Setting up SSH Tunneling

In computer networking, a computer decides what to do with an incoming network packet according to the "port" it arrived on. The port is simply a number attached to the packet. Certain ports are reserved for specific functions, for example packets arriving on port 22 are assumed to be intended for the SSH handler, so the computer passes those packets to SSH to interpret. Other port numbers are available to use for whatever you like, and as long as the same port is not used for different things on the same computer, everything works.

With SSH Tunneling, you will start an SSH session between your workstation and the bastion host hpc.nyu.edu, and instruct that session to create a tunnel. Your workstation will make one end of the tunnel, at "localhost, port 8023" ("localhost" is the computer's name for itself, so packets arriving at your workstation port 8023 will be sent into the tunnel). The bastion host will make the other end of the tunnel, at "prince.hpc.nyu.edu, port 22", so anything coming through the tunnel will be forwarded to the normal SSH port (22) of Prince. The fact that your workstation cannot see Prince does not matter, it only needs to see its end of the tunnel.

The following diagram illustrates the process. It looks complex, but only requires 2 steps: the blue text shows what happens when you create the tunnel (step 1) and the green arrows indicate using the tunnel (step 2).

You only need to do step 1 once, and then you can use the tunnel (step 2) as many times as you like - for example, you might have two terminal sessions and a WinSCP session all using the same tunnel created with step 1.

In these instructions we are using port 8023. If it happens that another program on your computer is watching this port (which is fairly unlikely) then it won’t work, and you’ll need to choose a different port number, eg 9020, and substitute that throughout these instructions. 4-digit numbers starting with an 8 or a 9 are usually good ones to choose.

(Mac only) Preparing your Mac for SSH Tunneling
We will be instructing your Mac to forward certain incoming packets to a tunnel, so first the Mac must be willing to accept the incoming packets at all. To enable this:

1. Open System Preferences and click Sharing.
2. Select the Remote Login checkbox.
3. Return to system preferences and click “Security” and then ”Firewall options” (if you are using Mavericks) or ”Advanced” (for older versions of OSX)
4. Uncheck “Block all incoming connections”.

We have a video guide of this process.

Also, make sure you have prepared your Mac for X in accordance with the instructions above.

Note that Linux users do not need to do this: remote logins are enabled by default under Linux.

**Setting up a tunnel you can reuse (the best approach)**

To avoid repeatedly setting up a tunnel, we write the details of the tunnel into your SSH configuration file. This is found in the hidden ".ssh/" directory under your home directory. To access it, first open a Terminal window.

The process is the same for Linux and Mac, and is demonstrated in the first minute and a half of this video guide (the remainder of the video demonstrates transferring files over the tunnel, which is covered here).

To see the contents of a directory, enter "ls -la" at the command prompt. The "a" is important, without it files and directories starting with "." will be hidden. We are looking for a directory called ".ssh".

If you do not have a ".ssh" directory, create one as follows. The permissions of this directory are important, hence the chmod command.

```
mkdir ~/.ssh
chmod 700 ~/.ssh
```

Using your favorite editor, open the file ".ssh/config". If you are not familiar with Unix-ish editors, we have some help here.

Add the following lines to ".ssh/config:

```
# first we create the tunnel, with instructions to pass incoming packets on ports 8020, 8021 and 8022 through it and to specific locations

Host hpctunnel
  HostName hpc.nyu.edu
  ForwardX11 yes
  LocalForward 8020 mercer.es.its.nyu.edu:22
  # you can add other hosts to the tunnel too - for example, to specifically use login node 1 on mercer:
  LocalForward 8023 mercer.es.its.nyu.edu:22
  # for the Hadoop cluster:
  LocalForward 8025 dumbo.es.its.nyu.edu:22
  User NetID

# next we create an alias for incoming packets on port 8023. The alias corresponds to where the tunnel forwards these packets

Host mercer
  HostName localhost
  Port 8023
  ForwardX11 yes
  User NetID

# repeat the last step for each LocalForward entry under "Host hpctunnel", eg:

Host mercer1
  HostName localhost
  Port 8024
  ForwardX11 yes
  User NetID

Host dumbo
  HostName localhost
  HostKeyAlias dumbo
  Port 8025
  ForwardX11 yes
  User NetID

---

Starting the tunnel

To create the tunnel, ssh to it with the following command:

```bash
$ ssh hpctunnel
```

**Important:** you must leave this window open for the tunnel to remain open. It is best to start a new terminal window for subsequent logins.

Logging in via the tunnel

1. **Open a new terminal window**

2. **Use ssh to log in to the cluster, as shown below. Note that you must use the short name defined above in your `.ssh/config` file, not the fully qualified domain name**

```bash
$ ssh mercer
```
Creating a once-off tunnel (not the best approach)

You can set up a once-off tunnel without editing .ssh/config by running the following command:

```
$ ssh -L 8023:mercer:22 NetID@hpc.nyu.edu
```

This is the equivalent to running "ssh hpctunnel" in the reusable tunnel instructions, but the port forwarding is specified on the command line. You can setup tunnelling to a second cluster (e.g. Babar) by adding a second -L option to the command line.

Logging in via a once-off tunnel

However this does not create the convenient alias, so when connecting with ssh, or scp or rsync, you must explicitly connect to that port on your workstation (localhost):

1. **Open a new terminal window**
2. Use ssh to log in to the cluster by logging into localhost on the appropriate port, as shown below.

```
$ ssh -X -p 8023 localhost
```

With a once-off tunnel we do need the -X flag to ssh, and also the -p flag (lowercase) to specify the port number.