1 Quick Demo on Running Remote Processes

Lab Objective: *The lab machines in TMCB 150 have been increasingly occupied by the senior ACME class running Jupyter notebooks for extended periods of time. To avoid conflict between students who want to use the same machines, this document aims to give brief instruction on how to* `ssh` *to the lab machines, remotely copy files to the machines using the* `scp` *command, and run* `.py` *files as background tasks with low priority to minimize the impact on other students computing.*

SSH

If you are on campus, you can use the Terminal command `ssh` to open a terminal on any lab machine that is turned on. This works even if someone else is currently using the machine. The syntax to `ssh` with a student account is as follows:

```
ssh connorjr@byu.local@acme13.byu.edu
connorjr@byu.local@acme13.byu.edus password:
Last login: Tue Sep  5 11:01:11 2017
[connorjr@byu.local@acme13 ~]$
```

This terminal can now be used to run processes on `acme13.byu.edu`. To exit the ssh connection, type `exit` and hit return. If you are not on campus, you cannot `ssh` to the lab machines without VPN access.

SCP

The Linux command `scp` allows you to copy files from one computer to another using `ssh`. This allows you to create and edit `.py` files and dependencies on your personal computer, copy them to a lab machine, and then run them there. You may copy files from one computer to the next with the following syntax:

```
scp -r mydirectory connorjr@byu.local@acme13.byu.edu:~/
```

The above command will recursively copy "mydirectory" in my current terminal from my machine to my home directory (that's the `~/` part) on `acme13.byu.edu`. Once the files are on the remote computer, you will have to use emacs or vim to edit them further.

Your files should save the results of your program to a file. This file can then be copied from the lab machine to your machine if you have `ssh` enabled on your personal computer. Linux has this enabled by default. Instructions for Windows and Mac are included below.

Enable SSH on Mac

Open `System Preferences` -> Click on `Sharing`-> Check the box for `Remote Login`. Under "`Remote Login: On`" it should now read: "To log in to this computer remotely, type "ssh user@ip".

Enable SSH on Windows

Google says go to `Control Panel` -> `System and Security` -> `Administrative Tools` -> and open `Services`. Locate the `sshd` service and click `Start the service`. If you can't find the service, you need to install it.

Copying with SCP from Lab Machine to Personal Machine

The syntax is basically the same as is given above. To copy the directory `mydirectory` from my home directory on `acme11.byu.edu` to the home directory on my machine, I would type:

```
scp -r connorjr@byu.local@acme11.byu.edu:~/mydirectory user@ip:~/
```

Running .py files

It is best to use the `top` or `htop` command to see if anyone else is running a computationally expensive program before you run one yourself. Once you are ready to run a command, use the `nice` command to run your program with lower priority. The syntax to run a python file with nice low priority in the background is as follows:

```
[connorjr@byu.local@acme13 ~]$  nice -n 20 python myfile &
```

The `nice` command takes in the input `-n 20` to give a nice value of 20. A regular user on the lab machines can use a `nice` value between 0 and 20, 0 being more priority and 20 being less. This will rarely cause your program to run significantly slower, but will give priority to an active user on the machine. The `&` on the end will run the program in the background.

You can find programs running in the background by using the command `jobs`. Appending `-l` to the `jobs` command will also list the process id number. An example is as follows:

```
[connorjr@byu.local@acme13 ~]$ jobs -l
[1]+ 21123 Running                 nice -n 20 python hello.py &
```

You can kill your job by using the `kill` command. For example, to kill the job above you would type:

```
[connorjr@byu.local@acme13 ~]$ kill 21123
[connorjr@byu.local@acme13 ~]$ jobs
[1]+  Terminated              nice -n 20 python hello.py
```

Be careful that you get the process id correct or you risk killing another process.

Once you have run the process in the background, you may exit your `ssh` connection. However, you will not be able to access your program through the `jobs` command after you leave. Instead, you will have to use `top` or `htop` to see your process id.

> **Note**
>
> It is good practice to have your program save whatever it can at intervals in case the computer is shut down or some other unforseen event stops it.

If you have any questions or something doesn't work for you, call or send a message to Connor Robertson at 949-245-5236.