

# AI-02045 CGNAT and Tailscale

Owner	© Rigel Arcayan
Tags	Project
Created time	@August 30, 2025 7:49 PM

CGNAT, or Carrier-Grade Network Address Translation, is a technique used by internet service providers (ISPs) to stretch the limited supply of IPv4 addresses by allowing multiple customers to share a single public IP address A B. It's like a digital apartment building where each tenant (customer) has their own private space, but they all share the same front door to the internet.

## 🧠 How CGNAT Works

- Your home network uses private IP addresses (like 192.168.x.x).
  - These are translated by your router using NAT to a public IP.
  - With CGNAT, the ISP adds another layer: it translates many customers' private IPs to one shared public IP, using port numbers to keep traffic sorted.

This setup is often called NAT444:

1. Your device's private IP
2. ISP's private IP
3. Public internet IP

## ⚙️ Why ISPs Use CGNAT

- 🌐 IPv4 exhaustion: There just aren't enough public IPv4 addresses to go around.
  - 💰 Cost-saving: Transitioning to IPv6 is expensive and complex.
  - 🛠️ Operational continuity: CGNAT lets ISPs keep things running while slowly adopting IPv6 C.

## ⚠️ Downsides

- 🚫 Port forwarding issues: Hosting servers or using peer-to-peer apps can be tricky or impossible.

- 🕵️ Reduced transparency: It's harder to trace traffic back to individual users.
- 🧩 Breaks end-to-end connectivity: Which can affect gaming, VoIP, and remote access.

If you're building protocols for autonomous home systems or server hosting, CGNAT can be a real constraint. Want help designing a workaround or testing if your ISP uses CGNAT? I can walk you through it.

### **Can tailscale punch my PLDT network to access my server at home**

Yes, Tailscale can absolutely punch through your PLDT network—even if it's behind CGNAT—and allow you to access your home server remotely. That's one of its superpowers. 🦸

#### Why Tailscale Works with CGNAT

Tailscale uses a mesh VPN built on WireGuard, which means:

- It doesn't rely on port forwarding or public IPs.
  - It establishes peer-to-peer encrypted tunnels using NAT traversal techniques like STUN and relays if needed.
  - It works even when both devices are behind CGNAT (like your PLDT fiber connection).

#### What You'll Need

- Tailscale installed on your home server (can be Linux, Windows, Raspberry Pi, etc.).
  - Tailscale installed on your remote device (laptop, phone, etc.).
  - Optional: Set your home server as a subnet router or exit node if you want broader access to your LAN or route all traffic through home.

#### Bonus: Subnet Routing

If your server is running Tailscale but other devices on your LAN aren't, you can enable subnet routing so your server acts as a gateway to the rest of your home network. This is perfect for accessing printers, NAS, or other non-Tailscale devices.

#### Real-World Use

Users on CGNAT-heavy ISPs like PLDT, Metronet, and others have successfully used Tailscale to bypass the limitations of port forwarding and DDNS A B C.

Would you like help setting up subnet routing or configuring your server as an exit node? I can walk you through it step-by-step.

### **What is CGNAT? What is Tailscale? What is Tailnet? Can Tailscale punch a hole in my PDLT router which uses CGNAT? How to setup Tailscale in Ubuntu and Proxmox?**

CGNAT (Carrier-Grade Network Address Translation) is a method used by internet service providers (ISPs) to share a single public IPv4 address among many customers. Due to the limited supply of IPv4 addresses, this allows ISPs to serve more users without needing a unique public IP for each one. From the user's perspective, this can prevent them from using services that require incoming connections or port forwarding, as the public IP and open ports are shared and managed by the ISP.

#### **What is Tailscale?**

Tailscale is a zero-configuration VPN service that creates a secure, private mesh network between your devices. Built on the open-source WireGuard protocol, it allows devices to connect directly to each other, regardless of their physical location or the network they're on. Instead of routing all traffic through a central server like a traditional VPN, Tailscale establishes encrypted peer-to-peer connections. It's often used for secure remote access, connecting home labs, or providing a seamless network for a distributed team.

#### **What is Tailnet?**

A tailnet is the private network created by Tailscale for an organization or individual. It's the secure, interconnected collection of all the users, devices, and resources that you've added to your Tailscale account. Each device in a tailnet is assigned a unique private IP address from the 100.64.0.0/10 range, which is used for internal communication within the network.

#### **Can Tailscale punch a hole in my PDLT router which uses CGNAT?**

Yes, Tailscale works with CGNAT. This is one of its core strengths. It doesn't "punch a hole" in the traditional sense of port forwarding, which is what CGNAT prevents. Instead, it uses a technique called NAT traversal. Tailscale's control servers help devices find each other and establish a direct connection, even when

both are behind a NAT or CGNAT router. If a direct peer-to-peer connection can't be established, Tailscale automatically falls back to using its worldwide network of relay servers (called DERP relays) to route the traffic. This ensures connectivity even in the most restrictive network environments.

## How to set up Tailscale in Ubuntu and Proxmox

The setup process for both Ubuntu and Proxmox is straightforward, as Proxmox is based on Debian/Ubuntu. The general steps are:

1. Install Tailscale

The recommended and simplest way to install Tailscale on Ubuntu and Proxmox is to use their one-line installation script. This script automatically adds the Tailscale repository and GPG key, then installs the package.

2. On your Ubuntu machine or Proxmox host, open a terminal and run the following command:

```
curl -fsSL https://tailscale.com/install.sh | sh
```

3. Authenticate the Device

After the installation is complete, you need to authenticate the device to your tailnet.

4. In the terminal, run the tailscale up command:

```
sudo tailscale up
```

5. This command will output a unique URL. Copy and paste this URL into a web browser on your computer.

6. Log in to your Tailscale account (if you're not already) and confirm the addition of the new device.

7. The device will now appear in your Tailscale admin console, and it will be part of your tailnet.

8. Proxmox-Specific Considerations

If you want to access the Proxmox Web UI securely over Tailscale, you can use Tailscale's serve functionality.

9. To access the Proxmox web interface without needing to open a port or use an insecure connection, run the following command on your Proxmox host:

```
sudo tailscale serve --bg https+insecure://localhost:8006
```

10. This command tells Tailscale to serve the Proxmox web interface on the default HTTPS port (443) using a valid, automatically generated certificate, making it securely accessible via its Tailscale IP or MagicDNS name. To allow devices on your tailnet to access other devices on your local network, you can configure the Proxmox host as a subnet router.
11. Enable IP forwarding on the Proxmox host by running:

```
echo 'net.ipv4.ip_forward = 1' | sudo tee -a /etc/sysctl.d/99-tailscale.conf
echo 'net.ipv6.conf.all.forwarding = 1' | sudo tee -a /etc/sysctl.d/99-tailscale.conf
sudo sysctl -p /etc/sysctl.d/99-tailscale.conf
```
12. Then, advertise the local subnet to your tailnet when running the tailscale up command, for example:

```
sudo tailscale up --advertise-routes=192.168.1.0/24
```

(Replace 192.168.1.0/24 with your local network's subnet).
13. Finally, go to the Tailscale admin console, find your Proxmox host, and enable the advertised subnet route. This will allow other devices on your tailnet to access all devices on your Proxmox host's local network.

**I have an OpenEMR running in my Ubuntu Server, how to configure so that any users can login to OpenEMR from anywhere in the world even if my PLDT CGNAT is preventing the access.**

That's a great question! It brings up a very common challenge with home servers. We can definitely work through this.

The problem you're facing with your PLDT CGNAT is that it prevents incoming connections, which is what you need to access your OpenEMR server from the outside world. To get around this, we can use a service that helps your server "reach out" to the internet, rather than waiting for the internet to connect to it. One of the most popular and easiest ways to do this is with a mesh VPN, and Tailscale is an excellent choice. It creates a secure, private network for your devices, and it's designed specifically to handle tricky network situations like CGNAT.

Let's break down the concepts so you understand how it all works. First, we need to understand a bit about how your network is set up and what CGNAT does. Can

you describe what you think CGNAT is and how it might be different from a normal home router setup?

### **The CGNAT is preventing me from getting the ip address which the world can see**

That's a perfect way to put it! You've got the core of the problem right there. With CGNAT, your router gets a private IP address from your internet service provider (ISP) that's only useful within their network. The public IP address that the world sees is shared by hundreds, or even thousands, of other users. So, when someone tries to connect to your public IP, the ISP's big router doesn't know which of those thousands of users the request is for, so it just drops the connection. This is why you can't just set up port forwarding.

So, to solve this, we need a way for your Ubuntu server to "call out" and establish a connection to a service on the internet, which then acts as a bridge. This is exactly what Tailscale does.

Here's our plan:

- We'll install Tailscale on your Ubuntu server.
- We'll then have your server join your private Tailscale network, which we call a tailnet.
- We'll then use that tailnet to access your OpenEMR server from your other devices, like your laptop or phone, regardless of where they are in the world. Sound good? Let's start with the first step. Can you tell me if you've already created a Tailscale account?

The video [How to Use Tailscale: Step-by-Step Setup Guide for Beginners](#) provides a comprehensive walkthrough of setting up Tailscale, which is essential for our next steps.

YouTube video views will be stored in your YouTube History, and your data will be stored and used by YouTube according to its Terms of Service

Perfect, Rigel. Here's a Mission Manual-style protocol for hosting a public WordPress site from your home server using Tailscale Funnel, even behind PLDT CGNAT. This guide blends technical precision with resilience logic—ideal for your autonomy framework.

---

 Protocol: Public WordPress Hosting via Tailscale Funnel (CGNAT-Resilient)

## Objective

Enable anonymous public access to a self-hosted WordPress site on a home server behind CGNAT using Tailscale Funnel.

---

## Prerequisites

Component	Description
Home server	Linux (Debian/Ubuntu preferred), running WordPress stack (Apache/Nginx + PHP + MySQL)
Tailscale account	Free Personal tier (up to 3 users, 100 devices)
Tailscale CLI	Installed and updated to v1.38.3 or later
Domain (optional)	Funnel provides a .ts.net subdomain; custom domain not required

---

## Step-by-Step Deployment

### 1. Install Tailscale on Your Server

```
curl -fsSL https://tailscale.com/install.sh | sh  
sudo tailscale up
```

Log in via browser when prompted. Confirm device appears in your Tailscale admin panel.

---

### 1. Enable Funnel in Tailnet Policy

Edit your tailnet policy file ( `ACLs` ) to allow Funnel access:

```
{  
"funnel": {  
"enabled": true  
}  
}
```

You can do this via the admin console under Access Controls.

---

### 1. Start Your WordPress Server

Ensure Apache or Nginx is serving your WordPress site locally on port 80 or 443.

Example for Apache:

```
sudo systemctl start apache2
```

---

## 1. Expose Site via Funnel

Use the Tailscale CLI to publish your site:

```
sudo tailscale funnel enable
sudo tailscale serve --https=443 /proxy/localhost:80/
```

This creates a public HTTPS endpoint like:

<https://your-server-name.ts.net>

---

## 1. Test Public Access

From any browser (outside your tailnet), visit the `.ts.net` URL. You should see your WordPress homepage.

---

### Security Notes

- Tailscale handles TLS certs automatically.
  - No need to open ports or configure PLDT router.
  - You can disable Funnel anytime via CLI:

```
sudo tailscale funnel disable
```

---

### Optional Enhancements

Feature	Purpose	Command	Example
---------	---------	---------	---------

Custom domain	Use Cloudflare to proxy .ts.net to your own domain	DNS CNAME setup	
---------------	--	-----------------	--

Exit node fallback	Route traffic through another Tailscale device	<code>tailscale up --exit-node=...</code>	
--------------------	--	---	--

Subnet routing	Access other LAN devices via server gateway	<code>tailscale up --advertise-routes=192.168.0.0/24</code>	
----------------	---	---	--

---

### Resilience Logic

If Funnel fails (e.g., CLI bug or policy misconfig):

- Fallback to Cloudflare Tunnel or Tunnelmole
    - Use ZeroTier for peer-to-peer access (not anonymous)
    - Document failure modes and recovery steps in Mission Manual
-

Would you like this protocol formatted into a checklist or embedded into your Mission Manual index? I can also help you test the `.ts.net` endpoint or simulate failure scenarios.