**SSH and Keys**

ssh [username@ip.address]

ssh kyle@192.165.40.140

This command will try to open a shell/terminal on the computer “192.165.40.140” by logging in with the username “kyle”. If your SSH keys are not set up correctly, you will be asked for a password. Follow the commands below to generate and add ssh keys.

ssh-keygen [arguments] – Used for generating a new set of SSH keys.

Check if you already have keys in the ~/.ssh folder before running:

ssh-keygen -t rsa -b 4096 -C "Name\_it\_here"

If you use all of the suggested settings when creating the keys (which is absolutely fine) then you will have a set of keys generated in the ~/.ssh/ directory called “id\_rsa” and “id\_rsa.pub”. Don’t share “id\_rsa”, this is your private key. It is ok to share “id\_rsa.pub”, this is your public key.

**Pipes |**

Pipe characters “|” direct the output of one program to be input into another program.

cat ~/.ssh/id\_rsa.pub | ssh user@ip.address 'cat >> .ssh/authorized\_keys'

This command, when run on your computer, will copy the public key generated from *ssh-keygen* to the list of authorized keys on the remote computer. If you are using Jetstream, you should instead add your public key to your Jetstream account through the online interface under “Settings > Advanced”. This will automatically add your ssh key to every instance you launch.

history | grep ssh

Normally, using the *history* command will print the last ~200 commands run to the terminal – but that’s a lot to sift through! Think of the *grep* command as “Ctrl + F” in the command line. In this case it will look for every line that contains “ssh” – this is great if you can’t remember the IP-address of your instance!

**Capture Output >**

The greater than symbol “>” can be used to capture the text output of a program and store it in the file specified.

ls > directory\_list.txt

We have already seen that *ls* will output the name of every file and folder in the current directory. By using the “>” the output is redirected to a file called “directory\_list.txt” that will store the output instead of displaying it in the terminal.

cat [genome1.fasta] [genome2.fasta] … [genome\_n.fasta] > all\_genomes.fasta

The cat command will concatenate multiple files into one output. By default, it will dump everything into the terminals standard output, which isn’t normally that useful. This is a common case where you may want to capture the output with “>” and write a new file.