Raspian 10 (buster)

Prepare SD card

- Download and install the Raspberry Pi Imager
- Copy “RaspBerryPi OS Lite” to the SD card

Install OS

- Boot from SD card
- Update all packages
- Enable SSH server in *Interface Options* and set location and keyboard in *Localisation Options*:

```
sudo raspi-config
```

- Change host name
- Config SSH and generate SSH keys
- Exchange public key files
- Install missing packets:

```
apt install autossh vim
```

Install AutoSSH

Install AutoSSH on a **remote_host** to connect to a **local_host** to allow connections from this **local_host** or **any_host** from any (other) location.

Setup remote_host

1. Verify access from local_host and any_host: SSH from **remote_host** to **local_host**:<ssh_port_local_host> with key pair authentication and establish tunnel, requires <ssh_port_local_host> to be open on local location:

```
remote_host$ ssh -R <port_to_access_remote_from_local>:localhost:22
user@<domain_name_of_local_host> -p <ssh_port_local_host>
```

2. Once confirmed, create file *autossh-tunnel.service* in /etc/systemd/system/:

```
[Unit]
Description=AutoSSH tunnel service Remote port
<port_to_access_remote_from_local> to local 22
After=network.target

[Service]
```

```
Environment="AUTOSSH_GATETIME=0"
ExecStart=/usr/bin/autossh -o "ServerAliveInterval 10" -o
"ServerAliveCountMax 3" -N -R
<port_to_access_remote_from_local>:localhost:22
user@<domain_name_of_local_host> -p <ssh_port_local_host> -i
/home/pi/.ssh/id_ecdsa

[Install]
WantedBy=multi-user.target
```

We need to tell SSH the identity file as systemd will run as root. The environment variable is added so the autossh service can run in the background.

3. Once we have the service file created start the service and enable it to run at boot:

```
remote_host$ sudo systemctl daemon-reload
remote_host$ sudo systemctl start autossh-tunnel.service
remote_host$ sudo systemctl enable autossh-tunnel.service
```

4. Trouble shoot:

```
sudo journalctl -u autossh-tunnel.service [-b]
```

Connect from local_host

1. SSH from **local_host** to **remote_host** through tunnel at <port_to_access_remote_from_local>

```
local_host$ ssh pi@localhost -p <port_to_access_remote_from_local>
```

Connect from any_host

1. SSH to **local_host** and establish tunnel from (any_host) <port_to_access_remote_from_local> to localhost:<port_to_access_remote_from_local>
2. SSH from any_host to localhost:<port_to_access_remote_from_local>

Connect to other services at remote location

Connections from any location cannot tunnel to other services on the remote network directly. Instead, we can open a (temporary) tunnel to any host on the remote network and add a tunnel to the same port to the connection from any_host to local_host.

1. Login to remote_host, then establish an additional tunnel:

```
remote_host$ ssh -R
<port_to_access_additional_service>:host_on_remote_network:<port_to_add
itional_service> user@<domain_name_of_local_host> -p
<ssh_port_local_host>
```

2. SSH to **local_host** and establish tunnel from (any_host) <port_to_access_additional_service> to localhost:<port_to_access_additional_service>
3. Connect from any_host to localhost:<port_to_access_additional_service>
4. Example: to access a https website on remote host 192.168.1.1, host_on_remote_network=192.168.1.1, <port_to_access_additional_service>=50443, <port_to_additional_service>=443

Links

- SSH tunneling with Autossh