Working on the Linux Machines

Benjamin Mayes
(`bdm8233@rit.edu`)

Revised on March 14, 2012

1 Connecting to the CS Machines

The machines in the ICLs are an incredibly useful tool, not only because they have the majority of software you will need for your classwork but also because they can be easily accessed and worked from remotely.

1.1 Software

1.1.1 Windows

Most students with Windows computers use a free program called *PuTTY* (see google for a download link) for accessing their shell accounts on the CS machines. To use PuTTY you want to select a machine (glados is a commonly used machine but you can use any of the machines from the labs) and use the ssh protocol to connect to the address `<your username>@<machine>.cs.rit.edu`. After you connect you will be prompted for a password, you want to use your CS account password. Many students also install *Xming* and enable X11 forwarding to use GUI software from the machine over the internet by navigating to *Connection* → *SSH* → *X11*, enable the checkbox for *X11 Forwarding* and setting the *X display location* to `localhost:0`

1.1.2 Mac and Linux

Mac and Linux users are able to open up a terminal and use `ssh` to connect to a remote machine, if the command is passed the `-X` flag then X11 forwarding will be enabled (use the command `man ssh` for more information).

1.2 Transferring Files

1.2.1 Gondor (Windows, Mac, and some flavors of Linux)

There are three good methods to transfer files from your machine to the CS machines. The first way is using the `gondor` smb front-end (this is only available on campus). Windows users can connect by typing \\gondor.cs.rit.edu\abc1234\ (replacing abc1234 with your username) in the run dialog (press `windows key+r`). Mac users can hit `command+k` in Finder and fill in the Server Address field with `smb://gondor.cs.rit.edu/abc1234` (again, replacing `abc1234` with their username) and use the link created on their desktop.

1.2.2 SSH-FTP (any)

The second command The other method involves using an SSH-ftp client (which is different from sftp) and connecting to any usable CS machine in the same way you would with PuTTY. *Filezilla* is a good platform-independent FTP client.

1.2.3 `scp` (Linux and Mac)

The third method is not available to Windows users as it involves using the `scp` command from a terminal, for more information on this command see section 2.1.

2 How to use the CS Machines

The command line has a relatively steep learning curve. With that being said in my experience those who know their way around it are more productive programmers. Your mileage may vary.

2.1 Common Commands

All of these examples are safe to type. Some commands such as `rm` can result in a loss of data.

Command	Description	Examples

man	Display a manual page.	`man man`
		`man ls`
		`man -k searchstring`
		`man stdio 3`

ls	Lists the files in a directory.	`ls`
		`ls -l`
		`ls -a`
		`lls -la`
		`lls -Cal /`

cd	Change the current directory you are in.	`cd`
		`cd myfolder`
		`cd ../`

mv	Move file(s) or rename a file.	mv file somefolder/ mv file newname mv ../file .
cp	Copy a file.	cp file1 file2 cp -r dir1 dir2
scp	Copy file(s) to and/or from remote machine(s).	scp abc1234@glados.cs.rit.edu:remote local scp -r ldir abc1234@kansas.rit.edu:rdir
mkdir	Create directories.	mkdir dir mkdir dir1 dir2 mkdir -p newdir1/newdir2
touch	Create empty file(s).	touch newfile.cpp touch file1 file2 touch "a space" file\ with\ spaces touch -d "Jan 1, 2000" oldfile
pwd	Obtain the **p**resent **w**orking **d**irectory.	pwd
cat	Con**cat**enate data in files.	cat file cat file1 file2 > file cd ‘echo "/etc" | cat‘
grep	Search for text in a file or collection of files.	grep --help grep -n "public static void main" *.java grep -in icl /etc/hosts grep -E "int [i|j|k] " *.cpp grep -l "int main(" *.c*
rm	Removes files.	rm --help rm file rm file1 file2 rm -rf directory rm -ri directory
echo	Write text to standard output.	echo echo "Hello, world!" echo -ne "\007" echo "input for a program" | ./prog echo -e "‘pwd‘\n‘ls‘"

Other useful commands: `top`, `ps`, `alias`, `expr`, `fg`, `bg`, `jobs`, `seq`, `test`, `if`, `while`, `for` (for more information on any of these commands view their man page)

2.2 Redirection and Other Tricks

It is possible to output the result of the command into a file, use a file as the input for a command, or use the output as one command as the input for another.

Characters	Description	Examples

`cmd > file`	Redirect program output to file.	`echo "Hello, world!" > hello.txt`
		`cat file1 file2 > newfile`
		`./myprog > stdoutlog`
		`./myprog 2> stderrlog`
		`./myprog &> log`

`cmd >> file`	Append program output to the end of the file.	`echo "Hello, world!" >> hello.txt`
		`cat file1 >> file2`

`cmd < file`	Redirect file to program input.	`cat < hello.txt`
		`./myprog < input.txt`
		`./myprog < in > out`

`cmd1 | cmd2`	Pipes redirect standard output of one program to the standard input of another program.	`cat file | ./myprog`

`‘cmd‘`	Backticks return the result of the output of the command within the backticks. When used alone they will execute the result.	`‘echo "ls"‘`
		`echo ‘ls‘`
		`ls ‘echo‘`

2.3 Useful Keystrokes

Keystroke	Description
ctrl+C	Sends SIGINT to the current process, usually resulting in termination.
ctrl+D	Signifies the end of keyboard input (`eof`). Some programs (`cat`) use this as a termination condition.
ctrl+Z	Suspends a process. The process can later be resumed using the `fg` command or `bg` to run it in the background, `jobs` will list all the current processes that are suspended or backgrounded.
ctrl+S	Suspends the current terminal session. (Not very useful but you will accidentally do it at least once in your life.)
ctrl+Q	Resumes the suspended terminal session.

2.4 Notes

2.4.1 File Names

Filenames are generally case-sensitive on the Linux machines. Additionally certain characters can be present in filenames that prove to be problematic when trying to access the file, the most commonly encountered example of this is the space character. In almost all commands when the space character is typed it signifies the end of the argument and creates another argument. The program sees this file name as two different arguments and does not know it should combine them together. To get around this the concept of an escape character is used. An escape character is a character that is generally unlikely to be encountered in the situation where it is used and signifies that the character that appears after it should be treated differently. In the case of the bash shell the escape character is a backslash. If a space (or any other symbolic character) is preceded by a backslash then it will be treated as the actual character instead of being parsed as a potentially special character.

2.5 Editors

There are a number of different editors present on the Linux machines. Although silly, a student's choice of editor manages to be the topic that causes some of the the most vehement debates between students. Here is my attempt at an moderately unbiased and incomprehensive look at your choice of editors.

2.5.1 emacs

Emacs is an incredibly powerful editor and has an added benefit of having a GUI front-end which will be used if X11 forwarding is enabled. It has less of a learning curve than VIM does but requires that you know a large number of keystrokes to use effectively without a GUI.

2.5.2 vim

Vim is a very light-weight editor that has a very high learning curve. To use effectively you have to familiarize yourself with somewhat confusing keystrokes and commands, a text interface, and a bit of scripting. With that being said, vim has incredibly customizable behavior, built-in support for macros, and a very large number of resources exist. Vim has been around for two decades (its predecessor vi has been around for almost twice that) and remains one of the most popular editors used among Linux users. GUI implementations exist but not on the Linux machines. Vim has built-in spell checking, code completion, and syntax highlighting but these features require some extra knowledge to use. For a crash course in vim you can use the command `vimtutor`.

2.5.3 nano (pico)

Simple and light-weight editor with syntax highlighting.

2.5.4 gedit

The "notepad" of the Linux machines, requires X11 forwarding. Complete with syntax highlighting, a spell checker, and a very simple interface.

3 Warnings, Caveats, and Solutions to Common Issues

3.1 Doing Coding Work from Your Computer

If you insist on doing non-portable (i.e. not Java or Python) coding work from home make sure that it compiles and runs correctly on the CS machines. It is fairly easy to write programs in Visual C++ that g++ will either not compile (namely those dealing with nested templates) or crash (segmentation fault) on the Linux machines but cause no issue in Visual C++ (this is due to different methods of memory mapping and different permissions in your user account). Note that in either of these cases the excuse of "it working on my machine" is not acceptable since in the vast majority of both cases it's the fault of the coder. CS4 is the stage where you, as the coder, have to start being mindful of differences between operating systems and even the differences between multiple architectures running the same operating system.

3.2 Accidentally Deleting Files

If you accidentally deleted or misplaced important files there is a chance that the System Administrators (70-3590, next to the women's restroom) may be able to recover your lost work. Prevention is the best medicine though: make backups, use version control, and lock or your machine before leaving it unattended.

3.3 A Machine is Not Working or is Inacccessible

If you encounter a machine that you cannot login to (physically or remotely) or notice that a machine appears to be running incredibly slowly (and the `top` command shows that some process is bogging down the machine) send an email to `problems@cs.rit.edu` or notify the System Administrators in person. Be sure to include the machine name and the lab the machine is in with a short but specific description of the problem. I have seen turnaround times of less than 10 minutes. **Never shut down a lab computer.**