OpenWRT WireGuard Server Setup guide using LuCi

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

This is guide is to setup WireGuard as a server.
A server is a WireGuard interface listening for incoming connections e.g. from your phone/laptop from outside.
A client is making an outbound connection to a WireGuard server.
But as WireGuard basically is a peer to peer connection it can be both "client" and "server" at the same time, and if you have this setup between two routers we are talking about a site-to-site setup

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.

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 =.*

As WireGuard is a routed solution all three involved subnets have to be different. So the Servers subnet, the WG subnet and the Clients subnet all have to be different!
As you often cannot choose the subnet of the client it is best to avoid using frequently used subnet for your routers IP address of e.g. 192.168.1.1/24 or 192.168.0.1/24

To be able to connect from outside your internet facing 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 is **not** something starting 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)

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

Index

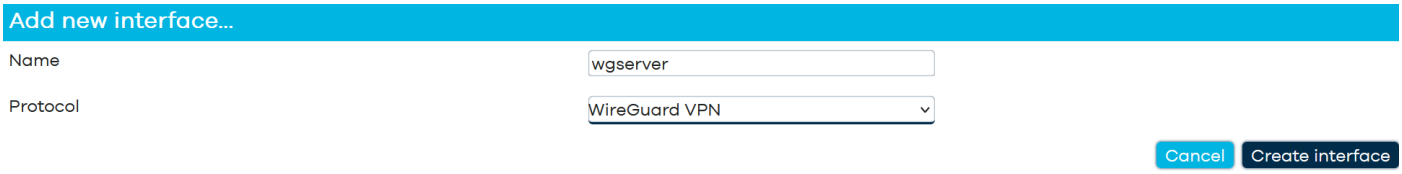
Server setup

Installation

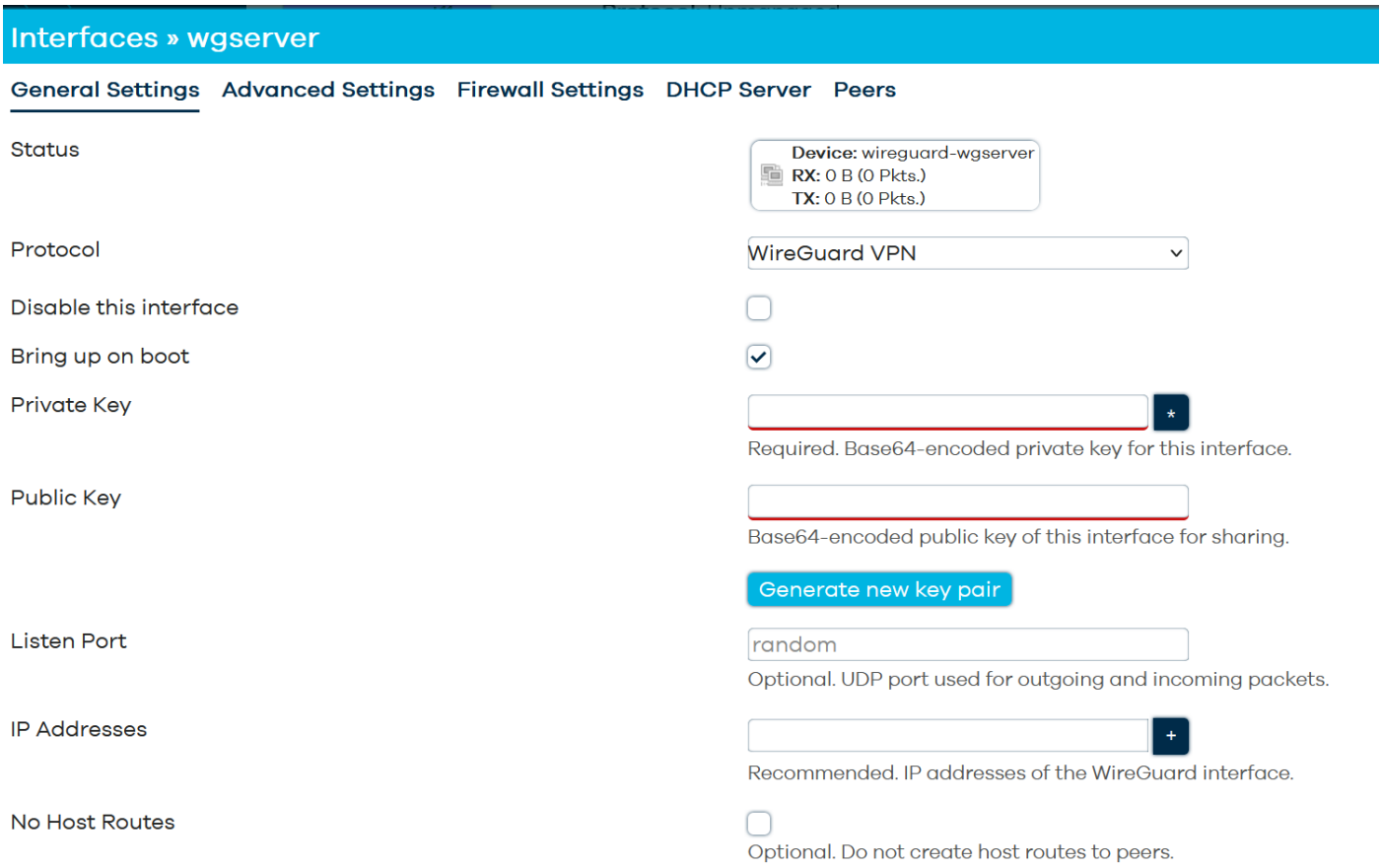
Install WireGuard:
LuCi > System > Software: click `*Update Lists*` to get the latest packages for your build
Install: `*luci-proto-wireguard*`, `*wireguard-tools*` and `*wg-installer-client*` (only necessary if you later want to install a client)

Create WireGuard Interface

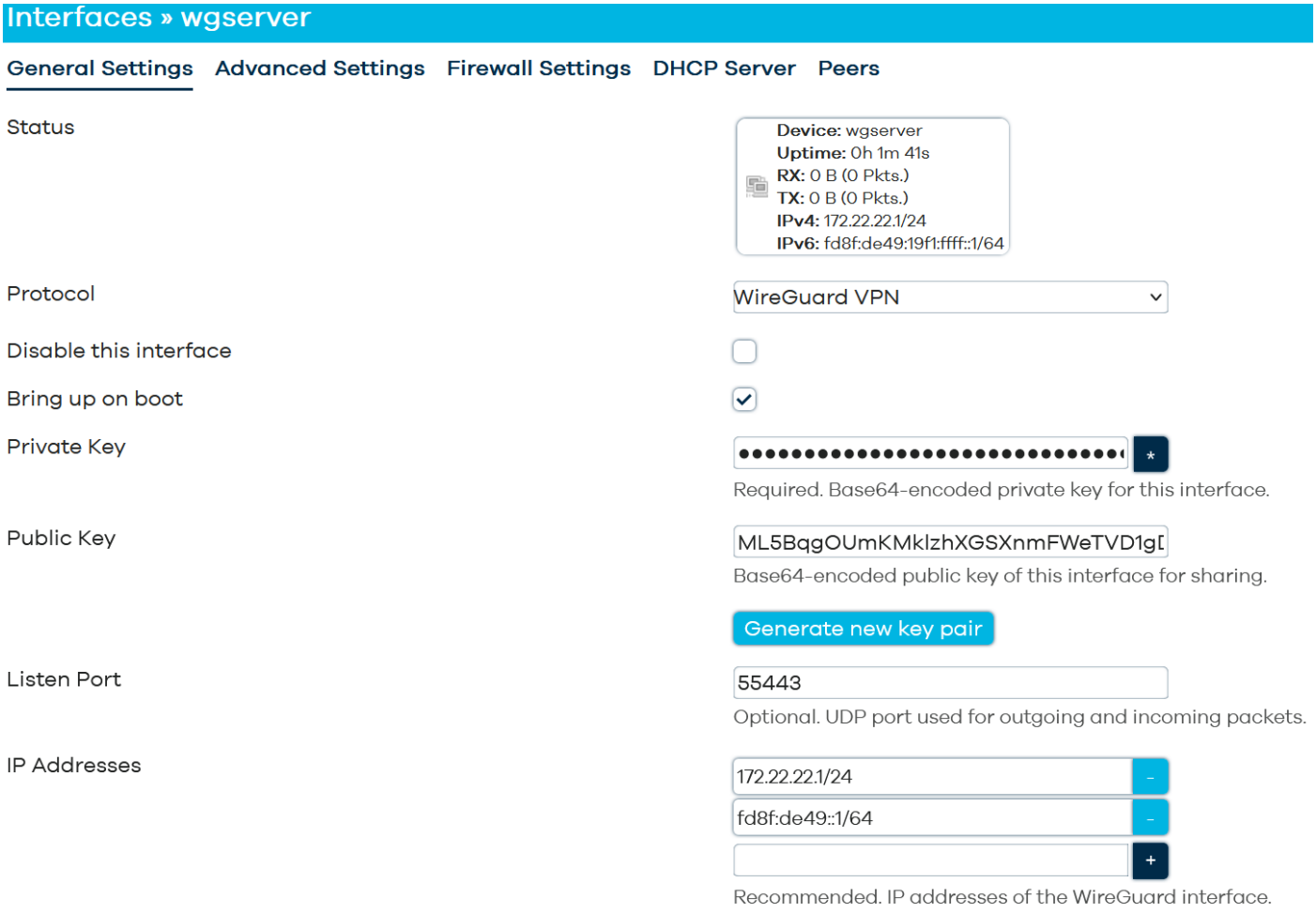
Next up we are going to create the WireGuard Interface:
Network > Interfaces on the bottom click: `Add New interface`



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:



Click: *Generate new key pair*
Listen port: *55443* , you can use any port with is not already taken.
IP Addresses: *172.22.22.1/24*, if you also want IPv6 use a ULA address e.g.: *fd8f:de49::1/64*, you can use an ULA calculator if you want



Save and then Save & Apply

/etc/config/network:

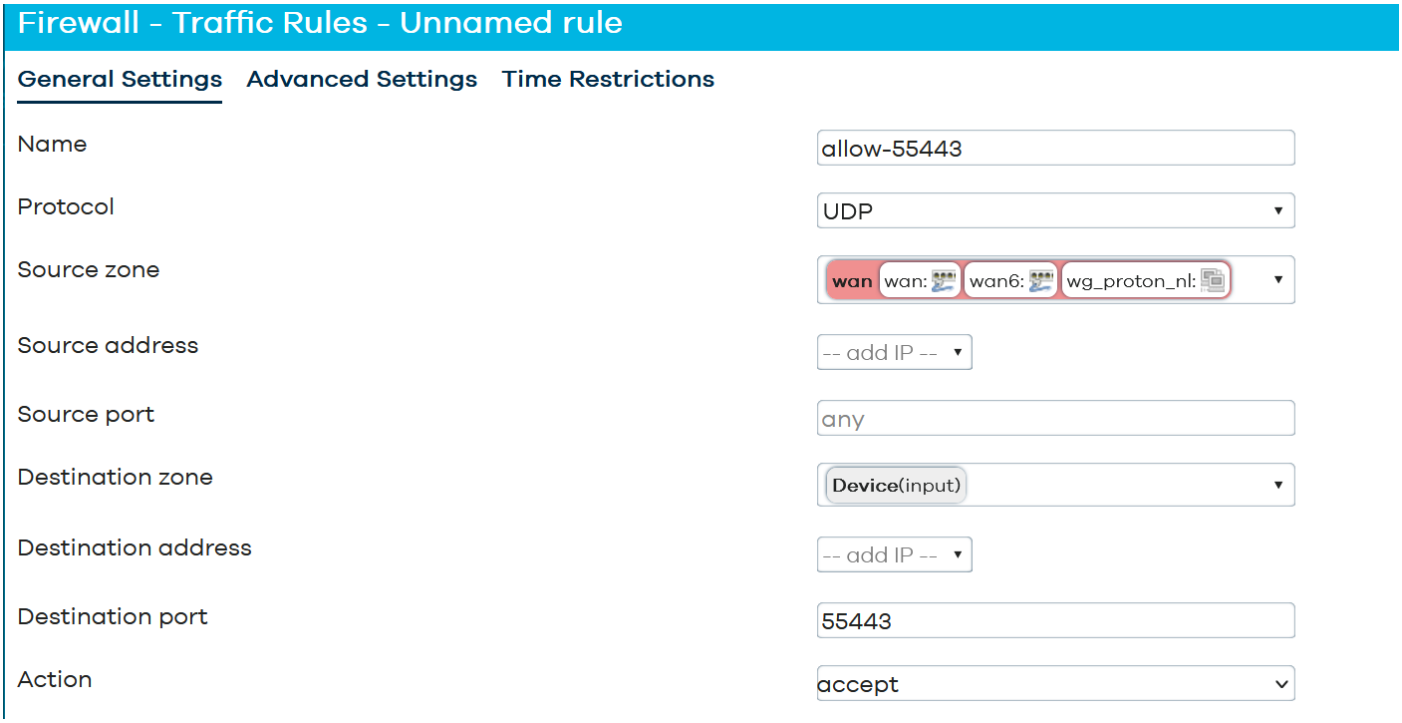
```
config interface 'wgserver'
        option proto 'wireguard'
        option private_key 'MIShxrFJZqAQ4UG/pql2Y+xgs+QP5FyAl57s3M7lhW0='
        option listen_port '55443'
        list addresses '172.22.22.1/24'
        list addresses 'fd8f:de49::1/64'
```

Firewall Setup

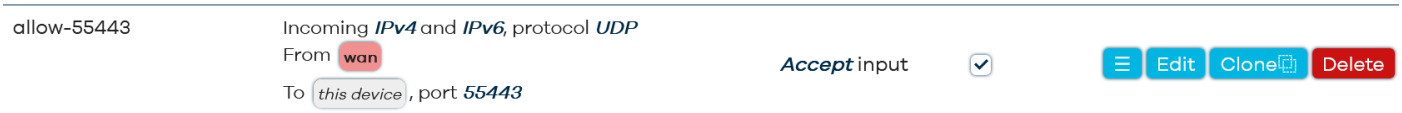
The firewall setup consist of three things:
1. Opening up the port (55443 in this example) with a traffic rule
2a. Allowing traffic for the wgserver the interface
3. Allow IPv6 internet for your wgserver clients, Optional if you have implemented IPv6 and want your cleints to have IPv6 internet

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

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



Save the rule and the result looks like this:



/etc/config/firewall:

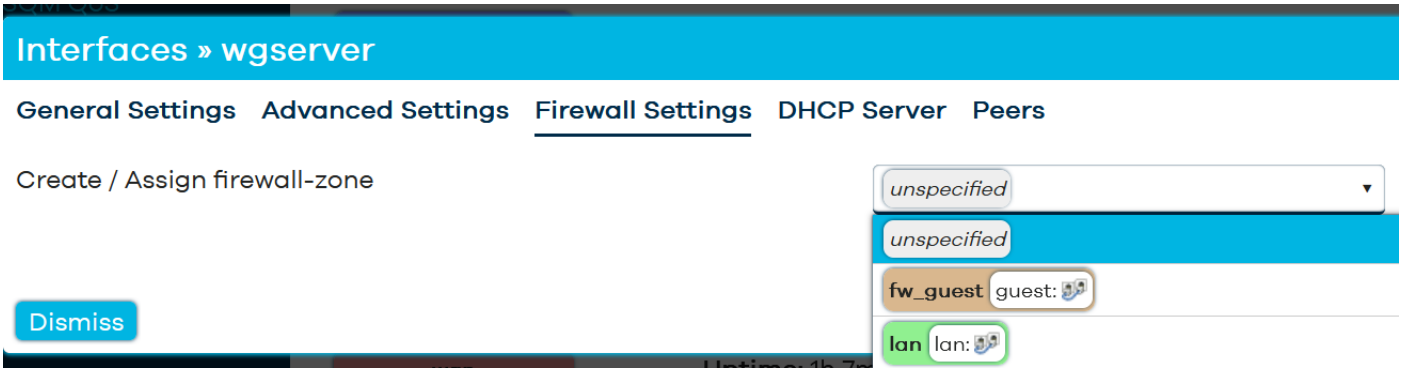
```
config rule
        option name 'allow-55443'
        list proto 'udp'
        option src 'wan'
        option dest_port '55443'
        option target 'ACCEPT'
```

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'
```

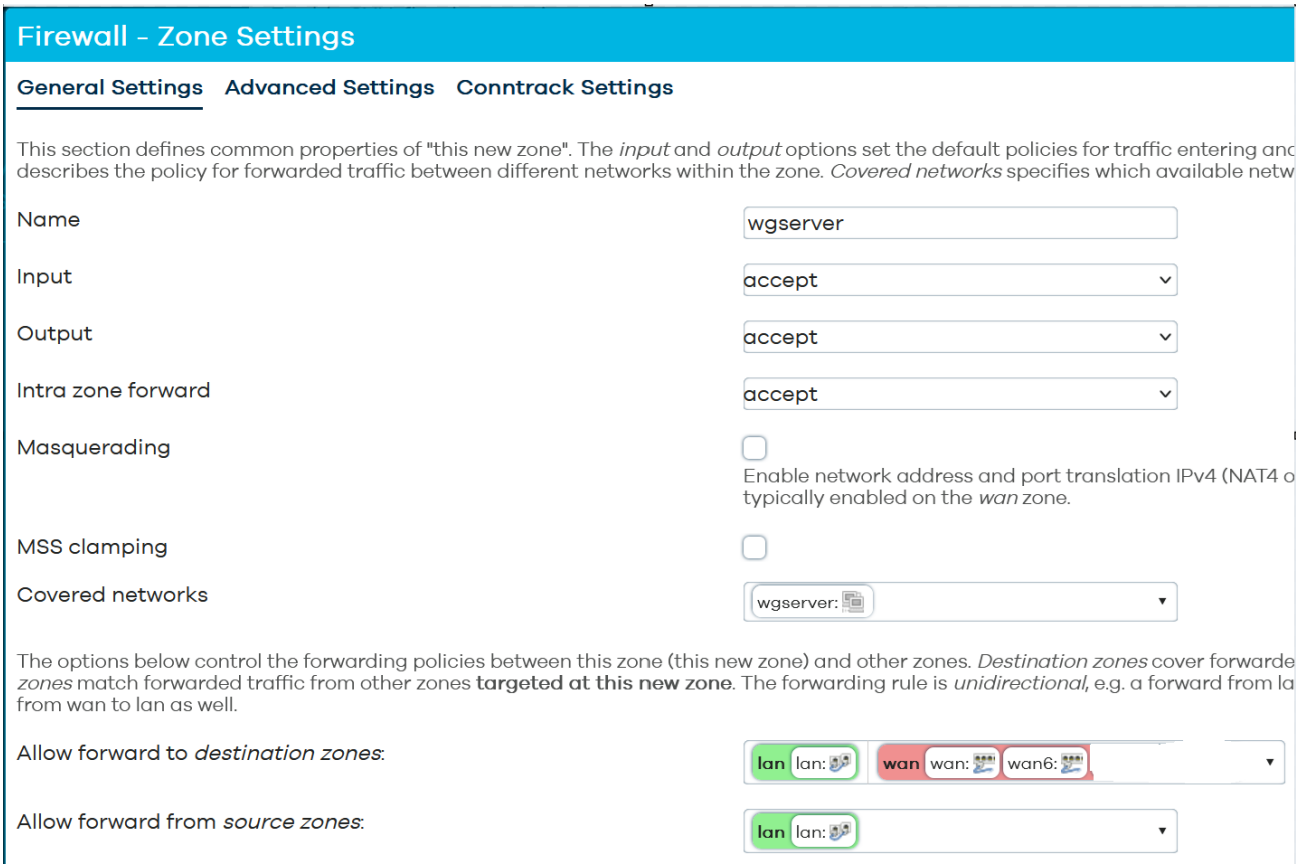
2b. Alternative setup

This can be used if you want a more finer grained control.
Instead of adding the wgserver interface to the LAN zone we crate 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)



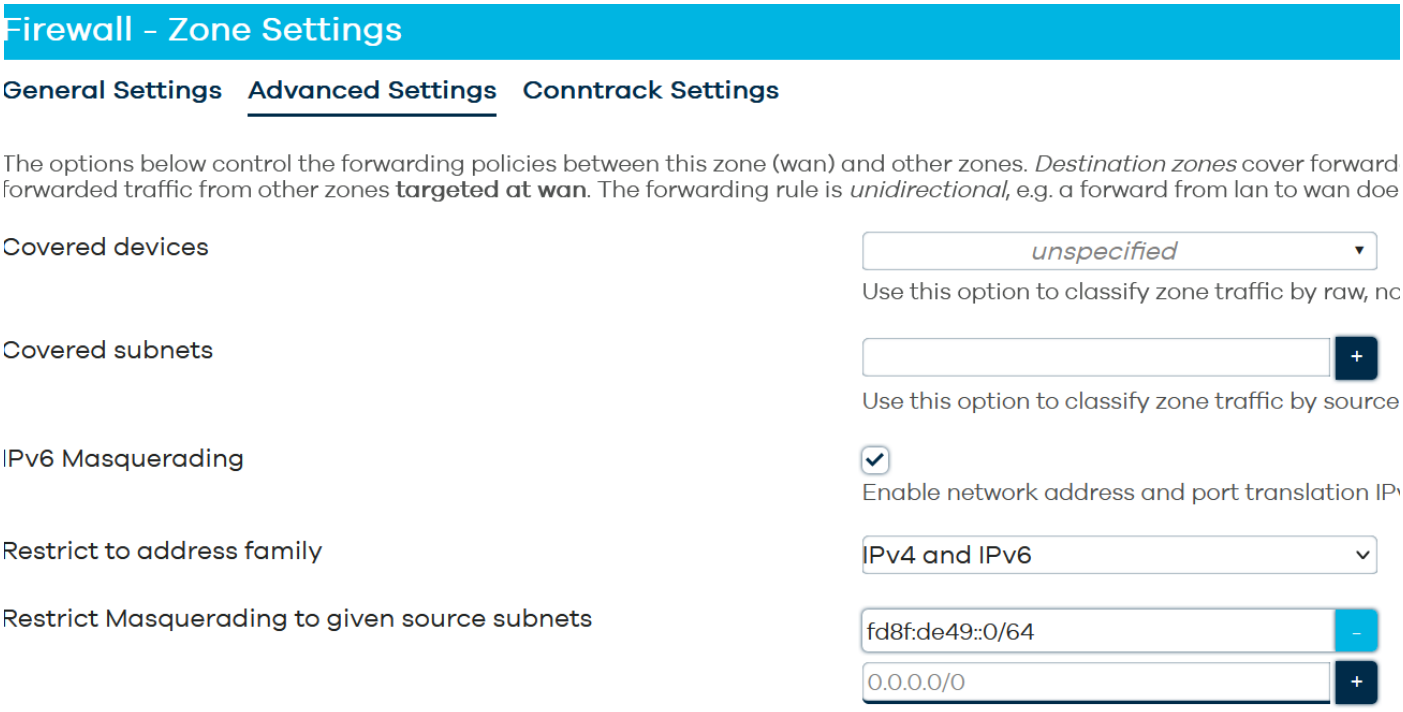

```
/etc/config/firewall:
config zone
        option name 'wgserver'
        option input 'ACCEPT'
        option output 'ACCEPT'
        option forward 'ACCEPT'
        list network 'wgserver'

config forwarding
        option dest 'lan'
        option src 'wgserver'

config forwarding
        option dest 'wgserver'
        option src 'lan'

config forwarding
        option src 'wgserver'
        option dest 'wan'
```

3. Allow IPv6 internet for wgserver clients

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)

Save and **Save & Apply**



/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:

Save and **Save & Apply**

/etc/config/network:

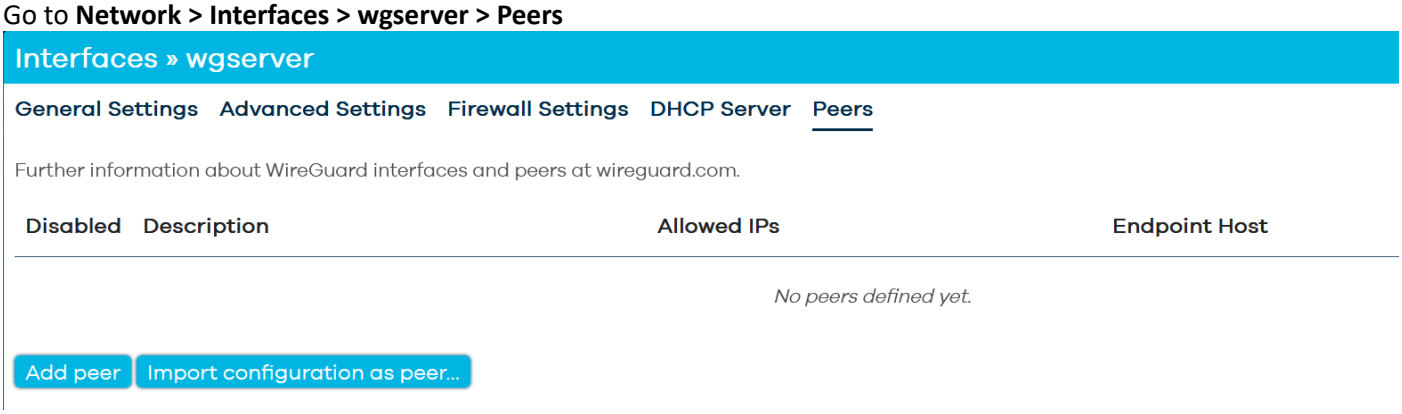
```
config interface 'wan6'
        option device 'wan'
        <>
        option sourcefilter '0'
```

Peer Setup

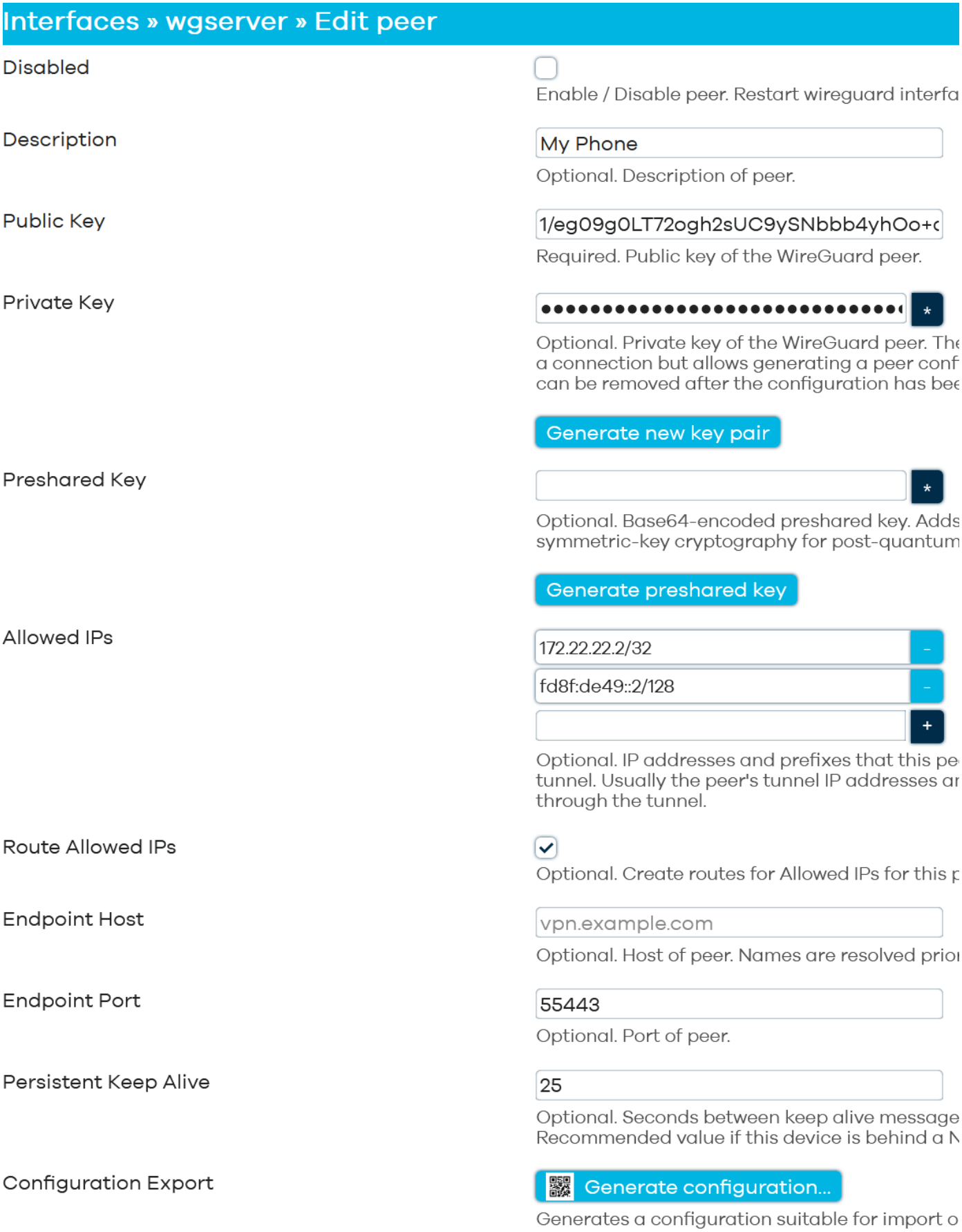
Next setup the peers for the WireGuard server.
Peers are the clients which connects from outside to the wgserver.
There are WireGuard clients for almost operating systems.

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!

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



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 this peer use 172.22.22.2/32, note the /32 mask. Subsequent peers will use .3/32 etc.
For IPv6 you add: *fd8f:de49::2/128*, note the /128 mask
Route Allowed IPs: *Enable*, Always enable this
Endpoint host: Leave blank
Endpoint port: *554433*, 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).

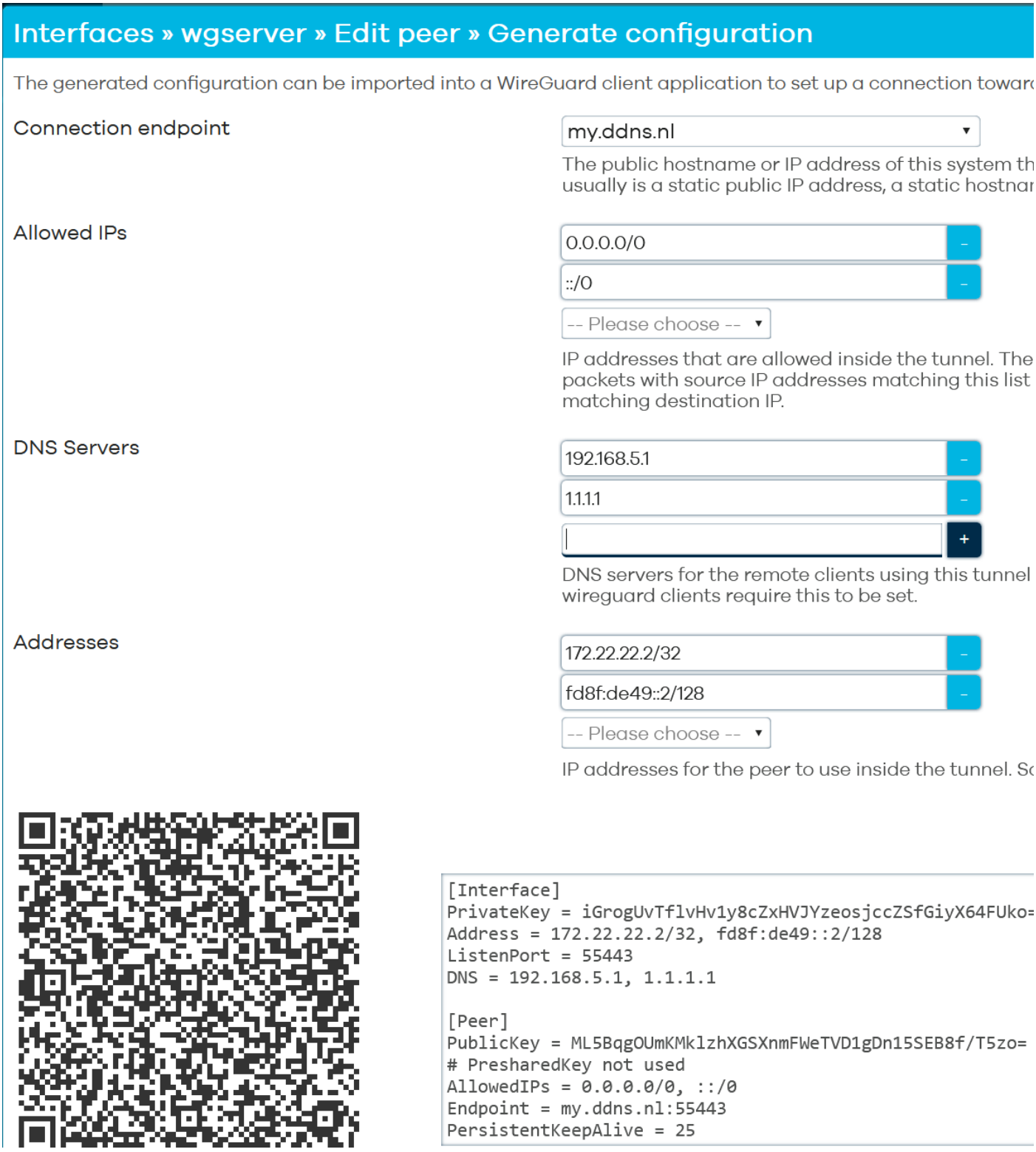


Save

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

Click: *Generate configurations*

Connection Endpoint: this is 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, not all clients can deal with this (rebind protection, using the wgserver interface (172.22.22.1) might help) but you router might also not listen on the wgserver interface so to be sure that you have got DNS resolution use *1.1.1.1*
Addresses: do not change




```
[Interface]
PrivateKey = iGrogUvTflvHv1y8cZxHVJYzeosjccZSfGiyX64FUko=
Address = 172.22.22.2/32, fd8f:de49::2/128
ListenPort = 55443
DNS = 192.168.5.1, 1.1.1.1

[Peer]
PublicKey = ML5BqgOUmKMklzhXGSXnmFWeTVD1gDn15SEB8f/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/eg09g0LT72ogh2sUC9ySNbbb4yhOo+q7kmDaAv1gM='
        option private_key 'iGrogUvTflvHv1y8cZxHVJYzeosjccZSfGiyX64FUko='
        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 via goolge play store , apple store Microsoft store or download from the WireGuard website.
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

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

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*

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 between networks

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

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 form 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 '/home1/192.168.1.1'

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/

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-i2JBtG4FE