**Operating systems:**

**Windows:**

**Basic Info of powershell:**

Powershell in windows is a command line which is more advanced than the cmd (command promp )

What ever the commands we use in the powershell are the aliases in the other cli’s ( command line interfaces )

To interact with the powershell we can use the 1. Aliases 2. Original powershell commands 3. Command.exe commands ( cmd commands )

Note: we can use cmd commands in powershell but they aren’t as powerful as the powershell commands.

Note: If you want to get help with the cmd commands then use /? Example dir/?

Each process in windows has some streams:

1. Stdout
2. Stderr
3. Stdin

Each command you type in the powershell takes some input and prints some output.

**Navigating through windows: ( commands in the commands section )**

Absolute path the path from the main or parent directory ( from the c: )

Wild card is a character that’s used to help select files based on a certain pattern.

In powershell if you want to do advanced pattern based selection then we can use regular expressions.

**Windows users and groups:**

A single windows machine can have many users in it and different types of users are put together in a group according to their powers. As if each and every user gains access to the system then they can install apps and can bloat the system.

Windows domain is a network of computers,users,files,etc that are added to a central database.

Windows has a local administrator account which has all the controls to the computer but it is disabled by default.

You can create your own account of administrator group instead of using the local administrator account and can use your administrative powers to maintain the system. This possible due to the windows feature called UAC ( user access control ) : A feature in windows that prevents unauthorized changes to a system.

Admin can change the password of the user and can prompt to change password when he/she logs on.

In Windows, files and directory permissions are assigned using Access Control Lists or ACLs. Specifically we’re going to be working with Discretionary Access Control Lists or DACLs.

Windows files and folders can also gave System Access Control Lists or SACLs assigned to them SACLs are used to tell Windows that it should use an event log to make a note of every time someone accesses a file or folder.

DACL can be assumed as a note of which who can use a file and what they can do with it. Each file or folder will have one or more owner and many DACLs.

**Package management in windows:**

All the applications, software and browsers and other such things are properly packages by the developers for us to install the packages come in all sort of shapes and sizes developers package software with the help of software compiling tools and the end result is a package.

In windows software is usually packaged in .exe file where executable file ( .exe ) Contain instructions for a computer to execute when they’re run. Windows software are created in the PE format.PE format exe files don’t contain only executable file but also some text files and also some pictures of that software. Software will also include a file called msi file Microsoft install package ( .msi ) is used to guide a program called the windows installer in the installation, maintenance, and removal of programs on the windows operating system.

Windows executable files are generally used to start or bootstrap the file or program in this case it might just contain a msi file which will start up the installer.

Sometimes executable files can also be used as a standalone file to bootstrap, run and execute a file or program.

If you want to take care of all the file dependencies of that file then you can use the standalone exe file. But if you want Windows to take up the responsibility of the file dependencies then we can use file guided by msi file.

Microsoft introduced a new platform to install programs called windows store. Windows store is an application repository warehouse where you can download and install windows store apps these programs use a format called APPX to package their content.

With the help of powershell we can automate the installation process.

**Archive**: comprised of one or more files that’s compressed in to a single file

Package archives: the core or source software files that are compressed into one file

Ex of package archives: .zip, .rar etc

Anything can be archived for example pics, vids etc.

Packages or software usually rely on other pieces of code.

A complex software may depend on many different types of small code work to work properly it has its dependencies on different types of small code work.

**Having dependencies**: Counting on other pieces of software to make an application work, since one bit of code depends on another, in order to work.

Libraries are a way to package a bunch of useful code that someone else wrote. In windows these libraries are called dll’s or dynamic link libraries. A single dll can be used by lots of many programs so to run multiple programs lot of memory is not needed as there will be no need to saving many dll’s in the memory. Generally a windows program will have many dependencies and all the dependencies are generally handled by the msi file in the installer package and the msi file will handle all the needs of the dependencies of that program

In windows these days dlls are handled by the side by side assemblies or sxs most of these files are stored in the c:\windows\Winsxs

To install a package we need to specify the package source first. And the best package source is chocolatey.

**Package manager:** A Package manager makes sure that the process of software installation, removal, update, and dependency management is as easy and automatic as possible.

Underneath the hood: Whenever we click on an installer media or an installer .exe file then what ever happens is based on the developers code and most of the programs won’t allow to look at the source code of what is happening but if the installer file has .msi or the exe file is being helped by the msi file then the msi file has some instructions or rules of how to execute the installer. The windows installer can understand their instructions the msi file is not an ordinary file it also has some databases and some instructions which the windows installer will keep track of in other file and make some instructions and this enables the users to uninstall the program whenever necessary. We can see what the program is doing with the help of Microsoft’s sysinternalstool has a program called process monitor which helps us to monitor what the installer is doing under the hood and what files it is changing or creating while installing the software.

The windows updates are handled by Microsoft get updated as cumulative which means the updates need not be updated from the beginning of the update but the latest updates are superceeded.

Windows updates are crucial for security purposes as Microsoft sends patches for the security holes and these updates are also crucial for new features and these days even driver updates are rolled out via Windows updates.

**File Systems:**

**Filesystem:** Filesystem in operating systems is used to keep track of files and file storage on a disk. Without file system operating system wouldn’t know how to keep track of files. Whenever we are using a new disk or storage device we need to add a file system.

Windows operating system uses NTFS file system. NTFS file system supports huge files ranging from 4gb and higher and NTFS supports disks more than 32gb.

NTFS file system is not supported for mac so the file system to be run on both mac, linux, and windows there is a file system called FAT 32 which is inferior to the NTFS but gets the work done.

To have supports for large sizes and inter-compatibility we can use a centralized storage solutions ( servers ).

A storage disk can be divided into different partitions a partition is nothing but a disk divided into different disks these are mostly used if we are using two different oses.

**Partition Table:** Partition table tells the OS how the disk is partitioned. How much space is used and allocated to it. which partition we moved from.

There are two main partitioning tables used

1. MBR ( Master Boot Record )
2. GPT ( GUID Partition Table )

MBR Partitioning table is used widely and it supports only max of 2TB and we can create only 4 primary partitions and these partitions need to be shrinked then made into logical partitions.

GPT Partitioning table is being used in new hardware and it solves all the problems of the MBR table. These partitioning is mostly used in UEFI bios.

During partitioning a drive there is an option called enable file compression which makes the drive to store info in zip format and whenever we want to use or read those files we need to extract them which burdens the CPU. Allocation unit size is the size of the block which will be used by the files to get stored in them. When ever the file is stored then the file will be stored in these pieces and the info will be retrieved from them. If we are saving small files then less allocation unit size is used so that less space is used and whenever we are dealing with large files then large allocation unit size is used so we only read some blocks of data by default the value is set to default and this is fine for general use. While formatting if we don’t use quick format then windows will check for errors and bad sectors in the disk.

**Mounting and Unmounting disks in Windows:**

Mounting is making something accessible to the computer, like a filesystem or a hard disk. Windows does this for us automatically. After using the file disk can be unmounted or ejected.

**Virtual Memory:** Virtual memory is how our OS provides the physical memory available in our computer to application that run on the computer this is done by mapping virtual to real memory addresses. Virtual memory allows us to use more memory than we have installed onto our computer this is done by allocating some space on the disk which is used to store inactive or seldom used services on windows and whenever they are used they are retrieved from the hard drive in windows this service is managed by memory manager the pages are saved in c:\ drive in special directory called page.sys windows automatically creates pages and manages it. Windows also allows to allocate how much amount of disk needed for paging through system properties.

**Files:** Files in Operating System are managed by File system and the file data (data in the file) and file meta data(name, location, owner, etc.) are managed by it. NTFS file system uses master file table (MFT) to keep everything straight. Every file in the volume has its entry in MFT including MFT itself usually there is a one to one correspondence to files and MFT itself if a file has a whole lot of attributes then there will be more than one record in the MFT whenever the files get deleted then the entries in the MFT get deleted and new files can replace those empty places. Whenever a file has its entry in the MFT then file will have an identifier called file record number which is the index of the files entry in the MFT

Shortcuts in the Operating are just an other files in the MFT but it has some reference to another file

NFTS also provides shortcuts with HARD and SYMBOLIC links

**Symbolic links** are type of links equivalent to shortcut to a file where the shortcut acts like substitutes of the original file.

**Hard links** are type of links to a file where the shortcut stores the file record number instead of the files original name in this way if name of the file gets changed then also the hard link can point to that file.

**Windows Disk usage:** The size of the disk used by files can be monitored by the computer management tool

It shows how much of the space is being used and how much is free to use in the Windows disk usage there is a tool called cleanmgr.exe to clean up the disk, old windows installation in that disk and what so ever.

It also has a tool called disk defragmentation to take all the files stored on a given disk, and reorganize them into

Neighboring locations which makes hard drive to work faster and efficiently but not to ssd’s but operating system also has trim for ssd’s

**Filesystem Repair:** Whenever we are copying files and folders to a thumb drive then the OS prompts us to eject the drive before removing the drive this is due to the DATA buffer system. DATA buffer system is a region of RAM that’s used to temporarily store data while it’s being moved around as RAM is much faster than hard drive. Now even if copying files to another drive completes there are chances of copying DATA buffer to the drive and during this period if power goes off or what ever happens then there will be chance of data loss or data corruption.

**Data corruption:** Data corruption in Windows is easily handled by NTFS file system as NTFS file system creates a log of changes done to the files and if an error occurs then the NTFS reverses the changes by referring the log created before this is called self-healing.

**Process Management:**

**Programs:** Programs are the software applications

**Processes**: Processes are the programs running a single program can have many processes.

Whenever we open up a program that process is gives a Process ID and with the help of the process ID kernel figures out which hardware resources are needed to run that process and the resources are assigned to that

Background processes/Daemon processes: are the processes that run in the background which are crucial for functioning of a system.

When windows starts up or boots up the first non-kernel process is session manager subsystem ( smss.exe ) which then starts winlogon.exe along with the Client/Server Runtime Subsystem ( csrss.exe )

In windows each new process needs a parent which tells operating system that new process needs to be made. The child process inherits some properties from parents like variables and settings which we can describe as environment. Which gives the child process a good start-up and after some time the chile process is on its own way. Unlike in linux processes in Windows chile processes can operate on themselves.

Process is made when a silent code ( program ) is poked and the silent code now shines. In Windows to check the process information we can use taskmanager or taskmgr.exe

**Managing processes:**

Process Explorer is a utility Microsoft created to let IT Support Specialists, system administrators and other users look at running processes.

**Resource Monitoring:**

**Powershell Commands:**

|  |  |  |
| --- | --- | --- |
| Si no. | Command | Descripiton |
| 1) | Ls ( list directory ) | ls command is used to list the directories in the opened directory |
| 2) | Get-help ls | Gives the brief summary of the command parameters |
| 3) | Get-help ls -full | Gives the brief and description as well as the examples of the command parameters |
| 4) | ls -Force | Shows the hidden files in the directory |
| 5) | pwd | Prints the working directory |
| 6) | cd | Change directory to go up a level or a down a level |
| 7) | cd.. | To down a level or to go back |
| 8) | cd..\ desktop | To change to directories in the directory |
| 9) | cd~\desktop | To change to directories in the directory |
| 10) | Tab completion | Write the first letter of the directory then press tab to automatically complete the name of the directory |
| 11) | mkdir | This command is used to create new directories |
| 12) | # tab | To scroll through previously used commands |
| 13) | Clear | Clears the powershell |
| 14) | History | Shows the history of the previously used commands |
| 15) | Cp | To copy files from one directory to another cp file then path to which where you want to copy ( doesn’t copy the inner files in the directory ) |
| 16) | Cp \*.py | To copy all the files which have .py |
| 17) | Del | To delete files from the directory |
| 18) | Cp -recurse | To copy the directory along with the inner directories within it |
| 19) | -verbose | To get the warnings or the intimations that work has been executed or has failed to do while doing some so and so task |
| 20) | Mv | To rename or move directories from one directory to another |
| 21) | rm | To remove the files or directories |
| 22) | cat | To view data in the file |
| 23) | More | To view data in a systematic order |
| 24) | Cat -head | To view only first few lines of the file |
| 25) | Cat -tail | To view only last few lines of the file |
| 26) | Select-string or sls | To look a particular data in the file |
| 27) | -filter | To look up some files in the directory containing some common name or suffix. |
| 28) | Echo | To print the information on the shell |
| 29) | > | To save the output of a command into a file  Note: each command in gives out output in the output stream of the windows so instead printing the output of the command on the powershell we can save the output to a file.  Note: If the file exists then the data will be sent into that file else a new file will be created. |
| 30) | >> | To append the output of a command into a file instead of re-writing it. |
| 31) | | pipe operator | To send output of one command into input of another command. |
| 32) | Get-help about\_redirection | Gives info about the redirection operator in windows powershell. |
| 33) | New-item | To create a new file in powershell. |
| 34) | Get-content | To print the content in the file |
| 35) | Set-content | To write information into the file |
| 36) | Compress-archive | To compress the files in the powershell |
| 37) | Get-alias | To see what the actual powershell command gets executed when we use the alias of that command. |
| 38) | $null | Stands nothing in powershell. |
| 39) | Get-localuser | To get info of local users active and description of them. |
| 40) | Get-localgroups | To get info of local group active and description of them. |
| 41) | Get-localgroupmember | To know what users are in that group.  Note: get-localgroupmember has parameter which takes group name and gives out desired output. |
| 42) | Set-localuser | Set |
| 43) | Compress-archive | To compress or archive a file or folder |
| 44) | Find-package | To locate software dependencies. |
| 45) | Register-packagesource | To register package source. |
| 44) | Get-process | To get list of processes running in windows |
| 45) | Uptime | Shows how time is the pc been running |
| 46) | Date | Shows date and time of the PC |
| 47) | $psversiontable | To know version of powershell. |

**Command.exe Commands ( Command prompt commands )**

|  |  |  |
| --- | --- | --- |
| No. | Command | Description |
| 1) | Diskpart | To format the drives in the command prompt.  Note: This will open another terminal window which will read diskpart. |
| 2) | List disk | To list the disks connected to the pc. |
| 3) | Select disk 1 | To select the disk 1 to format. |
| 4) | Clean | To remove all partition and file formatting from the disk. |
| 5) | Create partition primary | To create a primary partition |
| 6) | Select partition | To select the partition of that disk. |
| 7) | Active | To set the disk as active. |
| 8) | Format fs=NTFS label=”name” quick | To format the drive with ntfs file type drive name as name and format method quick. |
| 9) | Mklink | To make symbolic or hard links to files.  Note: to make hard links /h is used. |
| 10) | Fsutil | To check the process of self-healing. |
| 11) | Chkdsk | To check the drives for any errors  Note: if you want to solve the error then you can use /f to solve. |
| 12) | Taskkill | To kill the task in CMD  Taskkill /pid pidno. To kill the task regarding to the pid no. |
| 13) | Tasklist | To get list of tasks running |
| 14) | shutdown | To shutdown the pc |
| 15) | Shutdown /r /o | To get into advanced boot menu |
| 16) | Shutdown /r /o /f /t 00 | To get straight into the advanced menu without restarting |
| 17) | Bootrec.exe /fixmbr | To fix mbr partitioning table which has caused windows boot loader to not work |
| 18) | Bootrec.exe /fixboot | To fix boot on efi systems. |

Situation1: When the boot of efi system is not working and Bootrec.exe /fixboot is not working.

St1: Diskpart (cmd)

St2: Listdisk (cmd)

St3: select disk 0 (cmd)

St4: list vol (cmd)

St5: select vol 1 (cmd)

St6: assign letter=v (cmd)

St7: go to V:\

St8: md \efi\microsoft\boot (cmd)

St9: bootrec /fixboot (cmd)

St10: bcdboot C:\windows /l en-us /s b: /f all (cmd)

St11: exit

St12: bootrec /fixboot (cmd)

Situation 2 to remove grub from efi systems

St1: Open Command prompt with adminstration privileges

st2: Diskpart

St3: List disk

st4: select disk

st5: select partition 2 ( system partition )

st6: assign letter=x ( to mount the partition )

st7: move to x:

st8: then list the directories in x:

st9: then go for uefi directory.

st10: then delete linux/android directory with rd ubuntu /s then press y to continue then restart.

Situation 3: linux won’t install grub after installing windows

If PC already has windows on it without additional partition of efi then linux won’t install grub onto that machine unless and untill you allocate an efi partition in the installation procedure.

**Linux:**

Need to learn:

services

kernel module

kernel and linux updates

ssh

sockets and pipes

Shell Scripting:

**Writing scripts**

In linux to escape writing same command again and again we can script (write that command onto a file) and then open file in terminal to use that command.

Open any text editor and create .sh file and write what ever commands you want to execute

Note: write “#!/bin/sh” first before any other commands.

Open that file in terminal to execute it.

Change attributes of the file to executable then the shell script will run.

**Commenting in scripts**

For inserting comments in scrips we use “#”

for multiline comments we use “ :’ ” for beginning of comment and “ ‘ ” for the ending of the comment

**Variables in Scipts:**

var\_name = value ex: name = “shell”

echo “$var\_name” ex: echo “$name”

Note: variables should only be given double quotes to print them or use them else they will be considered as strings

**Giving values from command line:**

To read the Bash user input, we use the built-in Bash command called read. It takes input from the user and assigns it to the variable. It reads only a single line from the Bash shell.

Syntax:

read <var> ex: read name

**Printing values of variables on command line:**

To print values of variables, we use the built-in Bash command called echo. It gives output.

Syntax:

echo <var> ex: echo $name # calling variable with $ is necessary

**Operators in Bash:**

There are 5 basic operators in bash/shell scripting:

Arithmetic Operators

Relational Operators

Boolean Operators

Bitwise Operators

File Test Operators

Arithmetic Operators : These operators are used to perform normal arithmetics/mathematical operations. There are 7 arithmetic operators:

Addition (+): Binary operation used to add two operands.

Subtraction (-): Binary operation used to subtract two operands.

Multiplication (\*) :Binary operation used to multiply two operands.

Division (/) :Binary operation used to divide two operands.

Modulus (%) :Binary operation used to find remainder of two operands.

Increment Operator (++) : Uniary operator used to increase the value of operand by one.

Decrement Operator (–) : Uniary operator used to decrease the value of a operand by one

Relational Operators : Relational operators are those operators which defines the relation between two operands. They give either true or false depending upon the relation. They are of 6 types:

‘==’ Operator : Double equal to operator compares the two operands. Its returns true is they are equal otherwise returns false.

‘!=’ Operator : Not Equal to operator return true if the two operands are not equal otherwise it returns false.

‘<' Operator : Less than operator returns true if first operand is lees than second operand otherwse returns false.

‘<=' Operator : Less than or equal to operator returns true if first operand is less than or equal to second operand otherwise returns false

‘>’ Operator : Greater than operator return true if the first operand is greater than the second operand otherwise return false.

‘>=’ Operator : Greater than or equal to operator returns true if first operand is greater than or equal to second operand otherwise returns false

Logical Operators : They are also known as boolean operators. These are used to perform logical operations. They are of 3 types:

Logical AND (&&) : This is a binary operator, which returns true if both the operands are true otherwise returns false.

Logical OR (||) : This is a binary operator, which returns true is either of the operand is true or both the operands are true and returns false if none of then is false.

Not Equal to (!) : This is a uninary operator which returns true if the operand is false and returns false if the operand is true.

Bitwise Operators : A bitwise operator is an operator used to perform bitwise operations on bit patterns. They are of 6 types:

Bitwise And (&) : Bitwise & operator performs binary AND operation bit by bit on the operands.

Bitwise OR (|) : Bitwise | operator performs binary OR operation bit by bit on the operands.

Bitwise XOR (^) : Bitwise ^ operator performs binary XOR operation bit by bit on the operands.

Bitwise compliment (~) : Bitwise ~ operator performs binary NOT operation bit by bit on the operand.

Left Shift (<<) : This operator shifts the bits of the left operand to left by number of times specified by right operand.

Right Shift (>>) : This operator shifts the bits of the left operand to right by number of times specified by right operand.

File Test Operator : These operators are used to test a particular property of a file.

-b operator : This operator check weather a file is a block special file or not. It returns true, if the file is a block special file otherwise false.

-c operator : This operator checks weather a file is a character special file or not. It returns true if it is a character special file otherwise false.

-d operator : This operator checks if the given directory exists or not. If it exits then operators returns true otherwise false.

-e operator : This operator checks weather the given file exits or not. If it exits this operator returns true otherwise false.

-r operator : This operator checks weather the given file has read access or not. If it has read access then it returns true otherwise false.

-w operator : This operator check weather the given file has write access or not. If it has write then it returns true otherwise false.

-x operator : This operator check weather the given file has execute access or not. If it has execute access then it returns true otherwise false.

-s operator : This operator checks the size of the given file. If the size of given file is greater than 0 then it returns true otherwise it false.

**Bash if, if-else, else-if, case statements:**

if Syntax:

if [ expression ]; ex: if [ $a > 10 ];

then then

statements echo “a is greater than 10”

fi if

if-else syntax:

Options for If statement in Bash Scripting

If statement contains many options to perform a specific task. These options can be used for file operations, string operations, etc. Following are the some mostly used options:

Options (Operators) Description

! EXPRESSION To check if EXPRESSION is false.

-n STRING To check if the length of STRING is greater than zero.

-z STRING To check if the length of STRING is zero (i.e., it is empty)

STRING1 == STRING2 To check if STRING1 is equal to STRING2.

STRING1 != STRING2 To check if STRING1 is not equal to STRING2.

INTEGER1 -eq INTEGER2 To check if INTEGER1 is numerically equal to INTEGER2.

INTEGER1 -gt INTEGER2 To check if INTEGER1 is numerically greater than INTEGER2.

INTEGER1 -lt INTEGER2 To check if INTEGER1 is numerically less than INTEGER2.

-d FILE To check if FILE exists and it is a directory.

-e FILE To check if FILE exists.

-r FILE To check if FILE exists and the read permission is granted.

-s FILE To check if FILE exists and its size is greater than zero (which means that it is not empty).

-w FILE To check if FILE exists and the write permission is granted.

x FILE To check if FILE exists and the execute permission is granted.

**Syntax nested if:**

if [exp];

then

statement

if [exp];

then

statement

fi

fi

**syntax if-else statement:**

if [ condition ];

then

<if block commands>

else

<else block commands>

fi

**syntax else-if statement:**

if [ condition ];

then

<commands>

elif [ condition ];

then

<commands>

else

<commands>

fi

**syntax case:**

case expression in

pattern\_1)

statements

;;

pattern\_2)

statements

;;

pattern\_3|pattern\_4|pattern\_5)

statements

;;

pattern-n)

statements

;;

\*)

statements

;;

esac

**For loop:**

**Syntax for-in:**

for variable in list

do

commands

done

**Syntax for c style:**

for ((i=1; i<=10; i++))

do

echo "$i"

done

**Syntax for in range:**

for num in {10..0..1}

do

echo $num

done

**While loop:**

while [ expressions ];

do

commands;

multiple commands;

done

**Untill loop:**

until [ expression ];

do

command1

command2

. . .

. . . .

commandN

done

**Functions:**

# function definition

function\_name () {

commands

}

# function call

function\_name

**Functions with arguments:**

# function definition

function\_name ()

{

$1

$2

$n

}

# function call

function\_name arg1 arg2 arg3

**Overriding functions:**

Linux tips:

1) If you don’t want a command to show up in the history then press a space and the write the command then the command will not show up in the history.

2) Each command has several options to it you need not to learn or practice with each option you can view all the option of a command by using

command –help ex: ls --help

this gives info of that command.

3) to trim ssd’s in linux you need to install util-linux package to unlock utility to trim ssd.

Command to trim ssd in linux is fstrim but is more of a service than a command so you would need to start the service.

Systemctl start fstrim.timer.

4) to know the time of boot command used: systemd-analyze time

5) to set an alias to a command

alias new\_name = old\_name

cpu:

lscpu command is used to check the current clock speed of processor.

Sockets and Pipes:

Users:

In linux there are different types of users. Like in Windows there are administrative users which have full access to the system and can do anything.

By default linux creates a user ( general user ) while installing it on to the system and it asks for a name.

Super user ( adminstrative user ) is necessary when we are installing packages from apt ( advanced package tool ) or for any other commands.

Nano:

In linux nano is a terminal application to create, edit and remove files form terminal.

File permissions:

In linux file permissions are depicted in the form of letters.

Ex of permissions for a text file

-rwxr—r-- hello.txt

the first – refers to type of file generally file have dashes and directories have d instead of dash.

The next three letters refer to permissions of root user ( super user )

r refers to read

w refers to write

x refers to execute

d refers to directory

c refers to character devices

b refers to block devices

the next three letter refer to permissions of general users and groups

Note: the dashes refer that the they don’t have the permssion

the last three letters refer to permissions of public users

In the ls -l command the output of the command can be broken down into different things.

-rw-r--r-- 1 root root 25 May 20 14:40 file.txt

the first coloum refers to permissions of the file

the second coloumn refers to owner of the file

the third coloumn refers to the group which the ownership of the file is being shared to

the fourth coloumn refers to time stamp of the file

the last coloumn refers to name of the file

in linux to change permissions of the file there is a command named chmod

chmod uses some numbers to change permissions of the file

4 refers to read

6 refers to read/write

7 refers to directory

- refers to nothing ( no permission )

**Diskpartitioning and formatting:**

In linux there are many different command line tools for formatting and one which supports mbr and gpt is parted tool

Parted tool has two different modes

1) interactive mode

2) command line mode

In linux what ever folder has a . in front of its name then the folder is hidden from the normal user to read into that folder or even look that folder then you need to use ls -a command.

Linux File systems:

Linux handles files differently than windows does.

In linux there are no c drives etc only a single drive

the root of the file system is / and this is the starting point of the file system this is just like c drive in windows

/bin contains common executables which are shared by the user and root

/boot contains actual linux kernel as well as some boot configuration

/dev contains files for physical devices and sudo devices ( as each and every thing in linux is a file maybe physical device or any of the thing is a file )

/etc contains system program configuration files and files of apt files ( apps which you’ve installed )

/home is the home directory of a user. ( its for non root users )

/lib /lib32 /lib64 contains library files which are used by the system and that will include shared folders

/lost+found folder is used by fsdk to reconstruct files which are caused by some file system corruption

/media folder is used when we connect a media usb drive then linux will mount usb drive in media folder

/mnt folder is the place where we mount network drives ( home servers ) and other such devices this is due to linux does not create c drives and etc

/opt folder is used to save programs in it

/proc is a virtual filesystem for info of resources, processes and more there is a folder for every process

/root is the home directory of root user

/sbin is similar to bin but contains executables for only adminstrators

/tmp is folder to save any temporary files which will be deleted after reboot

/usr is like a mini linux file system which contains whatever / has but this is geared for the users

/var is folder for variable files for various purposes,ex, logs, tz data etc

Find:

Find is a command to files in a directory. Find has some flags to find the files in the directory.

Note: Find is recursive it means that find will find all the files way back in the root directory.

Grep:

Grep command is used to find info within a file.

Grep command takes two flags.

Git: Git is a free and open source distributed version conrtol system designed to handle form small to very large projects with speed and efficiency. Git allows many people to work on a single code.

Package Management:

Installing debian packages in linux is not so easy as there will be missing dependencies not like in windows which has all dependencies built into it. You need to install all the dependencies to install a package to install package along with dependencies ( literally to escape from hassle ) download gdebi which automatically installs all dependencies from the repositories along with the package.

In linux you can’t install a standalone package without installing its dependencies first in some cases a single package needs lot of dependencies so to deal with that linux has some package managers, Which automatically install dependencies.

Linux also has ppa Personal Package Archive is a software repository for uploading source apckages to be built and published as an Advanced Packaging Tool repository by Lauchpad.

Underneath the hood:

Unlike windows linux gives us full control of software installation and even gives us the full control of the source code to change or edit the code generally software installations have setup scripts which tell us what the code will do to our computer ( which programs are compiled, updation of directories etc ).

Devices and Drivers:

In linux every thing is considered a file even hardware devices whenever a device is connected to computer a device file is created in /dev directory.

There are many files in /dev folder some have different attributes

b refers to block devices

c refers to character devices\

Character devices: Like a keyboard or a mouse, transmit data character by character.

Block devices: Like USB drives, hard drives and CDORMs, transfer blocks of data; a data block is just a unit of data storage.

Linux also makes files for storage devices like hard drives and thumb drives

/ dev / sda is the first device the computer looked into ( more like drive 0 )

/ dev / sdb is the second device the computer looked into ( more like drive 1 )

/ dev / sda1 is the first partition of the first device ( more like c drive )

/ dev / sda2 is the second partition of the first device ( more like d drive )

In linux device drivers are not stored in /dev directory sometimes they are part of our linux kernel. A lot of support for hardware is built into the kernel and if we plug in anything it just works without needing to install device drivers. But for some devices which don’t have support for kernel they have kernel module.

Linux Bash Commands:

Linux bash can be used by typing the command name on the shell

linux commands are divided into 3 parts

* the command
* options required by the command
* the command arguments if required

general form of a linux command { command [ -option(s) ] [ arguments ] }

NOTE: Options MUST come after the command and before any command arguments. Options SHOULD NOT appear after the main argument(s). However, some options can have their own arguments! Historically, UNIX commands have been fairly standard in the way that they use options but there are variations - so be aware!

Since the introduction of UNIX System V, Release 3, any new commands must obey a particular syntax governed by the following rules:

* Command names must be between 2 and 9 characters in length
* Command names must be comprised of lowercase characters and digits
* Option names must be one character in length
* All options are preceded by a hyphen (-)
* Options without arguments may be grouped after the hyphen
* The first option argument, following an option, must be preceded by white space. For example **-o sfile** is valid but **-osfile** is illegal.
* Option arguments are not optional
* If an option takes more than one argument then they must be separated by commas with no spaces, or if spaces are used the string must be included in double quotes ("). For example, both of the following are acceptable:

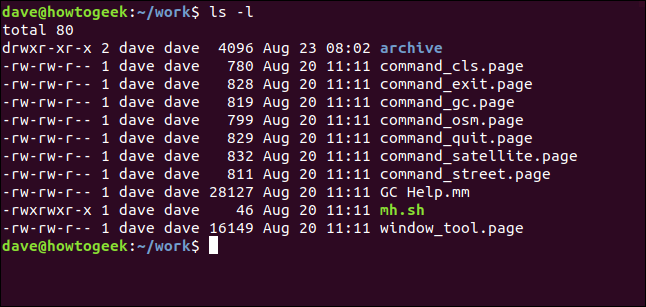
**-f past,now,next** and **-f "past now next"**

* All options must precede other arguments on the command line
* A double hyphen -- may be used to indicate the end of the option list
* The order of the options are order independent
* The order of arguments may be important
* A single hyphen - is used to mean standard input

You should bear in mind that commands established before System V, Release 3, do not conform to all of the above rules.

**Changing file permissions:**

File permission of a file are changed with the help of “chmod” command



There are seven columns in the given image, which are defined as follows:

The first column represents the file type and file permissions. Every file row begins with the file type and then specifies the access permissions associated with the files. These are the following types of files with their specific characters:

Regular file (-)

Directory (d)

Link (l)

Special File (c)

Socket (s)

Named pipe (p)

Block device (b)

The second column represents the number of memory blocks.

The third column represents the owner of the file or the superuser, who has the administrating power.

The fourth column represents the group of owner/superuser.

The fifth column represents the file size.

The sixth column represents the date and time when the file was created or lastly modified.

The last column represents the name of the file or the directory.

The next nine characters represent the settings for the three sets of permissions.

The first three characters show the permissions for the user who owns the file (user permissions).

The middle three characters show the permissions for members of the file’s group (group permissions).

The last three characters show the permissions for anyone not in the first two categories (other permissions).

To use chmod to set permissions, we need to tell it:

Who: Who we are setting permissions for.

What: What change are we making? Are we adding or removing the permission?

Which: Which of the permissions are we setting?

We use indicators to represent these values, and form short “permissions statements” such as u+x, where “u” means ” user” (who), “+” means add (what), and “x” means the execute permission (which).

The “who” values we can use are:

u: User, meaning the owner of the file.

g: Group, meaning members of the group the file belongs to.

o: Others, meaning people not governed by the u and g permissions.

a: All, meaning all of the above.

If none of these are used, chmod behaves as if “a” had been used.

The “what” values we can use are:

–: Minus sign. Removes the permission.

+: Plus sign. Grants the permission. The permission is added to the existing permissions. If you want to have this permission and only this permission set, use the = option, described below.

=: Equals sign. Set a permission and remove others.

The “which ” values we can use are:

r: The read permission.

w: The write permission.

x: The execute permission.

Syntax of chmod: chmod [class][operator][permission] file\_name

ex: chmod u+w hello.txt

|  |  |  |
| --- | --- | --- |
| Si no. | Command | Descripiton |
| 1) | Ls ( list directory ) | ls command is used to list the directories in the opened directory |
| 2) | ./hello | ./ refers to current directory and ./hello refers to open or run file |
| 3) | Ls -l | To search through directory for files with extra information. |
| 4) | ls --Force | Shows the hidden files in the directory |
| 5) | pwd | Prints the working directory |
| 6) | cd | Change directory to go up a level or a down a level |
| 7) | Cd .. | To down a level or to go back |
| 8) | Cd ..\ desktop | To change to directories in the directory |
| 9) | Cd ~\desktop | To change to directories in the directory |
| 10) | Tab completion | Write the first letter of the directory then press tab to automatically complete the name of the directory |
| 11) | mkdir | This command is used to create new directories |
| 12) | tab | To scroll through previously used commands |
| 13) | Clear | Clears the powershell |
| 14) | History | Shows the history of the previously used commands |
| 15) | Cp | To copy files from one directory to another cp file then path to which where you want to copy ( doesn’t copy the inner files in the directory ) |
| 16) | Cp \*.py | To copy all the files which have .py |
| 17) | Del | To delete files from the directory |
| 18) | Cp -recurse | To copy the directory along with the inner directories within it |
| 19) | -verbose | To get the warnings or the intimations that work has been executed or has failed to do while doing some so and so task |
| 20) | Mv | To rename or move directories from one directory to another |
| 21) | rm | To remove the files or directories |
| 22) | cat | To view data in the file |
| 23) | More | To view data in a systematic order |
| 24) | Cat -head | To view only first few lines of the file |
| 25) | Cat -tail | To view only last few lines of the file |
| 26) | Select-string or sls | To look a particular data in the file |
| 27) | -filter | To look up some files in the directory containing some common name or suffix. |
| 28) | Echo | To print the information on the shell |
| 29) | > | To save the output of a command into a file  Note: each command in gives out output in the output stream of the windows so instead printing the output of the command on the powershell we can save the output to a file.  Note: If the file exists then the data will be sent into that file else a new file will be created. |
| 30) | >> | To append the output of a command into a file instead of re-writing it. |
| 31) | | pipe operator | To send output of one command into input of another command. |
|  | Tee hello.txt | Tee command is used when you want the output to be displayed along with the output to be saved in a file.  Note: Tee doesn’t have it’s own output you have to catch output to it this is done by using pipe operator  Best case to use Tee:  ls *home*/vedhanth | tee output.txt  Note: in the above case ls gives it’s output to tee command and tee command writes output of ls command to output.txt as well as prints the output on the terminal. |
|  | Wc | Wc command is used to print word count of another command or a file. |
|  | Sudo su | To switch user from normal user to super user ( administrator ). |
|  | Sudo Apt-get install bluefish | To Install blue fish through terminal |
|  | Sudo Apt-cache search chrome | To search into repository for chrome |
|  | Sudo Apt-cache policy chrome | To search in the system for chrome |
|  | Sudo apt-get remove chrome | To remove chrome from the system |
|  | Sudo apt update | To |
|  | Sudo dpkg -i ./home/vedhanth/chrome.deb | To install .deb files from terminal. |
|  | Sudo apt-install -f | To install all dependencies of upcoming installations |
|  | Sudo dpkg -l | To list all the installed packages |
|  | Sudo dpkg -r chrome.deb | To remove chrome package |
|  | Sudo dpkg –purge chrome | To completely remove package along with its dependencies. |
|  | Sudo dpkg -l | grep chrome | To look for a particular package installed in computer. |
|  | Sudo apt-get update | To update repositories. |
|  | Sudo apt-get upgrade | To upgrade package installed in our system to newer versions. |
|  | Nano hello.txt | To create a text file from terminal |
|  | Sudo chown root:vedhanth file.txt | To change ownership of the file |
|  | Sudo chmod 664  file.txt | To change permissions of the file.  Note: the info of numbers is given above  Note: The placement of numbers is  first number refers to permission of root  second number refers to permission of user  third number refers to permission of public user. |
|  | Touch file.txt | To create a file named file.txt without using nano editor. |
|  | Find -type f -name “\*.txt” | To find a file having .txt as its name.  Note: -type f flag is used to find file types  -name flag is used to find file with the help of the given name. |
|  | Find -type f -iname “\*.txt” | To find a file having .txt as its name ignoring case sensitivity. |
|  | Find -type f -perm 0664 | To find files having permission having 0664 |
|  | Find -type f – size +1M | To find files having more than 1M space |
|  | Find -type f -not -iname “\*.txt” | To find files which are not .txt type. |
|  | Find -type f -maxdepth 1 -iname “\*.txt” | To remove recursive property of find command by giving maxdepth flag |
|  | Grep -i “flag” ./\* | To find “flag” in all files of current directory. |
|  | Grep -i -n “flag” | To find “flag” in all files of current directory. -n gives which line the “flag” is located. |
|  | find -type f -iname "\*.txt" -exec grep -i "flag" {} + | To find “flag” in only .txt files by using find command along with grep |
|  | Tee -a file.txt | To copy input/output to a file  Note: in this example we are appending input to file. |
|  | Top | To see top processes running on PC. |
|  | Ps aux | To see all processes running on PC. |
|  | Kill 7700 | To kill the process containing pid of 7700. |
|  | Crontab -e | To schedule task.  Note: crontab open in nano text edttor then you can mention time/date/ and command to execute. |
|  | 12 12 \* \* \* ls > helo.txt | This command is written in crontab which is opened in nano.  This command will send output of ls command running in directory x to helo.txt at 12 pm 12 min irrespective of day month and year. |
|  | Man ls | Gives description of ls command.  Note: this is similar to –help parameter. |
|  | E4defrag /home | E4defrag command is used to defrag directories. |
|  | E4defrag -c /home | This command is used to check if a directory needs defragmentation. |
|  | 7z e files.zip | To extract files from archive, |
|  | Uname | Shows info of operating system |
|  | Parted -l | To check which drives are connected to the computer |
|  | Sudo apt autoclean | To remove apt cache. |
|  | Sudo apt clean | To clean apt cache. |
|  | Sudo apt autoremove | To remove unused dependencies. |
|  | Sudo apt install gufw | To install gui fire wall. |
|  | Sudo alien /fullpath/filename.rpm | To convert rpm packages to debian packages |