Introduction

Last February I gave a talk on Wireguard. Wireguard ia a technology that started on Linux and spread widely to other OSes. It now also runs on Microsoft Windows, MacOS, Android, iOS, BSD, and others.

You may also recall all the fancy scripts, key generation, and configuration I had to do on each Wireguard host to get everything up and running. And one of the drawbacks is that if everything wasn't perfect, nothing happened.

Wireguard does take some effort to understand and get working well. A company, Tailscale, recognised that a lot of people didn't want to bother, risk getting frustrated, or screwing something up. They decided to build a product on top of Wireguard also called Tailscale. Tailscale the product encapsulates a lot of the tricky bits hiding it from most users taking away most of Wireguard's pain points.

Like Wireguard, Tailscale is obviously available for Linux, but the software also runs on MacOS, iOS, MS Windows, and Android.

Tailscale software that runs on the client (a/k/a node) is open source. It's released under the BSD 3-clause license on Github.

There's the other side though that runs on Tailscale's servers called a "coordination server". The coordination server handles the key management and configuration of your mesh network of nodes. That software is not open source. However, there is an open source version of it called Headscale, so not all is lost if you want to self-host your own coordination server.

Even though the coordination server is not free as in freedom, it is free as in beer for most all home users. You can register up to 20 nodes in your mesh network without costing you anything. If for some reason you need more, you can go up to 100 nodes for $4 a month.

But keep in mind unregistered machines on your network don't count towards your limit – only the registered nodes. And your nodes can still talk to all the unregistered machines just like they could before. This is helpful for unsupported machines like wifi access points, printers, network storage devices, webcams, etc.

Last month I gave a talk about setting up your home router as a OpenVPN server. After my talk, Rich was successful in doing this for himself with his work router.

One of the problems though with classic VPNs is that all networking traffic has to flow through the VPN hub or gateway. With Wireguard, you can configure it as a classic VPN topology or as a mesh where each node can talk directly to each other. Tailscale implements its topology as a mesh, so traffic traverses a minimal number of hops reducing delays and improving bandwidth.

See slide #4 of my OpenVPN presentation

Nifty Tailscale Features

Unique IP Addresses

When your nodes register, they get assigned their own unique and permanent IP addresses that any other node in your network can use to directly communicate with them.

Punches through NATs

No need to worry about your machines being NATted. Tailscale figures out how to make all your nodes visible to each other routing packets traversing through routers and behind firewalls.

Because of how Tailscale traverses NATting, you can also use Tailscale in virtual hosts running on your main machines.

Jumps over Firewalls

Like Wireguard, Tailscale uses UDP for exchange of its network packets. However, if UDP is blocked by a firewall, that won't stop Tailscale. Tailscale will use an encrypted TCP relay over HTTPS, known as a DERP (Designated Encrypted Relay for Packets).

Share Nodes with other Users

You can share your Nodes with other Tailscale users.

If you have say a multimedia server you'd like a friend to use, you can grant them access to add your Node to their Tailnet.

Drawbacks of Tailscale

Performance

When I first set up Tailscale on my home network, I was using wifi between my desktops and Raspberry Pi. When I connected directly via wifi versus Tailscale network over my wifi, the performance difference of transfer speeds was small, only a 7% loss in throughput. However, when using wired I saw much larger drops, as much as 87%.

I'm not sure yet why such wide variations occur. It seems that the higher the bandwidth of the connection and the higher the CPU power of the hosts, the higher the loss in throughput.

Does Not Play Well with Others

I encountered another drawback in that Tailscale "does not play well with others". When trying to use Tailscale with Fedora's firewall daemon, they kept mangling each other's `nftables` wirewall rules. This is a known bug.

I also had some poor interactions between my Wireguard configuration and Tailscale. I'm not sure what was causing that, so I just turned off my Wireguard configuration.

DEMO!

Setting up Tailscale

On Tailscale's web site, you have to register to set up your account. You can't use just any email address though. You have to use a site that provides Single Sign-On (SSO) via one of the supported providers. That includes Google, Microsoft, GitHub, Okta, OneLogin and others.

I've already created an account and will show you my home setup.

As you can see, I have 7 machines registered as Nodes. I have two exit nodes, `do1` and `li1`, and one Subnet Router, `pi4`.

`do1` is Digital Ocean droplet running in New York City. `li1` is an Linode instance running in Dallas, Texas.

I even have my iPhone registered as a Node showing up as `quentins-iphone`.

Registering a new Node

For this demo, I created another Linode instance `li2`. It's been boot and setup my my account, but for now knows nothing of Tailscale.

I'll show you what it takes to add the Tailscale software and set up a new Node.

In the `tssetup` script, you can see all it takes is 4 commands. The last command `tailscale up`, when invoked the first time, will ask me to register by going to a URL.

Then we can refresh our machines list and see the new Node.

Running `tailscale status` will show the Node participating in the network. We can then network with any Node in the list.

(If connected, use g5 to start tailscale to run as a remote host on the Tailnet with:

```
sudo tailscale up --reset \
        --accept-routes \
        --exit-node=100.90.51.61 \
        --exit-node-allow-lan-access=true
```

Go to `https://whatismyipaddress.com/` to show IP address and geolocation.