Linux CLI

General
Ch 1

Bash: Bourne again Shell, replaced sh written by Steve Bourne

* bash converts text into operations to be used by the OS.
* terminal emulators communicate with Bash while running a GUI

:D the most recent 500 commands are remembered and can be called on easily!

Multiple console sessions:
even with no terminal running, consoles can be accessed by ctrl-alt-F1 through F6 (one of these will be the GUI.)

File System
Ch 2

Commands

pwd: print working directory
ls: list directory contents
cd: change directory

Structure of Linux

/ root
all other files

Linux is built in a hierarchy file style, similar to a tree branching out with directories and subdirectories.

Absolute vs. Relative Path names

relative path is based on current PWD,
Absolute is straight from root.

relative ./Documents/folder
Absolute /home/user/Documents/folder

Navigation Ch 3

* cd tips!
cd with no args goes to home directory
cd .. previous directory
cd ~username changes pwd to users home

** files in linux do not need an extension, linux knows what the file is.
ex. 123.jpg = 123

* Commands
file - determines file type
less - view contents of a file

command arguments
short options one dash followed by a letter
ex ls -l

long options two dashes followed by a word.
ex ls --help

short options can be strung together
ex ls -alR

Linux Filesystem Break Down
- see Fig a

Symbolic Links
(soft link, symlink)
basically having multiple names for one file.

* hard links, later on in notes

Manipulating Files Ch 4

commands
cp - copy file
mv - move file/rename
mkdir - create directories
rm - remove files
ln - create hard and symlink

Fig b

Wildcards (globbing)
* - any characters
? - any single character
[characters] - any characters in this list
[!characters] characters not in the list
[[:class:]] any character that is a member of the specified class.

Fig c

Classes
[:alnum:] alphanumeric
[:alpha:] aphabetical
[:digit:] numeric
[:lower:] lowercase
[:upper:] uppercase

Fig d

Wildcard Examples

pattern	matches
*	all files
g*	all files starting g
b*.txt	all txt files starting b
Data???	file starting Data with 3 characters
[abc]*	any file with abc beginning

* ranges

** wildcards work in the GUI as well

Commands explained

mkdir: makes directories
Can be used to make 1 directory
* mkdir directory
or can make multiple at once
* mkdir dir1 dir2 dir3 ...

cp: copies file & directories
copy one item to another
* cp item1 item2
copy multiple items to one directory
* cp item1 item2 direct

fig e

Common options

option	meaning
-a --archive	copies atributes of file
-i --interative	promps for overwrite
-r --recursive	copies contents of directories
-u --update	only copies files that are new or nonexistant

Options Con't

-v - verbose displays information message on action.

Pg 29 Examples

mv: move or rename

1 file

* mv f1 f2

multi file

* mv f1 f2 f3 dir1

options

-i

-u

-v

ex pg 30

rm: remove files

* rm dir

* rm dir1 dir2 ...

options

-i

-r deletes directory and all contents

-f forces and ignores non existant commands

-v

ln: create links Pg 32

hard links cannot refrence a directory or a file on seperate partitions

* ln file link hard link

* ln -s file link soft link

Symbolic Links

created to overcome shortcomings of hard links.

works like short cuts :D

Refrence Pg 33 for practice

Play ground notes

Links can be created with the Gnome KDE GUIs

Ch 5 Working with Cmnds

Commands introduced

type - tells how a cmnds name is interpreted
which - display which program will be executed (shows location)
man - manual page
apropos - displays apropriate commands
info - displays commands info entry
whatis - ~~brief~~ brief discription of a command
alias - create an alias command

What are commands?

- one of 4 things

1. executable programs such as /usr/bin. Compiled C and C++ programs or programs scripting languages such as Python or Ruby.

2. Shell builtins. Programs such as cd and [illegible] that are built into Bash.

3. Shell Functions. Mini Shell scripts built into the environment

4. Alias. User defined command

Getting Command Documentation

For Bash builtins, type help [illegible] command*, this brings up the help documentation for that command.

command --help
most commands have a help page

man command
manual page for program

man page breakdown
1. User commands
2. Program interfaces for Kernal system call
3. Program interfaces C library
4. Special files such as drivers/nodes
5. File formats
6. Games and amusements
7. Miscellaneous
8. System Admin

* to search for man section use
man command section

apropos (man -k)
shows possibly helpful man page entries.

What is command
gives a very brief discription of a command.

info command
pulls up GNU info page for command

** Cool trick!
combining commands on a line!
use semi colons to seperate commands.
ex: cd ~[redacted]; mkdir hi; ls hi; rm -r hi

Create a command!

1. See if name is available
   type name
2. create command
   alias name='command here'
   no space between name and "=" or comand

To remove alias use unalias
1. unalias name
Bam! alias gone! :D

* aliases disappear once the shell session is ended, unless appended to the environment config.

Ch 6 Redirection

Commands

cat - concatenate files
sort - sort lines of txt
uniq - report or omit repeated lines
wc - print new line, word, and byte count for each file
grep - print matching lines
head - output first part of a file
tail - output last part of a file
tee - read from standard input and write to standard output files

Standard input/output/error
* everything is a file.
Bash take commands from stdin
prints results to stdout
errors go to stdout

I can redirect commands to print to a file instead of std out by directing the

printing like this:

`ls -alR > ~/Desktop/ls.txt`

this causes the ls to print to a txt file on the desktop.

* this will erase and rewrite the file each time. To append it use >> to redirect.

** file streams vs. file descriptor

stdin	=	0	Bash internally
stdout	=	1	views files as
stderr	=	2	these

Redirecting errors.

same as stdout except 2 must precede >:

`ls / 2> errorlog.txt`

To print both to a single txt file, use 1 of 2 methods:

* stdout must come 1st

1. `ls / > log.txt 2>&1` — redirects stdout then redirects err to stdout (& symbol)
2. `ls / &> log.txt` — redirects both

to redirect errors to a different file, use op 1 and specify file at end.

Redirecting stdin

reverse the arrow back to the command

`cat < test.txt`

instead of taking commands from stdin, the input came from file test.txt.

Piping

takes the stdout of one command and makes it the stdin for another:

`command 1 | command 2`

Filtering

basically piping multiple commands.

cmd 1 | cmd 2 | cmd 3

useful for organizing and processing data.

Tee

Adds a "T" pipe to piping. Prints to a specified file and allows data to flow down the pipe.

ls / | tee test.txt | sort

Ch 7 Seeing the World as the Shell Sees it. (Expansions)

Commands

echo - display a line of txt

Expansion

~~Basically parsing command before executing~~

additional functionality of the shell

examples:

wildcards

~

Arithmetic: $((math))

Brace: { input }

may be nested

very useful for making many files

Parameter

Command Sub: $(command)

uses command stdout as expansion

Quoting

formating in a sense. Selectively supresses expansions

- Double quotes

supresses all except $ \ ` (word split/math) Cmd sub/param/math

Single quotes supress all

escape characters

used to eliminate a single expansion:

bad/file/name

/ is used to escape

* also has special notation

\a	bell
\b	backspace
\n	newline (line feed)
\r	carriage return
\t	tab

Ch 8 Adv. Keyboard Tricks

Commands

clear - clean screen

history - displays history

Cursor Movement

ctrl a	beggining of line
ctrl e	end of line
ctrl f	right → key
ctrl b	left ← key
alt f	forward one word
alt b	back one word
ctrl l	clear

Key/Action

Txt edit

ctrl d	delete character
ctrl t	transpose character pre
alt t	transpose word pre
alt l	lower case word
alt u	upper case word

Cut and Paste

Ctrl k kill (cuts) text to end of line
Ctrl u kill to begining of line
Alt d kill to end of word
Alt backspace kill to beg of word
Ctrl y paste

Completion
tab
works for paths/cmnds/variables
Pg 73 refrence

In Bash man pg

History
hist expansion is pretty cool:
!88 bang plus hist number to be executed

* Searching history
ctrl r begin search
enter execute cmnd
ctrl j copy cmnd to line
ctrl c cancel search

Pg 75 refrence

** Script Check it out

Ch 9 Permissions

*** Cool tip!
Linux is multi user, meaning more than one user may login and use the computer at once

Cmnds
id - display user id
chmod - change a file's mode

umask - set default file permission
su - login as another user
sudo - super user do
chown - change file ownership
chgrp - change file group ownership
passwd - change password

Id's
uid - user
gid - primary group id
groups - groups

* etc/passwd - user acc
etc/group - group info
etc/shadow - passwords

pg 74 ref for rwx
80 also!
82 octal

pg 90

Very important Chapter!

Ch 10 processes

Commands

ps ~~snap~~ snapshot of current processes
top display tasks
jobs list active jobs
bg place job in background
fg place job in foreground
kill send signal to process
killall kill process by name
shutdown shutdown/reboot

10-1 Pg 97

Controlling processes

interupting - nicely asking to terminate (ctrl-c)
background - runs in background (&) (bg %)
foreground - allows for keyboard input (fg %)
pausing - allows for program edit (ctrl-z)

kill [-sig] pid
killall [-sig] program (kills multiple instances)

Ch 11
The Environment

Commands

printenv - print environment
set - set shell options
export - export environment to subsequently executed programs
alias - alias command

* The basic data sets:
env var - everything else
shell var - bash data

Set shows all
printenv shows only environment

** One sys start, bash reads startup files which defines the default env. More in home directory customize to us.

ch 12
V_i

Ch 13
Prompt

Trivial prompt edit
prompt = $PS1

Ch 14
Package Mngmt

Package Management is installing, maintaing, removing etc of software

Repository types
general
testing
development
etc

Dependencies
shared libraries used to support many programs.

High/low lvl package tools

low	high
dpkg	apt/aptitude
rpm	yum
	pacman

man pages will be very useful

low lvl used to check for packages on system?

** Cool :)
looking for which package installed a certain file is possible with low lvl.

Ch 15
Storage Media

Commands

mount - mount filesystem
umount - unmount
fdisk - partition table manager
fsck - check and repair a filesystem
mkfs - make filesystem
fd format - format floppy
dd - write block oriented data directly to a device
mkisofs - create iso 9660 image
cdrecord - write data to optical disk
md5sum - md5 check