CS 137 Lab Guide

Victoria Sakhnini

Fall 2023

Table of Contents:

Note: If you find any errors/typos or have suggestions for improvement, please email vsakhnini@uwaterloo.ca

1 Introduction

This document is very useful to complete A0 and get ready for all future tasks. Reading this document is **strongly recommended**.

In CS 137, we will be writing C programs, which are compiled and linked using gcc, and submitted for auto-marking on the Marmoset testing server. There are many different and acceptable ways to write, compile, test, and store the source code.

2 Password

[Before logging on the computers in the CS labs, you need to change your password. Any accounts with unchanged passwords will be shut down after a week. Once changed, your new password will be used to log into any CS lab Mac or **student.cs** account.]

To change your **student.cs** password, follow this link (https://student.cs.uwaterloo.ca/password/auth/). Enter your WatIAM (Quest) username and password, and click OK.

Read the instructions for choosing a password, then type your password in the first box. When you are finished typing in the box, read the message to the right of the box to determine if your password is acceptable according to the requirements. If it is not, you must try again until you get "**Looks OK!** " Retype your new password in the second box. When you get "**Looks OK!** " you can save your password by clicking the "**Save**" button.

3 Connect to student Linux Environment

3.1 Using the terminal on a Mac

After logging into the Mac, click the "**Terminal**" icon as pictured. It should be found in your dock. If not, open the Launchpad and navigate to Other/Terminal.

Now you want to login to our student Linux environment:

type in "`ssh -X` (optional) `YourQuestUserName@linux.student.cs.uwaterloo.ca`" (for example my Quest username is `vsakhnin`) and press enter. (Note: -X enables X11 forwarding, which gives you the ability to interact with graphics. if you use this option, try typing in "`xeyes`" to see what will happen!)

You should be prompted to enter your password. Enter your password that you just set up previously and press enter.

3.2 Using PuTTY on Windows machines

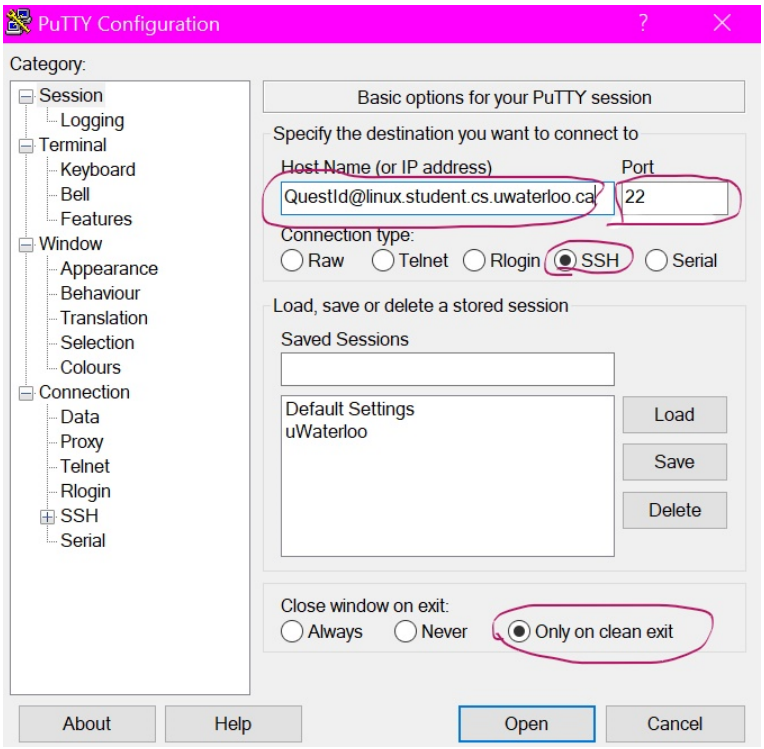
3.2.1 What is PuTTY?

PuTTY is a free and open-source terminal emulator, serial console and network file transfer application. It supports several network protocols, including SCP, SSH, Telnet, rlogin, and raw socket connection. It can also connect to a serial port.

3.2.2 How do we use PuTTY?

First, you need to install PuTTY on Windows.

After installing PuTTY on your Windows computer, you will need to set up a profile as seen on the next page:



Settings for PuTTY:

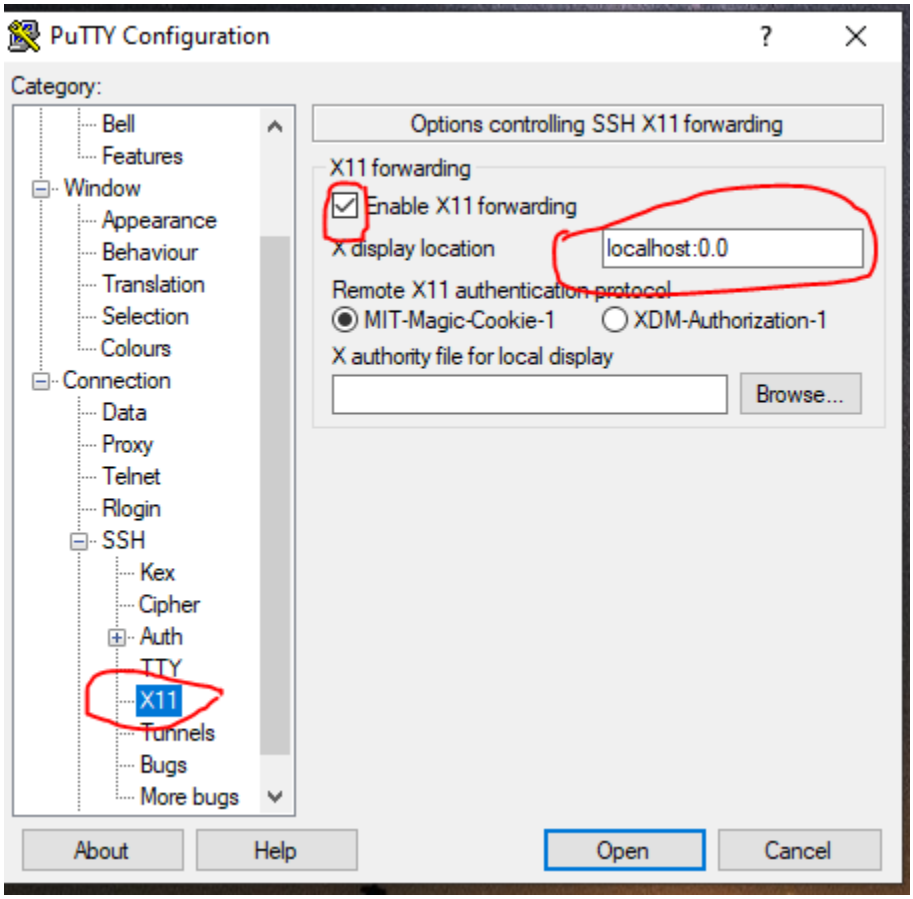
Host name: `QuestUserName@linux.student.cs.uwaterloo.ca` (where "`QuestUserName`" is YOUR Quest username of up to 8 characters) (for example my Quest username is `vsakhnin`)

- Port number: `22`
- Connection Type: `SSH`
- Close window on exit: `Only on clean exit`

Make sure to save the session profile so that you do not have to reinput the settings every time you want to access your student environment. Typically, a window will pop up whenever you first login into a new session, simply click **Accept**. Enter your **student.cs** password when prompted and hit enter (you will not see any keyboard inputs on screen).

Voila, you are now logged into your student environment on your Windows computer.

If you want to enable **X11 Forwarding**, you need to download **Xming** and run it first. Then in Putty you need to ensure the following settings:


PuTTY Configuration
Category:
Bell
Features
Window
Appearance
Behaviour
Translation
Selection
Colours
Connection
Data
Proxy
Telnet
Rlogin
SSH
Kex
Cipher
Auth
TTY
X11
Tunnels
Bugs
More bugs
Options controlling SSH X11 forwarding
X11 forwarding
Enable X11 forwarding
X display location
localhost:0.0
Remote X11 authentication protocol
MIT-Magic-Cookie-1
XDM-Authorization-1
X authority file for local display
Browse...
About
Help
Open
Cancel

4 Basic UNIX knowledge

Before we get into the commands, we want you to understand what is a directory.

A directory is a location for storing files on your computer. Directories are found in a hierarchical file system, such as Linux, MS-DOS, OS/2, and Unix.

In a GUI such as Microsoft Windows, directories are referred to as folders. However, a directory and folder are synonymous.

Some useful UNIX commands

Command	Example	Description
1. ls	ls	Lists files in current directory
	ls -alF	List in long format
2. cd	cd tempdir	Change directory to tempdir
	cd ..	Move back one directory
	cd ∼/cs137	Move into the directory named cs137 in your home directory
3. mkdir	mkdir cs137	Make a directory called cs137
4. cp	cp file1 dir	Copy file into a directory named dir
5. mv	mv old.html new.html	Move or rename files
6. man	man ls	Online manual (help) about command "ls"
7. pwd	pwd	See your current working directory
8. cat	cat file1	display the contents of a file named file1

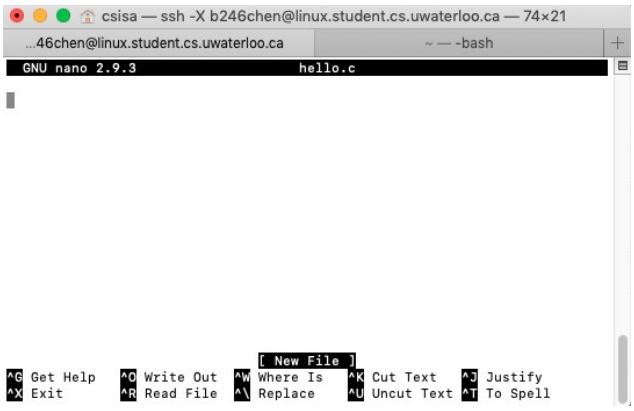
These are seven commands that you really need to know in order to get started with UNIX. They are probably similar to commands you already know for another operating system. I've created an optional chapter **Linux Shell** that includes more commands and details if you would like to learn more about it.

5 Writing and compiling our code in student Linux environment

After you have successfully logged into your student Linux environment:

1. Make a cs137 directory by using the following command "`mkdir cs137`"
2. Now, change directories by using "`cd cs137`"
3. Open an editor called "**nano**" using the command "`nano hello.c`".

(Note: You don't have to use nano, it is just a text editor. Other popular text editors are "**vi**" and "**emacs**")



Write the code for assignment 0 using nano.

1. You can start typing your code right away!
2. Type out your code.
3. Press **Ctrl + O** to save your code. (notice prompts on bottom of Terminal; "^" means the Ctrl key)
4. Press **Ctrl + X** to Exit
5. Press **Y** to save
6. Press **Enter**

Do NOT use software like Microsoft Word and other Rich Text Editing software since they often embed characters not recognized by compilers.

Hooray, you have successfully created your first (or *n* + 1 th) C source code file! Next, we want to compile this code into an executable called "**hello**":

1. Make sure the file you want to compile is saved in your current directory
2. Run the command "`gcc -std=c11 -o hello hello.c`". (if you're curious: `gcc` is the compiler command, `-std=c11` indicates what version of `gcc` we want to use, `-o` lets us enter a name for the executable file (default is **a.out**)
3. If `gcc` is not found, try `ssh` into `studentusername@linux.student.cs.uwaterloo.ca` if not done already
4. Run the executable by using the command "`./hello`" (the "`dot`" and the "`slash`" tell the terminal to look for a program called "**hello**" in the current directory).

6 Submitting the code to Marmoset

6.1 Do it the tedious but easy way

Type "`/u/cs_build/bin/marmoset_submit CS137 a0 hello.c`" into your terminal, and you will be greeted by a confirmation message.

Next time, if you want to submit anything to marmoset, type the exact command except things that you need to change.

For example, "`a0`" needs to be changed to the name of whatever assignment you want to submit and "`hello.c`" needs to be changed to the name of your source code.

6.2 Do it the not so short and difficult way

Change directory to "~" (type `cd ~`) to go back to your home directory.

Using "**nano**" or other text editors to open a file called "**.profile**". and add the following line to the end of the file,

```
alias marmoset_submit=/u/cs_build/bin/marmoset_submit
```

Save the file and quit.

Now type: `source ~/.profile` to run it and update the environment.

Now, type "`marmoset_submit`", you should see the message

"`Usage: marmoset_submit course project file [...]`".

If not, you need to check the steps above.

Now you can simply just type "`marmoset_submit CS137 a0 hello.c`" to submit your file to Marmoset from now and later on.

6.3 Additional Notes

If you are submitting more than one file, you can use the command "`zip`" create a zip file, and change the command accordingly.

ZIP is a compression and file packaging utility for Unix. Each file is stored in single .zip

.zip-filename file with the extension .zip.

zip is used to compress the files to reduce file size and also used as file package utility. zip is available in many operating systems like Unix, Linux, Windows etc.

Syntax:

```
zip [options] zipfile files_list
```

Syntax for Creating a zip file:

```
zip  myfile.zip filename1.txt   filename2.txt
```

7 Marmoset FAQs

What is Marmoset?

Marmoset is an automatic testing server. When you submit an assignment, it will judge your solution and automatically assign a grade. You'll be able to see the grade almost immediately.

How does Marmoset judge my program?

Marmoset will run your program against a few test cases. For each test case, it will compare your program output with the expected output. If the two outputs match EXACTLY, then you pass the test case. Your mark on the assignment problem will be the number of test cases you passed, out of the total number of test cases.

Note: Marmoset has a time limit for how long it will test your code for. This is to prevent inefficient code from slowing down the entire system. Therefore, it is possible to not pass a test case because your code "timed-out". This also emphasizes the importance of efficient code.

How many times can I submit a solution?

You can submit as many times as you like before the due date. However, this is not the best practice for learning, you should debug and test your solutions completely before you submit it. Don’t rely on marmoset to give you test results as you won’t know exactly what went wrong, and won’t test all the cases (check the next question)

What are public, private, and release tests?

There are three types of test cases on Marmoset: public, private, and release tests. We will **not** be using release tests in CS137! Public and private tests both count towards your assignment grade.

Public tests are the ones you see when you've just submitted your program. If you fail a public test, we usually tell you what the test case is. The test cases are usually straight forward. We want to make sure that your program compiles and is basically working.

After the due date, we will reveal the result of private tests on your programs. Private tests are more complicated and sophisticated than public tests. Since you will not see the result of private tests until due date, you should test your code thoroughly before you submit it.

Am I penalized for incorrect submissions?

At CS 137, we take your *best* submission for each assignment problem. It doesn't matter if you submitted once, 3 times, or 30 times. As long as you have one perfect submission, you will get perfect on that problem.

Help! Marmoset cannot compile my program!

You can view your submission to see the compilation error.

So far, the most common cause of compile error is **incorrect filenames**. Marmoset is very picky about the filenames of your submissions, so please read the assignments carefully and make sure that your file has the right name.

Also, the file system on Marmoset is case-sensitive. If we ask you to submit 'hello.c', please ensure that your file name is not 'Hello.c' instead.

Help! I got a 0 on my submission!

The most common cause for a mark of 0 is **output formatting**.

Some students like to prompt the user when they write a C program. So, instead of

```
scanf ("%d", &number);
```

They say

```
printf ("Please enter a number!\n");
scanf ("%d", &number);
```

Absolutely do not output extra text! If an assignment problem has sample inputs and outputs, then your program output must match those, down to the last character!

8. Copying Files

If you are working in the Linux environment provided by the School, there might be times when you want to copy files from this remote machine to your local machine. How you do it will depend on which computer/OS you are using and how you chose to connect to the Linux environment:

8.1 Using the scp command

If you have access to a command prompt that supports the `scp` command you can use it to transfer files back and forth between a local and remote computer.

- Open a terminal and run the command `man scp`. This gives you the manual entry for the `scp` command. Read it.
- At its core, the `scp` command has the following format:

  ```
  scp source destination
  ```

  This will copy the file(s) from the source location to the destination location. Each of source and destination can be of the form [[user@]host1:]file1. Some examples follow:

  - `scp hello.c linux.student.cs.uwaterloo.ca:cs137/`. copies file `hello.c` from the current directory of the local machine (on which this command is being executed) to the `cs137` directory on the l`inux.student.cs.uwaterloo.ca` remote location. It assumes that the `userid` on both the local and remote machines is the same.
  - `scp vsakhnin@linux.student.cs.uwaterloo.ca:cs137/a0/a0.pdf .` will copy the file `cs137/a0/a0.pdf` from `vsakhnin`'s student account to the current directory (note the `.` to indicate the current directory) on the computer this command was executed.

8.2 Graphical Interfaces for Secure File Transfer

Every OS has some form of secure graphical file transfer application. Just search for "secure file transfer application". For example, if you had a Mac, you could search for "Mac OS X secure file transfer application".

Popular applications for Windows are winscp and FileZilla.

8.3 Using SSHFS to mount your student account as a drive

SSHFS (a filesystem client based upon SSH) mounts the specified account as a drive/folder like Samba, but it uses SSH. You will be able to work with the account's files in a Finder window.

8.4 Getting Access:

You just need to be able to SSH into the account.

If you want to use this on your local machine, you will have to install SSHFS first. See

https://www.digitalocean.com/community/tutorials/how-to-use-sshfs-to-mount-remote-file-systems-over-ssh

for instructions covering Ubuntu/Debian, Windows, and Mac OS X machines.

The following instructions are for the Mac, but will be similar for the other operating systems.

1. Open a Terminal and run these 3 commands.

   You don't need to SSH into anything, just run them when the Terminal opens. The third command will ask for your password because it connects to your own account.

   (a) `mkdir -p ~/yourQuestID`

   (b) `sshfs yourQuestID@linux.student.cs.uwaterloo.ca: ~/yourQuestID -o volname=yourQuestID-ssh`

   (c) Type in your password to your account in the CS student environment if you haven't set up an SSH key. (If you need to reset your password, go to https://www.student.cs.uwaterloo.ca/password/ first.)

   Explanation: The first command creates a folder where your account will be mounted. Your files will show up in these folders. The -p means "do nothing if the folders already exist." This only needs to be done once, unless you're working in the Mac labs where your environment is temporary.

   The second command mounts your own account to the folder /yourQuestID. After connecting, an icon may appear on your desktop with the name yourQuestID-ssh that you can double-click upon to see your own files.

2. Your account should show up as drives on the Desktop. If not, they should show up in your home folder (go to the Finder menu bar, click Go then Home). You can disconnect from the account by right-clicking the drive/folder, and choosing Eject.

3. (optional): To save time, you can copy-paste the 3 commands into a file such as sshfs.sh (if you've set up an SSH key, you'll only need the first 2 commands). Then run the commands by typing:

   ```
   bash sshfs.sh
   ```

   This way, you won't have to type the commands each time you log back in to a lab Mac. If you're not using the Mac lab, it's probably easier to just set up the second command as an alias.