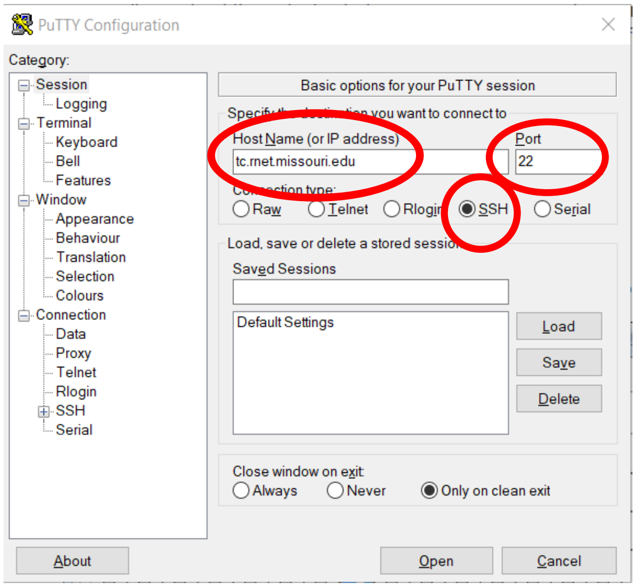
# CS 1050 – Lab 1

**Intro:** For this course we will coding in C and will be programming in a Unix environment (if you don't know what that means you're okay). To connect to the server that our code will be hosted on, we will use PuTTY – this should already be downloaded on all lab computers so there's no installation necessary. It's a free piece of software so you can download it and use it on your personal computers (if it's a Windows device – Mac users will use the built-in Terminal). For this lab, we will be going through some basic Unix commands, writing a Hello World! program, and learning about how to use a File Transfer Protocol (FTP) client.

**Connecting to the Server:**

First, open PuTTY by hitting the windows key (⊞) and then typing "putty" to search for the program. After the program opens up type in tc.rnet.missouri.edu for the Host Name (make sure that SSH is selected and the port is 22 (picture below).



PuTTY Configuration

Basic options for your PuTTY session

Host Name (or IP address): tc.rnet.missouri.edu

Port: 22

Connection type: ○ Raw ○ Telnet ○ Rlogin ● SSH ○ Serial

Load, save or delete a stored session

Saved Sessions: Default Settings

Close window on exit: ○ Always ○ Never ● Only on clean exit

**NOTE:** When prompted if you trust the source you are connecting to hit yes.

After connecting you should be prompted with text that shows “login as: ” – type your pawprint (your pawprint is a string 5-6 with digits and characters – NOT your student number) and hit enter. It will then ask you for your password (this will be the same password that you use for MyZou). Note that when you are typing your password it will not show the yellow box moving – but you *are* still entering it – you just can’t see it.

**Basic Unix Commands:**

Here are the basic Unix commands you will need to know to create directories, navigate through them, and create files/code.

List of commands:

**mkdir *directoryname*** – “mkdir” this stands for make directory will create a new directory (think of it like a folder on your computer) with the name that you give it.

**ls** – “ls” stands for list and will show all of the files/directories in the directory you are currently in.

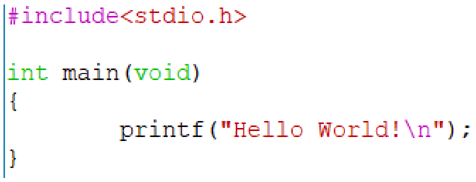
**cd *directoryname*** – “cd” stands for change directory and will move you into the directory that you specify. If you want to move back a directory you would type “cd ..”

**pwd** – “pwd” stands for print working directory and will show you the current “path” of directories you are in.

**vim *filename.c*** – “vim” is a text editor and will be the primary way that we will edit our code. By typing “vim lab1.c” it will create a file called “lab1.c” and will automatically take us into that file. To re-enter a file that currently exists you would still type “vim lab1.c” to enter that file.

**Steps to creating our Hello World! program:**

1. Type vim lab1.c (make sure that the only space that you have is in-between vim and lab1.c – also make sure that you have the “.c” at the end of lab1 – otherwise your code won’t work).
2. From here you will need to hit the letter “i” on your keyboard to be able to type any code. In the bottom right of your screen you should see “-- INSERT --" this shows that you are now able to start typing.
3. From here, type the code shown below *exactly* as it appears – capitalization matters and in most places the spacing will also matter.



```c
#include<stdio.h>

int main(void)
{
        printf("Hello World!\n");
}
```

Explanation of the code:

#include<stdio.h> – will “include” a pre-built library of different functions that we can use – this is not the only library that exists but it’s the only one we need for this program. Libraries are incredibly useful as they save us a *ton* of time since we don’t need to code everything ourselves thanks to them.

int main(void) – think of this as the "driver" portion of our code – when we compile the code it will know to look for a function called "main" in order to know where to start the program.

printf("Hello World!\n"); – printf will print whatever is in the quotation marks. \n tells the code to then add a new line wherever we put it.

4. After typing the code, in order to compile it we will need to exit the code and return to our terminal screen. To do that, hit the ESC key – after hitting the key you should no longer see "-- INSERT --" in the bottom left hand corner of your screen. From here, type :wq this stands for write, quit which will save the code and then exit it. If you ever horribly mess up your code and want to revert to your last save you can type :edit! and it will revert all of your changes to the last save of your code – there is no way to get your changes back after typing that so only do that if you know for sure you want to revert your code.
5. From here, we should be back on our terminal screen (where we were prior to typing vim lab1.c). In order to compile our code, type the following: compile lab1.c – this will make our code "readable" to the computer.
6. After doing that if you type "ls" you should now see an executable called "a.out" – there are ways to rename our executable, but for this course we'll leave it alone. From here, type ./a.out – this will execute our code and you should see Hello World! displayed if everything worked properly.
7. Now you're ready to submit your code! In order to submit the code, type the following command *exactly* as shown.

```
mucs submit 1050 lab lab1.c
```

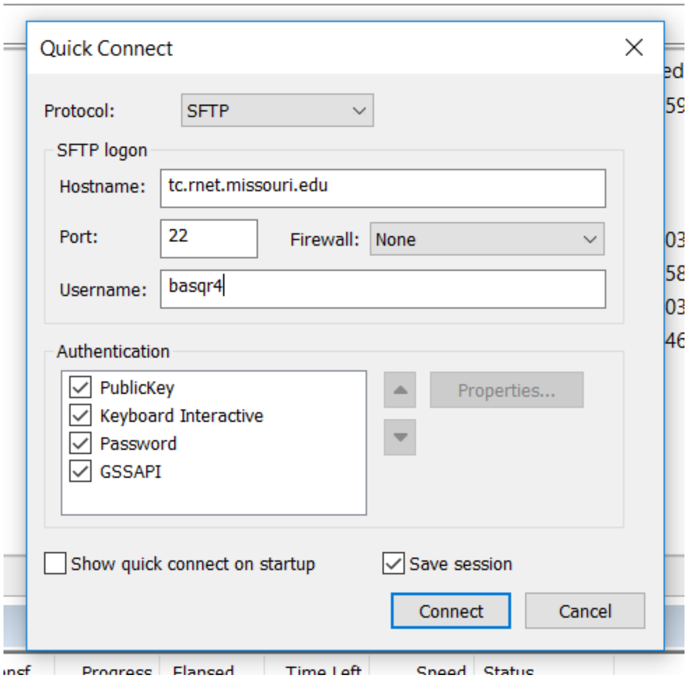
NOTE: if you named your file something other than lab1.c you will need to change "lab1.c" in the above command to whatever you named your file.

8. After submitting you should receive text saying the submission was successful!

### Using a File Transfer Protocol (FTP) client

For this lab we will be using SecureFX as our FTP client since that's what is installed on the lab computers – for your personal computers I'd suggest using FileZilla as that's a free FTP client. The purpose of a FTP client is for moving files between different sources, for us we'll be transferring files from the tc.rnet server to our computers and vice versa.

After opening the program you'll be asked if you want a password – if we were working in industry we definitely would want to set one, but since this is just for class we're fine without one. From there you will have the following box pop up:



Quick Connect

Protocol: SFTP

SFTP logon

Hostname: tc.rnet.missouri.edu

Port: 22 Firewall: None

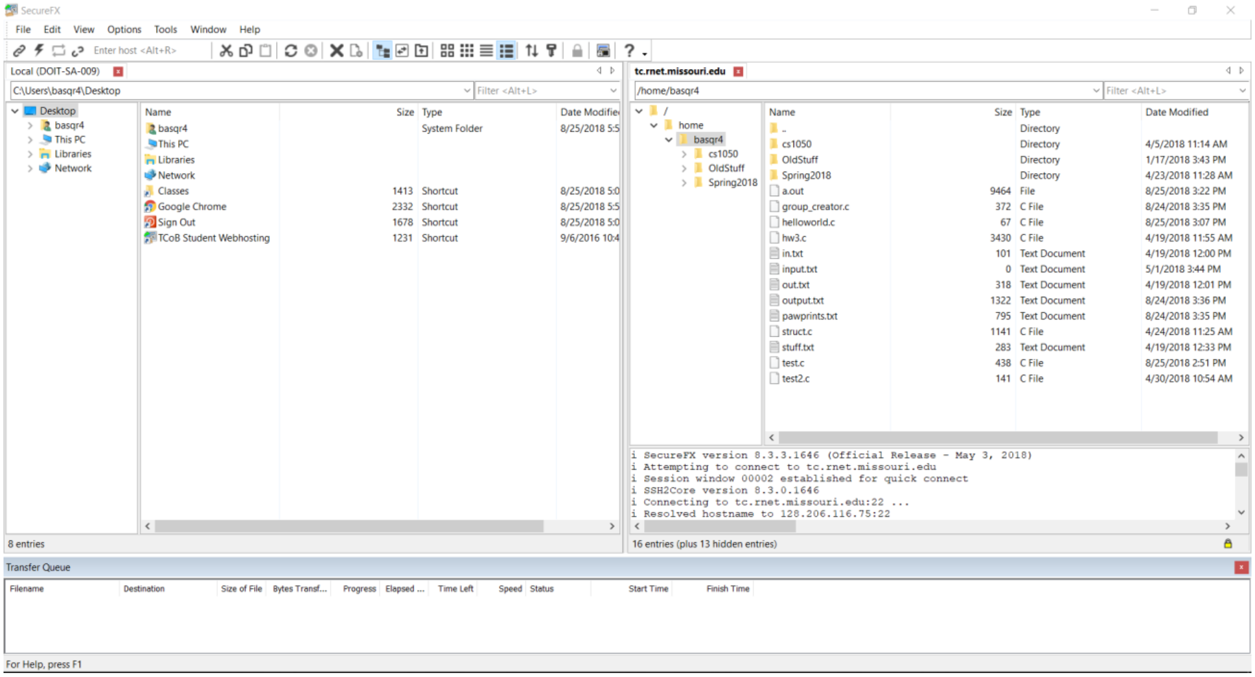
Username: basqr4

Authentication

☑ PublicKey
☑ Keyboard Interactive
☑ Password
☑ GSSAPI

☐ Show quick connect on startup ☑ Save session

Enter the information as shown above (substituting your pawprint for the one I have shown). You'll (likely) be asked if you trust the source – again, click accept & save. From there enter your password.



C:\Users\basqr4\Desktop

- Desktop
  - basqr4
  - This PC
  - Libraries
  - Network

| Name | Size | Type | Date Modified |
|---|---|---|---|
| basqr4 | | System Folder | 8/25/2018 5:5 |
| This PC | | | |
| Libraries | | | |
| Network | | | |
| Classes | 1413 | Shortcut | 8/25/2018 5:0 |
| Google Chrome | 2332 | Shortcut | 8/25/2018 5:5 |
| Sign Out | 1678 | Shortcut | 8/25/2018 5:0 |
| TCoB Student Webhosting | 1231 | Shortcut | 9/6/2016 10:4 |

8 entries

/home/basqr4

- /
  - home
    - basqr4
      - cs1050
      - OldStuff
      - Spring2018

| Name | Size | Type | Date Modified |
|---|---|---|---|
| .. | | Directory | |
| cs1050 | | Directory | 4/5/2018 11:14 AM |
| OldStuff | | Directory | 1/17/2018 3:43 PM |
| Spring2018 | | Directory | 4/23/2018 11:28 AM |
| a.out | 9464 | File | 8/25/2018 3:22 PM |
| group_creator.c | 372 | C File | 8/24/2018 3:35 PM |
| helloworld.c | 67 | C File | 8/25/2018 3:07 PM |
| hw3.c | 3430 | C File | 4/19/2018 11:55 AM |
| in.txt | 101 | Text Document | 4/19/2018 12:00 PM |
| input.txt | 0 | Text Document | 5/1/2018 3:44 PM |
| out.txt | 318 | Text Document | 4/19/2018 12:01 PM |
| output.txt | 1322 | Text Document | 8/24/2018 3:36 PM |
| pawprints.txt | 795 | Text Document | 8/24/2018 3:35 PM |
| struct.c | 1141 | C File | 4/24/2018 11:25 AM |
| stuff.txt | 283 | Text Document | 4/19/2018 12:33 PM |
| test.c | 438 | C File | 8/25/2018 2:51 PM |
| test2.c | 141 | C File | 4/30/2018 10:54 AM |

```
i SecureFX version 8.3.3.1646 (Official Release - May 3, 2018)
i Attempting to connect to tc.rnet.missouri.edu
i Session window 00002 established for quick connect
i SSH2Core version 8.3.0.1646
i Connecting to tc.rnet.missouri.edu:22 ...
i Resolved hostname to 128.206.116.75:22
```

16 entries (plus 13 hidden entries)

| Filename | Destination | Size of File | Bytes Transf... | Progress | Elapsed ... | Time Left | Speed | Status | Start Time | Finish Time |
|---|---|---|---|---|---|---|---|---|---|---|

For Help, press F1

You should see a screen similar to this one if you've completed the above steps. The files on the left side are what's on your computer and the files on the right are what are on your personal portion of tc.rnet – to transfer a file simply click and drag that file from one side to the other!

For the purposes of CS 1050 that's about all you'll need to know for how an FTP client works! Using FileZilla is similar to using SecureFX and you should be able to transfer what you've learned here to FileZilla.