Remote Access via OpenSSH

Homelab Club 2/10/2025

Why?

Do I even need to make a slide for this?

Example

I self-host Bitwarden for passwords and use clients on my desktop and my phone.

I update my UMD password and save my password to Bitwarden using my desktop.

I go to campus, but now I can’t sign into eduroam because I changed my password.

Unfortunately, I forgot to open Bitwarden on my phone before I left home, so I don’t have the new password :(

Remote access saves the day!

Remote access saves the day!

I open the Wireguard app on my phone to connect to my homelab network, letting me reach my Bitwarden server. Now I can sync and get the latest credentials.

Woohoo!

As an aside: Exposing services to the Internet

Instead of exposing a VPN, I could expose Bitwarden directly to the Internet. However, this is not a great idea unless you are purposefully expecting the public to use it.

- Not all services you host may be well audited and free of known vulnerabilities. Think hobby-project repos that take user input, built using old versions of popular libraries, and haven't been updated in a couple years (but it does the one cool thing you want to do, so you really really really want to host it!)
- Even with well-maintained apps, if a vulnerability is found and fixed in an update, you need to keep on top of updates. The more services you host, the harder it may be to keep on top of updates for each individual service.
- Exposing services directly can make you a target when getting port-scanned, as tools can usually return what software is being hosted on a port (and sometimes the version!!)

Instead: Maintain one well-vetted endpoint

6.1.4 OpenVPN security in four pictures: (another aside, this is a great document as a general homelab intro)

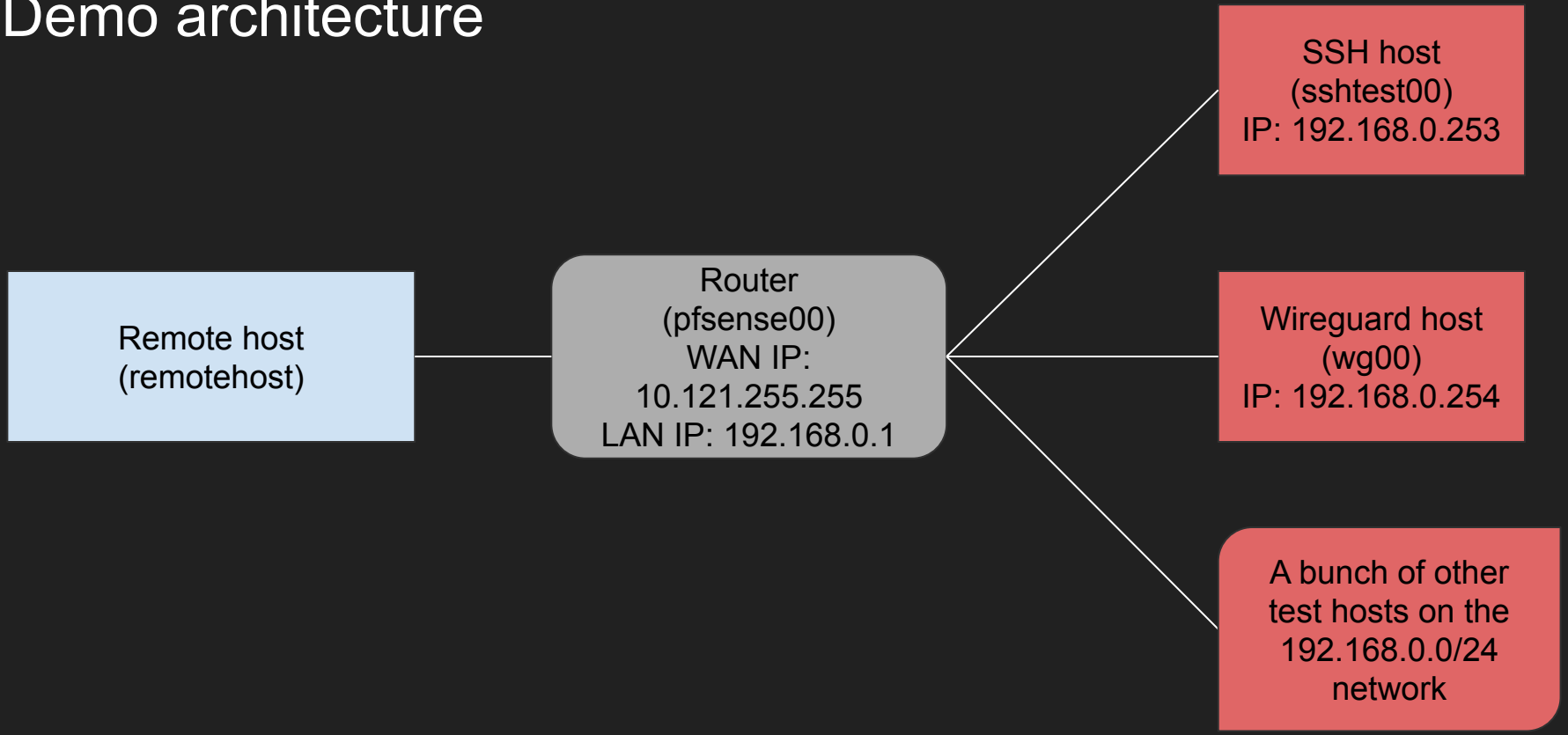
- Port scanners will only see a single endpoint and won't be able to make guesses to what software you're running inside unless they manage to break in.
- As long as you don't accidentally backdoor yourself some other way (installing malicious software), you shouldn't have to worry about direct attacks from outside.

Two common options

SSH bastion with SSH keys ← Today's focus

VPN (e.g. OpenVPN, IPSec, WireGuard)

Demo architecture



SSH

Protocol for securing network services through encryption. Applications can build SSH into their services (e.g. SFTP)

Though it is a protocol, most people talking about SSH are usually referring to the OpenSSH client/server.

- Improved over its predecessors by encrypting data over the line, so you aren’t sending plaintext credentials when logging in.
- Allows for passwordless connection using client-side keys.
- SSH Tunneling/Proxying (we will talk about this later)

OpenSSH Setup Demo

OpenSSH Setup general steps

1. Install OpenSSH package (e.g. Debian -> openssh-server, Fedora -> sshd)
2. Ensure OpenSSH server is running (For systemd distros, systemctl status sshd)
3. Connect from an OpenSSH client
    a. If remote, you will need configure the port forwarding rule on your router to the SSH server (default 22/TCP).

SSH keys

The SSH server has keys associated with it (host keys) so the client can be assured that the server they are connecting to is the same one as before.

- Of course, if someone copies the host keys to a different server, they can spoof that host. Keep these safe!

The SSH client can also present a key to the server to allow for login. This is what we are going to set up today.

- By using client keys instead of passwords, you can feel more assured that you won’t get owned by a brute-force password attack.

SSH client key setup

ssh-keygen

By default, on MacOS, it will set up an ed25519 key with comment “user@hostname” to identify it.

Algorithm can be overridden with the “-t” argument (might be required for older systems)
Comment can be overridden with the “-C” argument

This command generates two files: a public key (give this to servers/services that you want to connect with over SSH), and a private key (DO NOT SHARE THIS! This stays on the client computer.)

SSH keys with passwords

Though an SSH key can be generated without giving it a password, you should **definitely** set a password.

Without a password, if your SSH private key is stolen, you have effectively given someone the keys to the city! Multiple layers of protection is a good thing.

An SSH key password is different from the typical password prompt when connecting to an OpenSSH server. The password is checked locally, so connecting to a malicious server will not endanger the password you use for your key. In contrast, there is a chance that a password in a typical password prompt could be compromised.

SSH client key setup (server side)

Copy the contents of your the public SSH key file you just generated into the ~/.ssh/authorized_keys file on the server (make the file if it doesn't exist). By default, the name will be <key_name>.pub, saved in the same directory where you saved the private key.

Highly recommended: Disable password login

In the SSH server config file (default /etc/ssh/sshd_config), add the following line (or edit the existing line in the config):
PasswordAuthentication no

Then, restart the sshd service. systemctl restart sshd

SSH Client Key Demo

SSH Tunneling/Proxying

The OpenSSH client can allow a client to access network resources that are accessible to the OpenSSH server.

-L [local_address]:<client_port>:<destination_address/hostname>:<destination_port>

e.g. Access a website hosted on the OpenSSH server on port 80, using port 8080 on the client side (because ports <1000 may be blocked by default)
-L 8080:localhost:80
e.g. Access Windows remote desktop port

-L 3389:mywindowshostname:3389

After this, you can connect to the service at localhost:client_port.

SSH Tunnel Demo

OpenSSH SOCKS Proxy

You can open a SOCKS proxy using OpenSSH. This is useful when you need to connect to multiple services and don’t want to set up a tunnel for each one.

ssh -D 4000 192.168.0.253

This will SSH into the host and will make a SOCKS proxy accessible from the client at localhost:4000 (you can choose whatever port number you want). Anything accessible from the server (192.168.0.253) can be accessed over this proxy.

Most browsers support SOCKS proxies, and most OS’s support it at the OS level.

SSH Proxy Demo

How do I get my home IP?

Remote access is only really good if you know where to connect to…

- Residential ISPs usually hand out addresses dynamically, so if your router resets, you will almost certainly get a new IP address.

A Dynamic DNS service is a common solution to this problem.

- Essentially you have a script that runs on a computer at your home that tells another service what your IP address is.
- DuckDNS is a popular free example. If you own your own domain name, your DNS service probably has an API you can use to do the same thing.

Internal/external DNS is a whole talk in and of itself.

Wireguard

At a later date…