Ubuntu-Linuxhint

- `shell` is a tool that is used to communicate with OS, run other programs
- `terminal` is a window, which is used to interact with the shell
- Linux shell and more *here*

Linux file system

- It has a hierarchical directory structure
- `Root` is the basic directory of Linux

Few Example Commands

- `pwd`: print working directory
- `cd`: change directory
- `cd /`: comes back to the previous working directory
- `ls`: list folders and files in the present working directory
- `ls .`: shows all the files and folders on the present directory
- `ls ..`: shows all the files and folders one directory above

Hard Links & Soft Links

- links are a handy way to create shortcuts to an original directory
- can be used to link libraries
- can be used to keep a copy of a single file in multiple places

- It’s a pseudo-file that points to the original files
- types of links:
    1. Hard links
    2. Soft links (symbolic links)
- `Hard links` can link only files, not directories. You can’t link a file from another disk and they refer to the same node as original source. It remains usable even original file is removed.
- `Soft links` can link to directories, files, on different disks also. The link will act as a broken link if original referred file or directory is deleted.

Creating Hard Link

- go to the original directory
- use command `ln <orignalFileName> <hardLinkName>`

Creating Soft Link

- go to the original directory
- use command `ln -s <orignalFileName> <softLinkName>`

List File (ls)

- ls used to list files
- `ls`: list all files and folders
- `ls -l`: list in long format
- `ls -a`: shows all the files including hidden ones (*starting with dot extension*)
- `ls -al`: shows a list of all files including hidden ones in long format
- `ls -lS`: sorts all the list (*the command has the uppercase letter ”S”*)
- `ls -lS > filename.txt`: creates a file with name `filename` and copies all the contents showed through the command `ls -lS` into it

- `man ls`: shows all the information about commands related to `ls`

File Permissions

- while performing command `ls -ls` we will get output like this

  ```
  -rw-rw-r--   1 pank pank 49153 Nov 24 17:57 package-lock.json
  drwxrwxr-x   2 pank pank  4096 Dec 25 12:33 workingFolder
  ```

  - the first column is related to permissions from the list
  - the hyphens are divided as owners, groups, public respectively
  - example: for `-rw-rw-r--`, owners have the privilege to read and write, groups have the privilege to read and write while the public have the privilege to read-only
  - meaning of the letters associated to permissions:
    1. `r`: read
    2. `w`: write
    3. `d`: directory
    4. `x`: execute
    5. `u`: user
    6. `g`: group
    7. `o`: other outside people

example command to add permission for `outside` to write also for the file `package-lock.json`

```
chmod o+w package-lock.json
```

- Then on executing the command `ls -ls`, we will see

```
-rw-rw-rw-   1 pank pank 49153 Nov 24 17:57 package-lock.json
```

example command to remove permission for `outside` to write also for the file `package-lock.json`

```
chmod o-w package-lock.json
```

- Then on executing the command `ls -ls`, we will see

```
-rw-rw-r-- 1 pank 49153 Nov 24 17:57 package-lock.json
```

to change the permissions at once for a user, group, outsider

- number related to the permissions:
    1. `0`: no permission
    2. `1`: execute
    3. `2`: write
    4. `4`: read
- example command: `chmod 754 package-lock.json`
    - here the numbers are related to the owner, group, and outsider respectively. So, the number 7 is for the owner, 5 for the group, and 4 for an outsider
    - If we break down the number 7 then it becomes `4+2+1` which means the owner has access to read, write, execute. Similarly for num 5, we will have `4+1` which means read and execute. And for others, num 4 is for read

Environment Variables

- A variable is a location for storing value. The value stored can be displayed, edited, deleted, and reset.

- Environment variables are the dynamic values that affect the process of a program on a computer. They exist in every computer system and their types may vary
- They can be created, edited, saved, and deleted
- It gives information about the system behavior
- There can be multiple environment variables like `$PATH`, `$USER`, `$HOME`
- To access values stored for each environment variable: `echo $path`
- To access all environment variables: `env`

Create and remove variables

- Create new variable: `variableName=values`
  - example: `PANKVAR=pank111111`
  - accessing the variable: `echo $PANKVAR`
- Removing the existing value: `unset variableName` (*don’t use $ while unset*)
  - example: `unset PANKVAR`

Editing Files

- creating a text file and writing into it: `echo "contents to add here" > filename.txt`
- editing a text file: `nano filename.txt`
- see the contents of the file: `cat filenam.txt`

Pseudo File System dev/proc/sys

- "pseudo-filesystem" means **a filesystem that doesn't have actual files** – rather, it has virtual entries that the filesystem itself makes up on the spot. For example, /proc

Dev (*Devices*)

- list all the devices system has: `ls /dev/` (*pwd: root*)
- The devices are not physical but the kernel has made the entries
- To access any of the devices, one needs to go through the device tree

Proc (*processes*)

- list all the process and their id: `ls /proc/`
- to see the details of the process running: `cd /proc/<anyProcessID> && ls`

Sys

- sysfs is **a pseudo file system provided by the Linux kernel** that exports information about various kernel subsystems, hardware devices, and associated device drivers from the kernel's device model to user space through virtual files.
- It includes all important directories like kernel, firmware
- Inside of the kernel, we can set the flags which can change the behavior of the OS

Find Files

- to search for the file with their name: `find . filename`
    - Dot . simply represents the current directory is is usually where you want to search.
- to search for specific name of the file: `find . -name "filename"`
- to find all the files containing the passed string in file system: `locate <textToSearch>`

Dot Files

- These are normally hidden files

- while navigating through the file systems, a single dot `.` refers to the current dir and dual dot `..` refers to the one directory up which is useful while navigating

Compression and Decompression

- To compress a file: `gzip filename`
- To decompress a file: `gunzip filename`
- create a compressed folder having files name.txt and hello.txt

```
tar cvf compressedFile.tar name.txt hello.txt
```

  - `cvf` means to create, dispaly, fileoption respectively
  - `tar` stands for tape archive, which is used by a large number of Linux/Unix system administrators to deal with tape drives backup.

- decompress a compressed folder

```
tar xvf compressedFile.tar
```

- `xvf` is the Unix-style, short method to implement –extract –verbose –file.

Touch Command

- It can be used to create files
- Can also be used to change the timestamp of the files
- To create a new file: `touch filename.fileformat`
  - The same command will change timestamp of the existing file

Create and Remove Directories

- Create directory: `mkdir directoryName`
- Remove directory: `rm -r  directoryName` or `rmdir directoryName`

Copy Paste Move and Rename Files in Linux

- copy the file contents: `cp filenameToPick filenameToCopyTo`
- rename the file: `mv fileName newName`
- move the file: `mv fileName directoryToMoveTo`

File name and Spaces

- If we use spaces to create files then it will create new files from the separation of the space.
    - To exclude the space in between, we need to use `\` before space or use `“ ”`

Auto-Completion in Linux

- It can b don by using `Tab` key
- Linux file system is case sensitive. To get proper result, make sure to use proper keywords

Keyboard Shortcuts in Linux

- `ctrl + shift + n`: create a new folder
- `Alt + HOME`: navigates to the home
- `Alt + F4`: closes the window
- `Ctrl + Alt + T`: opens terminal
- `Alt + F2`: prompt to write command
- `Ctrl + D`: removes the line

Command Line History in Linux

- list history of commands: `history`
- execute command of particular index: `!indexOfCommand` (*no space after "!"*)
- see small list of the commands only: `history | less` (max *500 commands*)

Head and Tail Commands

- Head gives upper part of the text file while Tail gives lower part of the text file
- Head: `head -number filename.txt`
- Tail: `tail -number filename.txt`
    - number is the num of part or proportion to read from the txt file
- Read 2 files together: `head file1.txt file2.txt`

wc Command in Linux

- It gives number of lines, words, characters from the text files
- command: `wc filename.txt`
- to know only characters: `wc -c filename.txt`
- to know only lines: `wc -l filename.txt`
- to know only words: `wc -w filename.txt`

Package Sources and Updating

- package refers to a compressed file archive containing all the files that come with a particular application
- to update repository: `sudo apt-get update`

Package Management Search Install Remove

- Search a package: `apt-cache search packageName`
- Install a package: `sudo apt-get install packageName`
- Uninstall a package: `sudo apt-get remove packageName`
- Completely remove package and files: `sudo apt-get purge packageName`

Logging

- Log files are a set of records that Linux maintains for the administrators to keep track of important events.

- logs are present in `/var/log` directory
- to see log contents: `cat /var/log/filename.log`
- to manage multiple systems than you can use centralized logins

Services

- Linux service is application running in the background waiting to be used or carrying out important tasks. example: apache, nginex
- list all services running on the background: `service --status-all`
    - `[+]` means that service is running and `[-]` means that service is not running
- check status of specific service: `service serviceName status`
    - green dot in the next line means the service is running

- stop particular service: `service serviceName stop`
- restart particular service: `service serviceName restart`

Processes

- Processes carry out tasks within OS. It's computer program in action
- list process running in background: `ps`

  ```
    PID TTY          TIME CMD
  36728 pts/0    00:00:02 bash
  41226 pts/0    00:00:00 ps
  ```

  - `PID` is unique id given to processes
  - `TTY` is terminal from where it is running
  - `CMD` is basic name of the process
  - Time is in HH:MM:SS
- To run particular process in background: `processNam &` (*It returns the process id*)

Utilities

- Is a tool to run command also known as commands
- example: ls, mkdir, cat, chmod

Kernel Modules

- These are drivers that can be loaded as needed or at boot time
- These are stored in `/lib/modules`
- It is not statically compiled into the kernel
- list all available modules: `lsmod`
  - It outputs column containing name of module, size and information to each drive

- removing modules: `sudo rmmod moduleName`
- installing modules: `sudo insmod pathToInstall/moduleName.extension` (*note that modules are listed in /lib/modules/ directory*)

Adding and Changes Users

- read list of groups on system: `cat /etc/group`
    - we will get output like: `pank:x:1000:`
    - `pank`: name of group
    - `x`: permission
    - `1000`: Id
- add new user: `sudo useradd -d /home/<userName> -s /bin/bash -g <nameOfGroup> -m <userName>`
    - `d`: for directory
    - `s`: for shell (*bash in this case*)
    - `g`: for group
    - `m`: name of user
- check the new user: `cat /etc/passwd`
    - outputs the new user in the list at bottom with group id

User Groups & User Privileges

- create user directly without specifying the group and directory: `sudo useradd -m <userName>`
- verify the user is created: `cat /etc/passwd`
    - outpost the new user in th list at bottom with group id
- create a group and add users to it: `sudo groupadd <groupName>`
- verify the group is created: `cat /etc/group`

    - outputs the new group in the list at bottom
- delete specific group: `sudo groupdel <groupName>`
- verify the group is removed: `cat /etc/group`
    - doesn't output the deleted group

Using Sudo

- stands for `superUser Do`
- to make changes in thee root, to update packages one needs `sudo` privilege

Network UI

- `ip link`: provides the ability to display link layer information, activate an interface, deactivate an interface, change link layer state flags, change MTU, the name of the interface, and even the hardware and Ethernet broadcast address.
    - `LOWER UP`: ensures the link to physical layer is made
    - `UP`: ready to be used
    - one without IP address ensures that the link to physical layer isn't made yet
- `ip addr`: lists the IP address of interfaces and system
- use `man <commandName>` to see the details of the particular command

DNS

- stands for `Domain Name System`
- Is responsible for translating internet domain and host name to IP address and vice versa
- we will have lookup request and lookup response while using DNS
- Forward lookup resolves hostname to IP address

- Revers lookup resolves IP address to the hostname

configure DNS server

  - `hostnamectl`: provides a proper API used to control Linux system hostname and change its related settings.
  - `hostnamectl set-hostname <URL>`: setting up hostname

- referring to the video would be better since i'm also not much familiar with networking concepts. *link*

Changing Nameservers

- referring to the video would be better since i'm also not much familiar with networking concepts. *link*

Basic Troubleshooting

- `ping -c5 8.8.8.8` : check internet usability with 5 req to `8.8.8.8`
- `traceeroute 8.8.8.8 ` : trace al the path
- `dig amazon.com`: get info about the website
- `netstat`: list of the active sockets and internet connections

Informational Utilities

- `arp -a`: address resolution protocol
- `route`: routing table
- `netsta -rn`: routing table

Capture Packets

- we need to install wireshark to do the operation `sudo apt-get install wireshark`
- rest refer to video *link*

IP Tables

- set of rules which defines behavior of the network and machine
- `sudo iptables -L`: view iptables

Netcat

- *things are bit jargon here since i don't have prior idea of networking concepts xD*
- to create server using netcat: `sudo nc -l -p <portnumber>`
- to create client using netcat: `nc localhost <portnumber>`

Installing Apache MySql and Php

- *the section was not relevant at that moment so i skipped the part*

Best Web Editors

- Just list of text editors

BASH Scripts

- Create a file and change user privilege
  1. Create bash script file: `nano filename.sh`
  2. Add `#!bin/bash` on first line. (*first line will always have path to the sell we want to execute*)

3. Add commands each line one

```
#!bin/bash

touch filename.txt
chmod 777 filename.txt
```

- To run the bash script: `bash  filename.sh`
- Make bash script executable: `chmod 775 filename.sh`

Python Scripts

- Need to have python installed on system
- One way to write the scripts in the bash directly with command `python3`
- Other way is to create the file and execute command from there
- Example:
    1. Create a python file with commands
    2. Run it through bash by `python filename`

C Program

- Ubuntu might already have C installed
- To verify run `gcc --version`
- If not have: `sudo apt install build-essntial`
- To run the C program script:
    1. Create a c file: `touch filename.c`
    2. Write the codes in file using `nano filename.c`
    3. Execute the file using `gcc filename.c -o outputFileName` (*need to create output file to show binary file*)

4. See result by `cat filename` or `./filename`

Review

- Really loved the course but i was not able to grasp stuffs from the networking parts