Max-Planck-Institut für Polymerforschung
Max Planck Institute for Polymer Research

Connecting to the MPI-P with OpenSSH

Robert Klein*

December 4, 2023

1 Setting SSH access on your (home) computer

Here is a config intended to achieve

- let you connect to your MPI-P Linux PC
- having to type the secret key's passphrase only at first use
- having to type the six-digit password for login2 only once per parallel session
- not having to type the password for pckrXXX at all
- work on Linux, Mac OS, and Windows

Please follow these steps to comfortably connect to the MPI-P.

1. Start a terminal window
2. if it doesn't exist, create a directory/folder ".ssh"

```
cd
install -d -m 0700 .ssh
```

3. save the files "id_ed25519" and "id_ed25519.pub" and move them to this folder and fix permissions for the file "id_ed25519":

*kleinrob@mpip-mainz.mpg.de

```
cd .ssh
mv ../Downloads/id_ed25519 ../Downloads/id_ed25519.pub .
chmod 0600 id_ed25519
```

- on a *WSL ("Windows Subsystem for Linux") Linux* you'll have to copy the files from you Windows download directory (I'm copying here instead of moving, as you'll probably want to reuse the keys for Windows openssh and/or MobaXTerm.):

```
# first, install debians own ssh:
sudo apt-get install ssh
# then proceed with the keys:
cd .ssh
cp /mnt/c/Users/$LOGNAME/Downloads/id_ed25519 .
cp /mnt/c/Users/$LOGNAME/Downloads/id_ed25519.pub .
chmod 0600 id_ed25519
```

4. On Linux systems change the passphrase for the secret key to the same password you use to login to your home computer, so Linux can add it to the SSH keychain automatically and you don't have to type the passphrase every time:

```
ssh-keygen -p -f ~/.ssh/id_ed25519
```

5. prepare sockets for multiplexed connections:

```
install -d -m 0700 ~/.ssh/sockets
```

6. Edit the file "config" (also in the .ssh directory) and insert the following lines (replase pckrXXX with the name of your Linux PC at the MPI-P and LOGNAME with your account name at the MPI-P):

```
HashKnownHosts No
# UseKeychain only exists on MacOS X.  Ignore on other OS's.
IgnoreUnknown UseKeychain
# Store my secret key's passphrases is MacOS X's keychain:
UseKeychain yes
# Load key into ssh-agent automatically:
AddKeysToAgent yes
# Additionally allow ssh key type ssh-ed25519 (prepend to key list):
PubkeyAcceptedKeyTypes ^ssh-ed25519

Host login2.mpip-mainz.mpg.de
    # Location of my ssh-ed25519 key:
    Identityfile ~/.ssh/id_ed25519
    # The "Control" directives for login2 allow me to log in to any
    # MPI-P Computer with only having to type the OTP password once
    # per login session.
    ControlMaster auto
```

```
    ControlPath ~/.ssh/sockets/%r@%h:%p

Host pckrXXX pclinux1
    User LOGNAME
    Hostname %h.poly.mpip-mainz.mpg.de
    # Forward the keys in the local ssh-agent to the ssh-agent on the
    # remote system:
    ForwardAgent yes
    # Connect to pckr's using an intermediate jump host.
    Proxyjump LOGNAME@login2.mpip-mainz.mpg.de
```

Note: If you're using Windows' OpenSSH, replace the "ProxyJump" line with this line:

```
ProxyCommand ssh.exe -l LOGNAME -W %h:%p login2.mpip-mainz.mpg.de
```

- Append the contents of id_ed25519.pub to the ~/.ssh/authorized_keys file on pckrXXX. If your home computer is a Linux machine you may be able to use this command:

```
ssh-copy-id -i ~/.ssh/id_ed25519.pub pclinux2:
```

2 Using the setup

Now you can connect to your MPIP desktop as well as to pclinux1: the latter is for co-workers without a Linux desktop at the MPI-P and in case of need e.g. when your office desktop is switched off.

The command to connect is e.g.

```
ssh pclinux1
```

3 Copying files

When you have set up your ssh access as outlined in 1, you can just *scp(1)* files from and to there:

```
# copy thesis tex file from office PC to home computer
scp pckrXXX:Documents/thesis.tex ~/Documents
# copy htesis bibliography from home computer to office PC
scp ~Documents/biblio.tex pckrXXX:Documents/
```

Please note, that the data of multiplexed connections go through the original connection. This may impact performance, e.g. when copying large files over the connection.

4 Establishing a tunnel to your Linux Office PC

Why? you might ask. Perhaps you want to run Jupyter Notebook using the power and disk space available to your Office PC and use the Web interface on your local computer.

Edit the file `~/.ssh/config` on your (home) computer and add the following lines:

```
# Inner tunnel to office PC
Host inner
    User USERNAME
    Hostname pckrXXX.poly.mpip-mainz.mpg.de
    ProxyJump LOGNAME@login2.mpip-mainz.mpg.de
    LocalForward 2222 pckrXXX.poly.mpip-mainz.mpg.de:22
```

Which will give you a listening SSH session at localhost port 2222 wich really ends up as pckrXXX port 22.

5 Using Jupyter Notebook running on your office PC

Note: If your'e using an office PC not your own, please use another port number for Jupyter Notebook, so you don't block your colleague's work – Use for example the last four digits of your numerical Account ID at the MPIP; that's the output of "`id -u`" without the "100" in the front and add, for example 9000. A short command to get a port number is "`echo $(( 9000 + $(id -u) - 1000000))`". Should the number already be in use, please add another 1000 or 2000. Thank you very much.

Edit the file `~/.ssh/config` on your (home) computer and add the following lines (replace the port number if necessary):

```
# Outer tunnel to office PC
Host jupyter
    User USERNAME
    Hostname localhost
    Port 2222
    LocalForward 9000 127.0.0.1:9000
```

now you first open the inner tunnel:

```
ssh inner
```

Start your Jupyter notebook in this SSH session, my port is 9000:

```
jupyter notebook --port 9000 --no-browser
```

Caveat: Try to use the same port number for the Jupyter Notebook *and* the LocalForward port, so you don't have to edit the Jupyter Notebook address.

Then start the connection to your Jupyter Notebook:

```
ssh jupyter
```

Now copy your Jupyter notebook URL into your local browser and start doing your Jupyter stuff.