

SSH Guidelines

The purpose of this document is to walk through how to remotely log in to the CS Lab machines to test your code. You will need to do this in your future classes, like CS 429 and CS439, so feel free to get ahead now! We will talk about how to SSH via Mac and Windows. We will talk about some essential commands needed to navigate the Linux terminal, and how to compile and test your code remotely. The compiling and testing will work if you are physically at the lab machines and want to run your code that way.

You will need a UTCS Account for this to work. To request an account, visit [this](#) link.

What's SSH and why would we use it?

SSH stands for Secure Shell. It's a secure way for you to remotely log into the CS machines from your laptop. This is meant for use if you have your code on your machine, but would like to compile and run it on the lab machines for testing.

When SSHing into a lab machine, you cannot use Eclipse or BlueJay to edit your files!!! If you wish to edit your files via SSH, there are some IDE's that will allow this on Mac, or you can use a text editor like vim, emacs, or sublime (there are many others).

When we grade, we use the lab machines. If your code compiles on your computer, but not on the lab machines, you will receive a zero for correctness. We're giving you the tools to check your work. Use them.

Computer Setup and Login

Mac:

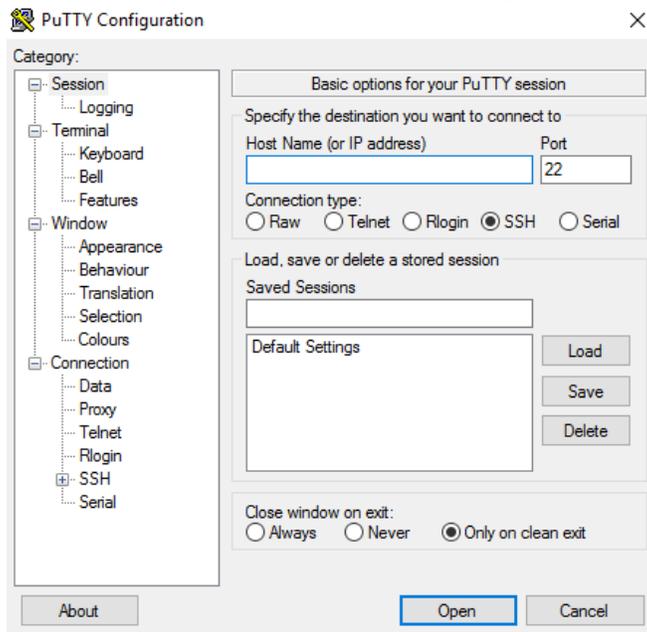
There are two options for Mac—you can either use the terminal, or you can install Putty. To use the terminal, follow these instructions. **To use Putty, look at the Windows instructions.**

1. When you open up the terminal, you'll see a prompt that looks something along the lines of:
myComputerName:
2. type in ssh CSLOGIN@cs_lab_machine.cs.utexas.edu
 - a. A list of all the CS Lab machines and their names can be found [here](#):
 - b. We recommend you don't use linux.cs.utexas.edu or skipper.cs.utexas.edu because those tend to have a very high load.
3. Press enter.
4. This should prompt you for your password. Type that in.
5. You're successfully logged into the lab machine!

Windows:

There are some SSH clients out there to help with SSHing into different computers—I recommend Putty. Download it [here](#) and install it on your machine

1. When you pull up Putty, it looks something like this:



2. Type the name of a lab machine in the Host Name box. Click “Open”.
 - a. A list of all the CS Lab machines and their names can be found here:
 - b. We recommend you don’t use linux.cs.utexas.edu or skipper.cs.utexas.edu because those tend to have a very high load.
3. Type in your CS Username. Press Enter.
6. Another option to login is to type in CSLOGIN@cs_lab_machine.cs.utexas.edu into the hostname at the beginning
4. Type in your CS password. Press Enter.
5. You’re SSH’d into your lab machine!

Navigating Linux

We’re going to go through a few useful commands for you to get around a Linux machine and compile/run your code.

1. First, type ls
 - a. This lists all of the folders and files that are in your home directory (folders are called directories).
2. To go into one a directory, type “cd <directory name>”
 - a. Suppose your directory name is “CS312_Project1”. You may not want to type that out each time you want to access anything in the directory. Linux has a great auto complete feature to help with this. If you type in “cd CS” and hit the Tab key, it autocompletes the folder name for you.
 - b. Suppose you had another directory in the same workspace called “CS314_Project1”. You can still use autocomplete. If you type in “cd CS” and hit the tab key twice, the terminal displays all the directories/files that start with those letters. If you want to access

“CS314_Project1”, simply hit the up arrow key. This will pull the last instruction you typed (and you can keep going up or down to hit different commands that you’ve typed). Fill in 314 and autocomplete with a tab.

3. Suppose you want to make a new directory to put all of your java files in. Type “mkdir <directory_name>”. This makes the directory.
4. If you want to remove this directory, type “rm <directory_name>”. Note that if there are any files in this directory, you’ll get an error. You can either go in and remove these files one by one, or you can recursively remove all the files in the directory using “rm -r <directory_name>”. Now, if you have other directories inside this directory, google may tell you to “rm -rf <directory_name>”. DO NOT DO THIS UNLESS YOU ARE 100% SURE YOU KNOW WHAT YOU ARE DOING. This will completely wipe all the contents of the directory without asking you to confirm. Companies have deleted part of the internet because of this (no I’m not kidding.... [Here’s](#) a link if you’re interested in stupid things people do:).
5. Now, we need to put our java files from our computer onto the lab machines
 - a. Mac
 - i. This is a little tricky because you have to actually type commands into the terminal to make this work.
 - ii. Type in “scp username@mac:<path to file> <filepath from home to where you want the file to go on the lab machine>”
 - iii. Suppose I had assignment Song.java on my machine at ~/Desktop. I want to put it in the folder CS Labs on the CS Machine. I would type in “scp myLogin@myMac:~/Desktop/Song.java ~/Documents/CS\ Labs”
 - iv. You can tab complete this, and I recommend you do so.
 - v. Now your files are on the lab machine!
 - b. Windows:
 - i. Download WinSCP [here](#)
 - ii. Log in to the lab machine like you did with putty. The left GUI (graphical user interface) will show the files on your machine, and the right will show files on the Lab machine.
 - iii. Drag and drop the files from your computer to the lab machine.
 - iv. Now your files are on the lab machine!
6. Now, go back to the terminal (if you were on windows). Navigate inside the folder with your java file.
 - a. Type in “javac <filename>.java”
 - i. This compiles the java file into byte code that the computer knows how to read.
 - ii. If you have any issues, the terminal will give you back some information regarding your compile error.
 - iii. If you have 10 files that you need to compile, you can type in “javac *.java”. The * is like a wildcard. This tells the computer to compile all the files that end with java (saving you a lot of tedious work).
 - b. Now to run the program. Type in “java <filename>”. You don’t need the .java extension here because you’re running the executable file generated when you compiled the code.

These are the basic techniques and setup to get your code up and running on the lab machines. The other option, if you don't want to go through all this, is to physically go to a lab machine and log in. You can either email yourself the java files or put them on a USB stick (or some other form of transfer). Once you have the files on the lab machine, run the "javac <filename>.java" and "javac <filename>" commands.