

Lab 1 - Going Remote

Logging Onto The College Server

We are going to do all of our coursework on a **remote** computer installed somewhere on campus by logging into the remote computer from our **local** computer (laptop). This will allow us to avoid the hassle of installing a code editor, a compiler, and a runtime environment to run the programs that we've written. More importantly, it will provide us a common system on which to learn.

To log into the remote computer we are going to use the applications named GitBash (for Windows users) and Terminal (for MacBook users). From here on out, if I ask you to run/launch/start your **terminal** application I will be referring to GitBash and Terminal.

Ok, open up your terminal application.

Terminal applications run a program called a **shell**. There are many different types of shell programs. A commonly used shell is called **bash**. bash stands for *Bourne Again Shell*. It was written by Jason Bourne. No it wasn't. It was written by Brian Fox as a replacement for the Bourne shell written by Stephan Bourne. Hence the play on words, Bourne (born) again.

When a shell is running, it displays a **system prompt**. The appearance of the system prompt depends on the shell that you are using, but it usually ends with a symbol like \$ or #.

We can enter **commands** at the shell's system prompt, and when we press the *enter/return* key, the commands are executed (aka run) on the computer. In this class, the commands that we will run will include the *name of a program* followed by *data that we want to pass to the program*.

The **ssh** program is used to log into a remote computer. ssh stands for *secure shell*. We can only log into a *remote computer* if we have access to a *user account* on the remote computer and know the remote computer's *name* (aka **host name**).

When we run `ssh` we will provide it with the *username* of the account that we have access to along with *the (host) name of the remote computer*.

When we run `ssh` it will create an **encrypted connection** to the *remote computer* and provide us with a new *command prompt* for the remote computer that we are logged onto. When we run commands on the remote computer's *command prompt*, the commands are executed on the remote machine - not on our laptops.

To summarize, we use a Terminal application that runs a shell (e.g. bash) which gives us a command prompt to run commands. We'll run the ssh program from the command prompt to log onto a remote computer and once connected will be given a new command prompt for the remote machine, from which we can execute commands on the remote machine.

Terminal -> **shell (bash)** -> **command prompt** -> **ssh** -> **command prompt** -> **commands**
local computer (laptop) -> **remote computer**

A **server** is a computer that *serves* data to the computers that connect to it. For example, a web server is a server that serves web pages to the computers (typically using a browser) that connect to it.

Here on campus, we have a server named **cs.bridgewater.edu**, that runs the **Linux** operating system. The Linux operating system was written by Linux Torvolds and is modeled after UNIX. It runs very similar to MacOS, and not at all similar to Windows. The Android operating system was build from a modified Linux **kernel**.

Each of us has an account on cs.bridgewater.edu. The username of your account is the same as your Bridgewater College username. Your BC username is the *part* of your email address *before* the @eagles.bridgewater.edu.

To log into cs.bridgewater.edu do the following.

1. In your terminal, type the following command, *omitting* the \$ and *replacing* username with your Bridgewater College username. Then press enter/return.

```
$ ssh username@cs.bridgewater.edu
```

2. You should see a prompt which asks you if you trust the connection. Type yes and press enter.
3. You will then be asked for your password. **Note that when you enter your password the cursor will not move - it appears that you are not typing, but you are.** Enter your Bridgewater College password and press enter.

If successful, you will be connected to cs.bridgewater.edu and you should see a new system prompt.

Set Up Git

Git is already installed on cs.bridgewater.edu, but we need to configure Git so that it knows our identity *and* we need to change the default branch name that Git uses from 'master' to 'main'.

Please use the instructions in the git-scm.com tutorial named [1.6 Getting Started – First-Time Git Setup](#) to set up your identity and change the default branch name from master to main.

Note that you only have to perform the instructions in the sections titled [Your Identity](#), [Your default branch name](#), and [Checking Your Settings](#).

Also note, that you should use your full name when setting user.name, but can use any valid email address when setting user.email.

Finally, if you haven't already, run the following command to verify your Git config settings.

```
$ git config --list
```