



# 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

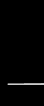
---

- 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

The screenshot shows the pfSense web interface. At the top, there's a navigation bar with tabs: System, Interfaces, Firewall, Services, VPN, Status, Diagnostics, and Help. A warning message is displayed: "WARNING: The 'admin' account password is set to the default value. Change the password in the User Manager." Below this, the breadcrumb path is "System / Package Manager / Available Packages". The "Available Packages" tab is selected. A search bar contains the text "wireguard". Below the search bar, there are two results listed in a table:

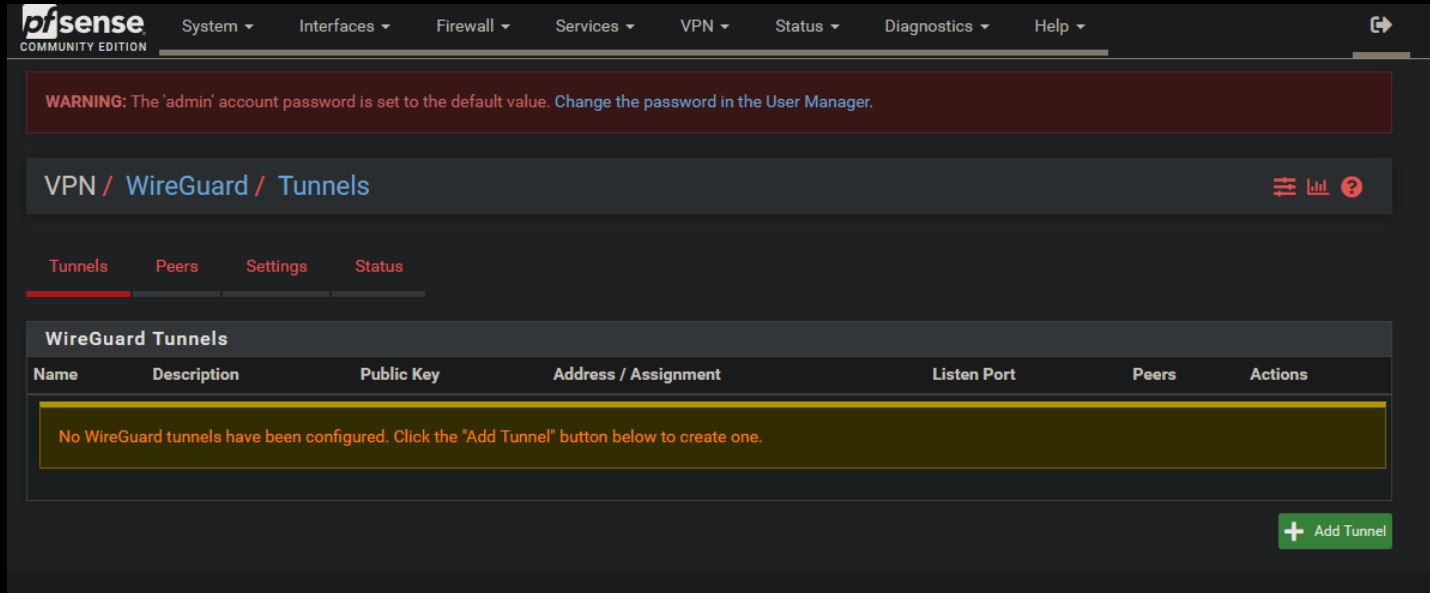
Name	Version	Description	Action
Tailscale	0.1.1_1	Tailscale is a mesh VPN alternative, based on WireGuard, that connects your computers, databases, and services together securely without any proxies.  Package Dependencies: tailscale-1.36.0	+ Install
WireGuard	0.1.6_2	WireGuard(R) is an extremely simple yet fast and modern VPN that utilizes state-of-the-art cryptography. It aims to be faster, simpler, leaner, and more useful than IPSec, while avoiding the massive headache. It intends to be considerably more performant than OpenVPN. WireGuard is designed as a general purpose VPN for running on embedded interfaces and super computers alike, fit for many different circumstances. Initially released for the Linux kernel, it is now cross-platform and widely deployable. It is currently under heavy development, but already it might be regarded as the most secure, easiest to use, and simplest VPN solution in the industry. This package is EXPERIMENTAL.  Package Dependencies: wireguard-tools-1.0.20210914_1 wireguard-kmod-0.0.20211105	+ Install

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



The screenshot shows the pfSense web interface. At the top is a navigation menu with items: System, Interfaces, Firewall, Services, VPN, Status, Diagnostics, and Help. The 'VPN' menu is expanded, showing 'WireGuard' and 'Tunnels'. Below the menu is a warning banner: 'WARNING: The 'admin' account password is set to the default value. Change the password in the User Manager.' The main content area is titled 'VPN / WireGuard / Tunnels'. It has four tabs: 'Tunnels', 'Peers', 'Settings', and 'Status'. The 'Tunnels' tab is selected. Below the tabs is a table titled 'WireGuard Tunnels'. The table has columns: Name, Description, Public Key, Address / Assignment, Listen Port, Peers, and Actions. The table is empty, and a message below it says: 'No WireGuard tunnels have been configured. Click the "Add Tunnel" button below to create one.' At the bottom right of the table is a green button with a plus icon and the text 'Add Tunnel'.

pfSense  
COMMUNITY EDITION

System ▾ Interfaces ▾ Firewall ▾ Services ▾ VPN ▾ Status ▾ Diagnostics ▾ Help ▾

WARNING: The 'admin' account password is set to the default value. [Change the password in the User Manager.](#)

VPN / WireGuard / Tunnels

Tunnels Peers Settings Status

WireGuard Tunnels

Name	Description	Public Key	Address / Assignment	Listen Port	Peers	Actions
No WireGuard tunnels have been configured. Click the "Add Tunnel" button below to create one.						

+ Add Tunnel

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

Select [+ Add Tunnel]

# Tunnel Config

## Tunnel Configuration (tun\_wg0)

Enable ☒ Enable Tunnel

Note: Tunnel must be enabled in order to be assigned to a pfSense interface.

Description

wireguard

Description for administrative reference (not parsed).

Listen Port

51820

Port used by this tunnel to communicate with peers.

Interface Keys

Private key for this tunnel. (Required)

Public key for this tunnel. (Copy)

Generate

New Keys

## Interface Configuration (tun\_wg0)

Assignment

Interface Assignments

Firewall Rules

WireGuard Interface Group

Hint

These interface addresses are only applicable for unassigned WireGuard tunnel interfaces.

Interface Addresses

10.0.0.1

/ 24

Description

IPv4 or IPv6 address assigned to the tunnel interface.

Description for administrative reference (not parsed).

Add Address

+ Add Address

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.

Once finished, click save. Then go to Peers.

Meaning I can use 10.0.0.2-10.0.0.254  
See [RCF1918](#).

# Peer Config

The WireGuard service is not running.

Tunnels Peers Settings Status

## Peer Configuration

Enable

☒ Enable Peer

Note: Uncheck this option to disable this peer without removing it from the list.

Tunnel

tun\_wg0 (wireguard) ▼

WireGuard tunnel for this peer. (Create a New Tunnel)

Description

Peer 1

Peer description for administrative reference (not parsed).

Dynamic Endpoint

☒ Dynamic

Note: Uncheck this option to assign an endpoint address and port for this peer.

Keep Alive

Keep Alive

Interval (in seconds) for Keep Alive packets sent to this peer.  
Default is empty (disabled).

Public Key

Public Key ⓘ

WireGuard public key for this peer.

Pre-shared Key

.....

Optional pre-shared key for this tunnel. (Copy)

Generate

New Pre-shared Key

## Address Configuration

Hint

Allowed IP entries here will be transformed into proper subnet start boundaries prior to validating and saving.

Allowed IPs

10.0.0.2 / 32 ▼

Description

IPv4 or IPv6 subnet or host reachable via this peer.

Description for administrative reference (not parsed).

Add Allowed IP

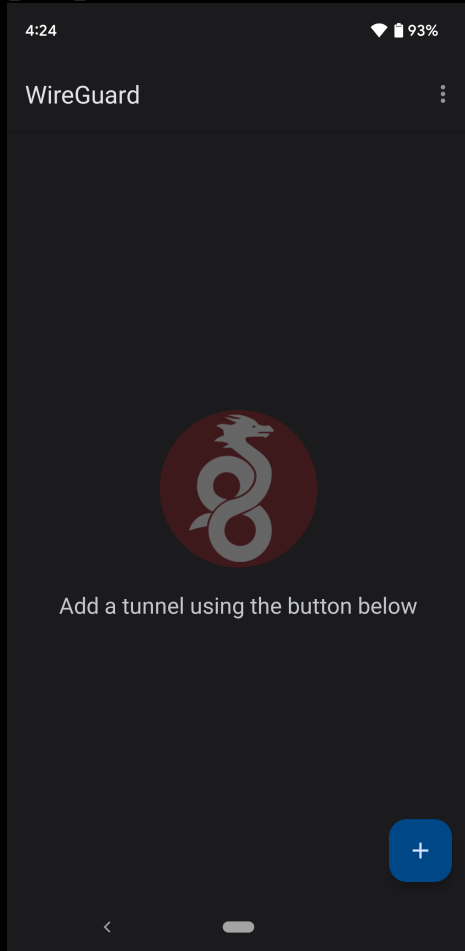
+ Add Allowed IP

Save Peer

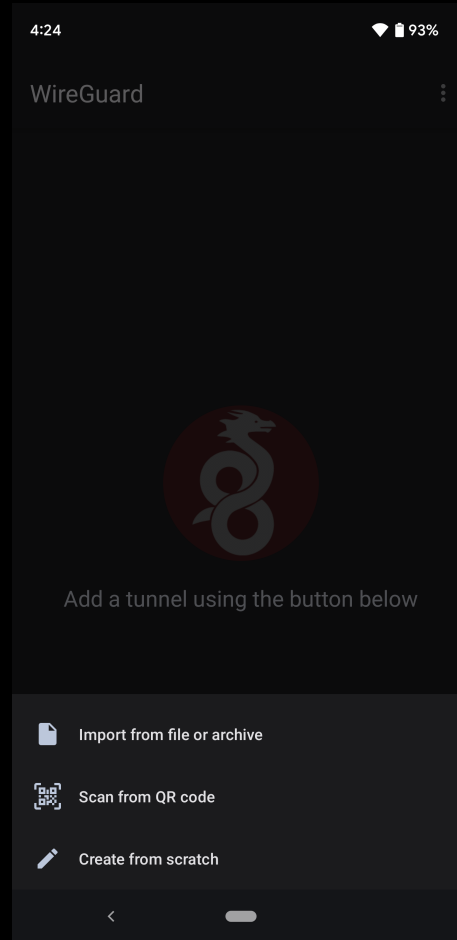
1. Enable peer.
2. Select the tunnel you created (in this case tun\_wg0).
3. Set a Description.
4. Keep the endpoint dynamic.
5. Keep Alive: leave blank.
6. Public Key\* this will be generate from your app.
7. Pre-Shared Key (optional but can provide quantum protection) Click [Generate] to create one.
8. Allowed IP's: Set this to an address that has not been used in your subnet yet.  
Ex: 10.0.0.2 (means only 10.0.0.2)



# App Config: Android



Click the +  
button



Select create from  
scratch

# App Config: Android

9:17 93%

← WireGuard

Interface

Name  
PfSense

Private key  
..... ↻

Public key  
MWrEDgJraUBP77BNvSi8ZqBDwds4LcCui7...

Addresses  
10.0.0.2/32

Listen port  
(random)

DNS servers  
10.0.0.1

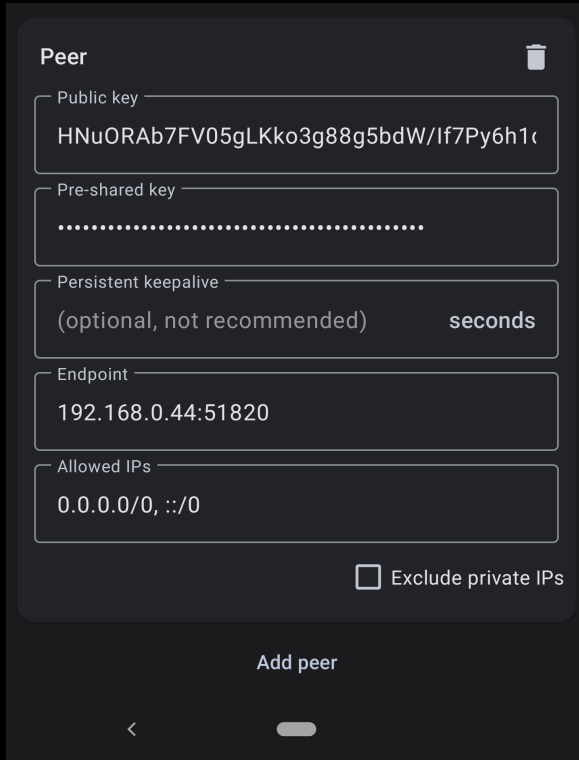
MTU  
(auto)

All Applications

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



The screenshot shows the 'Add peer' screen in the WireGuard Android app. The interface is dark-themed. At the top, there's a title 'Peer' with a trash icon. Below it are five input fields: 'Public key' containing 'HNuORAb7FV05gLKko3g88g5bdW/If7Py6h1...', 'Pre-shared key' with a series of dots, 'Persistent keepalive' with '(optional, not recommended)' and a 'seconds' label, 'Endpoint' containing '192.168.0.44:51820', and 'Allowed IPs' containing '0.0.0.0/0, ::/0'. At the bottom of the form is a checkbox labeled 'Exclude private IPs' which is unchecked. Below the form is a button labeled 'Add peer'. At the very bottom are navigation icons: a back arrow and a home indicator bar.

Peer

Public key  
HNuORAb7FV05gLKko3g88g5bdW/If7Py6h1...

Pre-shared key  
.....

Persistent keepalive  
(optional, not recommended) seconds

Endpoint  
192.168.0.44:51820

Allowed IPs  
0.0.0.0/0, ::/0

☐ Exclude private IPs

Add peer

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. Endpoint: 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

Firewall / Rules / WAN

Floating WireGuard WAN

Rules (Drag to Change Order)

<input type="checkbox"/>	States	Protocol	Source	Port	Destination	Port	Gateway	Queue	Schedule	Description	Actions
<input checked="" type="checkbox"/>	0 / 1.21 MiB	*	*	*	WAN Address	443 80	*	*		Anti-Lockout Rule	
<input checked="" type="checkbox"/>	0 / 384 B	*	Reserved Not assigned by IANA	*	*	*	*	*		Block bogon networks	

No rules are currently defined for this interface  
All incoming connections on this interface will be blocked until pass rules are added. Click the button to add a new rule.

Add Add Delete Save Separator

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 V”

Select Firewall → Rules from the main menu.

# Firewall Config

Edit Firewall Rule

Action

Pass

Choose what to do with packets that match the criteria specified below.  
Hint: the difference between block and reject is that with reject, a packet (TCP RST or ICMP port unreachable for UDP) is returned to the sender, whereas with block the packet is dropped silently. In either case, the original packet is discarded.

Disabled

☐ Disable this rule

Set this option to disable this rule without removing it from the list.

Interface

WAN

Choose the interface from which packets must come to match this rule.

Address Family

IPv4

Select the Internet Protocol version this rule applies to.

Protocol

UDP

Choose which IP protocol this rule should match.

Source

Source

☐ Invert match

any

Source Address

/

Display Advanced

The Source Port Range for a connection is typically random and almost never equal to the destination port. In most cases this setting must remain at its default value, any.

Destination

Destination

☐ Invert match

WAN address

Destination Address

/

Destination Port Range

(other)

51820

(other)

51820

From

Custom

To

Custom

Specify the destination port or port range for this rule. The "To" field may be left empty if only filtering a single port.

Extra Options

Log

☒ Log packets that are handled by this rule

Hint: the firewall has limited local log space. Don't turn on logging for everything. If doing a lot of logging, consider using a remote syslog server (see the Status: System Logs: Settings page).

Description

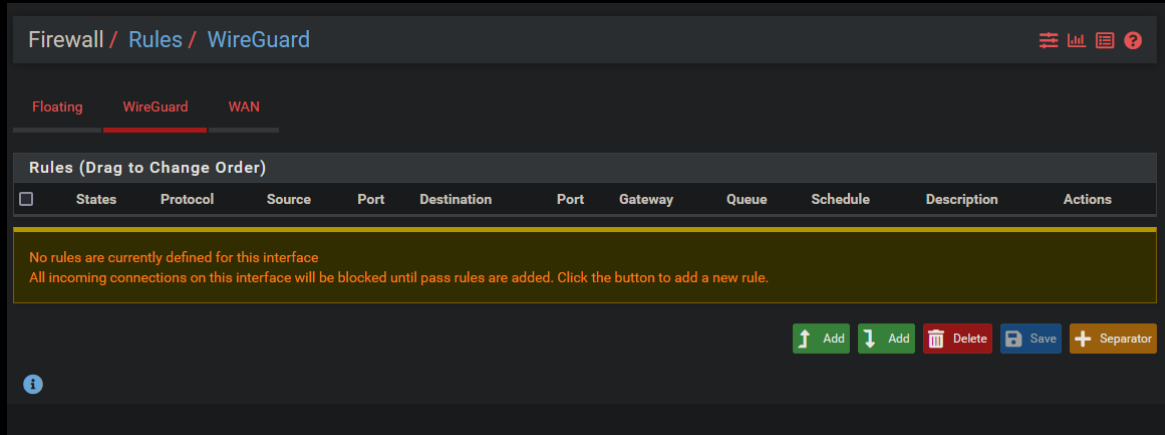
Wireguard - Allow WAN 51820

A description may be entered here for administrative reference. A maximum of 50 characters will be used in the ruleset and displayed in the firewall.

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 apply 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

Edit Firewall Rule

Action

Pass

Choose what to do with packets that match the criteria specified below.  
Hint: the difference between block and reject is that with reject, a packet (TCP RST or ICMP port unreachable for UDP) is returned to the sender, whereas with block the packet is dropped silently. In either case, the original packet is discarded.

Disabled

☐ Disable this rule

Set this option to disable this rule without removing it from the list.

Interface

WireGuard

Choose the interface from which packets must come to match this rule.

Address Family

IPv4

Select the Internet Protocol version this rule applies to.

Protocol

Any

Choose which IP protocol this rule should match.

Source

Source

☐ Invert match

any

Source Address

/

Destination

Destination

☐ Invert match

any

Destination Address

/

Extra Options

Log

☐ Log packets that are handled by this rule

Hint: the firewall has limited local log space. Don't turn on logging for everything. If doing a lot of logging, consider using a remote syslog server (see the [Status: System Logs: Settings](#) page).

Description

Allow all

A description may be entered here for administrative reference. A maximum of 52 characters will be used in the ruleset and displayed in the firewall log.

Advanced Options

Display Advanced

Save

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

Port Forward 1:1 Outbound NAT

## Outbound NAT Mode

Mode



Automatic outbound NAT rule generation.  
(IPsec passthrough included)



Hybrid Outbound NAT rule generation.  
(Automatic Outbound NAT + rules below)



Manual Outbound NAT rule generation.  
(AON - Advanced Outbound NAT)



Disable Outbound NAT rule generation.  
(No Outbound NAT rules)

 Save

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.



pfSense

COMMUNITY EDITION

System

Interfaces

Firewall

Services

VPN

Status

Diagnostics

Help

WARNING: The 'admin' account password is set to the default value. Change the password in the User Manager.

Firewall / NAT / Outbound

Port Forward

1:1

Outbound

NPt

Outbound NAT Mode

Mode

☒

☐

☐

☐

Automatic outbound NAT rule generation.  
(IPsec passthrough included)

Hybrid Outbound NAT rule generation.  
(Automatic Outbound NAT + rules below)

Manual Outbound NAT rule generation.  
(AON - Advanced Outbound NAT)

Disable Outbound NAT rule generation.  
(No Outbound NAT rules)

Save

Mappings

☐

Interface

Source

Source Port

Destination

Destination Port

NAT Address

NAT Port

Static Port

Description

Actions

↑ Add

↓ Add

Delete

Save

Automatic Rules:

Interface

Source

Source Port

Destination

Destination Port

NAT Address

NAT Port

Static Port

Description

✓

WAN

127.0.0.0/8 ::1/128

\*

\*

500

WAN address

\*

✓

Auto created rule for ISAKMP

✓

WAN

127.0.0.0/8 ::1/128

\*

\*

\*

WAN address

\*

✕

Auto created rule

i

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

17

# Firewall NAT Outbound

**Edit Advanced Outbound NAT Entry**

**Disabled** ☐ Disable this rule

**Do not NAT** ☐ Enabling this option will disable NAT for traffic matching this rule and stop processing Outbound NAT rules  
In most cases this option is not required.

**Interface** WAN  
The interface on which traffic is matched as it exits the firewall. In most cases this is "WAN" or another externally-connected interface.

**Address Family** IPv4  
Select the Internet Protocol version this rule applies to.

**Protocol** any  
Choose which protocol this rule should match. In most cases "any" is specified.

**Source** Network 10.0.0.1 / 24   
Type Source network for the outbound NAT mapping. Port or Range

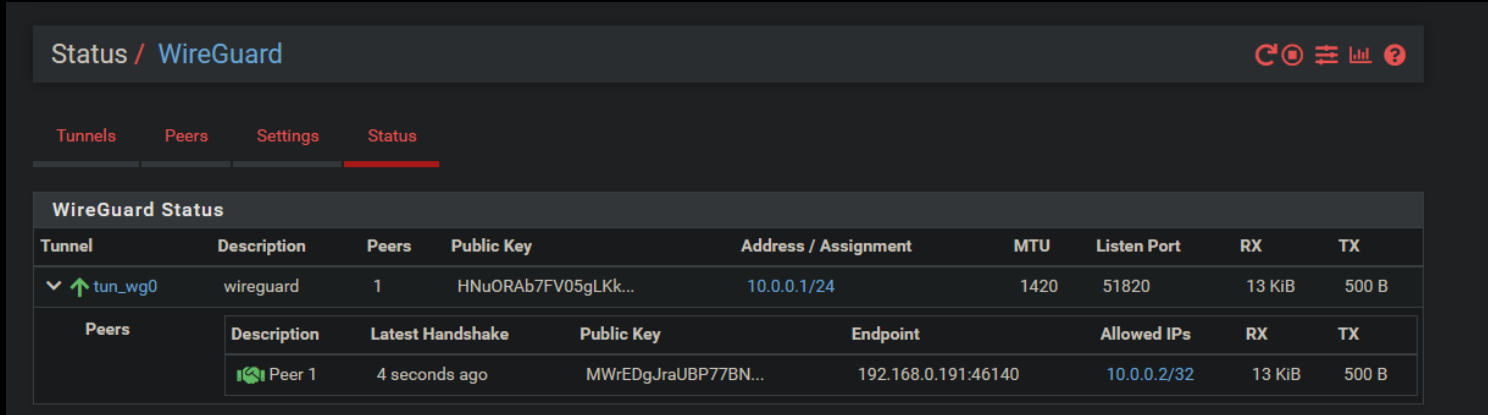
**Destination** Any  / 24   
Type Destination network for the outbound NAT mapping. Port or Range

☐ Not  
Invert the sense of the destination match.

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



The screenshot shows the 'Status / WireGuard' page of a VPN management interface. At the top, there are tabs for 'Tunnels', 'Peers', 'Settings', and 'Status', with 'Status' being the active tab. Below the tabs is a 'WireGuard Status' section containing two tables. The first table lists the 'tun\_wg0' tunnel with details like description, peers, public key, address, MTU, listen port, and RX/TX statistics. The second table, titled 'Peers', shows a single peer named 'Peer 1' with its latest handshake time, public key, endpoint, allowed IPs, and RX/TX statistics.

Status / WireGuard

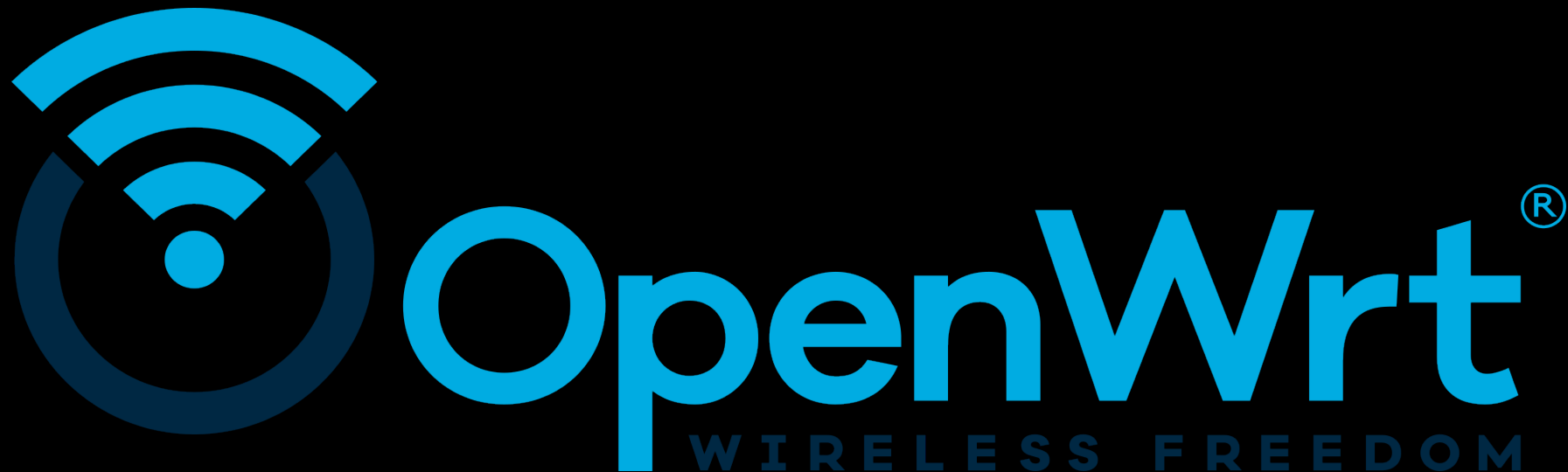
Tunnels Peers Settings Status

WireGuard Status

Tunnel	Description	Peers	Public Key	Address / Assignment	MTU	Listen Port	RX	TX
▼ ↑ tun_wg0	wireguard	1	HNuORAb7FV05gLkK...	10.0.0.1/24	1420	51820	13 KiB	500 B

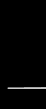
Peers	Description	Latest Handshake	Public Key	Endpoint	Allowed IPs	RX	TX
	Peer 1	4 seconds ago	MWrEDgJraUBP77BN...	192.168.0.191:46140	10.0.0.2/32	13 KiB	500 B

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-23.018.72712-6d712c3) to root...
Downloading https://downloads.openwrt.org/releases/22.03.4/packages/x86_64/luci/luci-app-wireguard_git-23.018.72712-6d712c3_all.ipk
Installing libuci-lua (2021-10-22-f84f49f0-6) to root...
Downloading https://downloads.openwrt.org/releases/22.03.4/packages/x86_64/base/libuci-lua_2021-10-22-f84f49f0-6_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-23.093.40597-10a1042_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

Interfaces » wg0

General Settings Advanced Settings Firewall Settings DHCP Server Peers

Status Device: wg0  
Uptime: 1h 40m 27s  
RX: 0 B (0 Pkts.)  
TX: 0 B (0 Pkts.)  
IPv4: 10.0.0.1/24

Protocol WireGuard VPN

Bring up on boot ☒

Private Key    
 Required. Base64-encoded private key for this interface.

Public Key k7pQPGQqFY+4WoxLni29W9krM   
 Base64-encoded public key of this interface for sharing.

Listen Port 51820   
 Optional. UDP port used for outgoing and incoming packets.

IP Addresses 10.0.0.1/24   
 Recommended. IP addresses of the WireGuard interface.

No Host Routes ☐  
 Optional. Do not create host routes to peers.

Import configuration   
 Imports settings from an existing WireGuard configuration file

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

Interfaces » wg0

General Settings Advanced Settings Firewall Settings DHCP Server Peers

Further information about WireGuard interfaces and peers at [wireguard.com](https://wireguard.com).

Description	Allowed IPs	Endpoint Host
No peers defined yet.		

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

# Option 1: Manual Peer Config

Interfaces » wg0 » Edit peer

Peer disabled ☐

Enable / Disable peer. Restart wireguard interface to apply changes.

Description

Optional. Description of peer.

Public Key

Required. Public key of the WireGuard peer.

Private Key

Optional. Private key of the WireGuard peer. The key is not required for establishing a connection but allows generating a peer configuration or QR code if available. It can be removed after the configuration has been exported.

Preshared Key

Optional. Base64-encoded preshared key. Adds in an additional layer of symmetric-key cryptography for post-quantum resistance.

Allowed IPs

Optional. IP addresses and prefixes that this peer is allowed to use inside the tunnel. Usually the peer's tunnel IP addresses and the networks the peer routes through the tunnel.

Route Allowed IPs ☐

Optional. Create routes for Allowed IPs for this peer.

Endpoint Host

Optional. Host of peer. Names are resolved prior to bringing up the interface.

Endpoint Port

Optional. Port of peer.

Persistent Keep Alive

Optional. Seconds between keep alive messages. Default is 0 (disabled). Recommended value if this device is

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

### Interface

Name

OpenWRT-1

Private key

..... 

Public key

bwS6UZG1X3+9lQw7rISWrZWeVA4P8fxvm...

Addresses

10.0.0.2/32

Listen port

(random)

DNS servers

MTU

(auto)

All Applications

Add peer

### Peer

Public key

Pre-shared key

(optional)

Persistent keepalive

(optional, not recommended) seconds

Endpoint

192.168.0.49:51829

Allowed IPs

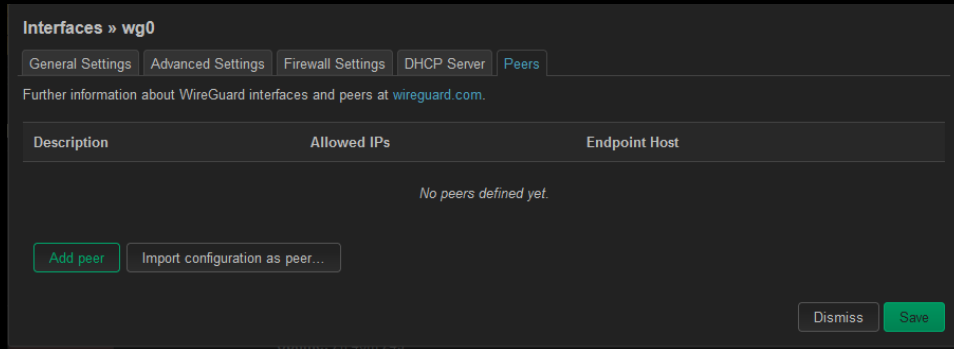
0.0.0.0/0

☐ Exclude private IPs

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

Interfaces » wg0 » Edit peer

Peer disabled ☐  
Enable / Disable peer. Restart wireguard interface to apply changes.

Description   
Optional. Description of peer.

Public Key   
Required. Public key of the WireGuard peer.

Private Key   
Optional. Private key of the WireGuard peer. The key is not required for establishing a connection but allows generating a peer configuration or QR code if available. It can be removed after the configuration has been exported.

Preshared Key   
Optional. Base64-encoded preshared key. Adds in an additional layer of symmetric-key cryptography for post-quantum resistance.

Allowed IPs    
Optional. IP addresses and prefixes that this peer is allowed to use inside the tunnel. Usually the peer's tunnel IP addresses and the networks the peer routes through the tunnel.

Route Allowed IPs ☐  
Optional. Create routes for Allowed IPs for this peer.

Endpoint Host   
Optional. Host of peer. Names are resolved prior to bringing up the interface.

Endpoint Port   
Optional. Port of peer.

Persistent Keep Alive   
Optional. Seconds between keep alive messages. Default is 0 (disabled). Recommended value if this device is behind a NAT is 25.

Configuration Export   
Generates a configuration suitable for import on a WireGuard peer


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


## Interfaces » wg0 » Edit peer » Generate configuration

The generated configuration can be imported into a WireGuard client application to setup a connection towards this device.

Connection endpoint

 The public hostname or IP address of this system the peer should connect to. This usually is a static public IP address, a static hostname or a DDNS domain.

Allowed IPs

 IP addresses that are allowed inside the tunnel. The peer will accept tunnelled packets with source IP addresses matching this list and route back packets with matching destination IP.



```
[Interface]
PrivateKey = 6A2c8AedNjwyTl6Ghk15iqMikXRStld/n5qRCrkl+kk=
# ListenPort not defined

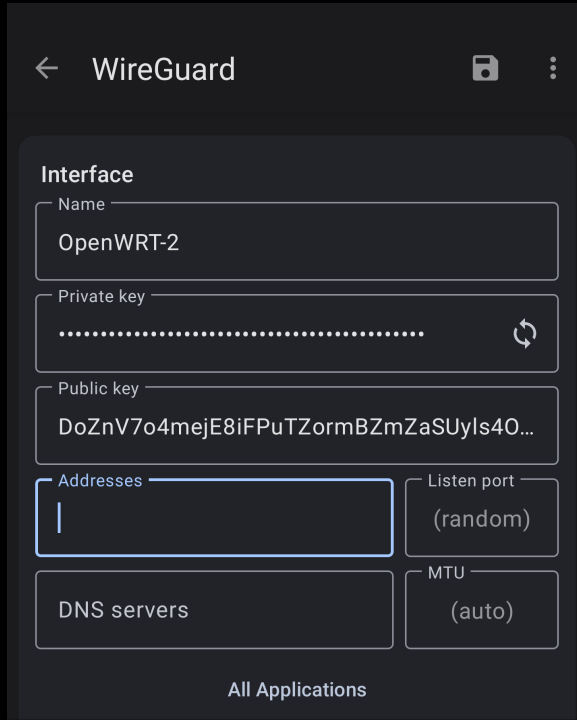
[Peer]
PublicKey = k7pQFGQoFY+4WoXLni29W9krMnRzyrcCv/Dt5N2MUwQ=
PresharedKey = QQ3Akq+z2LkNvXjtSdWmGG1Y47w/B8FplesgcVgFfo=
AllowedIPs = 0.0.0.0/0, ::0
Endpoint = 192.168.0.49:51820
# PersistentKeepAlive not defined
```

[Back to peer configuration](#)

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



WireGuard

**Interface**

Name  
OpenWRT-2

Private key  
.....

Public key  
DoZnV7o4mejE8iFPuTZormBZmZaSuyIs40...

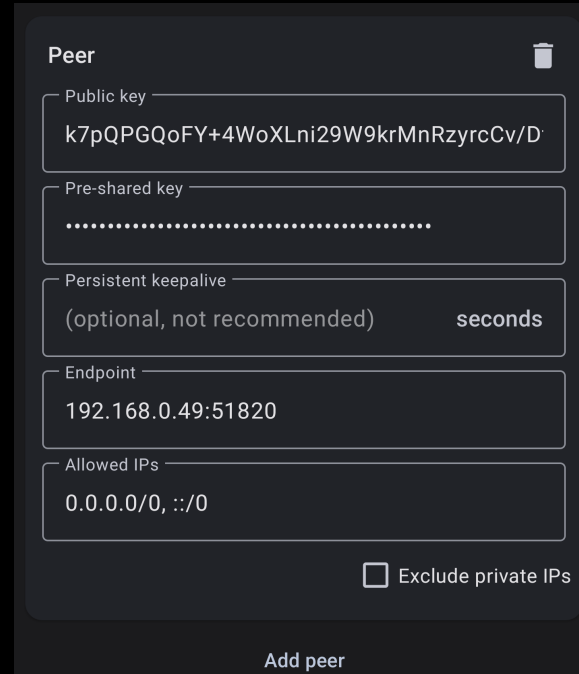
Addresses  
|

Listen port  
(random)

DNS servers

MTU  
(auto)

All Applications



Peer

Public key  
k7pQPGQoFY+4WoXLni29W9krMnRzyrcCv/D

Pre-shared key  
.....

Persistent keepalive  
(optional, not recommended) seconds

Endpoint  
192.168.0.49:51820

Allowed IPs  
0.0.0.0/0, ::/0

☐ Exclude private IPs

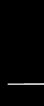
Add peer

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

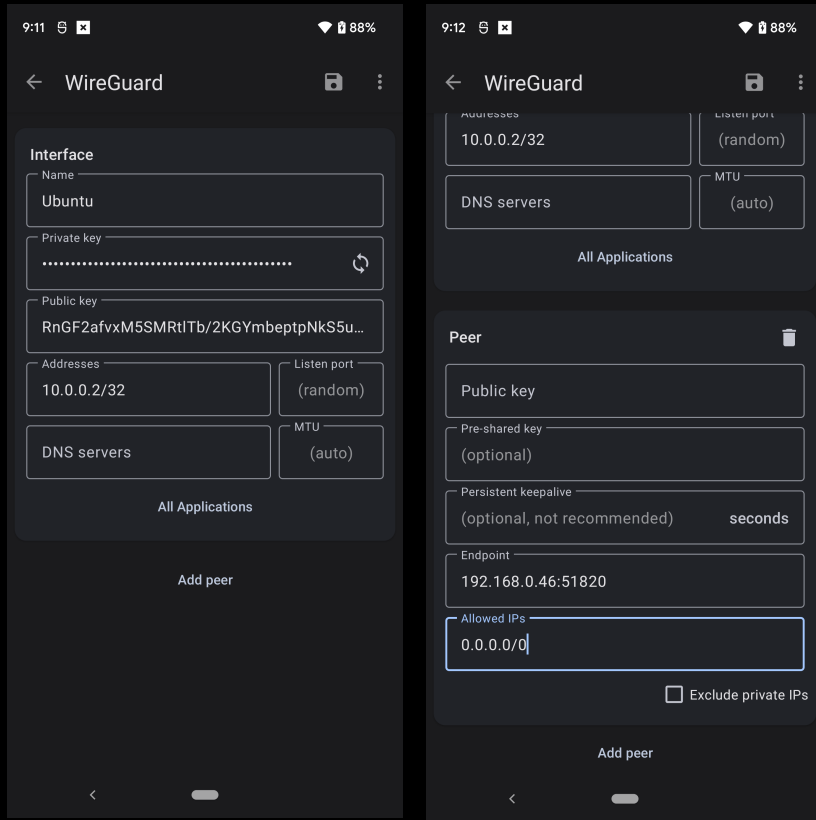


# wg0.conf continued

```
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

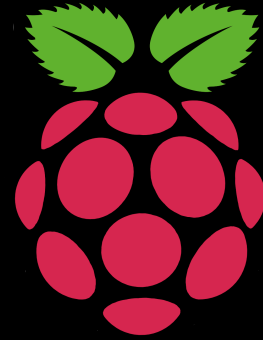
# Firewall

---

- `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.

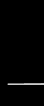
# Enable Wireguard

- `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

OpenWrt

Status ▾System ▾Network ▾Logout

REFRESHING

No password set!

There is no password set on this router. Please configure a root password to protect the web interface.

Wireless Overview

radio0

Cypress CYW43455 802.11bgn

Device is not active

RestartScanAdd

disabled

SSID: OpenWrt | Mode: Master

Wireless is disabled

EnableEditRemove

Associated Stations

Network	MAC address	Host	Signal / Noise	RX Rate / TX Rate
No information available				

Save & Apply ▾SaveReset

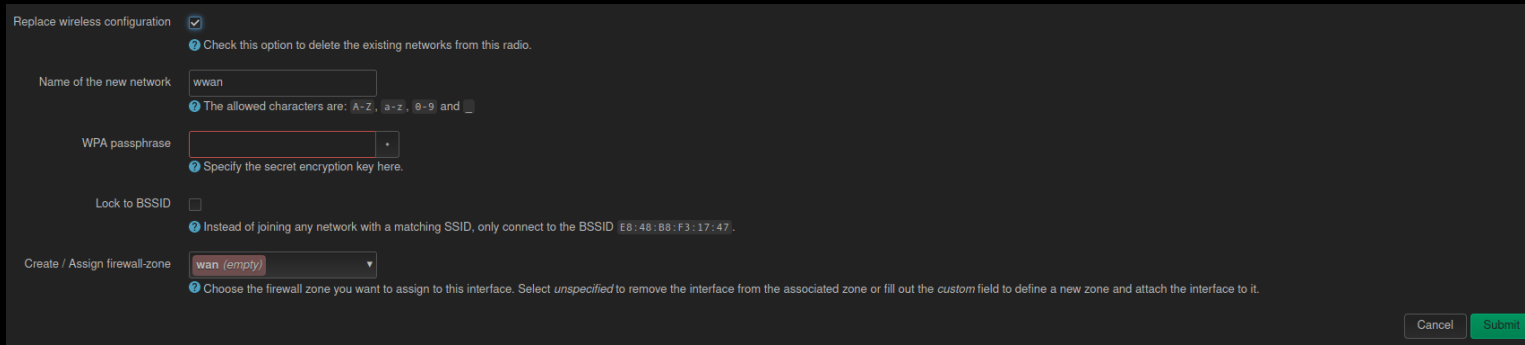
Network → Wireless

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.

48



# Radio0 Network



The screenshot shows a configuration window for a radio network. It has a dark grey background with white text. At the top, there's a section 'Replace wireless configuration' with a checked checkbox. Below it, a help icon and text state: 'Check this option to delete the existing networks from this radio.' The next section is 'Name of the new network' with a text input field containing 'wwan'. A help icon and text below it state: 'The allowed characters are: A-Z, a-z, 0-9, and \_'. The third section is 'WPA passphrase' with a password input field. A help icon and text below it state: 'Specify the secret encryption key here.' The fourth section is 'Lock to BSSID' with an unchecked checkbox. A help icon and text below it state: 'Instead of joining any network with a matching SSID, only connect to the BSSID E8:48:B8:F3:17:47.' The final section is 'Create / Assign firewall-zone' with a dropdown menu showing 'wan (empty)'. A help icon and text below it state: 'Choose the firewall zone you want to assign to this interface. Select unspecified to remove the interface from the associated zone or fill out the custom field to define a new zone and attach the interface to it.' At the bottom right, there are two buttons: 'Cancel' and 'Submit'.

Replace wireless configuration ☒

Check this option to delete the existing networks from this radio.

Name of the new network

The allowed characters are: A-Z, a-z, 0-9, and \_

WPA passphrase

Specify the secret encryption key here.

Lock to BSSID ☐

Instead of joining any network with a matching SSID, only connect to the BSSID E8:48:B8:F3:17:47.

Create / Assign firewall-zone

Choose the firewall zone you want to assign to this interface. Select *unspecified* to remove the interface from the associated zone or fill out the *custom* field to define a new zone and attach the interface to it.

Cancel Submit

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

---


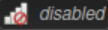
```
root@OpenWrt:~# lsusb
3us 001 Device 003: ID 0424:ec00
3us 001 Device 002: ID 0424:9514
3us 001 Device 001: ID 1d6b:0002 Linux 5.10.176 dwc_otg_hcd DWC OTG Controller
root@OpenWrt:~# █
```

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

 radio1	<b>Generic 802.11bg</b> <i>Device is not active</i>	Restart	Scan	Add
 disabled	<b>SSID: OpenWrt   Mode: Master</b> <i>Wireless is disabled</i>	Enable	Edit	Remove

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

# Radio1 Config: Continued

Device Configuration

General Setup

Advanced Settings

Status

Mode: Master | SSID: OpenWrt  
- dBm Wireless is not associated

Wireless network is enabled

Disable

Operating frequency

Mode

Channel

Legacy | 1 (2412 Mhz)

Allow legacy 802.11b rates

☐

Legacy or badly behaving devices may require legacy 802.11b rates to interoperate. Airtime efficiency may be significantly reduced where these are used. It is recommended to not allow 802.11b rates where possible.

Maximum transmit power

driver default

 - Current power: unknown

Specifies the maximum transmit power the wireless radio may use. Depending on regulatory requirements and wireless usage, the actual transmit power may be reduced by the driver.

Interface Configuration

General Setup

Wireless Security

MAC-Filter

Advanced Settings

Mode

Access Point

ESSID

OpenWrt

Network

wwan

Choose the network(s) you want to attach to this wireless interface or fill out the custom field to define a new network.

Hide ESSID

☐

Where the ESSID is hidden, clients may fail to roam and airtime efficiency may be significantly reduced.

WMM Mode

☒

Where Wi-Fi Multimedia (WMM) Mode QoS is disabled, clients may be limited to 802.11a/802.11g rates.

Dismiss

Save

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

Select Wireless Security.

# Radio1 Config: Continued

**Interface Configuration**

General Setup | **Wireless Security** | MAC-Filter | Advanced Settings | WLAN roaming

Encryption: WPA2-PSK/WPA3-SAE Mixed M ▾

Key:  \*

802.11w Management Frame Protection: Optional ▾  
 ⓘ Note: Some wireless drivers do not fully support 802.11w. E.g. mwlwifi may have problems

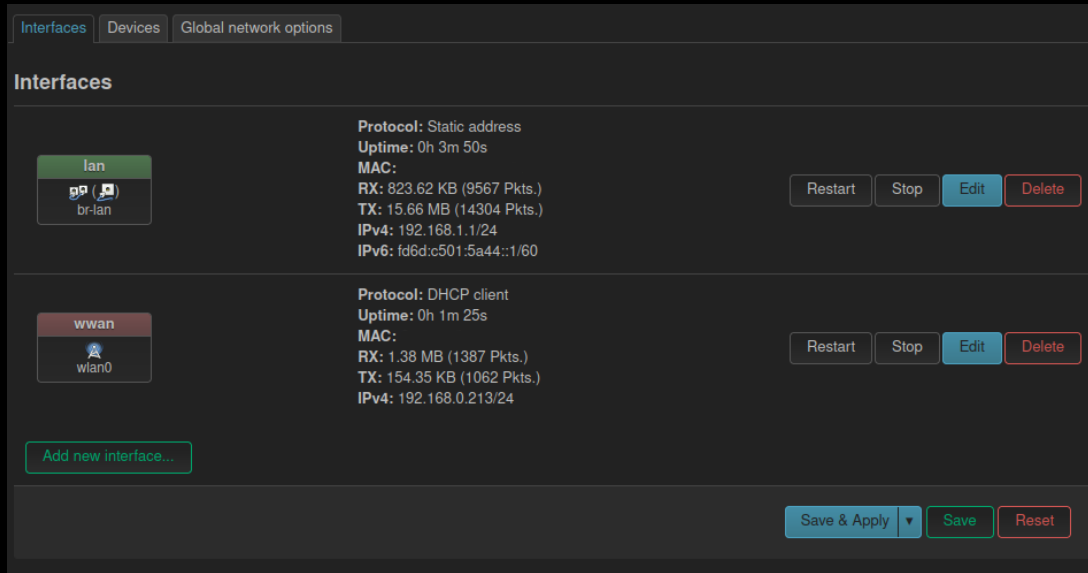
802.11w maximum timeout:   
 ⓘ 802.11w Association SA Query maximum timeout

802.11w retry timeout:   
 ⓘ 802.11w Association SA Query retry timeout

Enable key reinstallation (KRACK) countermeasures: ☐  
 ⓘ Complicates key reinstallation attacks on the client side by disabling retransmission of EAPOL-Key frames that are used to install keys. This workaround might cause interoperability issues and reduced robustness of key negotiation especially in environments with heavy traffic load.

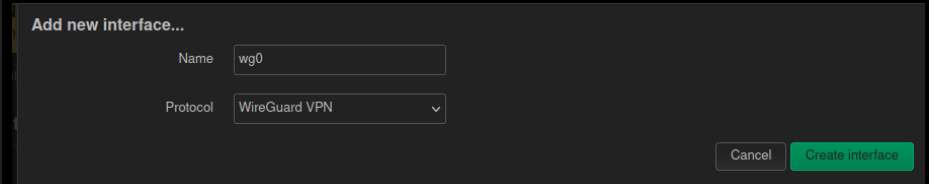
1. Encryption: WPA2-PSK/WPA3-SAE Mixed Mode
2. Key: This should be no less than 16 characters
3. Click Save
4. Then on the main Wireless Screen click save and apply

# 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

Interfaces » wg0

General Settings Advanced Settings Firewall Settings DHCP Server Peers

Status  Device: wireguard-wg0  
RX: 0 B (0 Pkts.)  
TX: 0 B (0 Pkts.)

Protocol WireGuard VPN

Bring up on boot ☒

Private Key  ⓘ Required. Base64-encoded private key for this interface.

Public Key STPK0rbj+WMf7gCXHaLee8D5Flt ⓘ Base64-encoded public key of this interface for sharing.

Listen Port  ⓘ Optional. UDP port used for outgoing and incoming packets.

IP Addresses  ⓘ Recommended. IP addresses of the WireGuard interface.

ⓘ Optional. Do not create host routes to peers.

Import configuration  ⓘ Imports settings from an existing WireGuard configuration file

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

The screenshot shows the 'wg0 Peer Config' interface with the following fields and descriptions:

- Description:** PFSense  
Optional. Description of peer.
- Public Key:** [Empty field]  
Required. Public key of the WireGuard peer.
- Private Key:** [Empty field]  
Optional. Private key of the WireGuard peer. The key is not required for establishing a connection but allows generating a peer configuration or QR code if available. It can be removed after the configuration has been exported.  
Generate new key pair
- Preshared Key:** [Empty field]  
Optional. Base64-encoded preshared key. Adds in an additional layer of symmetric-key cryptography for post-quantum resistance.  
Generate preshared key
- Allowed IPs:** 0.0.0.0/0, ::0  
Optional. IP addresses and prefixes that this peer is allowed to use inside the tunnel. Usually the peer's tunnel IP addresses and the networks the peer routes through the tunnel.
- Route Allowed IPs:** ☒  
Optional. Create routes for Allowed IPs for this peer.
- Endpoint Host:** 192.168.0.44  
Optional. Host of peer. Names are resolved prior to bringing up the interface.
- Endpoint Port:** 51820  
Optional. Port of peer.
- Persistent Keep Alive:** 0  
Optional. Seconds between keep alive messages. Default is 0 (disabled). Recommended value if this device is behind a NAT is 25.
- Configuration Export:** Generate configuration...

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.