Final Notes

Lecture 1

• What is linux:

o Linux is a kernel. A kernel is the core of an operating system.

o A linux distribution is any OS that uses linux kernel.

o It is multitasking OS.

o Largest collaborative project in history

o It’s a modular system, which means all its components are separated from each other.

o There are 2 main linux distribions: Debian and Redhat (Total more than 150)

o Its free

• Arcitecture of linux

o Kernel: the core

o Daemons: programs run in the background independently

o Shells: interface that accepts user input and translate into machine language for kernel

o Graphical Desktop Environment: collection of software that user can see and use

o Linux file structure:

Hierarchical treelike structure

Everything is a file

• Device files

• Directory files

• Binary files

• Regular files

Highest point of the structure is called root (/)

GNU public license

• The GPL is a free, copyleft license for software that guarantees end users the freedom to run,

study, share and modify the softweare.

• 3 versions of GPL: v1, v2, v3

• Linux is released under the GNU GPL v2

• Linux is considered to be open source and Free soft.

UBUNTU

• It is a linux distribution, free for everyone.

• It is suibale for both desktop and server use

• it is shipped in a stable and regular release cycle:

o Regular or Not-LTS: Shipped every 6 months and supported for 9 months

o LTS (Long-Term Support): Shipped every years with the support of 5 years.

• Examples of distribution based on UBUNTU: Linux Lite, linux mint, Elementary OS, Pop OS,

Parrot OS, SteamOS. Kali Linux, linux deepin.

• Linux is modelled on the Unix operating system.

Lecture 2

Virtualization:

• Using multiple operating systems at the same time in a same computer.

• 2 General types of virtualizations: Server-based and Client-based.

• Reduces costs by decreasing the buying costs of physical hardware.

Raspberry Pi

• It is a low cost, credit-card sized computer that plugs into a computer monitor or TV, and uses a

standard keyboard and mouse. It’s capable of doing everything ypu’d expect a desktop

computer to do.

• The Raspberry Pi foundation is registered educational charity based in the UK.

• There are 4 versions of Raspberry Pi.

• Raspberry Pi 400 is the latest version that comes with a dedicated keyboard.

Lecture 3

• GUI:

• DE: Desktop Environment

The Bash Shell

• The GNU bash shell is a program that provides interactive accesss to the Linux system.

• It runs as a regular program and is normally started whenever a user logs in into a

terminal.

• Most Linux distribution use Bash Shell.

Bash Shortcut

(have to include the screenshot)

• Copy and paste: Ctrl shift c and ctrl shift v

• !! – run last command

• !blah – run the most recent command that starts with blah

Shell Prompt

• When you launch a terminal, you see something like this:

ubuntu@ubuntu:-$

- Command history
- DPKG
- APT- Advanced package tool
- Ubuntu PPAs (Personal Package Archieves)

** LINUX Dictionary Structures **

• File system: Linux organizes its files in what is called a Hierarchical director system (tree-

like pattern or folders)

• Filesystem Hierarchy Standard (FHS) specifies requirement and guidelines

Types of pathnames:

• Absolute path: States the full pathname starting from the root (/). Always starts from

the root

• Relative path: Specifies the immediate path or the child path.

CD: Changes the current directory

• Cd + destination

• Single dot (.): represent the current working directory, double dot (..): go one step back;

• Cd../home; cd../.. (two steps back)

• cd ~ (easiest way)

Lecture 4

- cd/ will takes to root

- shortcuts of cd

- cd/usr/share/theme – absolute path (starts on the root)

- ls to see the lists

- cd Music / (relative path to home directory)

- mkdir is to make dir

- mkdir ~/drinks is an easy way to create a dir

- String: holds a data type.

- Mkdir (-p) will be in mids meaning parent folder.

- To create a file: touch filename

-

INODES (index files)

• A data structure that contains all the information about a file except the file name and

its content.

• Every file in the file system has an inode

• To know the the created file location: ls -I ~/filename.

Hard Links

• To create a hard link: ln file ~/Downloads/FileHL

o abd007@cis240-fall21:~$ ln file1 Downloads/file1-hl

o abd007@cis240-fall21:~$ ls -i file1 Downloads/file1-hl -1

o 656004 Downloads/file1-hl

o 656004 file1

o abd007@cis240-fall21:~$

• The hard link needs to be the same file system to create links between them. (ext – ext= link; fat32 -ext= no link)

• It's the direct connection to the file

Soft Links

• Soft connection between files.

• Connect to the hard link and not directly to the data/file.

• To create a symbolic link: ln -s fileName fileNameSL

• The advantage of soft li'nks is that they can point to files that are stored in different positions

o **abd007@cis240-fall21:~$ ln -s file1 softhere**

o **abd007@cis240-fall21:~$ stat file1**

o **File: file1**

o **Size: 0**

Blocks: 0

IO Block: 4096 regular empty file

o **Device: 805h/2053d Inode: 656004 Links: 3**

o **Access: (0664/-rw-rw-r--) Uid: (1000/ abd007) Gid: (1000/ abd007)**

o **Access: 2021-10-20 17:50:16.249998976 -0400**

o **Modify: 2021-10-20 17:50:16.249998976 -0400**

o **Change: 2021-10-20 18:01:29.494487106 -0400**

o **Birth: -**

o **abd007@cis240-fall21:~$ ls -l softhere**

o **lrwxrwxrwx 1 abd007 abd007 5 Oct 20 18:14 softhere -> file1**

o **abd007@cis240-fall21:~$**

Using wildcards/ File globbing

• Wildcard represents letters and characters used to specify a filename for searches

• File globbing is the processing of pattern matching using wildcards.\

• The wildcards are officially called metacharacter wildcards

The * wildcard:

The ? wildcard

• The '?' wildcard meta character matches precisely one character. You might need the

question mark to minimize a long list of files names down to a few.

The [] character

• The brackets wildcard match a single character in a range.

LECTURE 5

Handling Text File

Cat

• The Cat command is used for displaying the content of a file.

• Cat is short of concatenate which is the command intended use.

• It means joining two strings together.

• Example: cat todo.md

Tac

• Tac displays the files from tail to head (reverse order).

• It can also concatenate two files

More

• The more command is a pager program used for displaying the content of a text file one

page at a time.

• Ex: more + file to view

• To display first 10 lines of a file: head /etc/passwd

• To display first 5 lines of a file: head -5 /etc/passwd

• **To Display last 5 lines: tail -5 /etc/passwd**

Cut

• Allows you to extract files from a specific field

• Display the last 5 users: tail -5 /etc/passwd | cut -d ':' -f 1

Sort

• Sort -o filename.txt oldname.txt (to sort the file and save with a new name)

• Sort with numeric data: sort -n filename.txt

• Check if a file is sorted: sort -c filename

• Sort in reverse order: sort -r filename

• Sort by column number: sort -k 2 fileName

• Remove duplicate user: sort -u filename

Grep

• The **Grep** command is used to match a string pattern from a file.

• Example: grep + option + Pattern to match + file

•

Match all lines that start with uppercase letters

grep "^[[:upper:]]" /etc/passwd

Match all lines that end with a digit

grep "[[:digit:]]$ data.csv"

Match only lines containing IPv4 addresses

grep -E

'[[:digit:]]{1,3}.[[:digit:]]{1,3}.[[:digit:]]{1,3}\

.[[:digit:]]{1,3}' ipaddresses

Match one word or the other.

grep -E 'hello|hi' file.txt

Grep can search for pattern sequences using {n}

Search all lines that contain a character repeated 3 times

grep -E "A{3}" file.txt

Search all lines that contain a phone number of the format 973-111-2222

grep "[[:digit:]]{3}[-][[:digit:]]{3}[-

][[:digit:]]{4}" file

The period character (.) is used to represent any single character. For example, search

for all lines that contain any word ending in "able" and has 3 characters before "able".

grep "...able" logbackup.log

• abd007@cis240-fall21:~$ man ls | grep "comma separated"

• output:

-m fill width with a comma separated list of entries

•

I/O (input/output) Redirection

LECTURE 6

VIM

• Open vim > insert mode to write (i) > save (Esc + : + w + filename+ enter) > save and quit (esc +

: + w + q) > overwrite and save (esc + : + q + !)

• To set line number: in commant mode: :set number

• No number: :set nonumber

TAR command:

• TAR command is used to create and extract archive.

• To create: tar + options + archive name + files to add to archive (tar + filename.tar +

• To extract: tar + options + file to extract

• To create an archive: tar -cf example.tar file1 file2 file3

• -f is always required

• * -v display the details. Not required

• To list the files in archive: tar tf filename.tar

• To add new file to the archive: tar rf filename.tar newfile.txt

• To update an existed file after edit: tar uf filename.tar newfile.txt

• To delete: tar --delete -f filename.tar filename.txt

• To move files to a new directory: tar -xf filename.tar -C newDir/

• To display all files including number in name for image and videos: tar cf allfiles.tar *.txt *[0-

9]*.{jpg,png,svg} Video.mp4

• To generate a text file and save the output: lorem > filename.txt

CPIO

• CPIO is used to extract an archive

• To create an archive: ls | cpio -ov > nature.cpio

• To extract:

Ar

• To create an archive: ar r archive.a *

• List of the files: ar t archive.a

• To add new file: ar r archive.a newfile.txt

• To delete a file: ar d archice.a newfile.txt

Gzip

• Gzip filename.txt

• Compress multiple files: gzip file1.txt file2.txt file3.txt

• Compress and keep the original: gzip -k file.txt

• Decompress: gzip -d filename.txt

• Decompress a file from another directory to the present working directory: gzip -dkc <

../otherDirectoryName/filename.txt.gz > presentFile.txt

File Permission:

* To execute a file: chmod u+x filename.sh

* To run the script: ./scriptname.sh

Symbolic mode

• u = user/owner, g = group, o = other

• r = read, w = write, x = execute

• To read, write, and execute: chmod u=rwx,g=rw,o=r

• Example: chmod u=rwe,g=rw,u=r filename.txt (user= rwe, group= rw, other = r)

Numeric mode

• Read = 4

• Write = 2

• Execute = 1

• RWe = 7, rw = 6, rx = 5

• Example: chmod 765 filename.txt (user= rwe, group= rw, other = r)

Lecture 7

Managing user accounts

• It involves adding, modifying and deleting user accounts and information.

• To add user acc: user add or adduser (user add is low level utility, later one is better command)

o **Example: sudo adduser abd007 (then create a password)**

• To modify user info: usermod

• To delete: userdel

o **Example: sudo userdel -r abd007**

• Following files are involved in the user creation process:

o **/etc/login.defs**

o **/etc/default/useradd**

o **/etc/skel/**

o **/etc/passwd**

o **/etc/shadow**

o **/etc/group**

• /etc/login.defs file: grep -ve ^$ /etc/login.defs | grep -v ^#

o **First grep command will suppress all empty lines, second grep will suppress all comments which are lines that start with the # symbol**

• To view the default parameters in the /etc/defau;t/useradd file: useradd -d or cat /etc/default/useradd

• The /etc/passwd file stores information about every avvount in a linux system.

• Each line = a user

• Entries in the passwd file contains I fields divided by a:

• To see students user info: grep student /etc/passwd

To view info about user's acc n passwd: getent passwd student, sudo getent shadow student

• To update passwd for current user: sudo passwd

• To update psaswd for another user: sudo passwd + username

• To lock and unlock account: suto passwd -l or pass -u

• Create a home directory for user: sudo usermod -md /home/abd007 abd007

• Loging with new user : su username

• Logit: exit

• Change the default shell: sudo usermod -s /bin/bash username

Managing Group:

• Cat /etc/passwd | grep "Adrian"

• Cat /etc/passwd | ^"Adrian"

• Sudo groups Adrian

Lecture 8

Shell Scripting

• Shell scirpts are examples of interpreted programs.

• The interpreter used is the BASH shell

• After creating and assigning execute permission, absolute or relative path must be entered to run it.

• Creating basic script:

o Start vim, enable line numbers, enter INSERT mode. (vim script.sh)

o #!/bin/bash

Echo "This is script"

Uname -a

o Save the file and name it (script.sh): :wq!

o To make it executable: chmod u+x script.sh

o To run: ./script.sh

o Echo -n "this is another line" (-n does not output a new line)

• Shell variables

o **Echo “this a script that shows info”**

Ouname -a

Echo “current home directory: $HOME”

Echo “The current shell is: $SHELL”

Echo “This is the user: $USER”

IF/Else statement

• **#!/bin/bash**

if pwd

then

echo “it worked”

else

echo” not working. Try again”

fi

•

• **N1 -eq n2 = checks if n1 is equal to n2**

• **N1 -ge n2 = checks if n1 is greater than or equal to n2**

• **N1 -gt n2 = checls if n1 is gretater than n2**

• **N1 -le n2 = checks if n1 is less than or equal n2**

• **N1 -ne n2 + checks if not equal**

•