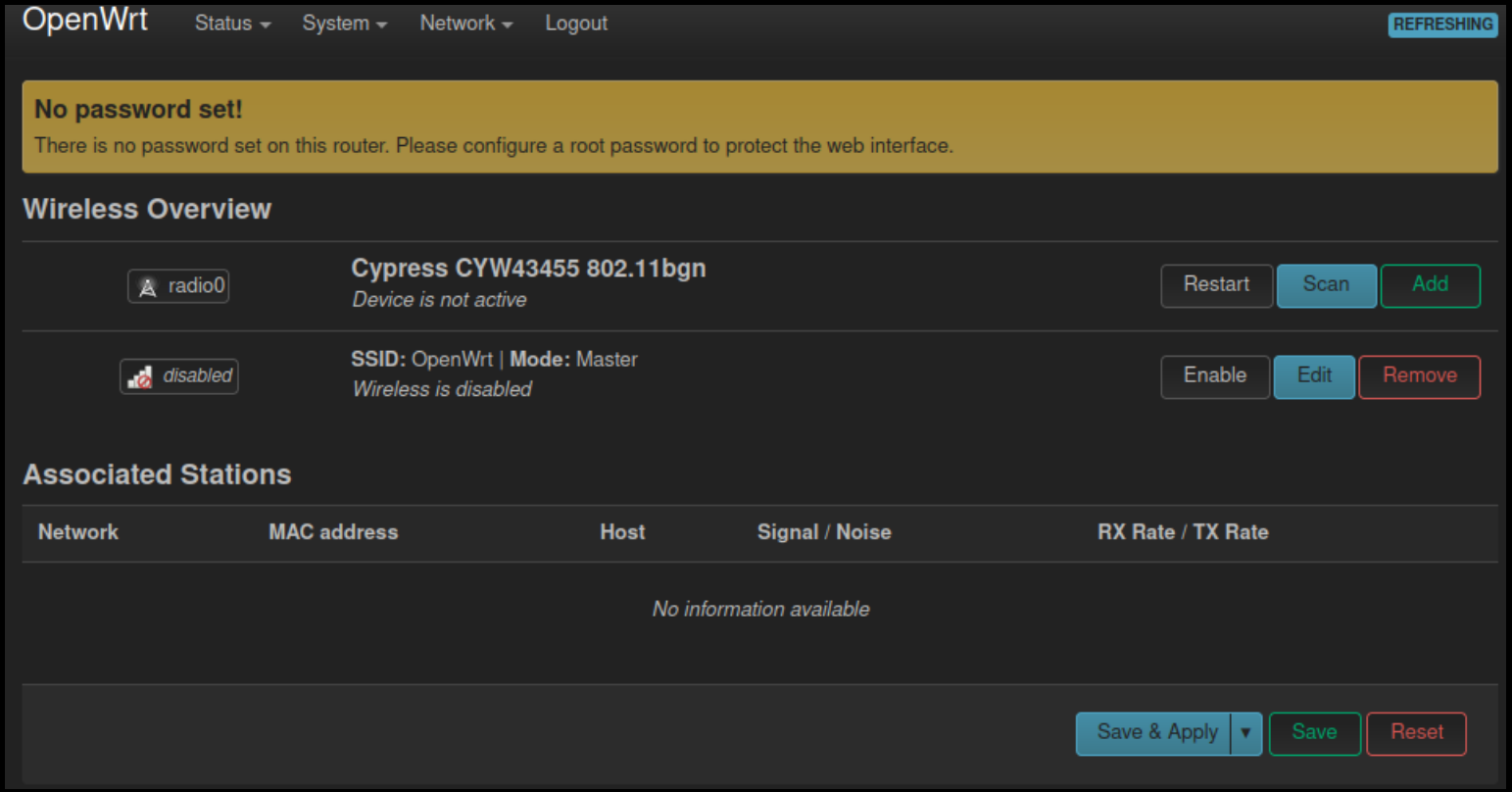
Pre-Requisite

- The travel router will be a Wireguard Peer, so you will need an existing Wireguard server setup.
- Any of the previous examples will work as the Wireguard server. This tutorial will only cover setting up the Peer, not the server, since those steps were covered in the previous slides.

Setup

- Power on your Raspberry Pi and connect a LAN cable to the ethernet port and connect your computer to that LAN cable.
- The router should give out a DHCP address. If not, you will have to set your IP to 192.168.1.2.
- OpenWRT uses 192.168.1.1 by default

Radio0



Network → Wireless

1. Radio0 is the on-board wifi of the Rpi. Enable this device and then scan for wifi.
2. This will give the Rpi network access.

Radio0 Network

1. After you've found your network select "Replace wireless configuration"
2. Set the name to wwan
3. Enter the WPA passphrase of your network and then click submit.
4. You will be brought to another config page, but just click save and do not make changes.

Update and Install Packages

- After you've connected the RPi to your local wifi you can update and install packages.
- Run opkg update to download the package repository list.
- Then install wireguard.
- opkg install wireguard-tools luci-app-wireguard

Install USB Drivers

- After you've installed Wireguard you will also need to install USB Drivers.
- Run: opkg install kmod-rt2800-lib kmod-rt2800-usb kmod-rt2x00-lib kmod-rt2x00-usb kmod-usb-core kmod-usb-uhci kmod-usb-ohci kmod-usb2 usbutils

Radio1

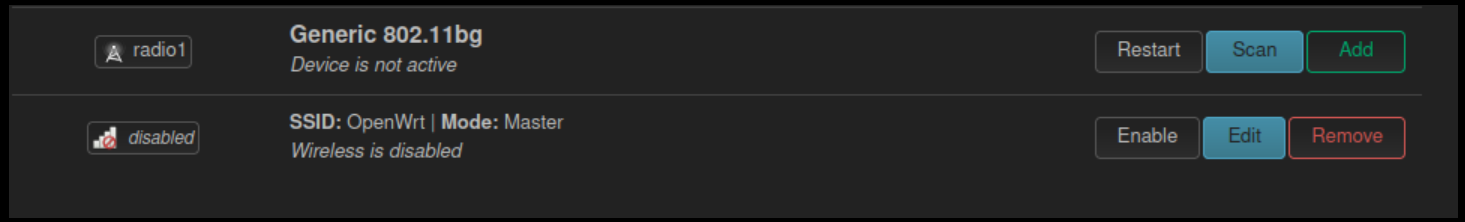
```
root@OpenWrt:~# lsusb
Bus 001 Device 003: ID 0424:ec00
Bus 001 Device 002: ID 0424:9514
Bus 001 Device 001: ID 1d6b:0002 Linux 5.10.176 dwc_otg_hcd DWC OTG Controller
root@OpenWrt:~#
```

Run lsusb to see what devices are connected to the RPi. After you have the list connect your wifi card to the USB port on your RPi.

```
root@OpenWrt:~# lsusb
Bus 001 Device 004: ID 148f:5370 Ralink 802.11 n WLAN
Bus 001 Device 003: ID 0424:ec00
Bus 001 Device 002: ID 0424:9514
Bus 001 Device 001: ID 1d6b:0002 Linux 5.10.176 dwc_otg_hcd DWC OTG Controller
root@OpenWrt:~#
```

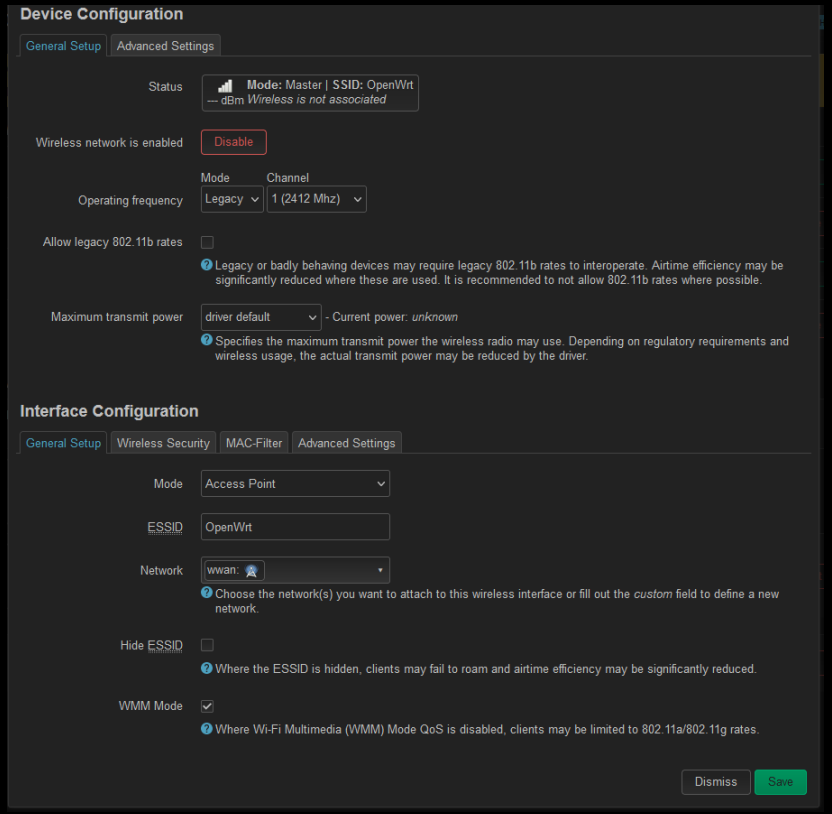
Run lsusb again to see if a new device has been discovered. In this case the Ralink 802.11 n WLAN device has been found. Now you can switch back to the GUI

Radio1 Config



Under Network→Wireless you should now see a second radio device (radio1). Click Edit to bring up the config menu.

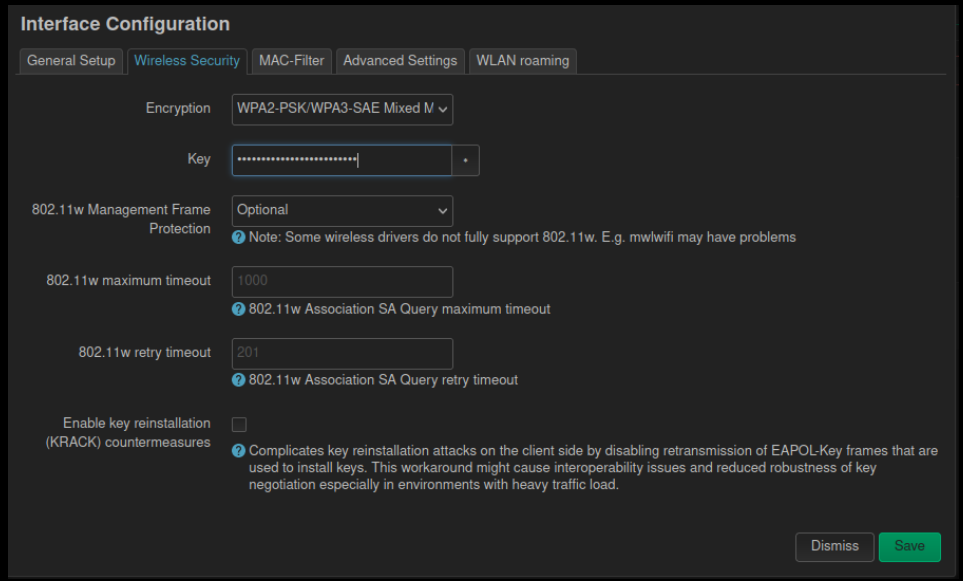
Radio1 Config: Continued



1. Mode: Access Point
2. ESSID: Anything you want (default OpenWRT)
3. Network: wwan

Select Wireless Security.

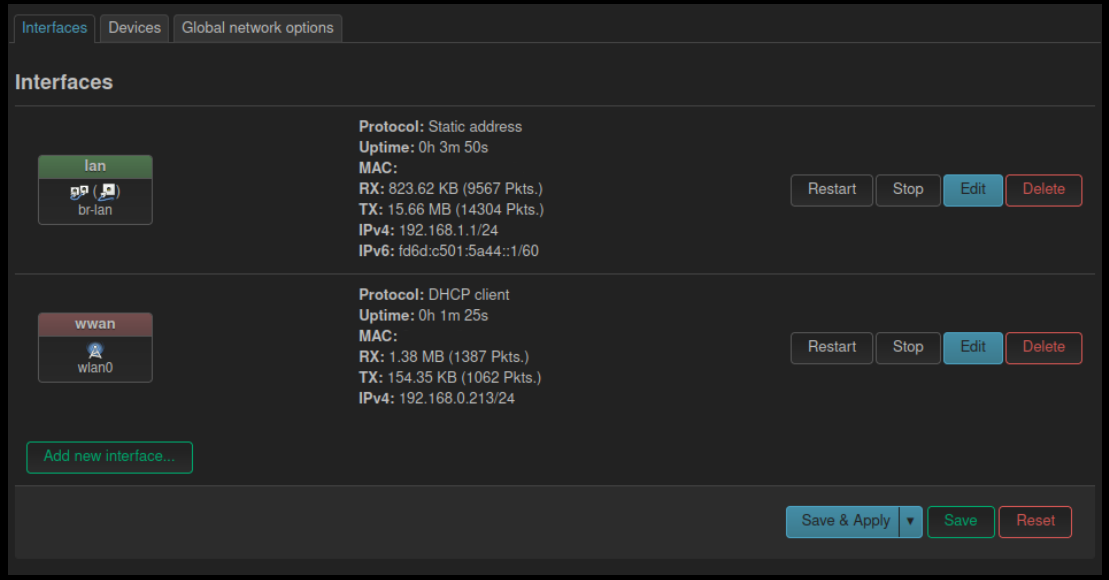
Radio1 Config: Continued



1. Encryption: WPA2-PSK/WPA3-SAW Mixed Mode
2. Key: This should be no less than 16 characters
3. Click Save

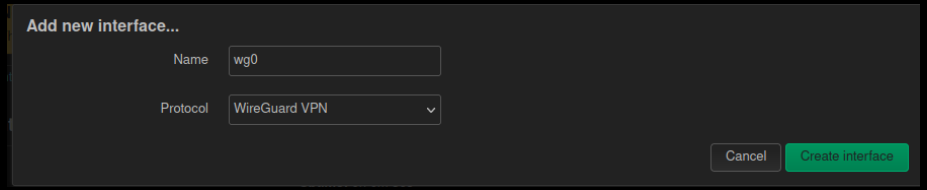
4. Then on the main Wireless Screen click save and apply

Wireguard Setup



Network → Interfaces

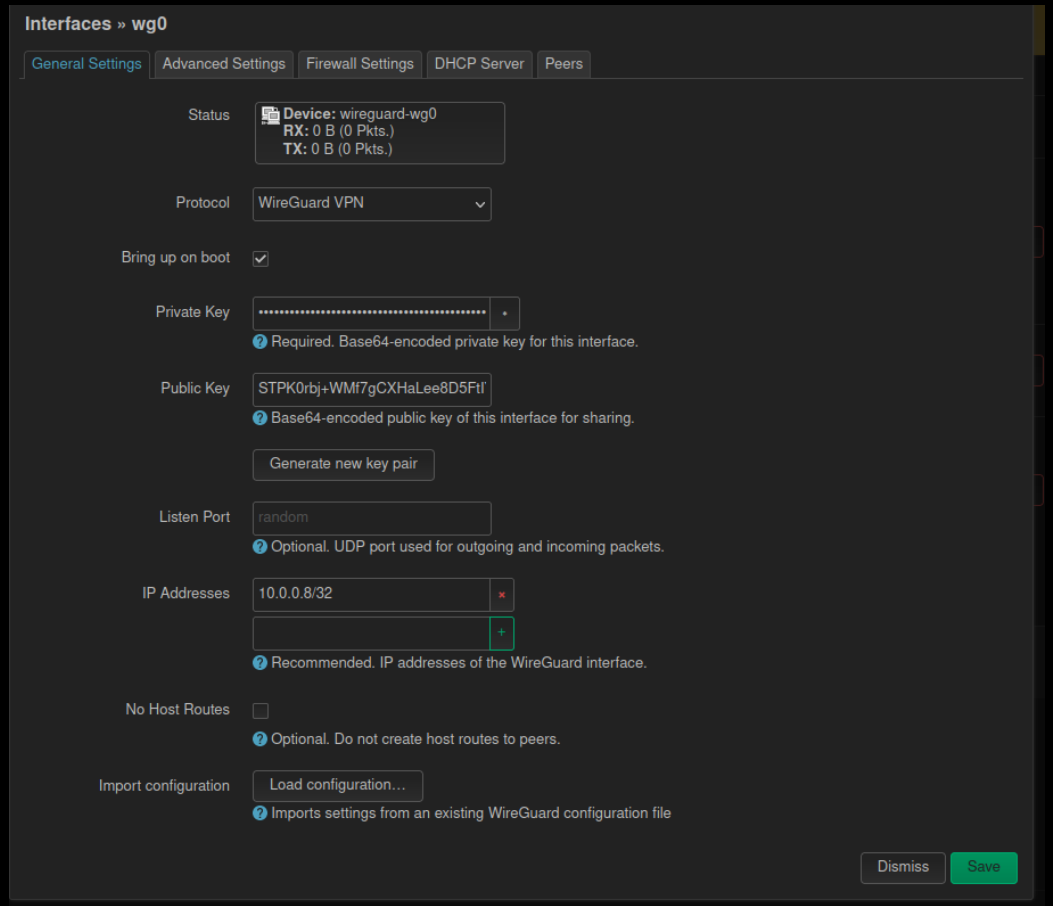
1. Select Add new interface



1. Name: wg0
2. Protocol: WireGuard VPN

NOTE: If WireGuard does not show in the protocol list you may have to reboot the device.

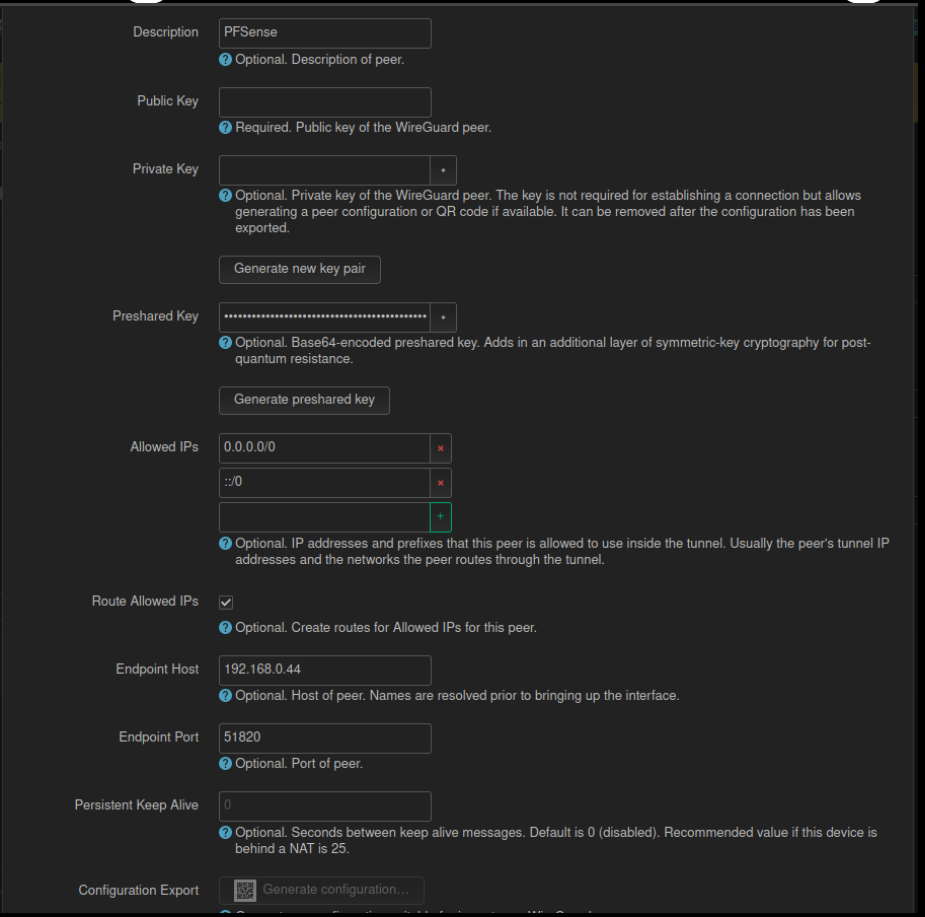
wg0 Config



1. Generate new Key Pair
2. Ip Addresses: This will be the IP of the Peer you configure on your WG server.

The Public Key here will also go on your WG server as the Peer Public Key.

wg0 Peer Config



1. Description: Anything you want
2. Public Key: Server’s public key
3. Private key: leave blank
4. Preshared Key: PSK generated from server (do not generate here)
5. Allowed IP’s: 0.0.0.0/0,::/0 (allow all IPv4 & IPv6 Traffic)
6. Route Allowed IPs: Check
7. Endpoint Host: Public IP of your server
8. Enpoint Port: Port that wireguard is running on your server.

Firewall

```
uci rename firewall.@zone[0]="lan"
uci rename firewall.@zone[1]="wan"
uci del_list firewall.wan.network=”wg0”
uci add_list firewall.wan.network="wg0"
uci commit firewall
/etc/init.d/firewall restart
```

1. Set zone 0 to lan
2. Set zone 1 to wan
3. Remove wg0 from wan network
4. Add wg0 to wan network
5. commit changes
6. restart the firewall

Verify

- After committing firewall changes you may have to reboot the RPi to get successful connection.
- Once the device comes back online you should see traffic on the Network → Interfaces page under the wg0 interface. Rx & Tx should have numbers.