LAB #1 – Introduction/Logon

This document covers several things you need to do before you can start your Computer Science coursework. We will start with logging into the OSU server, then we will do some basic UNIX/Linux commands, then some text editing and finally our first C++ program.

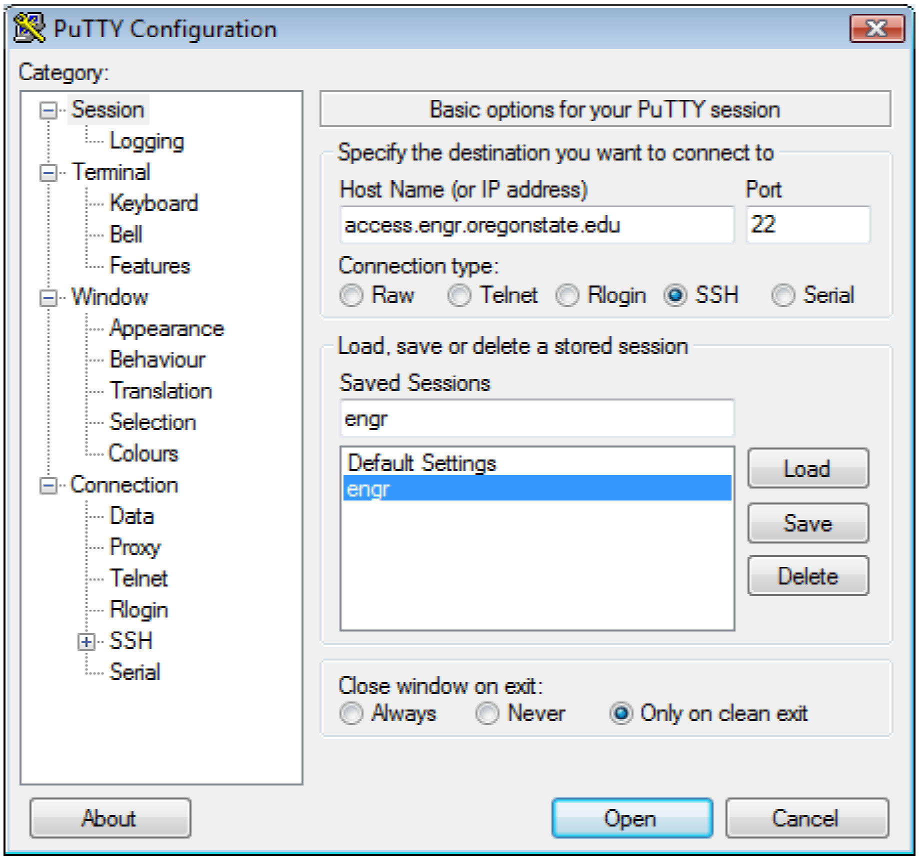
First, you will need to setup your engineering account. To do this, go to the TEACH website and create a new account:
https://secure.engr.oregonstate.edu:8000/teach.php?type=want_auth

Once you have an engineering account, download and install **PuTTY**, the secure shell (ssh) client from OSU: http://oregonstate.edu/helpdocs/software/recommended-software/osuware.

Once you download PuTTY, the file is an .exe that can be opened without being installed.

If you are using a Mac or Linux laptop, then you probably have a terminal with ssh built into the OS. Open the terminal, and type **ssh username@access.engr.oregonstate.edu** at the prompt.
Then type the following address in the hostname: **access.engr.oregonstate.edu.**

I suggest you load the engr host name under a saved session by typing something in the **Saved Sessions** text area and pressing the **Load** button. This makes it easier to save specific properties to the session such as font, background color, text color, etc. Then, you can simply select the saved session each time before pressing Open.



Specify the destination you want to connect to

Host Name (or IP address): access.engr.oregonstate.edu

Port: 22

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

Load, save or delete a stored session

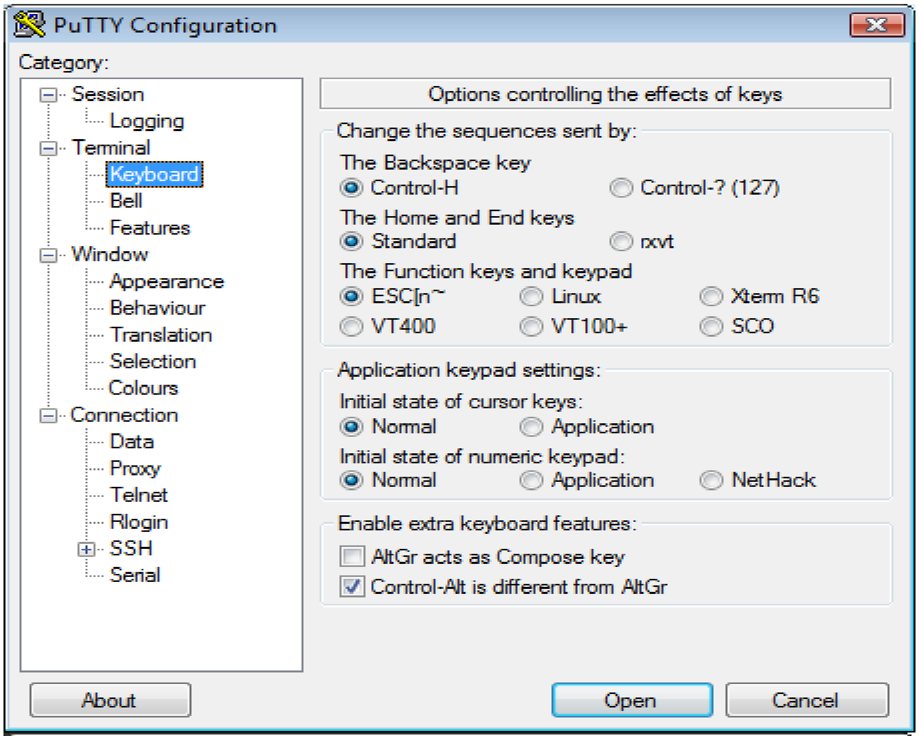
Saved Sessions: engr

Default Settings
engr

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

Two properties you may want to change immediately in the ENGR session is **The Backspace key** located under the **Terminal** -> **Keyboard** tab. This will eliminate those annoying ^? when pressing the backspace key in the vim text editor.

You can also eliminate the **Access denied** message that pops in Putty up after entering your user name when logging into the system by removing the check mark in the **Attempt GSSAPI authentication** box under the **Connection** -> **SSH** -> **Auth** -> **GSSAPI** tab (See first image on next page).



Options controlling the effects of keys

Change the sequences sent by:

The Backspace key
● Control-H ○ Control-? (127)

The Home and End keys
● Standard ○ rxvt

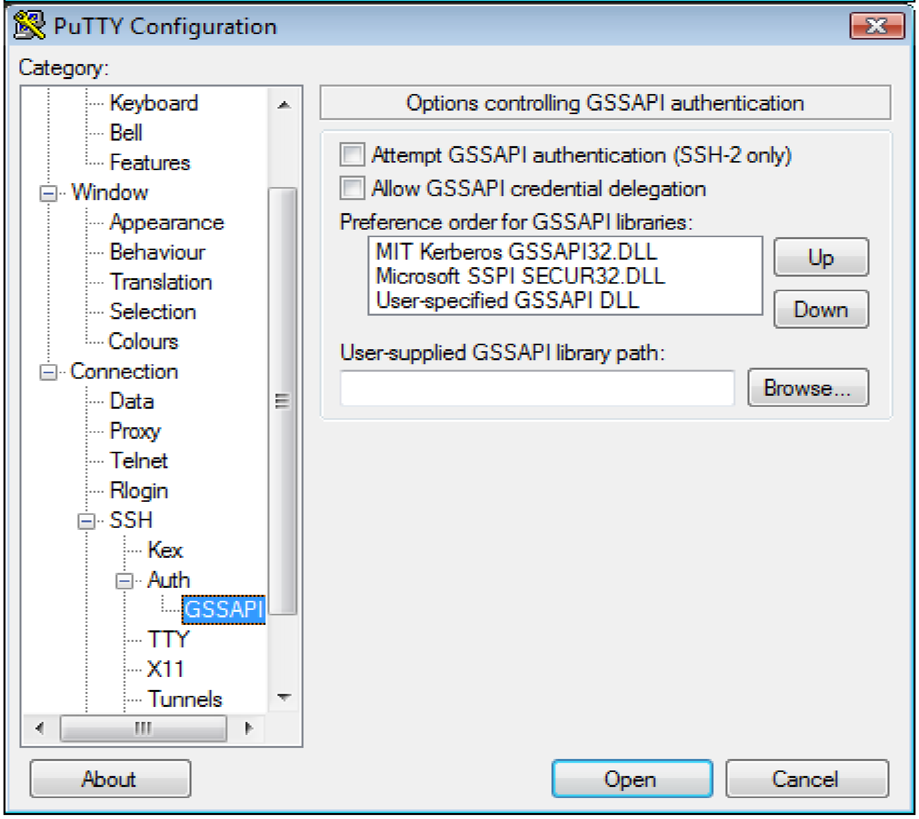
The Function keys and keypad
● ESC[n~ ○ Linux ○ Xterm R6
○ VT400 ○ VT100+ ○ SCO

Application keypad settings:

Initial state of cursor keys:
● Normal ○ Application

Initial state of numeric keypad:
● Normal ○ Application ○ NetHack

Enable extra keyboard features:
□ AltGr acts as Compose key
☑ Control-Alt is different from AltGr



Options controlling GSSAPI authentication

□ Attempt GSSAPI authentication (SSH-2 only)
□ Allow GSSAPI credential delegation

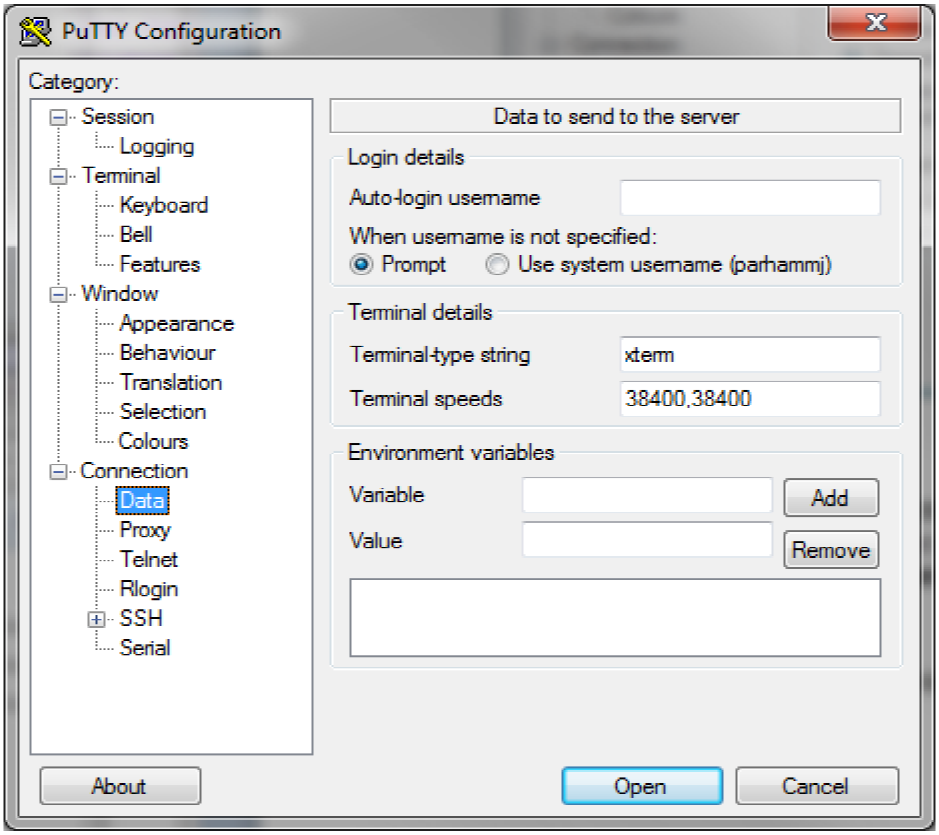
Preference order for GSSAPI libraries:
MIT Kerberos GSSAPI32.DLL
Microsoft SSPI SECUR32.DLL
User-specified GSSAPI DLL

Up
Down

User-supplied GSSAPI library path:

Browse...

Another property you might want to check is the terminal type you are using. Go to the **Connection -> Data** tab, and type **xterm** in the **Terminal-type string** text area (See second image on next page). This terminal type ensures that color coding is used for your programs in the vim text editor. Don't worry if you don't understand what this means right now, you will later.



Login details

Auto-login username

When username is not specified:
● Prompt ○ Use system username (parhammj)

Terminal details

Terminal-type string: xterm

Terminal speeds: 38400,38400

Environment variables

Variable

Value

Make sure you go back to the **Session** tab and press the **Save** button before opening the ENGR session. This will save these properties and ensure that you do not have to do ALL this again.

Now, open the ENGR session you just created, and enter your username and password at the prompt. ****Note**: You will not see anything as you type your password. This is a security feature in Linux!!!

This opens a terminal window, which is similar to a Windows command window that presents you with a DOS prompt, but instead of being on your local Windows machine, you are on a remote Linux machine located somewhere else. This is why you are prompted for a username and password to get into the machine because we don't want just anyone on the ENGR machines!

After a successful logon, you should get a prompt like the one below:
flip1 ~ %

At the prompt, type the following commands to look at your files and directories in your home directory. Note the differences.

```
ls
ls -a
```

ls -l
ls –al

When you use the **ls –a** command, you should notice a '**.**' and '**..**' directory listed. The '**.**' directory refers to your current directory, and the '**..**' directory refers to one directory above your current directory. The '**~**' refers to your home directory.

UNIX/Linux provides you manual pages for the commands. You are encouraged to read these manual pages when you have questions about a specific command or want more details about the options to use with a command.

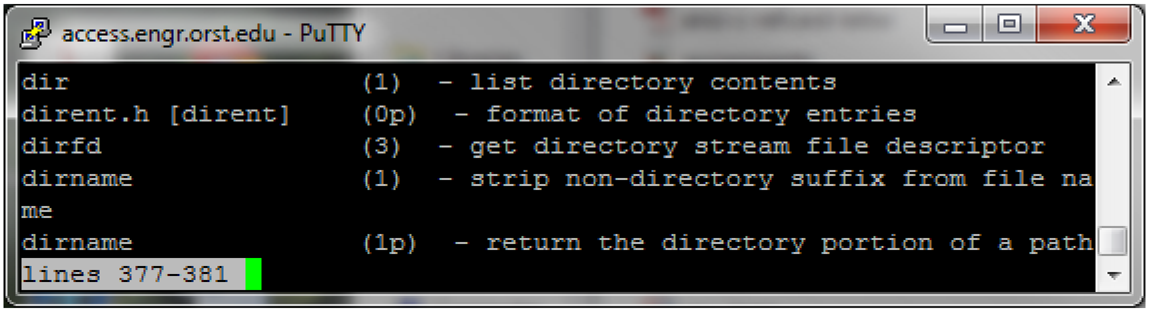
Type in this command: **man ls.** Use the up and down arrows to scroll through the content and note how slow it is. Use the **space bar** to scroll forward through the manual pages one page at a time. Now press **b** to scroll backwards one page at a time. Press **q** to quit the manual page.

UNIX/Linux also provides a way to find out what command to use for a certain task. You can find the appropriate command typing '**apropos**' and a keyword. This command will search the man pages and return commands that contain the keyword.

Type in this command: **apropos editor.** You will see many lines of text which list all of the editing options you have. At the bottom of the list you will see vim. That is an editor we will introduce later.

Now type in this command: **apropos directory**. You will notice that you get more text than what can fit in your terminal window. To view data a page at a time in your terminal, you can **pipe** the command contents through another command called **less**. This will allow you to scroll through the pages using **space bar, b, and q** just as you did with the manual pages.

Try this command: **apropos directory | less**



```
dir                    (1)   - list directory contents
dirent.h [dirent]      (0p)   - format of directory entries
dirfd                  (3)   - get directory stream file descriptor
dirname                (1)   - strip non-directory suffix from file na
me
dirname                (1p)   - return the directory portion of a path
lines 377-381
```

Let's make sure you have your engineer email messages forwarded to an account that you check often. This is where announcements and grading comments/reports are sent. You have several options for doing this.

You can implicitly modify the Linux system file by going through TEACH:
https://secure.engr.oregonstate.edu:8000/teach.php?type=want_auth

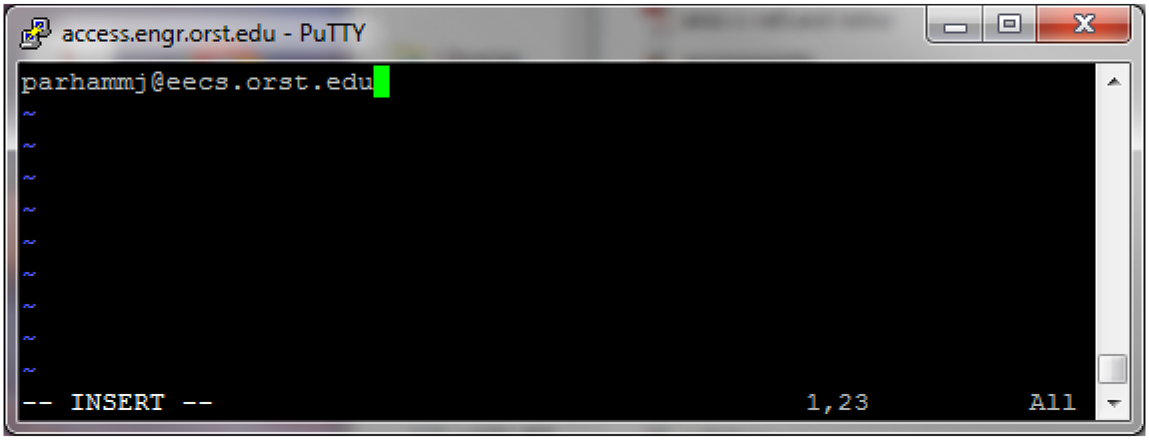
OR

You can explicitly modify/create a **.forward** system file. This system file is used to forward your email to a different location, e.g. **username@onid.orst.edu**. In order to create this file, we are going to use a default UNIX text editor.

One of the default editors that come with the UNIX operating system is vim (**v**isual editor **im**proved). Type in this command: **vim ~/.forward.** This places you in the vim text editor, editing a file called .forward.

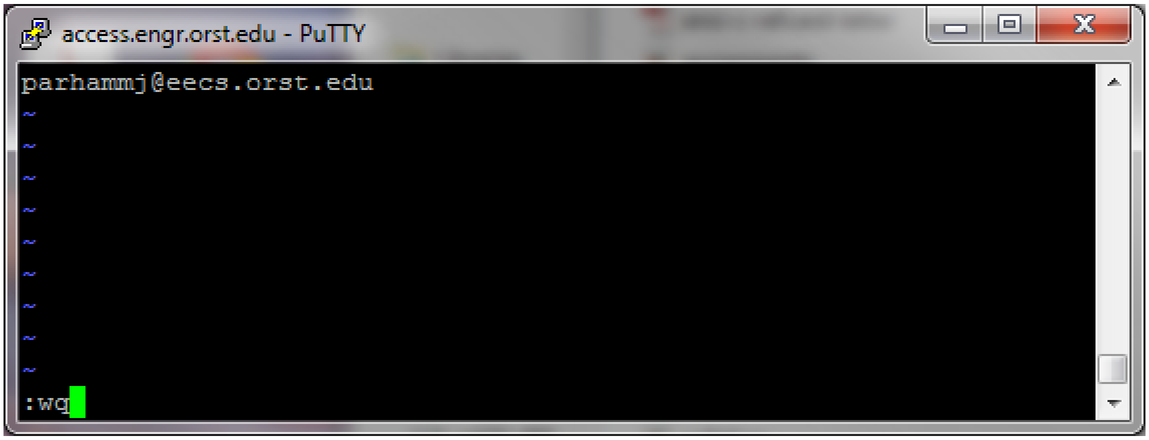
vim has 2 modes we are interested in for now: command and insert. When in vim, press **i** to put yourself in insert mode where you can type text, and you will notice the word -- INSERT -- appear at the bottom of the editor.

Now, type the email address to use when forwarding you engineer email. In this example the user is forwarding engineer email to her eecs email.



```
parhammj@eecs.orst.edu
~
~
~
~
~
~
~
~
~
-- INSERT --                                         1,23          All
```

To save the text you have typed, you need to press **Esc** to go back into command mode, and now you can type **:wq** to write/save the text to the file and quit (press enter after the command). You can also type **:w** and enter to write the file without quitting out of vim or **:q** and enter to quit without saving.



```
parhammj@eecs.orst.edu
~
~
~
~
~
~
~
~
~
:wq
```

Below is an overview of vim and a set of useful commands. Remember you can **man vim** to see the manual pages for vim.

The UNIX vim editor is a full screen editor and has two modes of operation:

- *Command mode* commands which cause action to be taken on the file, and
- *Insert mode* in which entered text is inserted into the file.

In the insert mode, every character typed is added to the text in the file; pressing the <Esc> (*Escape*) key turns off the Insert mode.

Basic Commands (in command mode):
:w<Return> - write out modified file to file named in original invocation
:wq<Return> - quit vim, writing out modified file to file named in original invocation
:q<Return> - quit (or exit) vim
:q!<Return> - quit vim without saving the latest changes
:0<Return> - move cursor to first line in file
:n<Return> - move cursor to line n
:$<Return> - move cursor to last line in file
:set number<Return> - insert line numbers into vi file

/search<Return> - find first occurrence of search from the location of cursor in the downward direction
?search<Return> – find first occurrence of search from the location of cursor in the upward direction
n – move cursor to next occurrence of last search pattern (**in direction of search**)

j [or down-arrow] - move cursor down one line
k [or up-arrow] move cursor up one line
h [or left-arrow] move cursor left one character
l [or right-arrow] move cursor right one character
0 (zero) - move cursor to start of current line (the one with the cursor)
$ - move cursor to end of current line
^ - move cursor to the first non-whitespace character in a line

w - move cursor to beginning of next word
b - move cursor back to beginning of preceding word
u – undo whatever you last did
x – delete current character
dd – delete current line
yy – yank line and put into buffer for pasting
p – paste text in buffer to line below cursor

i – enter insert mode and enter text before the cursor
a - enter insert mode and append text after cursor
o - enter insert mode and enter text on line below cursor
<Esc> - get out of insert mode and enter command mode

After you create your .forward file, make a directory in your home directory named labs, and change into the labs directory.
Type in: **mkdir labs**
Now type in: **cd labs**

Create a directory in your labs directory named lab1, and change into the lab1 directory.
Type in: **mkdir lab1**
Now type in: **cd lab1**

Find out what your current directory is with the **p**rint **w**orking **d**irectory command.
Type in: **pwd**

You can go back/up a directory by using two periods/dots together, and you can go back to your home directory by using the tilde, ~. Use the **pwd** command to confirm you are back in your home directory.
Type in: **cd ..**
Type in: **cd ~**
Type in: **pwd**

Now, change into the labs/lab1directory by using your **up arrow key** to take you through the history of commands you've used in the past. You should see the **cd labs** and **cd lab1** command you typed earlier. You can also change directly into the lab1 directory by using **cd labs/lab1**.

A good rule of thumb is to use **pwd** at any time to determine where you are, in case you forget☺ Also, don't be scared to use **ls** as often as you need to see a listing of your current directory!

Use the vim editor to create a C++ file containing your first C++ program.
Type in: **vim hello.cpp**

Write the infamous "hello world" program as your first piece of C++ code.

```
#include <iostream>

int main() {

  std::cout << “Hello World!” << std::endl;

  return 0;
}
```

Compile and execute your C++ “hello world” program.
Type in: **g++ hello.cpp –o hello**
Type in: **./hello**

You should see “Hello World!” (without the quotes).

Logout of the remote machine.
Type in: **exit (or logout)**

You will need to transfer files from the engr server to your home computer and vice versa. You have two options for this. You can use a free SFTP client such as Filezilla (available at http://oregonstate.edu/helpdocs/software/recommended-software/osuware), or you can map a network drive. This allows you to directly work off the server as if it were a disk drive on your computer. You can follow these instructions to map a network drive for Windows or MacOS.

Windows:
http://engineering.oregonstate.edu/computing/fileaccess/windows_file_sharing/#map_network_drive
MacOS:
http://engineering.oregonstate.edu/computing/fileaccess/smb/

If you want to use the drive off campus, then you must download the **Cisco VPN Client** from OSU: http://oregonstate.edu/helpdocs/network/vpn-campus-access

You may want to put these programs on a flash drive to carry with you in your backpack. This will help you get around from any computer without needing your laptop all the time.