Setting up SSH and SSHFS for IN5050

Spring 2023

Prerequisites

SSHFS installation

Debian

```
$ sudo apt-get install sshfs
```

Mac

```
$ brew cask install osxfuse
$ brew install sshfs
```

SSH key generation

We will assume that you have generated a SSH key pair, and that is located in its default location; `~/.ssh/id_rsa` and `~/.ssh/id_rsa.pub`. Generate key pair with the command `ssh-keygen`. You will be fine just by using the default values (aka press enter for each step).

```
$ ssh-keygen
```

Con iguring SSH

The IN5050 machines are only reachable through the 129.240.66.57 machine, which is only reachable through the IFI machines. A primitive workflow would be to SSH in to the IFI machines, then to the `129.240.66.57` machine and then to the concrete machine (Tegras or x86). This is tedious and can result in different issues, especially if we are using SSHFS. A solution is to set up a so called ProxyJump.

1. Open/create `~/.ssh/config`.
2. Enter the following:

```
Host ifi
     User siverto
     Hostname loft.ifi.uio.no
     Forwardx11 yes
     Forwardx11trusted yes
     compression yes

 Host in5050
     User in5050-g02
     Hostname 129.240.66.57
     ProxyJump uio

 Host tegra-1
     User in5050-g02
     Hostname tegra-1
     ProxyJump in5050

 Host tegra-2
     User in5050-g02
     Hostname tegra-2
     ProxyJump in5050

 Host tegra-3
     User in5050-g02
     Hostname tegra-3
     ProxyJump in5050

 Host tegra-4
     User in5050-g02
     Hostname tegra-4
     ProxyJump in5050

 Host x86-1
     User in5050-g02
     Hostname in5050-2014-10
     ProxyJump in5050

 Host x86-2
     User in5050-g02
     Hostname in5050-2014-11
     ProxyJump in5050
```

Note that we are using the username `siverto` for the IFI machines, and `in5050-g02` for the IN5050 machines. You have to change this accordingly. We have also selected `loft.ifi.uio.no` instead of `login.ifi.uio.no`. `login.ifi.uio.no` will assign you to a computer based on available resources, and it can cause problems, especially with SSHFS. Our suggestion is that you find a computer by SSH-ing in to `login.ifi.uio.no` and replace `loft` with your given machine.

3. Try to SSH to the different machines:

```
# IFI
$ ssh ifi

# IN5050
$ ssh in5050

# Tegras
$ ssh tegra-1
$ ssh tegra-2
$ ssh tegra-3
$ ssh tegra-4

# X86
$ ssh x86-1
$ ssh x86-2
```

   You will be asked to enter your password for each hop. This is of course tedious, and luckily we have a solution!

4. We can add our personal computer's public SSH key to the machine's lists of authorized keys (`~/.ssh/authorized_keys`). You can manually copy the content of your `~/.ssh/id_rsa.pub` and paste it in each machine's `~/.ssh/authorized_keys`, but we can also do it by command:

```
$ cat ~/.ssh/id_rsa.pub | ssh ifi -T "cat >> ~/.ssh/authorized_keys"
```

   Just change `ifi` with the host you want to add the keys to. You should now be able to use SSH to the given machines without having to manually enter the password every time.

Setting up SSHFS

We want to be able to use our local text editor on the remote code. This can be done by mounting the IN5050 machines to our local machine with SSHFS. We will demonstrate how we can mount one of the Tegras to our local machine.

1. Create a directory on our local machine. For this example we will call it `~/tegra`.

   ```
   $ mkdir ~/tegra
   ```

2. Create a directory on our tegra. We will name it `~/in5050`.

   ```
   $ ssh tegra-1 "mkdir ~/in5050"
   ```

3. Mount the Tegra's `~/in5050` to your local `~/tegra` by

   ```
   $ sshfs tegra-1:in5050 ~/tegra -o reconnect
   ```

   Changes you do inside `~/tegra` will now by passed to `~/in5050`, and vice versa.

4. You can unmount the directory with

   ```
   # Debian
   $ fusermount -u ~/tegra

   # Mac
   $ umount ~/tegra
   ```