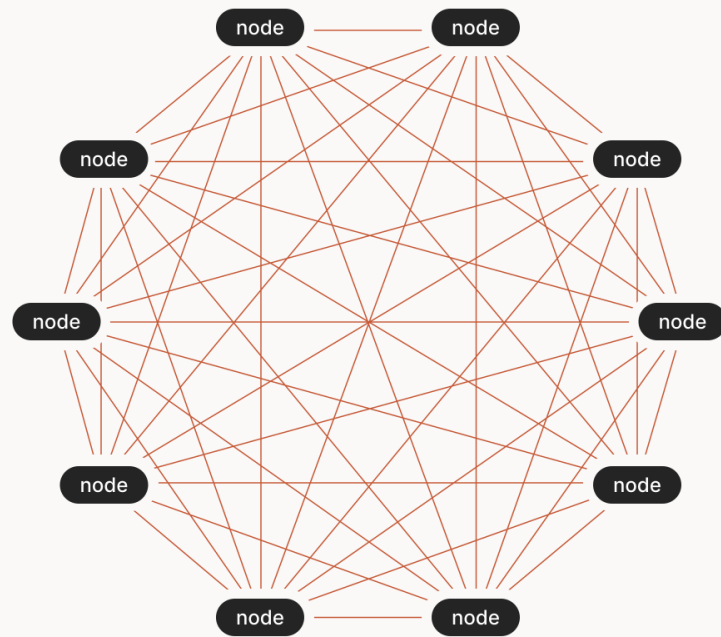


Setting up Tailscale VPN In My Home Network



About Tailscale

Tailscale utilizes a mesh VPN creating a private, encrypted network, what they call a *tailnet*. Tailscale spans all devices, regardless of the physical location. Tailscale employs WireGuard, a modern VPN protocol, which handles the encrypted transport between devices. In contrast to a traditional VPN where every packet is routed through a central concentrator, Tailscale devices are connected to each other, forming a peer-to-peer mesh.



$$n(n-1) = 90 \text{ WireGuard endpoints (for 45 connections)}$$

A coordination server distributes public keys and access policies allowing machines to find each other and traverse NAT.

NOTE: *The encrypted traffic itself flows between devices, and never touches Tailscale's infrastructure.*

As machines join the tailnet, authentication is handled through an identity provider (e.g., Google, Microsoft, Okta, GitHub, etc.) rather than sharing a static key or password, so access is tied to a user account that can be granted or revoked centrally. ACL policy is configurable to define which users & machines are allowed to reach which devices & ports.

As a result, the network behaves like a single LAN with capability to connect servers, laptops, VMs, cloud instances all within one accessible network, from anywhere in the world.

Initial Setup

Within Proxmox Virtual Environment, I created a Linux container (LXC) as my initial tailscale node— *tailscaleprd1*. For the LXC specifications, I provisioned the LXC with 4 cores and 8GB RAM, and 8GB of disk space.

Container 1000 (tailscaleprd1) on node 'pve1' vpn

Summary

> Console

Resources

Network

DNS

Options

Task History

Backup

Replication

Snapshots

Firewall

Permissions

Add

Edit

Remove

Volume Action

Revert

Memory	8.00 GiB
Swap	8.00 GiB
Cores	4
Root Disk	local:1000/vm-1000-disk-0.raw,size=8G

TUN Access & Tailscale Interface Creation

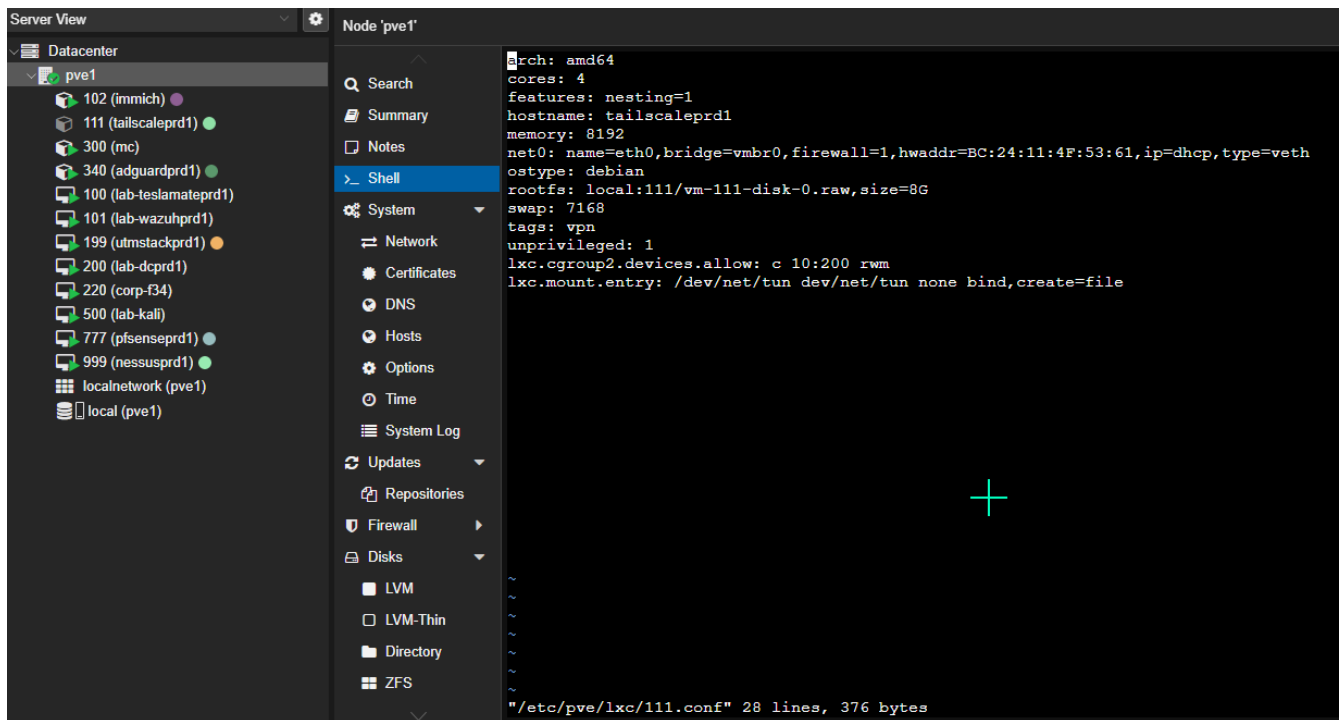
By default, the unprivileged LXC doesn't expose the host's `/dev/net/tun` device, which is required for Tailscale to create the virtual interface for the VPN tunnel. The following settings, this will grant permission to the TUN device, and mount the interface allowing Tailscale to reach it.

To do this, through Proxmox's host node shell, use the following command to edit the `.conf` of the LXC:

```
root@pve1:~# nano /etc/pve/lxc/[PCT ID].conf
```

Add & save the following lines:

```
lxc.cgroup2.devices.allow: c 10:200 rwm
lxc.mount.entry: /dev/net/tun dev/net/tun none bind,create=file
```



DHCP Reservation

Though not required, I configured a DHCP reservation within my router to reserve the IP address assigned to *tailscaleprd1*.

Grabbing the machines IP:

```
rhunt@tailscaleprd1:~$ ip a
2: eth0@if81: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc noqueue state UP
    link/ether bc:24:11:54:92:5b brd ff:ff:ff:ff:ff:ff link-netnsid 0
    inet 192.168.1.196/24 brd 192.168.1.255 scope global dynamic eth0
        [REDACTED]
```

Configuring DHCP reservation within ExperienceOS:

DHCP RESERVATION

DHCP Reservation

DHCP reservation leases a permanent DHCP allocated address to a client.

Select Device or manually enter a MAC address

☒ Device
 ☐ MAC Address

tailscaleprd1(192.168.1.196)

Input an IP address (192.168.1.100~192.168.1.249) to associate with a MAC address

IP Address

192.168.1.196

Apply

DHCP Reservation List (4)

Details of all current DHCP leases assigned to attached clients are displayed in the table below.

Device Name	MAC Address	IP Address
adguardprd1(192.168.1.104)	bc:24:11:78:1d:38	192.168.1.104
RE505 Extender(192.168.1.126)	bc:c4:93:ee:8e:49	192.168.1.126
	bc:24:11:21:06:47	192.168.1.157
nessusprd1(192.168.1.211)	bc:24:11:35:8e:6b	192.168.1.211

Install Tailscale

Now that TUN access is granted, and the tailscale interface is created, we can spin up tailscale in the *tailscaleprd1* node with one command: **NOTE: This command is for Linux machines only.**

```
rhunt@tailscaleprd1:~$ curl -fsSL https://tailscale.com/install.sh | sh
```

```

Reading state information... Done
The following NEW packages will be installed:
  sudo
0 upgraded, 1 newly installed, 0 to remove and 0 not upgraded.
Need to get 1890 kB of archives.
After this operation, 6199 kB of additional disk space will be used.
Ign:1 http://deb.debian.org/debian bookworm/main amd64 sudo amd64 1.9.13p3-1+deb12u2
Get:1 http://deb.debian.org/debian bookworm/main amd64 sudo amd64 1.9.13p3-1+deb12u2 [1890 kB]
Fetched 1890 kB in 0s (3987 kB/s)
Selecting previously unselected package sudo.
(Reading database ... 19157 files and directories currently installed.)
Preparing to unpack .../sudo_1.9.13p3-1+deb12u2_amd64.deb ...
Unpacking sudo (1.9.13p3-1+deb12u2) ...
Setting up sudo (1.9.13p3-1+deb12u2) ...
Processing triggers for man-db (2.11.2-2) ...
Processing triggers for libc-bin (2.36-9+deb12u13) ...
root@tailscaleprd1:~# ^[[200~curl -fsSL https://tailscale.com/install.sh | sh~^C
root@tailscaleprd1:~# curl -fsSL https://tailscale.com/install.sh | sh
Installing Tailscale for debian bookworm, using method apt
+ mkdir -p --mode=0755 /usr/share/keyrings
+ curl -fsSL https://pkgs.tailscale.com/stable/debian/bookworm.noarmor.gpg
+ tee /usr/share/keyrings/tailscale-archive-keyring.gpg
+ chmod 0644 /usr/share/keyrings/tailscale-archive-keyring.gpg
+ curl -fsSL https://pkgs.tailscale.com/stable/debian/bookworm.tailscale-keyring.list
+ tee /etc/apt/sources.list.d/tailscale.list
# Tailscale packages for debian bookworm
deb [signed-by=/usr/share/keyrings/tailscale-archive-keyring.gpg] https://pkgs.tailscale.com/stable/debian bookworm main
+ chmod 0644 /etc/apt/sources.list.d/tailscale.list
+ apt-get update
Get:1 http://deb.debian.org/debian bookworm InRelease [151 kB]
Get:2 http://security.debian.org bookworm-security InRelease [48.0 kB]
Get:3 http://deb.debian.org/debian bookworm-updates InRelease [55.4 kB]
Get:4 https://pkgs.tailscale.com/stable/debian bookworm InRelease
Get:5 http://security.debian.org bookworm-security/main amd64 Packages [312 kB]
Get:6 http://security.debian.org bookworm-security/main Translation-en [191 kB]
Get:7 http://security.debian.org bookworm-security/contrib Translation-en [652 B]

```

Once installation is complete, you can run the following command to invoke the tailscale process `tailscale up --ssh`

```
rhunt@tailscaleprd1:~$ tailscale up --ssh
```

To authenticate, visit

[https://login.tailscale.com/a/\[REDACTED\]](https://login.tailscale.com/a/[REDACTED])

Visit the designated URL and authenticate through your desired Identity Provider (IdP).



Log in to connect a device to your tailnet.

Sign in

OR

 Sign in with Google

 Sign in with Microsoft

 Sign in with GitHub

 Sign in with Apple

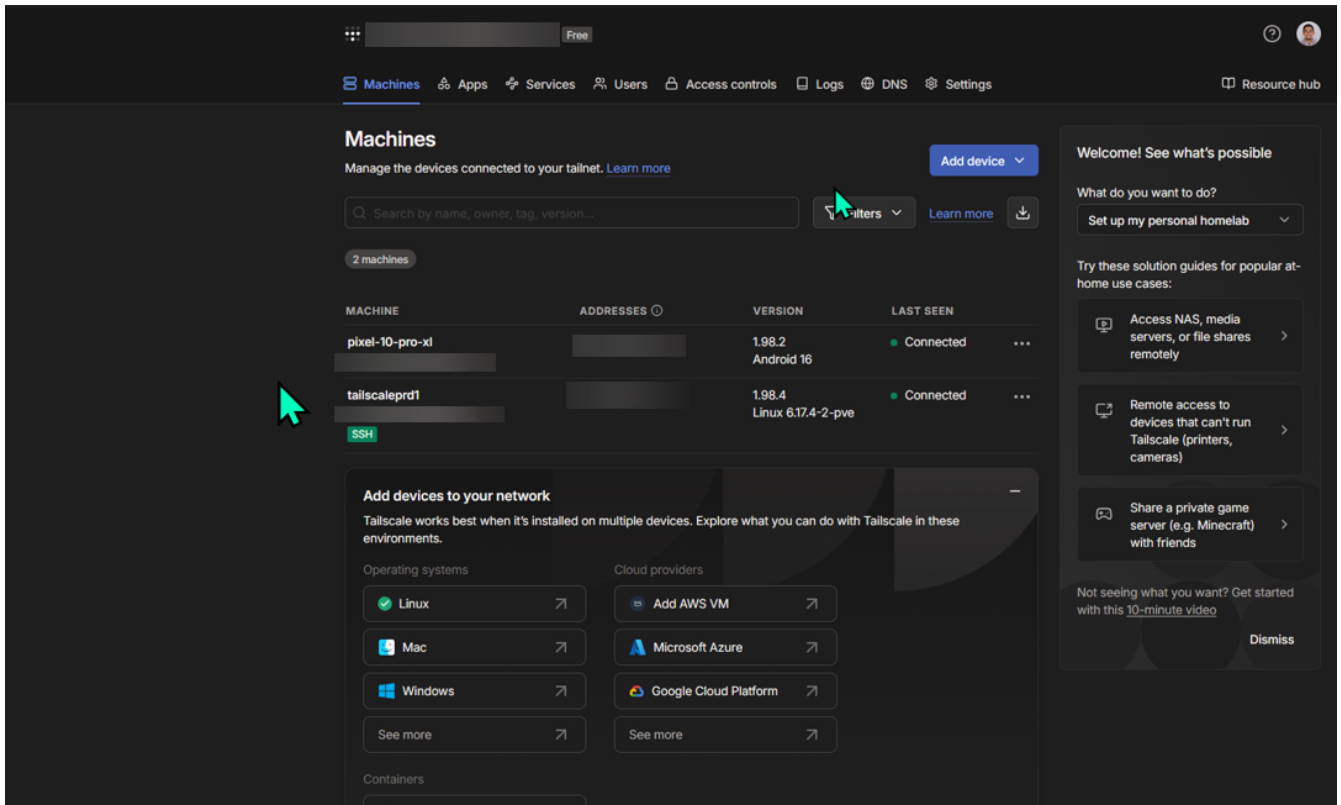
 Sign in with a passkey

Alternatively, use a [QR code](#).

First time? Learn more at tailscale.com.

By clicking the buttons above, you acknowledge that you have read, understood, and agree to Tailscale's [Terms of Service](#) and [Privacy Policy](#).

Once authenticated, you will see a similar screen to connect a second device.



In my case, I wanted VPN access to my homelab, specifically my Proxmox web GUI, so I deployed tailscale on my Proxmox host node— `pve1`. We can do this by repeating the same install command as before.

```

+ Mobaxterm Personal Edition v25.4 +
+ (SSH client, X server and network tools) +

SSH session to root@192.168.1.251
+ Direct SSH : ✓
+ SSH compression : ✓
+ SSH-browser : ✓
+ X11-forwarding : ✗ (disabled or not supported by server)
+ For more info, ctrl+click on help or visit our website.

Linux pve1 6.17.4-2-pve #1 SMP PREEMPT_DYNAMIC PMX 6.17.4-2 (2025-12-19T07:49Z) x86_64

The programs included with the Debian GNU/Linux system are free software;
the exact distribution terms for each program are described in the
individual files in /usr/share/doc/*/*copyright.

Debian GNU/Linux comes with ABSOLUTELY NO WARRANTY, to the extent
permitted by applicable law.
Last login: Mon May 25 10:57:56 2026 from 192.168.1.213
root@pve1:~# curl -fsSL https://tailscale.com/install.sh | sh
Installing Tailscale for debian trixie, using method apt
+ mkdir -p -mode=0555 /usr/share/keyrings
+ curl -fsSL https://pkgs.tailscale.com/stable/debian/trixie-noarmor.gpg
+ tee /usr/share/keyrings/tailscale-archive-keyring.gpg
+ chmod 0644 /usr/share/keyrings/tailscale-archive-keyring.gpg
+ curl -fsSL https://pkgs.tailscale.com/stable/debian/trixie-tailscale-keyring.list
+ tee /etc/apt/sources.list.d/tailscale.list
# Tailscale packages for debian trixie
deb [signed-by=/usr/share/keyrings/tailscale-archive-keyring.gpg] https://pkgs.tailscale.com/stable/debian trixie main
+ chmod 0644 /etc/apt/sources.list.d/tailscale.list
+ apt-get update
Hit:1 http://deb.debian.org/debian trixie InRelease
Get:2 http://deb.debian.org/debian trixie-updates InRelease [47.3 kB]
Get:3 https://security.debian.org/debian-security trixie-security InRelease [43.4 kB]
Get:4 https://pkgs.tailscale.com/stable/debian trixie InRelease
Get:5 https://security.debian.org/debian-security trixie-security/main amd64 Packages [215 kB]
Get:6 https://pkgs.tailscale.com/stable/debian trixie/main amd64 Packages [15.0 kB]
Get:7 https://security.debian.org/debian-security trixie-security/main Translation-en [130 kB]
Get:8 https://pkgs.tailscale.com/stable/debian trixie/main all Packages [354 B]
Hit:9 https://download.proxmox.com/debian/pve trixie InRelease
Fetched 458 kB in 5s (86.7 kB/s)
Reading package lists... Done
+ [ -n ]
+ apt-get install -y tailscale tailscale-archive-keyring
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
The following NEW packages will be installed:
  tailscale tailscale-archive-keyring
0 upgraded, 2 newly installed, 0 to remove and 172 not upgraded.
Need to get 37.1 MB of archives.
After this operation, 73.0 MB of additional disk space will be used.
Get:1 https://pkgs.tailscale.com/stable/debian trixie/main amd64 tailscale amd64 1.98.4 [37.1 MB]
Get:2 https://pkgs.tailscale.com/stable/debian trixie/main all tailscale-archive-keyring all 1.35.181 [3,082 B]

```

After authenticating the second device to the tailnet, you now have an encrypted WireGuard tunnel.

Subnet Router vs. Adding Individual Devices

Initially, I kept repeating the process by adding more devices to the tailscale, however I realized this wasn't my goal— my goal is to have an encrypted VPN tunnel to my entire network, where all devices can communicate with each other regardless of physical location.

After researching further into Tailscale capabilities, I came across [Tailscale Subnet Routers](#), which provided what I was looking for.

Per Tailscale documentation, Tailscale subnet routers allow you to extend your tailnet to include devices that don't or can't run the Tailscale client, acting as a gateway between your tailnet and physical subnets, enabling secure access to legacy devices, entire networks, or services without installing Tailscale everywhere.

Setting up Subnet Router within Tailnet

To achieve the subnet router functionality within my tailnet, you must enable IP forwarding within the kernel. To do so, you can execute the following commands in your initial tailscale node (in my case, *tailscaleprd1*).

```
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
```

```
root@tailscaleprd1:~# echo 'net.ipv4.ip_forward = 1' | tee -a /etc/sysctl.d/99-tailscale.conf
echo 'net.ipv6.conf.all.forwarding = 1' | tee -a /etc/sysctl.d/99-tailscale.conf
sysctl -p /etc/sysctl.d/99-tailscale.conf
net.ipv4.ip_forward = 1
net.ipv6.conf.all.forwarding = 1
bash: sysctl: command not found
root@tailscaleprd1:~# sysctl
bash: sysctl: command not found
root@tailscaleprd1:~# tailscale set --advertise-routes=192.168.1.0/24
root@tailscaleprd1:~# tailscale status
100.87.231.53 tailscaleprd1 ryananthunt@ linux -
100.104.206.29 pixel-10-pro-xl ryananthunt@ android -
100.84.239.72 pve1 ryananthunt@ linux -
100.109.57.65 win11 ryananthunt@ windows -

# Health check:
# - Subnet routing is enabled, but IP forwarding is disabled. Check that IP forwarding is enabled on your machine.
root@tailscaleprd1:~#
```

After executing the IP forwarding settings to `etc/sysctl.d/99-tailscale.conf` and running `sysctl -p` to apply them returned a command not found error and a warning that IP forwarding is disabled. This was due to `sysctl` residing in `/usr/sbin`, which wasn't included in the shell's `$PATH`. To fix this, I executed the absolute path to apply

the settings:

```
/sbin/sysctl -p /etc/sysctl.d/99-tailscale.conf
```

To validate the output of `/etc/sysctl.d/99-tailscale.conf` and ensure the IP forwarding settings applied correctly, you can run the following commands, which should return a value of 1.

```
rhunt@tailscaleprd1:~$ cat /proc/sys/net/ipv4/ip_forward
cat /proc/sys/net/ipv6/conf/all/forwarding
1
1
```

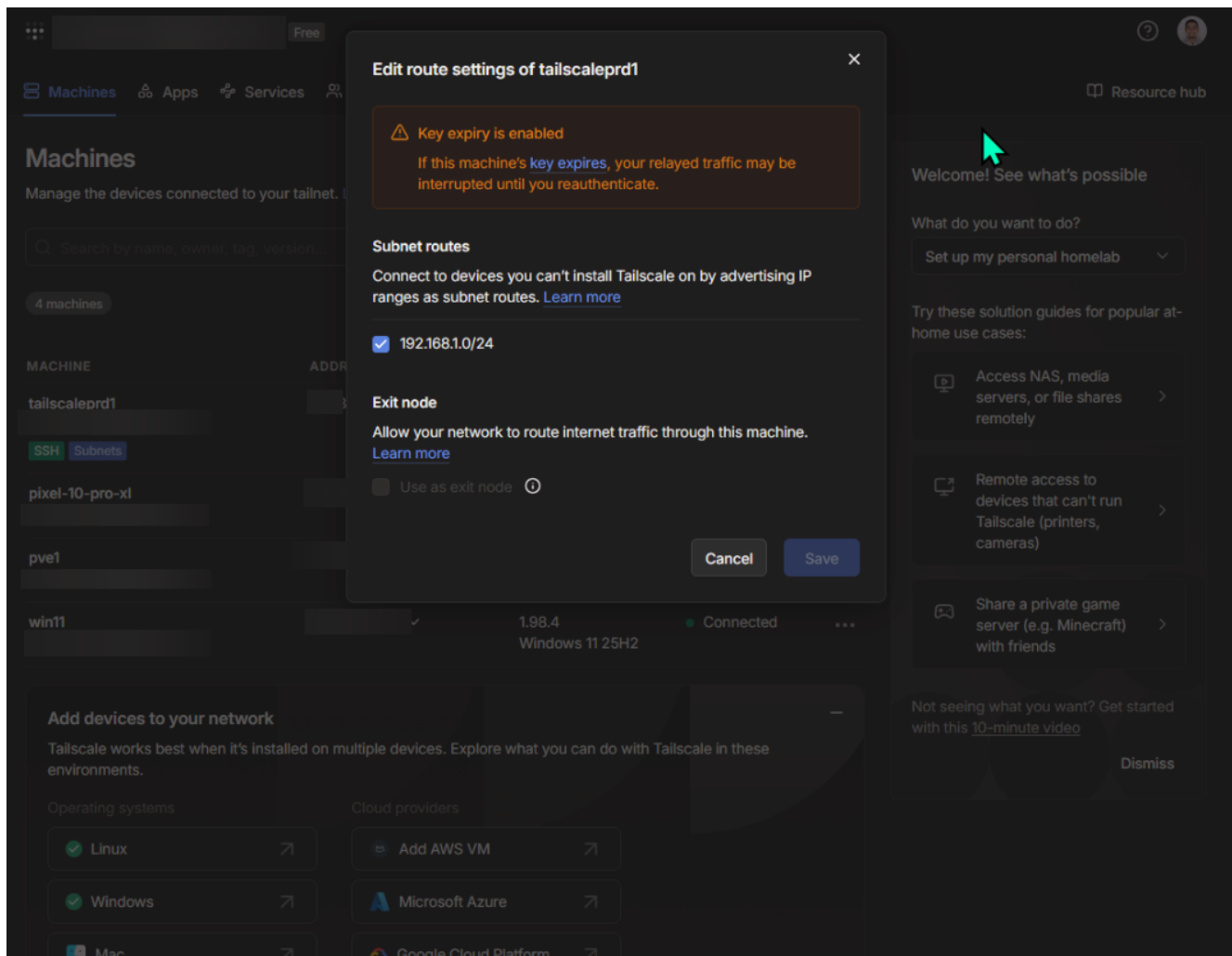
After IP forwarding settings are set, you must advertise your subnet routes via the following command:

```
sudo tailscale set --advertise-routes=192.0.2.0/24,198.51.100.0/24
```

Make sure to replace the command with active subnets in your own environment.

Once done, you can verify the status of tailscale through `tailscale status`.

Lastly, the IP route must be approved within the Tailscale Admin Center.



As the route is now approved, the tailnet is now advertising the subnet `192.168.1.0/24` network to the tailnet and forwards traffic to it. Devices within the tailnet can now reach hosts on that LAN as if they were local, regardless of the location of the physical device *without requiring Tailscale running themselves*.

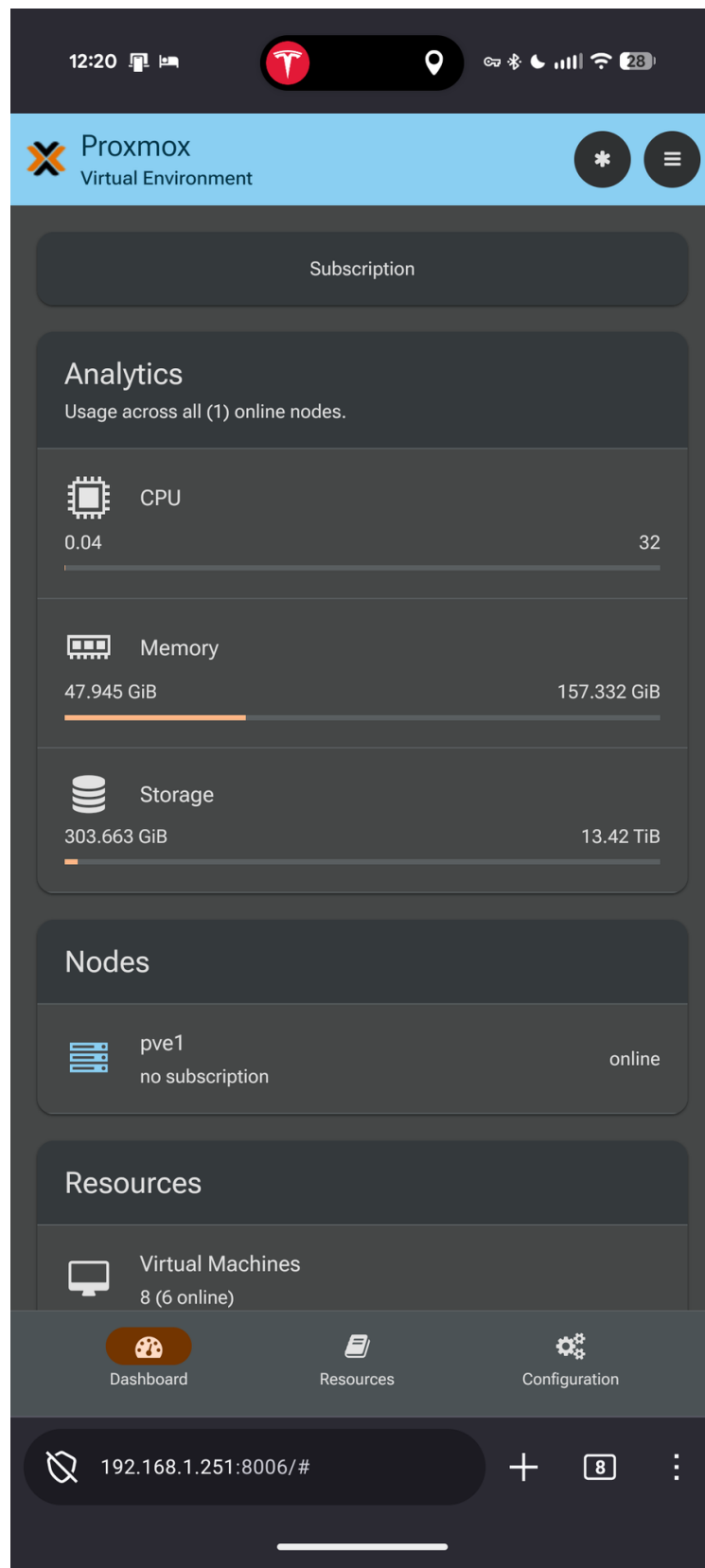
Custom DNS — AdGuard

A side benefit of routing traffic through the tailnet is that devices connected remotely also use my home DNS server, [AdGuard](#), for ad and tracker blocking — the same filtering that applies on the local network follows you wherever you connect from.

This is configured in the Tailscale Admin Center under DNS → Custom Nameservers, pointing to the LAN IP of the AdGuard instance.

Accessing Devices via Mobile

Now that the tailnet can access any device on the network, I tested this out by installing and connecting to my tailnet via my phone, which I was able to access my Proxmox Web GUI.



Windows RDP Workaround

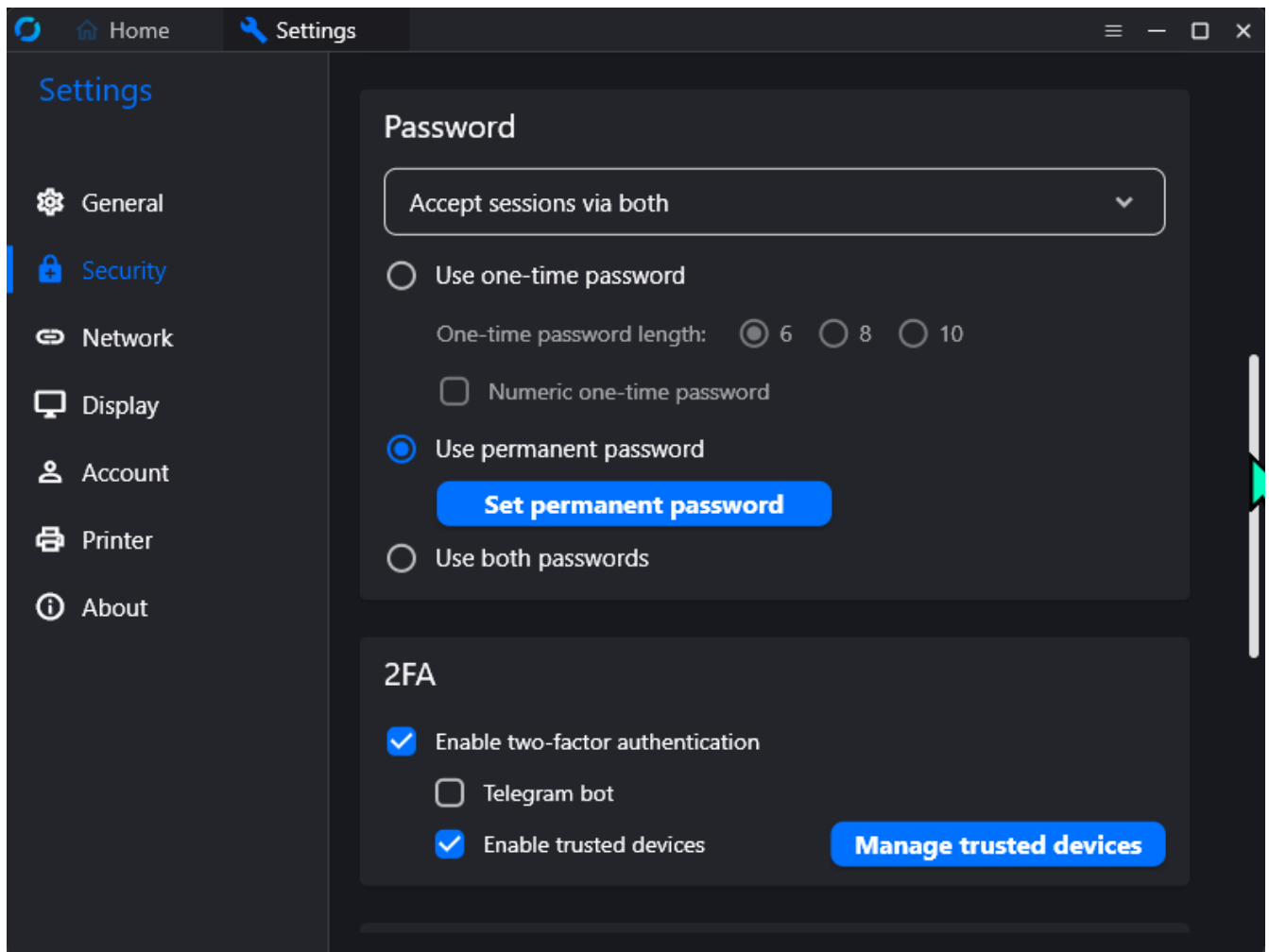
One of the features I was looking to gain was being able to control my Windows PC remotely, which is possible through Tailscale. However, this is not possible with Windows 11 Home Edition as it does not support incoming Remote Desktop Connections.

So, I explored some other solutions and stumbled upon [RustDesk](#), which is an open-source remote control alternative popular for self-hosting and security.

I installed the RustDesk .msi on both my Windows computers, and started getting into the security settings.

RustDesk Security Settings

As this needs to work remotely, the first thing I did was set a strong permanent unattended password and set up 2FA for any incoming connections.



Additionally, I enabled Direct IP access over a port I specified to allow for connecting directly to the machine IP or tailnet assigned IP.

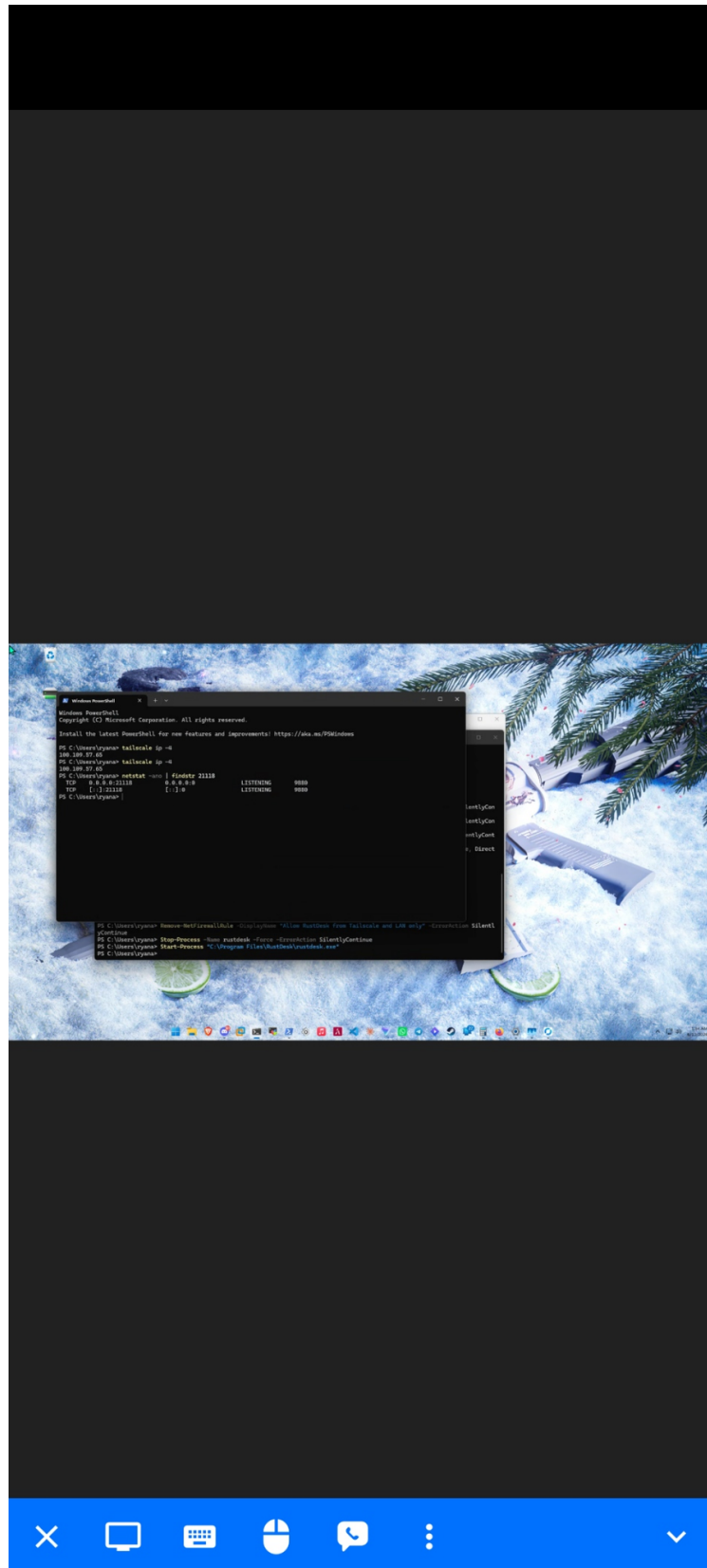
Limiting IP Ranges - Firewall

Even though 2FA is enabled for any incoming connection, I wanted an implicit deny for any incoming RustDesk connections from outside my 192.x network, and my 100.x tailscale network. To do this, I set up a Windows Firewall Rule via PowerShell using the following command:

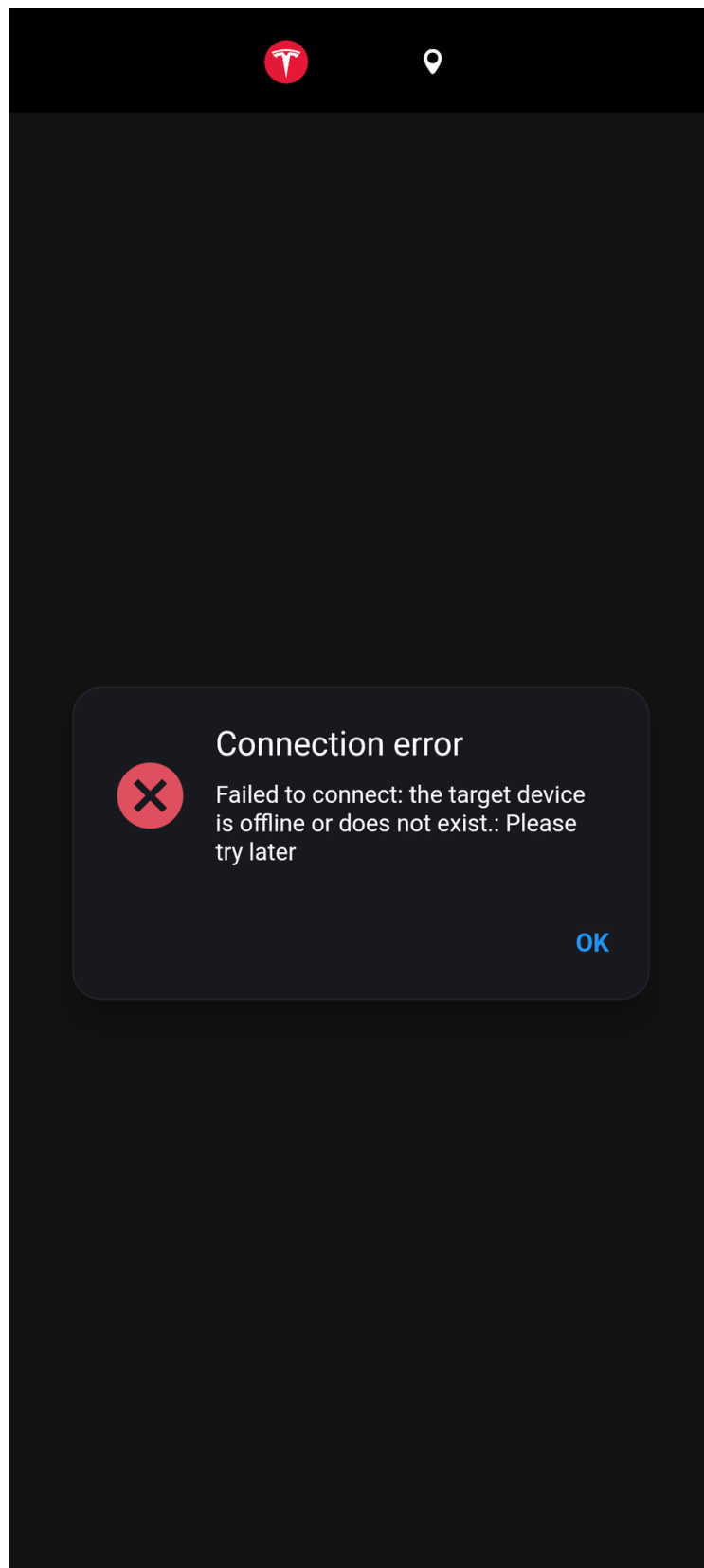
```
PS C:\Users\ryana> New-NetFirewallRule `
>> -DisplayName "Allow RustDesk Direct IP from Tailscale and LAN" `
>> -Direction Inbound `
>> -Action Allow `
>> -Protocol TCP `
>> -LocalPort 21118 `
>> -RemoteAddress 100.64.0.0/10,192.168.1.0/24
```

This rule effectively blocks any connection outside of the 100.x and 192.x networks.

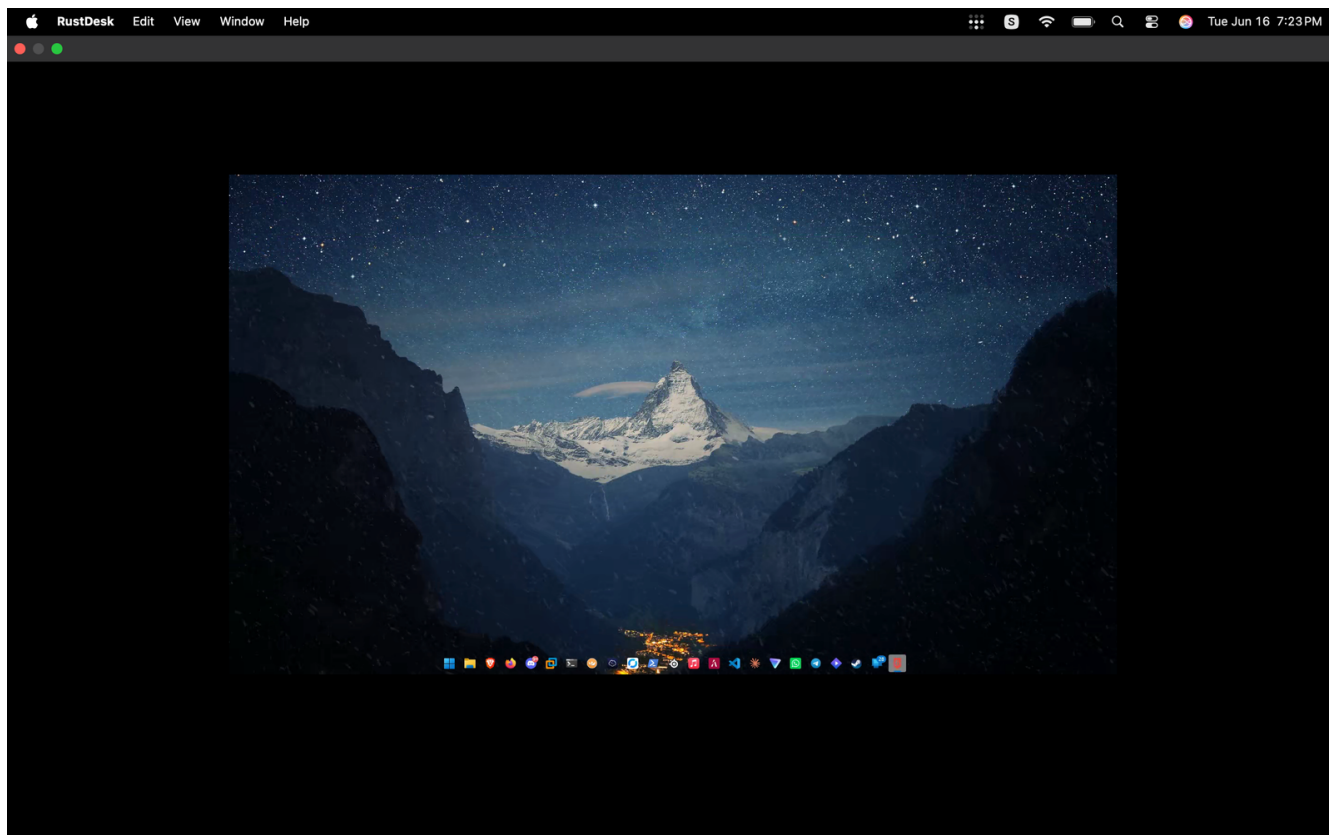
To test this and ensure it worked, I connected to Tailscale on my phone, and RDP'd into my main PC, which was allowed through the firewall.



And, to ensure it is blocking unknown incoming connections appropriately, I disconnected from the Tailscale VPN on my phone, disconnected from WiFi, and attempted the RustDesk connection, which was effectively blocked.



Accessing my Windows 11 PC via RustDesk:



Closing Thoughts

There's a lot left to explore within Tailscale, but ultimately I'm really happy with setting up this VPN to access my home network, limit incoming RDP/RustDesk connections.



hunts.dev

Originally published on hunts.dev

<https://hunts.dev/blog/tailscale/>