

OpenWRT WireGuard Server Setup guide using LuCi

Latest iteration of this guide can be found at: <https://github.com/egc112/OpenWRT-egc-add-on/tree/main/notes>

There you can also find a guide to setup [WireGuard as a Client](#).

Version 15

Introduction

[WireGuard](#) is an open-source VPN solution written in C by [Jason Donenfeld](#) and [others](#), aiming to fix many of the problems that have plagued other modern server-to-server VPN offerings like IPSec/IKEv2, OpenVPN, or L2TP.

It many ways it can be seen as a replacement for OpenVPN.

It has three advantages over OpenVPN, it is much faster especially on lower-spec hardware such as Soho routers (my own R7800 goes from 85 Mb/s on OpenVPN to 300 Mb/s with WireGuard), it is easy to setup if you know how, the guides will help you with that and it has a very small code base (about 4000 lines) so that it can easily be reviewed and checked for vulnerabilities.

Some key points about WireGuard:

- Layer 3 only no bridging
- UDP only punches through firewall
- Like SSH authenticated keys
- Executes in Linux Kernel
- Static routing

WireGuard basically is a peer to peer connection, it can be both "Client" and "Server" (if you have this setup at the same time between two routers we are talking about a site-to-site setup).

Although WireGuard is a peer to peer setup, the configuration is different when using it as a Server, listening for incoming connections e.g. from your phone/laptop from outside, or as a Client making an outbound connection to e.g. a WireGuard server of your VPN provider.

So for practical purposes we still distinguish between a Server and a Client setup

This guide is about setting up a WireGuard Server on a router in basic gateway mode (WAN connected to the internet or to an other router and on its own subnet) where it listens for incoming connections of your phone, laptop or other router.

This guide is based upon OpenWRT 24.10 but also should work on 23.05 and Main builds and uses LuCi to set things up but the resulting **config files** are also listed. The screenshots are made with OpenWRT2020 theme but that is not much different from the default theme.

General Remarks

The most important parts of WireGuard are the public/private keys and the Allowed IP.

The public key is distributed to the peers.

The Allowed IP serves two roles, the first is that the allowed IP is used to know which of the peers public keys (if there is more than one peer) should be used to encrypt the packets.

Therefore the Allowed IP's must be unique for each peer!

The second one is security, if WireGuard detects a source IP which is not in the Allowed IP's, the packets are discarded.

The keys are 32 bytes long and can be easily represented in Base64 encoding in 44 *characters the last character is always an =*.

Check all involved subnets

As WireGuard is a routed solution **all three involved subnets have to be different and non overlapping**. So the routers subnet, the WireGuard subnet and the Clients subnet all have to be different.

As you often cannot choose the subnet of the WireGuard client, it is best to **avoid** using frequently used subnets for your router e.g. 192.168.1.1/24 or 192.168.0.1/24 but if you cannot easily change the routers subnet then leave it and hope for the best.

For the WireGuard subnet 172.22.22.1/24 is chosen in this example which is not often used.

Check for Public IP address

To be able to connect from outside, your router must have a public IP address (either IPv4 and/or IPv6).

Check if your router has a proper Public IPv4 address with (from command line):

ifstatus wan | grep address

A public IP address does **not** start with 192.168.X.X, 10.X.X.X, 172.16-31.X.X or a [CGNAT address](#) (IP addresses from 100.64.0.0 to 100.127.255.255)

Another way to test is to compare the routers IP address (*ifstatus wan | grep address*) with the address from *ipleak.net* it should show the same address for a Public IPv4 address. If it is not the same you might not have a Public IPv4 address and the router is not reachable with IPv4.

If your router is behind another router then:

- Check if that router has a Public IP address.
- Check if and how you can Port Forward from that router to your router which is going to run the WireGuard Server.

If you are behind CGNAT, so do not have a public IPv4 address and do not have a public IPv6 address (check with: *ifstatus wan6*) or using IPv6 is not applicable then you have to involve a commercial third party to get a public IP address.

This can be a VPN provider which supports port forwarding (e.g. ProtonVPN), or you can rent a Virtual Private Server (I have an Oracle VPS which can be had for free, see at the bottom of this guide), or use things like [Netbird](#), [Zerotier](#), [Cloudflared](#), [Tailscale](#) or [ngrok](#) and there are more, I have setup Netbird on several OpenWRT and Windows and Linux clients and it works well, see my notes [about setting up Netbird on OpenWRT](#) and the [Netbird support thread](#).

If your Public IP address is non static, e.g. it can change, then look into using [DDNS](#).

Test from outside

Proper **testing** can only be done **from outside** e.g. with your phone or laptop on cellular data or from a friends/neighbors internet.

OpenWRT WireGuard wiki

Other useful information can be found in the [OpenWRT WireGuard wiki](#).

Index

Introduction.....	1
General Remarks.....	1
Check all involved subnets.....	1
Check for Public IP address.....	2
Test from outside.....	2
OpenWRT WireGuard wiki.....	2
Server setup.....	4
Installation.....	4
Create WireGuard Interface.....	4
Firewall Setup.....	6
1. Opening up the port (55443 in this example) with a traffic rule.....	6
Luci > Network > Firewall > Traffic Rules.....	6
2a. Allowing traffic for the wgserver's the interface.....	7
2b. Alternative setup.....	7
3. Optional, Allow IPv6 internet for wgserver clients connecting to your router.....	9
Peer (Client) Setup.....	10
Setup WireGuard on your Client.....	13
Advanced Section.....	14
Allow seamless access to LAN clients.....	14
Site-to-site setup.....	15
On the server side (site A, subnet 192.168.5.0/24):.....	15
On the client side (site B, subnet 192.168.9.0/24):.....	15
DNSMasq resolution for clients connecting (peers).....	15
Multi-site setup.....	16
Hub and spoke.....	16
Mesh.....	17
WireGuard server on a BridgeAP.....	17
Simultaneous WireGuard Server and WireGuard Client.....	17
Asking for Help.....	19
Miscellaneous.....	19
Setup IPv6 on a bridgedAP.....	19
References.....	20
Setup Oracle free OpenVPN cloud server.....	20
Amazon Web services (AWS).....	20

Server setup

Installation

Install WireGuard:

LuCi > System > Software: click *Update Lists* to get the latest packages for your build

Install: *luci-proto-wireguard* (wireguard-tools should be installed automatically)

Create WireGuard Interface

Next up we are going to create the WireGuard Interface:

Network > Interfaces on the bottom click: **Add New interface**

Add new interface...

Name	<input type="text" value="wgserver"/>
Protocol	<input type="text" value="WireGuard VPN"/>

- **Name:** give the interface a name (hyphens are not allowed and the name has to be less than 15 characters!)
- **Protocol:** *WireGuard VPN*
- **Click:** *Create interface* and the Interface configuration screen should appear:

Interfaces » wgserver

General Settings Advanced Settings Firewall Settings DHCP Server Peers

Status	<div>Device: wireguard-wgserver RX: 0 B (0 Pkts.) TX: 0 B (0 Pkts.)</div>
Protocol	<input type="text" value="WireGuard VPN"/>
Disable this interface	<input type="checkbox"/>
Bring up on boot	<input checked="" type="checkbox"/>
Private Key	<input type="text"/> Required. Base64-encoded private key for this interface.
Public Key	<input type="text"/> Base64-encoded public key of this interface for sharing. <input type="button" value="Generate new key pair"/>
Listen Port	<input type="text" value="random"/> Optional. UDP port used for outgoing and incoming packets.
IP Addresses	<input type="text"/> Recommended. IP addresses of the WireGuard interface.
No Host Routes	<input type="checkbox"/> Optional. Do not create host routes to peers.

- **Click:** *Generate new key pair*
- **Listen port:** *55443*, (you can use any port which is not already taken but then you have to replace 55443 mentioned everywhere with your own port).
- **IP Addresses:** *172.22.22.1/24*, if you also want to use IPv6 use a [ULA address](#) e.g.: *fd8f:de49::1/64*, you can use an [ULA calculator](#) if you want.

Status

Device: wgserver
Uptime: 0h 1m 41s
RX: 0 B (0 Pkts.)
TX: 0 B (0 Pkts.)
IPv4: 172.22.22.1/24
IPv6: fd8f:de49:19f1:ffff::1/64

Protocol

WireGuard VPN

Disable this interface

☐

Bring up on boot

☒

Private Key

.....*

Required. Base64-encoded private key for this interface.

Public Key

ML5BqgOUmKMklzhXGSXnmFWeTVD1gI

Base64-encoded public key of this interface for sharing.

Generate new key pair

Listen Port

55443

Optional. UDP port used for outgoing and incoming packets.

IP Addresses

172.22.22.1/24

fd8f:de49::1/64

Recommended. IP addresses of the WireGuard interface.

Save and then Save & Apply

wgserver



wgserver

Protocol: WireGuard VPN
Uptime: 0h 2m 41s
RX: 0 B (0 Pkts.)
TX: 0 B (0 Pkts.)
IPv4: 172.22.22.1/24
IPv6: fd8f:de49:19f1:ffff::1/64

Restart Stop Edit Delete

```
/etc/config/network:
config interface 'wgserver'
    option proto 'wireguard'
    option private_key 'MIShrFJZqAQ4UGxcvXZVDRFDdS57s3M7lhW0='
    option listen_port '55443'
    list addresses '172.22.22.1/24'
    list addresses 'fd8f:de49::1/64'
```

Firewall Setup

The firewall setup consist of three things:

1. Opening up the port (55443 in this example) with a traffic rule
- 2a. Allowing traffic for the *wgserver* interface
3. Optional: Allow IPv6 internet for your wgserver clients, if you have implemented IPv6 and want your clients to have IPv6 internet

1. Opening up the port (55443 in this example) with a traffic rule

Luci > Network > Firewall > Traffic Rules

- **Add** new traffic rule
- **Name:** *allow-55443*
- **Protocol:** *UDP*, click drop down button and disable TCP
- **Source zone:** *WAN*
- **Destination zone:** *Device (input)*
- **Destination port:** *55443*, the port the wgserver interface listens on which is set on the wgserver interface
- **Action:** *accept*
-

The traffic rule, by default, applies to IPv4 and IPv6, you can restrict the rule to IPv4 on the Advanced Tab

Firewall - Traffic Rules - Unnamed rule

General Settings

Advanced Settings

Time Restrictions

Name

allow-55443

Protocol

UDP

Source zone

wan wan: wan6: wg_proton_nl:

Source address

-- add IP --

Source port

any

Destination zone

Device(input)

Destination address

-- add IP --

Destination port

55443

Action

accept

Save the rule and the result looks like this:

allow-55443	Incoming <i>IPv4</i> and <i>IPv6</i> , protocol <i>UDP</i>		
	From <i>wan</i>	Accept input	<input checked="" type="checkbox"/>
	To <i>this device</i> , port <i>55443</i>		

```
/etc/config/firewall:
config rule
    option name 'allow-55443'
    list proto 'udp'
    option src 'wan'
    option dest_port '55443'
    option target 'ACCEPT'
```

2a. Allowing traffic for the wgserver's the interface

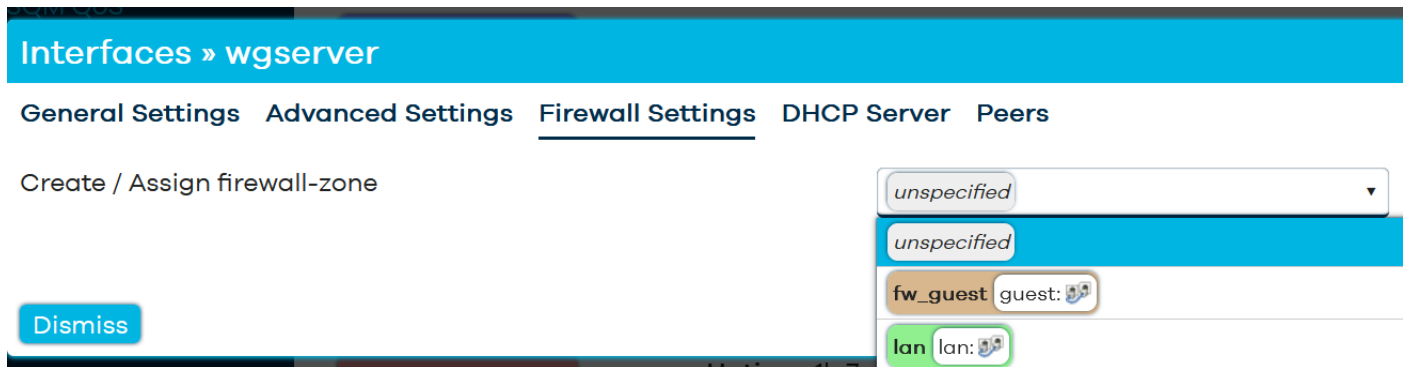
The easiest method is to edit the wg server interface.

Network > Interfaces and click the *edit* button on the *wgserver interface*

Go to **Firewall settings**:

Click on the drop down button and click on *lan*, this will add the wgserver interface to the lan zone

Save and Save & Apply



```
/etc/config/firewall:
config zone
    option name 'lan'
    option input 'ACCEPT'
    option output 'ACCEPT'
    option forward 'ACCEPT'
    list network 'lan'
    list network 'wgserver'
```

2b. Alternative setup

This can be used **instead** of 2a if you want a more finer grained control.

Instead of adding the wgserver interface to the LAN zone, create a separate firewall zone:

Go to **Network > Firewall** and

Click **Add**

- **Name:** *wgserver* (or a name to your liking)
- **Input:** *accept* (unless you do not want your wgserver clients to have access to your router)
- **Output:** *accept* (always set accept)
- **Intra zone Forward:** *accept* (unless yo do not want your wgserver clients to be able to communicate with each other)
- **Covered networks:** *wgserver* (this is the wgserver interface)
- **allow forward to destination zones:**
 - *lan zone* (to allow to connect to your lan clients)
 - *wan zone* (optional to give your wg server clients internet access via your router)
- **allow forward from source zones:**
 - *lan zone* (optional only to allow bidirectional traffic e.g. in case you have a site-to-site setup, ote that your wg server clients als0 have to allow this traffic)

Firewall - Zone Settings

General Settings Advanced Settings Conntrack Settings

This section defines common properties of "this new zone". The *input* and *output* options set the default policies for traffic entering and describes the policy for forwarded traffic between different networks within the zone. *Covered networks* specifies which available netw

Name	<input type="text" value="wgserver"/>
Input	<input type="text" value="accept"/>
Output	<input type="text" value="accept"/>
Intra zone forward	<input type="text" value="accept"/>
Masquerading	<input type="checkbox"/> Enable network address and port translation IPv4 (NAT4) typically enabled on the <i>wan</i> zone.
MSS clamping	<input type="checkbox"/>
Covered networks	<input type="text" value="wgserver:"/>

The options below control the forwarding policies between this zone (this new zone) and other zones. *Destination zones* cover forwarded *zones* match forwarded traffic from other zones **targeted at this new zone**. The forwarding rule is *unidirectional*, e.g. a forward from lan from wan to lan as well.

Allow forward to *destination zones*:

Allow forward from *source zones*:

/etc/config/firewall:

config zone

```
option name 'wgserver'
option input 'ACCEPT'      #use REJECT if you do not want access to the router
option output 'ACCEPT'
option forward 'ACCEPT'
list network 'wgserver'
```

config forwarding

```
# to give WG clients access to your LAN clients
# if you want more fine grained control remove this and make a traffic rule see below
option src 'wgserver'
option dest 'lan'
```

config forwarding

```
# to give your LAN client access to WG clients needed for site-to-site setup
option src 'lan'
option dest 'wgserver'
```

config forwarding

```
# to give your WG client access to the internet via the router
option src 'wgserver'
option dest 'wan'
```

#Extra traffic rule for access to individual LAN clients if there is no general forwarding:

config rule

```
option name 'access-from-wg'
option src 'wgserver'
list dest_ip '192.168.5.199'
option target 'ACCEPT'
list proto 'all'
option dest '*'
```


3. Optional, Allow IPv6 internet for wgserver clients connecting to your router

If IPv6 is implemented and you want wgserver clients to have internet access via IPv6 then Masquerading should be enabled for the WireGuard subnet out of the WAN.

Network > Firewall > **Wan zone** : **click edit**

goto *Advanced Settings*:

- **IPv6 Masquerading**: *Enable*

Restrict Masquerading to given source subnets: *fd8f:de49::0/64* ([WireGuard IPv6 subnet from the interface](#)).

Save and **Save & Apply**

Firewall - Zone Settings

General Settings **Advanced Settings** **Conntrack Settings**

The options below control the forwarding policies between this zone (wan) and other zones. *Destination zones* cover forward forwarded traffic from other zones **targeted at wan**. The forwarding rule is *unidirectional*, e.g. a forward from lan to wan doe

Covered devices	<div>unspecified</div> <div>Use this option to classify zone traffic by raw, nc</div>
Covered subnets	<div></div> <div>Use this option to classify zone traffic by source</div>
IPv6 Masquerading	<div><input checked="" type="checkbox"/></div> <div>Enable network address and port translation IPv</div>
Restrict to address family	<div>IPv4 and IPv6</div>
Restrict Masquerading to given source subnets	<div>fd8f:de49::0/64</div> <div>0.0.0.0/0</div>

```
/etc/config/firewall:
config zone
    option name 'wan'
    <>
    option masq6 '1'
    list masq_src 'fd8f:de49::0/64'
```

Furthermore there is no standard default route for IPv6 as "source" routing is used so source routing should be disabled.

Network > Interfaces > WAN6: **click edit**

goto *Advanced Settings*:

Disable Source Routing by removing the tick:

IPv6 source routing ☐ Automatically handle multiple uplink interfaces using source-based policy routing.

Save and **Save & Apply**

```
/etc/config/network:
config interface 'wan6'
    option device 'wan'
    <>
    option sourcefilter '0'
```

Peer (Client) Setup

Next setup the peers for the WireGuard server.

Peers are the clients which connects from outside to the wgserver.

There are WireGuard clients for almost operating systems.

We are going to setup one Peer but you can of course add as many as you want, note that you can reuse this one peer for multiple clients but you can only connect one at a time! So usually you make a peer for each client you want to connect to your server

Note: when adding a second peer later, be sure to restart the network (*service network restart*) or reboot the router to take effect.

Go to **Network > Interfaces > wgserver > Peers**

Interfaces » wgserver

[General Settings](#) [Advanced Settings](#) [Firewall Settings](#) [DHCP Server](#) [Peers](#)

Further information about WireGuard interfaces and peers at wireguard.com.

Disabled	Description	Allowed IPs	Endpoint Host
No peers defined yet.			

[Add peer](#) [Import configuration as peer...](#)

Click: *Add Peer*

- **Description:** give a name for your Peer
- **Click *Generate new key pair***, the keys for the peer will be filled in.
- **Allowed IPs:** 172.22.22.2/32, the wgserver has this address 172.22.22.1/24, all peers should have an address in this subnet, so for the first peer use 172.22.22.2/32, note the /32 mask.
Subsequent peers (clients) will use .3/32 etc.
For IPv6 you add: fd8f:de49::2/128, note the /128 mask
- WireGuard uses [crypto key routing](#) meaning the peers addresses have to be unique so the addresses cannot have overlap (long answer: overlap is possible as long as the addresses are unique).
- **Route Allowed IPs:** *Enable*, Always enable this!
- **Endpoint host:** *Leave blank*
- **Endpoint port:** 55443, this is the listening port of the wgserver (only used to make your config).
- **Persistent keep alive:** 25, most clients are behind NAT so to keep the connection open use persistent keep alive (only used to make your config).

Interfaces » wgserver » Edit peer

Disabled

☐

Enable / Disable peer. Restart wireguard interfa

Description

My Phone

Optional. Description of peer.

Public Key

1/eg09gOLT72ogh2sUC9ySNbbb4yhOo+c

Required. Public key of the WireGuard peer.

Private Key

.....*

Optional. Private key of the WireGuard peer. The a connection but allows generating a peer conf can be removed after the configuration has bee

Generate new key pair

Preshared Key

*

Optional. Base64-encoded preshared key. Adds symmetric-key cryptography for post-quantum

Generate preshared key

Allowed IPs

172.22.22.2/32-

fd8f:de49::2/128-

+

Optional. IP addresses and prefixes that this pe tunnel. Usually the peer's tunnel IP addresses ar through the tunnel.

Route Allowed IPs

☒

Optional. Create routes for Allowed IPs for this p

Endpoint Host

vpn.example.com

Optional. Host of peer. Names are resolved prior

Endpoint Port

55443


Optional. Port of peer.

Persistent Keep Alive

25

Optional. Seconds between keep alive message Recommended value if this device is behind a N

Configuration Export

 Generate configuration...

Generates a configuration suitable for import o

Save

Open the peer again by clicking on **Edit**.

Click: *Generate configurations*

- **Connection Endpoint:** Add the WAN IP address or DDNS address your wgserver listens on
- **Allowed IPs:** standard `0.0.0.0/0, ::/0`, which means all traffic from your wg client will use the tunnel
- **DNS server:** standard your routers IP address, but not all clients can deal with this (because of rebind protection, using the wgserver's interface IP (172.22.22.1) might help) but you router might not listen on the wgserver's interface or only listens for local subnets (option localservice '0') **so to be sure that you have got DNS resolution use 1.1.1.1** but if you have setup DNS hijacking (either manually or through e.g. Adblock or https-dns-proxy) that might also play tricks .
- **Addresses:** do not change
- **ListenPort:** The listen port is automatically added to the peers config and is the same as the listen port of the server, however it is better to remove the listen port on your client (your phone, laptop etc.), this way the client chooses its own listen port which will not collide with existing listen ports. This has to be done on the client itself after you uploaded the config.

Interfaces » wgserver » Edit peer » Generate configuration

The generated configuration can be imported into a WireGuard client application to set up a connection towards

Connection endpoint

my.ddns.nl ▼

The public hostname or IP address of this system that usually is a static public IP address, a static hostname

Allowed IPs

0.0.0.0/0 -

::/0 -

-- Please choose -- ▼

IP addresses that are allowed inside the tunnel. The packets with source IP addresses matching this list matching destination IP.

DNS Servers

192.168.5.1 -

1.1.1.1 -

+

DNS servers for the remote clients using this tunnel wireguard clients require this to be set.

Addresses

172.22.22.2/32 -

fd8f:de49::2/128 -

-- Please choose -- ▼

IP addresses for the peer to use inside the tunnel. See



```
[Interface]
PrivateKey = iGrogUvTflvHv1y...bcZSfGiyX64FUko=
Address = 172.22.22.2/32, fd8f:de49::2/128
ListenPort = 55443
DNS = 192.168.5.1, 1.1.1.1

[Peer]
PublicKey = ML5BqgOUmKMk1...Dn15SEB8f/T5zo=
# PresharedKey not used
AllowedIPs = 0.0.0.0/0, ::/0
Endpoint = my.ddns.nl:55443
PersistentKeepAlive = 25
```

```
/etc/config/network:
config wireguard_wgserver
    option description 'My Phone'
    option public_key '1/eg09g0LT72ogh2s7kmDaAv1gM='
    option private_key 'iGrogUvTflvHv1ySfGiyX64FUko='
    list allowed_ips '172.22.22.2/32'
    list allowed_ips 'fd8f:de49::2/128'
    option route_allowed_ips '1'
    option endpoint_port '55443'
    option persistent_keepalive '25'
```

Setup WireGuard on your Client

Setup WireGuard on your client (phone/laptop etc) by downloading the WireGuard app.

See for available apps and how to download: <https://www.wireguard.com/install/>

For OpenWRT see: <https://github.com/egc112/OpenWRT-egc-add-on/tree/main/notes>

For DDWRT see: <https://forum.dd-wrt.com/phpBB2/viewtopic.php?t=327397>

You can import the settings with the QR code or copy the text and paste in a file, name it *peer-172.22.22.2.conf* which can be used to import in your wg client.

Make sure the listen port is removed and start with using a public DNS server e.g. 1.1.1.1 as not all clients can deal with using the wgserver as DNS server

Finish by Saving and Applying everything and do a reboot!

Now see if you can connect from outside e.g. with your phone or laptop on cellular.

Note that your LAN clients will not always allow traffic from a foreign subnet, in that case you have to tweak the firewall of said lan clients to allow traffic from 172.22.22.0/24 (the wg servers subnet), or masquerade this traffic

[Allow seamless access to LAN clients](#)

Advanced Section

Allow seamless access to LAN clients

Your LAN clients might not accept traffic from your WG clients because traffic comes from another subnet and LAN clients might have their own firewall which blocks non local traffic.

The best way to solve this is to tweak the firewall of local clients to accept traffic from the WG subnet.

For Windows:

[How to Add IP Address in Windows Firewall](#)

Step 1) On the Start menu, Click 'Windows Firewall with Advanced Security'.

Step 2) Click the 'Advanced settings' option in the sidebar.

Step 3) On the left side, click the option 'Inbound Rules'.

Step 4) On the right, under the section 'Actions', click on the option 'New Rule'. Windows Firewall shows you the New Inbound Rule Wizard.

Step 5) A new window will open and Select the 'custom' option and click Next.

Step 6) In the left-hand side again, go to the option 'Scope'.

Step 7) Add the IP address and click on the 'Ok' button.


But if that is not feasible you can masquerade the WireGuard traffic which comes out of the router.

Simplest method is to use [option 2b](#) for setting up the firewall and Enable Masquerading on the LAN interface.

However this Masquerades all traffic so better is to only Masquerade WG traffic with the following firewall NAT rule:
Network > Firewall > NAT rules:

Firewall - NAT Rules - SNAT-WGserver

General Settings Advanced Settings Time Restrictions

Name	SNAT-WGserver
Restrict to address family	automatic
Protocol	Any
Outbound zone	lan lan: 
Source address	172.22.22.0/24 <small>Match forwarded traffic from this IP or range.</small>
Destination address	any <small>Match forwarded traffic directed at the given IP address.</small>
Action	MASQUERADE - Automatically rewrite to outbound interface IP

```
/etc/config/firewall:
```

```
config nat
    option name 'SNAT-WGserver'
    option src 'lan'
    option src_ip '172.22.22.0/24' # the WG subnet, note the .0 at the end
    option target 'MASQUERADE'
    list proto 'all'
    option enabled '1'
```

With Masquerading WG traffic you loose logging and access control, so I am not a fan of it, but in a typical SoHo setup, where you trust your users it might not be not a big deal.

Site-to-site setup

Although WireGuard is a peer to peer connection we still make a distinction between a server, listening for incoming connections and a client which initiates a connection to a server via an endpoint.

A site-to-site setup is the ultimate peer to peer setup in which the WireGuard interfaces are used to make a connection between two routers for bidirectional traffic.

Prerequisites:

All involved subnets need to be unique, so both routers must be on a different subnets and the wg subnet also must be different!

To start just setup one side as a server (Site A) and the other side as a client (Site B), check that you have a working connection.

WireGuard subnet: *172.22.22.0/24*

On the server side (site A, subnet *192.168.5.0/24*):

Network > Interfaces> wgserver interface: edit > Peers > edit Peer of side B

Peer setup of side B:

Allowed IPs: Add whole subnet of site B: *192.168.9.0/24*

It is perfectly possible to add more peers e.g. your phone etc. By connecting your phone to this WireGuard server you are also connected to site B.

On the client side (site B, subnet *192.168.9.0/24*):

Network > Firewall

WAN zone: **remove** *wgclient interface* from the WAN zone

LAN zone: **add** *wgclient interface* to the LAN zone

Network > Interfaces> wgclient interface: edit > Peers > edit Peer of side A

Peer setup of side A:

- Allowed IPs:
1. **Remove** *0.0.0.0/0 and ::0/0*
 2. **Add** the whole subnet of side A: *192.168.5.0/24*
 3. **Add** the whole subnet of WireGuard: *172.22.22.0/24*

DNSMasq resolution for clients connecting (peers)

It is perfectly possible to use DNSMasq for local DNS resolution between both routers.

Prerequisites: the domain names must be different

For a proper setup if both sides are OpenWRT routers four things are important

The first is to make sure that the DNS server can actually process queries from the other side.

DNSmasq has to listen on all interfaces so also on the WG interface, by default this is the case but if you changed that then you have to add the WG interface as listen interface.

The second is that DNSMasq has to answer non local request coming from the other side.

For this disable Local Service only (DNSMasq: *-local-service*):

```
Luci DNS-DHCP > Filter > Local service only : untick/disable, \
or in /etc/config/dhcp > config dnsmasq:
    option localservice '0'
```

The third is that DNSMasq is now also using a DNS server with a local RFC1918 address.

Dnsmasq has rebind protection which shield you from using local addresses as that can be used to spoof DNS.

So disable Rebind Protection:

```
Luci DNS-DHCP > Filter > Rebind protection untick/disable
```

```
/etc/config/dhcp > config dnsmasq
    option rebind_protection '0'
```

instead of disabling Rebind protection you can also whitelist the domain of the other side

```
Luci DNS-DHCP > Filter > Domain Whitelist "set name of domain of other side"
```

```
/etc/config/dhcp > config dnsmasq
    list rebind_domain 'set name of domain of other side'
```

The fourth is that you have to instruct DNSMasq which server it has to use to resolve the domain of the other side, this assumes you have set a different domain name for each side e.g. *lan5* (router is 192.168.5.1) and *lan9* (router is 192.168.9.1)

```
On router lan5 add: server=/lan9/192.168.9.1
```

```
/etc/config/dhcp > config dnsmasq:
    list server '/lan9/192.168.9.1'
```

```
On lan9 : server=/lan5/192.168.5.1
```

```
/etc/config/dhcp > config dnsmasq:
    list server '/lan5/192.168.5.1'
```

Additional tips:

It's possible to make reverse lookup work (e.g. `dig -x 192.168.5.2`).

```
config dnsmasq:
    list server '/5.168.192.in-addr.arpa/192.168.5.1'
    list server '/9.168.192.in-addr.arpa/192.168.9.1'
```

It's possible to set search domains so one can use "short-name" (e.g. from within `site_a` you can `ping server2` and it would ping `server2.site_b` provided `server2` does not also exist in `site_a`).

```
config dhcp 'lan'
    list dhcp_option '119,lan5,lan9'
```

Multi-site setup

Hub and spoke

You can use a hub and spoke setup where site 1 is the hub and site 2 and 3 are the spokes, connection from 2 to 3 is routed via site 1.

Site 1 is a classic server setup (wg interface added to the LAN zone, so no MASQUERADE and allowing incoming WG port).

Site 1 has two peers, site 2 and site 3. Each peer has the subnet and wg address of the respective router as allowed ips.

Sites 2 and 3 are setup as a client with respect to that they have one peer (site 1) and endpoint set to site 1 but they are servers in the sense that they should allow incoming traffic basically as a site-to-site setup, so WG interface added to the LAN zone.

Furthermore site 2 has the subnet of site 1 and site 3 as Allowed IPs and site 3 has as Allowed IPs the subnet of site 1 and 2.

Both site 2 and 3 also have the whole wg subnet as allowed IPs.

Mesh

Alternative is a mesh setup where all sites connect to all other sites, of course each site must be reachable via the internet.

Basically all sites are setup as a server with peers to all other sites, but these peers have an endpoint and make a connection, you use PBR on each site to do the routing.

Again each site has just one tunnel.

In your case this mesh setup might be the easier solution provided that each site is reachable via the internet.

Note both for this mesh setup and hub and spoke use as WG address a unique address in the same subnet and make sure all subnets are different.

WireGuard server on a BridgeAP

Prerequisites:

- Main router has a [public WAN IP addresses](#)
- Main router can do port forwarding, set this up to port forward the servers listen port to the Openwrt routers IP address.

First double check that you have setup your BridgedAP correctly see:

<https://openwrt.org/docs/guide-user/network/wifi/wifiextenders/bridgedap>

Setup the server as outlined in this guide and use the [Alternative Firewall setup](#) with the WireGuard interface on its own firewall zone.

You need a Firewall Forward rule from lan to wireguard zone and from wireguard zone to lan zone.

No need for a traffic rule for the listen port.

Enable Masquerading on the LAN firewall zone, if you have [IPv6 enabled](#) also for IPv6.

Disable source routing on the lan6 interface so that there is a default route for IPv6

Simultaneous WireGuard Server and WireGuard Client

If you have a WireGuard client running on the router with default route via the VPN then also running a WireGuard Server on the router is not possible without some form of [Policy Based Routing](#)., because the traffic for the Server is entering via the WAN and also has to return via the WAN while the default route is via the VPN.

Easy method is to install the [PBR app](#) with its default settings. The PBR app will automatically look for the WG servers listen-port (=source port) and will route that traffic out via the WAN. But you need at least version 1.1.8-r10.

You can upgrade PBR with the following instructions: <https://docs.openwrt.melmac.net/>

Note 1: Your WireGuard client must not have a listen-port set!

Note 2: In the PBR app, the WireGuard interface must **not** be added to Supported or Ignored interfaces option.

Note 3: You can check if it is properly setup with (from command line): *ip rule* that should show an ip rule routing the Servers listen port.

Alternatively you can use a script to route the Servers listen port via the WAN:

<https://github.com/egc112/OpenWRT-egc-add-on/tree/main/pbr-via-wan>

Netifd as of 26-May -2025 supports sport and dport functionality but currently only Main branch and on 24.10.2!

This can be used to create a routing table with default route via the wan which has the listen port of the WireGuard server interface set to this table to route the WireGuard server traffic out via the WAN:

/etc/config/network:

```
config route
    option interface 'wan'
    option target '0.0.0.0/0'
    option table '102'
    option gateway '<set gateway of wan interface>' #ifstatus wan | grep nexthop
```

```
config rule
    # for source port
    option sport '55443' #replace with your chosen listen port
    #table number to use for lookup
    option lookup '102'
```

Asking for Help

You can ask for help at the [OpenWRT forum](#).

If you do it helps if we can have a look at your configs, so please connect to your OpenWRT device [using ssh](#) and copy the output of the following commands and post it on the forum using the "Preformatted text </>" button



Remember to redact keys, passwords, MAC addresses and any public IP addresses you may have:

- `ubus call system board`
- `cat /etc/config/network`
- `cat /etc/config/wireless`
- `cat /etc/config/firewall`
- `wg show`

Miscellaneous

Setup IPv6 on a bridgedAP

/etc/config/network:

```
config interface 'lan6'
    option ifname '@lan'
    option proto 'dhcpv6'
    option reqprefix 'no'
    #option reqprefix '62'    #for ipv6 guest interface
    option sourcefilter '0'  # disable source routing for WG server routing of IPv6
```

References

<https://openwrt.org/docs/guide-user/services/vpn/wireguard/start>

<https://wiki.dd-wrt.com/wiki/index.php/Wireguard>

<https://www.wireguard.com/quickstart/>

<https://www.wireguard.com/>

<https://github.com/pirate/wireguard-docs>

<https://www.wireguard.com/papers/wireguard.pdf>

<https://wiki.archlinux.org/index.php/WireGuard>

<https://stackoverflow.com/questions/65178004/what-does-ip-4-rule-add-table-main-suppress-prefixlength-0-meaning>

ipv6:

<https://angristan.xyz/2019/01/how-to-setup-vpn-server-wireguard-nat-ipv6/>

<https://try.popho.be/wg.html>

suppress prefix length and wg quick

<https://ro-che.info/articles/2021-02-27-linux-routing>

<https://stackoverflow.com/questions/65178004/what-does-ip-4-rule-add-table-main-suppress-prefixlength-0-meaning>

Packet flow:

<https://www.procustodibus.com/blog/2021/01/wireguard-endpoints-and-ip-addresses/>

Logging:

Compile kernel with dynamic debugging:

```
| Location: |
| -> Global build settings |
| -> Kernel build options |
| (6) -> Compile the kernel with dynamic printk (KERNEL_DYNAMIC_DEBUG [=n])
```

```
# modprobe wireguard
```

```
# echo module wireguard +p > /sys/kernel/debug/dynamic_debug/control
```

View with:

```
# dmesg
```

<https://www.procustodibus.com/blog/2021/03/wireguard-logs/>

VXLAN over WireGuard

<https://forum.openwrt.org/t/vxlan-breaks-on-dependent-network-restart/240274/2>

WireGuard server in the cloud

Setup Oracle free OpenVPN cloud server

<https://www.youtube.com/watch?v=E-CLtExRzX8>

<https://mateo.cogeanu.com/2020/wireguard-vpn-pihole-on-free-oracle-cloud/>

Amazon Web services (AWS)

<https://www.youtube.com/watch?v=m-iJBtG4FE>