Intro Lab

C W Liew

August 27, 2019

Goals

The goals of this lab are to learn about some of the tools and processes that you will be using this semester. This includes:

- how to use (some) commands in the terminal (bash)
  - mkdir
  - gcc
- how to use ssh to connect to a remote machine
- how to use ssh to connect to a remote machine and run windowing programs (ones that create their own windows)
- how to setup ssh with public/private keys so that you do not have to enter passwords
- how to use scp to transfer files between machines
- how to compile C source file(s) into an executable program and run them:
  - compile a C file into an executable
  - compile multiple C files into an executable
  - compile a C file into an object file
  - compile a C file into assembler (X86)
  - compile multiple C object files into an executable (linking)

Resources

Some resources that you can look up (there are many available on the Internet):

- Linux tutorial - `https://www.youtube.com/watch?v=YHFzr-akOas`, `http://linuxcommand.org/lc3_learning_the_shell.php`
- bash tutorial - `https://www.howtoforge.com/tutorial/linux-shell-scripting-lessons/`
- ssh tutorial - `https://www.digitalocean.com/community/tutorials/ssh-essentials-working-with-ssh-servers-clients-and-keys`
- gcc tutorial - `https://courses.cs.washington.edu/courses/cse451/99wi/Section/gccintro.html`

Assignment

Your assignment for this lab consists of completing the following steps (all to be done in terminal):

1. use *ssh* to login to the remote computer (139.147.9.179) and account that have been created for you. Please see the instructor for the account login and password.

2. on your local computer (probably your laptop), use *ssh-keygen* to create a public/private key pair. Use an empty passphrase when generating the public/private key pair.

3. on the remote computer, create a folder (directory) called **.ssh**

4. use *scp* to copy the **public** key (id_rsa.pub) into the **.ssh** folder on the remote computer

5. check that you performed the previous steps correctly by using *ssh* to login to the remote computer. The remote computer should not prompt you for a password. If you are prompted for a password, then something was done incorrectly. You will need to fix it.

6. Download the file **hello.c** from moodle onto your laptop. On the remote computer, create a directory **lab01a**.

7. Copy the file **hello.c** to the **lab01a** directory on the remote computer.

8. On the remote computer in the **lab01a** directory:

   (a) use *gcc* (with the appropriate options - c,s,o,) to:

      i. compile an executable. Test this by running the program (*./a.out*)

      ii. compile an assembler language version of **hello.c**. The generated file is **hello.s**.

      iii. compile an object file version of **hello.c**. The generated file is **hello.o**.

      iv. compile the object file (**hello.o**) into an executable called **lab01**. Run the program (*./lab01*)

   (b) zip the folder **lab01a** using the command *gzip* with the -r option

9. Use *scp* to copy the zip file from the remote computer to your laptop

10. Upload the zip to moodle for your submission to part a.

11. Download the files **sum.c** and **add.c** from moodle to your laptop.

12. On the remote computer, create a folder **lab01b** and copy the files **sum.c** and **add.c** into it.

13. On the remote computer in the **lab01b** directory:

    (a) compile the files **sum.c** and **add.c** into an executable called **add**. Run it.

    (b) compile each of the files **sum.c** and **add.c** into their respective object files.

    (c) compile the object files into an executable file called **add**. (This is also called **linking**).

    (d) zip the folder **lab01b**.

14. Copy the zip file from the remote computer to your laptop and upload it to moodle as your submission for lab01b.

15. Set up your login on the remote computer to be accessible from another account. Assuming this account name is **foobar**, you will need to add (append) the *public key* of **foobar** to the file *authorized_keys* in the **.ssh** folder on your account on the remote machine. Try this with a fellow CS 203 student. Get their public key and add it to your remote computer. Now both you and the other student should be able to login to your account on the remote computer. When you have it working, remove the other student's public key (you don't want to give them permanent access to your account) and add my public key (it's on moodle as liew.pub) to your account on the remote machine.

16. You might it more convenient to edit files on the remote computer. There are two commonly used editors that have GUIs - they are *vim* and *emacs*. The user interfaces are radically different and you can pick the one you like. You can also install (for yourself) any other editor that you want to use. To use programs with GUIs, you have to use the -Y option with ssh as in:

    ```
    ssh -Y liew@139.147.9.179
    ```

    However, your computer must be running an X-server. This comes with MacOS but not with Windows. There are several free and paid software packages for Windows that support this - your mileage will vary.

Hint: It will probably be easier to create 2 terminal windows on your laptop and connect to the remote computer with one of them. This way, you will have a shell on your local computer and one on the remote computer.