Introducing Tailscale Funnel

Brad Fitzpatrick, Maisem Ali, Sonia Appasamy, Anton Tolchanov and Shayne Sweeney on November 17, 2022

Tailscale lets you put all your devices on their own private tailnet so they can reach each other, ACLs permitting. Usually that's nice and comforting, knowing that all your devices can then be isolated from the internet, without any ports needing to be open to the world.

Sometimes, though, you need something from the big, scary, non-Tailscale internet to be able to reach your device.

Maybe you need to receive a webhook from GitHub. Maybe you want to briefly test a website you're working on using a coworker's phone. Or maybe you even want to host your personal blog or a small Mastodon server on your own computer.

For any of that to work, though, you'll need an address the other parties can access. Shockingly, the whole world doesn't use Tailscale. (We're working on that.) So you'll need a publicly routable IP address, a TLS cert (hopefully!), and then necessarily a DNS name for the cert. Tailscale gives you a DNS name and supports your Tailscale node getting its own Let's Encrypt cert for that DNS name, but your Tailscale IP addresses aren't publicly routable, so those webhooks from GitHub or ActivityPub toots can't reach you. You're not really on the internet without a public IP address.

Yes, you could spin up a $5/month VM somewhere and forward a port from its public internet IP to your tailnet with one line in your `rinetd.conf` file. But is that fun? Do you really need a(nother) Linux VM in your life?

Here's something more fun: Tailscale Funnel. You can now expose things from your Tailscale node to the big scary internet and we'll tunnel it in to you, over Tailscale.

How it works

"My VPN is exposed to the internet!!?" we hear you screaming. We're also the worrying sort, so let's walk through how it works. Hopefully you'll find it less scary.

First off, rest assured that Tailscale Funnel is all off by default and double opt-in: It needs to be both enabled in the Tailscale admin console by a tailnet admin and enabled on the device running Tailscale.

When enabled, two things happen:

The first thing we do is set up public DNS records for your `node.tailnet.ts.net` MagicDNS name to point to public IP addresses of new servers we're now running. These new Funnel frontends (*fun*ends?) are georeplicated around the world, similar to how we run DERP servers around the world. Tailscale Funnel runs on distinct services, VMs, and networks from DERP, but they're similar in that they're both hosted by Tailscale. (Like DERP, which you can run yourself, you could also do an `rinetd` thing yourself for this, if you find that more fun.)

The second thing we do is add those Funnel ingress nodes to your tailnet's list of Tailscale peers. On nodes where Tailscale Funnel is enabled you'll see them in `tailscale serve status --json`. Those peers will be named `funnel-ingress-node` and are sent with a bit set marking them as funnel peers. That bit prevents them from having any packet-level access to your tailnet. The only thing they're allowed to do is offer your node a funneled TCP connection, which your node can accept or reject, depending on how it's configured.

(That magic bit will make an appearance in a future blog post; stay tuned!)

The way the Funnel ingress nodes are allowed to send a connection offer to your nodes is using Tailscale's inter-node "peerapi" mechanism that we originally added for Taildrop. With peerapi, each Tailscale node allocates a reserved ephemeral port number to be its inter-node RPC port. Those peerapi RPCs are then just HTTP requests over that port. TCP connections to that port are then intercepted by Tailscale after the WireGuard® decryption, before they hit your operating system. In fact, they're never delivered to your operating system: We handle the packets and TCP internally with gVisor's netstack, like we do for Tailscale SSH.

When somebody goes to `node.tailnet.ts.net` in their browser (or other client), a traditional DNS response then points to one of our funnel VMs, ideally in a region near your node.

We then accept those TCP connections from end users (which must be TLS), look at the SNI name in the TLS ClientHello, and then proxy those encrypted TCP connections to your Tailscale node over Tailscale itself. Notably, we're only proxying a TCP connection (which we verified has a valid SNI name in it); Tailscale Funnel is not doing any TLS termination. While it's true that we could in theory terminate TLS (as we own `ts.net` and could get our own Let's Encrypt certs for it), we don't want to, and you can verify in the public Certificate Transparency logs that we aren't.

So, we're proxying a TCP connection to your node. But remember, we don't have packet-level access to your nodes, so we're not just proxying our public TCP port 443 to your node's port 443. Instead, our Funnel ingress nodes send one of those aforementioned "peerapi" requests to your node: an HTTP request where the request says the source IP:port and target SNI name and port.

Your node then receives that peerapi HTTP request and decides for itself, based on configuration that lives only on your Tailscale node: Does it want that TCP connection for that tuple? If not, it rejects it. If so, what should it do with it?

At a high level, there are two main things Tailscale Funnel can do with that incoming connection. In either case, something on your device has to terminate TLS.

The first thing you can do is just pass off the TCP connection to a local webserver and let that webserver do the HTTPS. Both Caddy and Apache have support for terminating TLS via Tailscale's certificate fetching mechanism, for example.

The second thing you can do is have your device's Tailscale daemon itself terminate TLS. Then it can reverse proxy the HTTP requests to a local non-HTTPS webserver. That is, you run a webserver on `localhost:8080` and we put it on the internet, complete with a public IP address, DNS, TLS cert, and HTTPS server. Now that's a fun tunnel, if we do say so ourselves.

Now in alpha

Tailscale Funnel is now in alpha. To start, we're going to limit the number of testers and ramp up a bit slowly as we build confidence that all the infrastructure is working. You can join the alpha by following this URL. The functionality will be available in a stable release starting in Tailscale v1.34.0 but meanwhile you'll need to be running a recent unstable build (v1.33.257 or later) to try it out. After alpha we'll start opening this up more widely, hopefully soon! In the meantime, check out the documentation.

← Back to index

tailscale