Chapter 2 Unix Utilities for non-programmers

1

YUAN LONG

CSC 3320 SYSTEM LEVEL PROGRAMMING

SPRING 2019

Updated based on original notes from Raj Sunderraman and Michael Weeks

Getting Started

- Obtain an account -OR- install Linux
- Logging in
  - locally: enter username and password
  - remotely: use the *ssh/telnet* utility to sign on to

  ```
  [GSUAD\ylong4@snowball csc3320]$ ssh ylong4@snowball.cs.gsu.edu
  ```

- Upon logging on you will see a prompt (usually $ or %) which is displayed by a special kind of program called a SHELL program.
- To set your password, use *passwd*

Shells

- Popular shells:
  - Bourne Shell
  - Korn Shell
  - C Shell
  - Bash (Bourne Again Shell)
- All have common core functionality; some differences.
- Each shell has its own programming language. (Shell programming).

Running Utilities

- To run a utility, simply type the name of the utility after the prompt, and press ***Enter*** key
- Some utilities: *date, man, clear, stty, passwd*
  - Utility: date [yymmddhhmm[.ss]]
  - Utility: clear
  - Utility: man [-s section] word

    man -k keyword
- To logout, enter CTRL-D (or *exit*)

Special Characters

- list meta characters with ***stty -a*** command

```
[GSUAD\ylong4@snowball csc3320]$ stty  -a
speed 38400 baud; rows 37; columns 96; line = 0;
intr = ^C; quit = ^\; erase = ^?; kill = ^U; eof = ^D; eol = <undef>; eol2 = <undef>;
swtch = <undef>; start = ^Q; stop = ^S; susp = ^Z; rprnt = ^R; werase = ^W; lnext = ^V;
flush = ^O; min = 1; time = 0;
-parenb -parodd cs8 -hupcl -cstopb cread -clocal -crtscts -cdtrdsr
-ignbrk -brkint -ignpar -parmrk -inpck -istrip -inlcr -igncr icrnl ixon -ixoff -iuclc -
ixany
-imaxbel -iutf8
opost -olcuc -ocrnl onlcr -onocr -onlret -ofill -ofdel nl0 cr0 tab0 bs0 vt0 ff0
isig icanon iexten echo echoe echok -echonl -noflsh -xcase -tostop -echoprt echoctl
echoke
```

- ^ means Control

- ^C means hold down control key (CTRL) and press the C key

Special Characters

Backspace

- erase: CTRL-?   erase character before cursor

- werase: CTRL-W   erase word before cursor

- kill: CTRL-U   erase entire line

- rprnt: CTRL-R   reprint line

- To test these:
  - type some input like "one two three four" at the terminal prompt
  - move cursor around
  - try the above characters

Special Characters

- intr: CTRL-C   interrupt running program
- susp: CTRL-Z   suspend running program
- stop: CTRL-S/CTRL-Q   stop printing to screen
- eof: CTRL-D   give program end of file
- To test these:
  - Try a command like *find /*
  - For *eof*, try *cat > testfile* then **CTRL-D** on new line

Some common UNIX utilities

Utilities	Feature
ls	List files
cat ,more, page, head, tail	Show file contents
mkdir, rmdir,cd, pwd	Working with directories
mv, cp, rm	Working with files
wc	Word count
file, groups, chgrp, chmod	About file permission
vi, emacs	Text editor

Pathnames

- The hierarchy from a starting directory to a target file or directory.
  - E.g.   /home/myFile , ../myFile, ./myFile
- **Absolute** pathname : relative to the root directory
- **Relative** pathname : relative to its current working directory

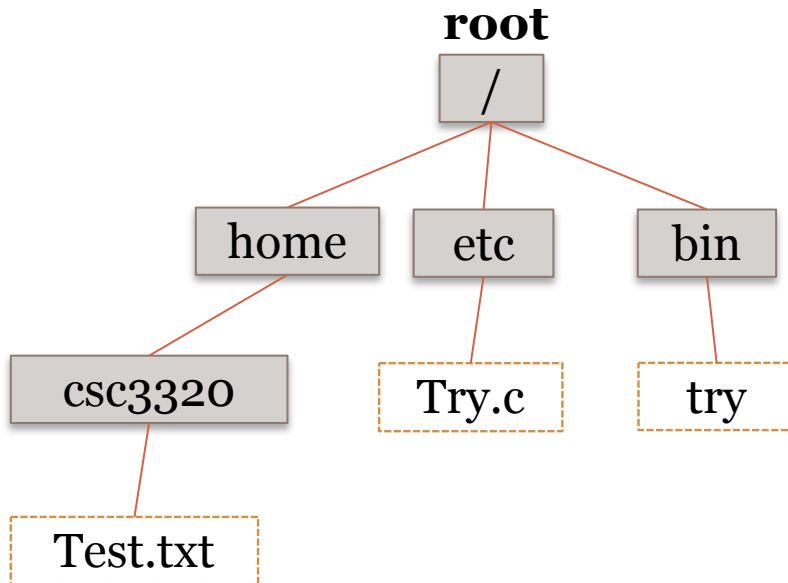```
/   refers to root directory
.   refers to current directory (example: ./a.out)
..   refers to parent directory (example: ../p2.cpp)
```

# Pathnames

**root**

```
        /
    /   |   \
 home  etc  bin
  |     |    |
csc3320 Try.c try
  |
Test.txt
```

File	Absolute Path
Test.txt	/home/csc3320/Test.txt
Try.c	/etc/Try.c
try	/bin/try

/   refers to root directory
.   refers to current directory (example: ./a.out)
..   refers to parent directory (example: ../p2.cpp)

Pathnames

root
/

home   etc   bin

csc3320   Try.c   try
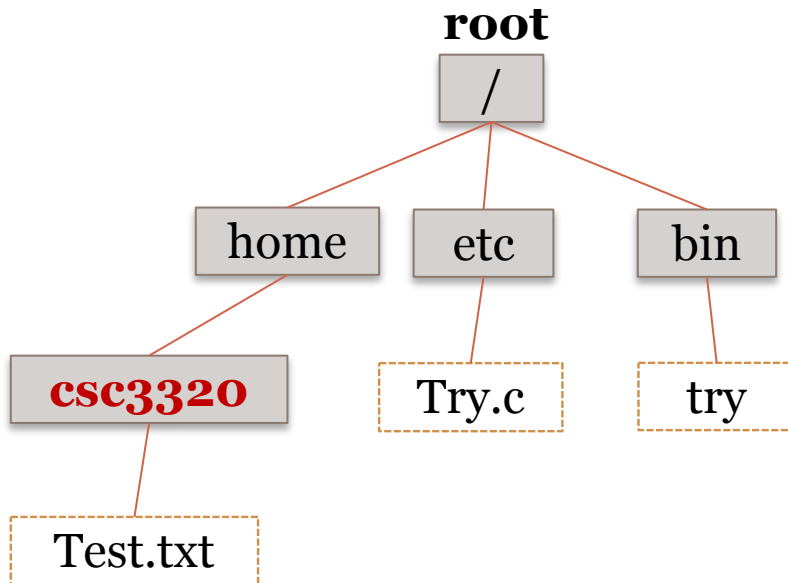
Test.txt

File	Relative Path
Test.txt	Test.txt or ./Test.txt
Try.c	../../etc/Try.c
try	../../bin/try

Assume csc3320 as the current working directory.

/   refers to root directory
.   refers to current directory (example: ./a.out)
..   refers to parent directory (example: ../p2.cpp)

Example of pwd

[GSUAD\ylong4@snowball csc3320]$ **pwd**
/home/local/GSUAD/ylong4/csc3320

[GSUAD\ylong4@snowball csc3320]$ **cd ..**

[GSUAD\ylong4@snowball ~]$ **pwd**
/home/local/GSUAD/ylong4

Example of *cat*

$ **cat > letter**
Hi Mom,

Please send me money!

David
^D

Special Character

$ **ls -l letter**
-rw-r--r--   1 raj   other   38 May 28 11:20
letter

List Files (*ls*)

- $ *ls* -*algFsdR*   *<file-spec-list>*

- options
  - ○ *a*: hidden files
  - ○ *l*: **long listing**
  - ○ *g*: group
  - ○ *F*: put character after file name indicating executable *, link @, directory /, socket =
  - ○ *s*: number of disk blocks
  - ○ *d*: dir details not contents
  - ○ *R*: recursive listing

> - The input can be a set of elements, separated by space
> - **File-spec** is the pathname for directory or file

Show File Contents (*cat, more*)

- $ *cat* *<file-spec-list>*
  - list contents of file(s) on screen without pause
- $ *more* *<file-spec-list>*
  - same as *cat*; pauses after each screen
  - **space bar** takes you to next screen
  - **q** quit
  - **enter** shows next line
  - **h** help key for more commands

Show File Contents (*page, head, tail*)

- $ *page*   *<file-spec-list>*
  - same as *more*; clears screen before each page
  - quicker
- $ *head –n <file-spec-list>*
  - display *n* lines from front of file
- $ *tail   –n <file-spec-list>*
  - display *n* lines from end of file

Working with Directories (*mkdir, cd, pwd*)

- $ *mkdir* *<dir-list>*
  - **m**a**k**e a new **dir**ectory
- $ *cd*   *<dir>*
  - built-in to shell
  - **c**hanges shell to a different **d**irectory
- $ *pwd*
  - **p**rint **w**orking **d**irectory

Rename Files (*mv*)

- Rename files (simply change labels in the file hierarchy)

- $ *mv* -*i* *<old>* *<new>*
  - *i* for inquire
  - (prompt if *<new>* already exists)

- $ *mv* -*i* *<file-spec>* *<dir>*

Move file to a new directory

- $ *mv* -*i* *<dir>* *<dir>*

Rename a directory

Copy Files (*cp*)

- $ *cp* *-i* *<old>* *<new>*
  - *i* for inquire
  - (prompt if *<new>* already exists)
- $ *cp* *-i* *<file-spec>* *<dir>*
- $ *cp* *-ir* *<dir>* *<dir>*
  - *r* for recursive

Removing Files (*rm, rmdir*)

- **r**e**m**ove (delete) file
- $ *rm* -*fir*   *<file-spec-list>*
  - **f**: force; inhibits all prompts/messages
  - **i**: inquire
  - **r**: recursive
- **Be careful!** This is a permanent deletion!
- **r**e**m**ove (delete) **dir**ectory
- $ *rmdir*   *<dir-list>*

Word Count (*wc*)

- **w**ord **c**ount

- $ *wc*   -*clw*   *<file-spec>*
  - **c** counts the bytes
  - **l**   counts the newline characters
  - **w** prints the word counts

File Attributes

```
$ ls  -lsF  test.java
4 -rw-r--r--. 1 GSUAD\ylong4 GSUAD\domain^users 247 Aug 22 13:36 test.java
```

- 4 : number of physical blocks
- -rw-r--r-- : file type (first char), permissions
  - file type (- regular, **d** dir, **b** buffered file disk drive, **c** unbuffered file terminal, **l** link, **p** pipe, **s** socket)
- 1 : hard link count
- GSUAD\ylong4 : file owner
- GSUAD\domain^users : file's group
- 247 : file size in bytes
- Aug 22 13:36 : file modification date
- test.java : file name

File Type (file)

- $ *file* *<file-spec-list>*

- Ascertains the type of file (ascii/binary etc.)

```
[GSUAD\ylong4@snowball csc3320]$ file testDir
testDir: directory
[GSUAD\ylong4@snowball csc3320]$ file Person.class
Person.class: compiled Java class data, version 50.0 (Java
1.6)
[GSUAD\ylong4@snowball csc3320]$ file *
Person.class: compiled Java class data, version 50.0 (Java
1.6)
test:          ELF 64-bit LSB executable, x86-64, version 1
(SYSV), dynamically linked (uses shared libs), for GNU/Linux
2.6.18, not stripped
test.c:        ASCII C program text
testDir:       directory
test.java:     ASCII C++ program text
```

> * matches all files in current directory

Groups (*groups,chgrp,newgrp*)

- list the groups a particular user belongs to
  - $ *groups* *<userid>*
- change the group a file belongs to
  - $ *chgrp -R <group-name>* *<file-spec>*
  - R stands for recursive
- create subshell with effective group id = group
  - $ *newgrp* *<group-name>*

- Example

I belong to two groups: **cs** and **music**

Currently, I am acting as an user in **cs**

  $ **date** > test1

  $ **newgrp** music   **"music" group shell**

  $ **date** > test2

  ^D   **terminate the new shell**

  $ **ls** –lg test1 test2

-rw-rw-r-- 1 **cs**   29 Aug 23 13:57   test1

-rw-rw-r-- 1 **music** 29   Aug 23 13:57 test2

Change File Permissions (chmod)

- change file <u>mode</u>; u,g,o,a r,w,x,s
  - **u**ser, **g**roup, **o**ther, **a**ll
  - **r**ead, **w**rite, **e**xecute, (**s**et)
  - plus (+) adds permissions
  - minus (-) subtracts

Change File Permissions

- Examples :

Assume the file permission is `rw-r--r--`

- $ chmod g+w <file-spec>
- $ chmod u-rw <file-spec>
- $ chmod u+w, g-r <file-spec>

Change parameters	Feature	File permission Result
g+w	Add group with write permission	rw-r**w**-r--
u-rw	Remove user read and write permission	---r--r--
u+x, g-r	Add execution permission for user, and remove read permission from group	rwx---r--

Change File Permissions

- ## Assign permissions absolutely using octal number
  - Example: Set the file permission of one file as **rw-r--r--**

    `$chmod 644 <file-spec>`

	User	**Group**	**Others**
setting	rw-	r--	r--
binary	110	100	100
octal	6	4	4

Change File Permissions

- # Another example with octal numbers

```
mweeks@carmaux:~$ ls -l example.txt
-rw-r--r-- 1 mweeks mweeks 51 2007-08-23 16:06
example.txt
mweeks@carmaux:~$ chmod 765 example.txt
mweeks@carmaux:~$ ls -l example.txt
-rwxrw-r-x 1 mweeks mweeks 51 2007-08-23 16:06
example.txt
```

- # Notice

    - 7 = 111 (binary) r "on", w "on", x "on"

    - 6 = 110 (binary) r "on", w "on", x "off"

    - 5 = 101 (binary) r "on", w "off", x "on"

The vi Editor

- **vi**sual text editor
- Very common on Unix systems
- Created by Bill Joy in 1976
- Many versions exist, including vim
  - vi improved

- Command mode vs Text Entry Mode
  - To enter text entry mode use the following commands:
    - i,I,a,A,o,O,R
  - to get back to command mode use ESC
- line ranges: 1,$ 1,.   .,$.,-2   5,12   .,.+12
- http://www.lagmonster.org/docs/vi.html

The vi Editor

- Cursor Movement:
  - up arrow   k
  - down arrow   j
  - left arrow   h
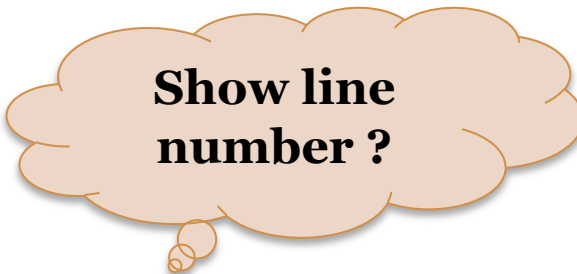 - right arrow   l
  - beginning of line   0
  - first non-whitespace char in line   ^
  - end of line $

- Cursor Movement:
  - next word   w
  - previous word (back)   b
- Goto line number *n*
  - :n
- Go to last line
  - :$

Show line number ?

Answer :set number

The vi Editor

- CTRL-D (down half screen)
- CTRL-U (up half screen)
- CTRL-F (forward one screen)
- CTRL-B (back one screen)

The vi Editor

- Deleting Text:
  - delete character under cursor   x
  - delete word   dw
  - delete line   dd   (also D)
  - delete many lines   :<range>d
- Replacing text:
  - replace char under cursor   r
  - replace word under cursor   cw
  - replace entire line   cc
  - substitute next 4 characters   4s

The vi Editor

- Pasting text:
  - :<range>y   (yank lines)
  - 5yy   (yank 5 lines)
  - p   (paste from buffer after cursor)
  - P   (paste from buffer before cursor)
  - **:n** then **p**   (after line *n*)

The vi Editor

- Searching:
  - search forward   /w
  - search backward   ?w
  - next   n
  - next in opposite direction   N

- Searching/Replacing:
  - :<range> s/old/new/   **Next occurrence in current line(s)**
  - :<range> s/old/new/g   **All occurrences in current line(s)**
  - : ~~<range>~~ %s/old/new/g   **All occurrences in file**

The vi Editor

- Saving/Loading:
  - write to file   :w <fname>
  - write again   :w
  - write   :<range>w <fname>
  - write and quit   :wq
  - edit new file   :e <fname>
  - read (insert) a file   :r <fname>

The vi Editor

- Misc
  - re-draw screen   CTRL-L
  - execute a shell command   :!command
- Quitting:
  - quit   :q
  - write and quit   :wq
  - quit without writing   :q!
  - exit and write if changed   :x

Review

- Shells and Special Characters
- Common Utilities
  - Working with Directories
  - Removing Files
- File Attributes/Types
- File permissions, groups, and owner
- The vi editor