^ → this means control
└→ cntrl. button

LINUX

(Ubuntu)

by Mohd Imran

→ To open terminal. click on activities and search terminal.

☆ **Terminal** :- its where all the commands are given

COMMANDS

(i) **ls** → directory के contents को list करता
→ ls is used to list files and directories within file system and shows detailed info about them
└→ eg. ls Downloads/ to list content of Downloads

(ii) **Pwd** → Print working directory
↳ To check कि हम कोनसी directory mein hai
└→ To see absolute path of directory

linux has tree structure instead of drives and folders.

cd *enter* → to get to home.

Date : / /
Page No.

Types Of Interfaces

(i) ⇒ GUI → Graphical user interface
(ii) ⇒ CLI → Command line interface.

(iii) / (slash) → root node.

(iv) cd → change directory (full form)
it will take you to sub directory

eg. → cd / → it will take us to '/'

cd ~ → it will take us to home user

Important directories in system

etc, bin, usr, tmp & dev will be present in all distribution

(v) use 'Tab' to auto complete a command.

(vi) cd .. → back or backspace to get back to a directory

★ Linux is Case sensitive

(vii) **mkdir** → to make new folder / directory

(viii) **touch** → touch name.txt or any other type file
→ used to make new file

Cat → cat name.txt to read one file

ix) **mv** → move command
↳ eg. mv name.txt folder
file to move | folder to which it needs to move.

x) **Cp** → copy command.
↳ eg. same as mv
cp -R dir1 dir2 → copy dir1 to dir2; create dir2 if not present

USERS IN LINUX

① **REGULAR USER** → can do all work in its home directory
↳ will have all permission, make any folder, read, write but in its own directory
↳ cannot enter into other user's directory

② **ROOT USER** or **SUPER USER** → has full access
↳ it can do anything in any directory
↳ aka Admin

Date: / /
Page No.

xi) **sudo 'any command'**

If a normal user get full access and becomes admin or root user then we can give command using sudo before every command in directories / areas where we didn't had access before.

sudo su

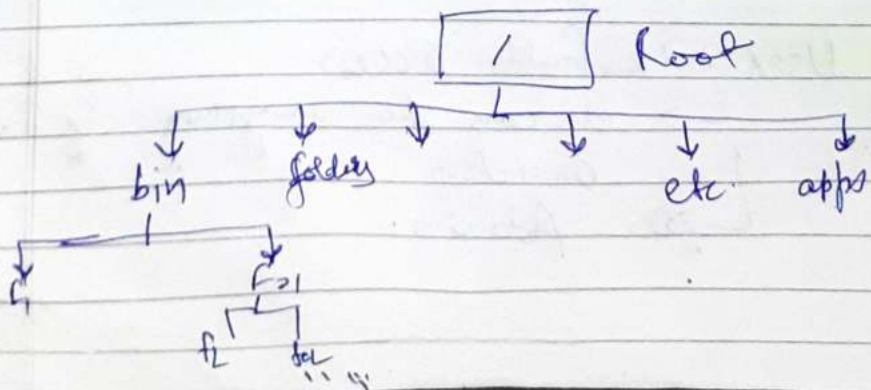
if we use this, then system always recognize us as super user

But its not recommended.

will convert $ to # or € means now we are super user

(3) **SERVICE USER** → services / servers account

ABSOLUTE Vs RELATIVE PATH



If I am in 'apps' and I want to go to some different folder 'fol' then we will use absolute path

e.g. `cd /bin/fol` → absolute path, starts with '/'

Relative path is used when we have to go in subdirectory of pwd.

⭐ $ → means regular user

(xii) apt → used to install softwares

(xiii) apt update → to get list of softwares that can be installed

(xiv) apt upgrade → it will actually install the softwares

(xv) • ls -R → will list all the content and directories in a directory.
↳ R means recursive.

ls -1 → to list in vertical view

(xvi) .file → file starting with '.' to a hidden file.

(xvii) ls -a → to list all files including hidden files in a directory

(xviii) ~~ls .ls --color~~
ls -l → long list format
ls -a → list all files including hiddenfiles
ls -r → sort in reverse order
ls -t → sort according to modification time.
ls -h → human readable
~~ls -alh~~ → long list all in human readable

(xix) clear → to get clear terminal
→ scroll up to see old command again

(xx) history → to get command history

(xxi) echo → it is used to print something on terminal only
↳ eg. ~~echo~~ echo jimmy
jimmy will be written/printed in terminal.

(xxii) ~~Print~~ Printf → same work as echo.
↳ eg. Printf "jimmy is my name"
jimmy is my name will be printed
↳ Printf "jimmy is my name \n"
↓ new line
new command after printing new line ~~#~~ diff connector

Control Shift + t → to open in on terminal

(xxiii) sudo apt install [any software name]

↳ used to install any softwares

& In desktop GUI
click on activities → search softwares
all softwares ← Ubuntu software.
store.

User Group & permissions

(xxiv) ls -l → list all permissions

there will be in format of

d r w x, r w x, r w x
owner/user, Group, other

where d → directory
r → read
w → write
x → execute

If any letter is replaced with '-' means that permission not granted.

~~types of permission~~

for eg.

- drwxr-xr-x

A folder which has read, write and execute permission for owner, but only read and execute permissions for group and others.

- -rw- rw- rw-

A file that can be read and written by anyone but not executed at all.

- -rw- r-- r--

A file that can be read and written by the user. but only read by the group and everyone else.

To change or set permission we use chmod command and chmod calculator

Search chmod calculator on google.

cd ../../ → 2 directories back
cd ../ → 1 directory back

(xxv) **chmod** → means change mode
↳ used to change mode or permissions

eg. chmod [permission no. from chmod calc] [file/group]

7 - execute, read & write
4 - read
2 - write
1 - execute

eg. 734
520

(xxvi) **top** → will tell about the processes which are taking most resources.
cntrl+C to exit top command.

(xxvii) **ps** → will list all the running process

(xxviii) **ps -a** → will list all the processes including background processes

(xxix) **kill** [Process ID]
↳ it's the numbers under PID in top command.
↳ to kill or close any process.

(xxx) **vim** → It is a text editor.
↳ eg. vim jimmy.txt

first we use sudo su
then after becoming super user.
→ apt install vim

it will instead vim editor
then we can use vim editor from commands

How to use Vim

→ first type ~~vim editor~~ vim filename

vim immy.txt

it will open file in vim editor

to type first press 'i' button
it will show insert, then we can type anything
i → Enter insert mode
Esc → to exit insert mode.
then,
:q → exit without saving
:wq → exit with saving

(xxxi) rm → remove command
↳ used to remove any file or directory
↳ e.g. rm filiname.

(xxxii) rmdir → used to remove a directory
↳ eg. rmdir directory name
↳ only used to delete empty directory.

rm -rf / → to delete and destroy the system
(never use this).

(xxxiii) rm -r → used to remove a directory and files inside it.

↳ eg. rm -r [directory]/*

rm -d → remove empty directories

(xxxiv) man → manual

↳ to know how a command is used

↳ eg man [command].

(xxxv) w → to know who is on machine

↳ will show uptime, no. of users, current time, cpu load, etc.

↳ :0 means sitting on local host.

who → same as 'w' but shows less info.

(xxxvi) netstat → to know what happening on the network

or ↳ to list all port, socket and everything on network

- netstat -a → to list all the connections
- netstat -at → to list only tcp connections
- netstat -au → to list only udp connection

- netstat -aut → to disable reverse dns lookup for faster output

- netstat -tnl → to check for listening tcp ports.

  -l → to check all open ports

  -unl → to see udp open port

- sudo netstat -nlpt → to get process name/ pid and user id.

  sudo netstat -nlpte → to get process id along with username

- net stat -t u p l n

  - t → tcp ports
  - u → udp ports
  - p → program attached to ports
  - l → listening ports
  - n → numeric address.

- netstat -s → to list statistics for all port
- netstat -rn → used to display Kernel routing info
- netstat -i → used to print network interfaces
  ↳ this will provide very raw data.

  for more human friendly version

  netstat -ic

- netstat -ct → output tcp connections continously

★ **DAEMON PROCESS** → It is a linux or unix program that runs in the background. Almost all daemons have names that end with letter "d". for e.g.

httpd the daemon that handles apache server.

or

sshd which handles SSH remote access connection.

ZOMBIE PROCESS

It is a process whose execution is completed but it still has an entry in the process table. Zombie process usually occurs for child processes, as the parent process still needs to read ~~from the process table~~ its child's exit status.

Once this is done using the wait system call, the Zombie process is eliminated from the process table. This is known as reaping the Zombie process.

GETTING THE SIZE OF DIRECTORY

(xxxvii) du → command used to know the space used by directories

(xxxviii) du -h → will show size of specified directory with subdirectories in human readable version.

. → short form of the directory we are in

Free → to check ram usage

(xxxix) du -sh → to show same the size of specified directory only without the subdirectories

(xl) du -ch → used to find grand total of the space.
or -c

Text Editors

- nano → simple to use.
- vim → advanced.

SHELL FEATURES

SHELL → it is type of unix, distribution
eg. Linux is a shell.

- Pipes
- Input and output Redirection

Pipes & Redirection

the pipe command lets you send the output of one command to another.

It can redirect the standard output, input or error of one process to another for further processing.

Piping character → | → between two command

command 1 | command 2 | ... | command N

> → output redirection
< → input redirection.

1.

All applications have three unique files that connect them to the outside world. These are :-

Standard Input	, Stdin	= 0
Standard output	, Stdout	= 1
Standard Error.	, Stderr	= 2

— Default values (will be used in specification of stream for redirection)

Date : / /
Page No.

Standard Input

Standard input is the default mechanism for getting into an interactive program. This is typically direct link to the keyboard in a terminal.

Standard Output

Standard output is the default for writing output from a program. This is typically a link to the output terminal but is often buffered for performance reasons. This can be especially important when the program is running over a slow network link.

Standard Error

It is an alternative mechanism for writing output from a program. This is typically a different link to the same output terminal but is unbuffered so that we can write errors immediately while the normal output can be written one line at a time.

* → all files in current directory
(g)ls *a → will list all files starting with a

★.2 Redirecting into files

This is typically done with '>' operator between the application to run and the file to write the output into.

for eg. we can send the output of the ls command into a file called files as follows.

```
ls > files
```

We can do this after any command, including needed arguments

```
ls -l *.txt > text-files
```

★.3 Appending into files.

When we use >, then we start writing the output to the file, replacing all of the contents. If needed we can also append/add to the file using '>>' instead

echo → used to display text / string
↳ [just more often runs.]

```
ls -1 *.txt > files
ls -1 *.text >> files
ls -1 *.log >> files
```

we can use this to build up files using as many of output if we need to including echo to exact output line.

```
echo text files > files
ls -1 *.txt >> files
echo log files >> files
ls -1 *.log >> files
```

4. Redirecting Standard Error.

On occasion we need to redirect standard error instead of standard output

When we use redirect operators, by default this applies to standard output. We can specify the stream to redirect by prefixing it with the stream ID ie 0, 1, 2.

For Eg.
to redirect standard error, from the cat command we would use 2>

```
cat does-not-exist 2> log.
```

We can also use &> to redirect standard output and standard error at the same time

✡.5 Redirecting into streams

Sometimes we want to redirect into a stream instead of a file. We can achieve this by using the stream ID (0,1,2), prefixed by & in place of the filename

for eg.

We can redirect into standard error by using >&2.

echo standard output >&1
echo standard error >&2

We can combine this with above to combine streams, by redirecting from one stream into another.

A common construct is to combine standard error into standard output. So they can be used together. We achieve this using 2>&1 - literally redirecting stream 2 into stream 1

ls -l 2>&1

★.6 Redirecting Multiple Streams

We can easily combine the above to redirect standard output and standard error at the same time. We simply combine two different redirects on the same command.

ls -l > stdout.log 2> stderr.log

This won't work as desired if we try to redirect both streams into the same file. What happens here is that both streams are redirected individually, and whichever comes second wins.
If we want to redirect both into the same files, then we can use &> as above

ls -l > log 2>&1

★.7 Reading from files

Sometimes we want to achieve the opposite - redirecting a file into an application. We do this using < operator
and the contents of the file will replace

the standard input of the application

```
wc < /usr/share/dict/words
```

When we do this, the only input that the application can receive comes from this source and it will all happen immediately

☆.8 Piping between Applications.

The final action that we can perform is to direct the output of one application into another one. This is commonly referred to as piping, and uses | operator instead.

```
ls | wc
```

This directly connects the standard output of our first ~~command~~ application into the standard input of the second one and then lets the data directly flow between them.

9. Handling Standard Error

The standard error isn't connected by default, so we'll get anything written to that stream appearing in the console.

if we want to redirect standard error as well then we can use the techniques from above to first redirect it into standard output and then pipe into the next application

```
$ ls i-dont-exist | wc

ls: i-dont-exist: No such file or directory
      0      0      0

$ ls i-dont-exist 2>&1 | wc
      1      7     44
```

10. Combining Pipes

When piping between applications, we can also build arbitrary chains where we are piping between many applications to achieve a result.

docker images | cut -d '' -f1 | tail -n+2 | sort -u | wc -l

docker images → Get a list of all docker images

cut -d '' -f1 → Cut this output to only return the first column, when column are space separated.

tail -n+2 → Limit this to start from line 2.

sort -u → Sort the list, only returning unique entries

wc -l → count the number of lines

Echo

Echo command in linux is used to display a line of text/string.

Example: echo "Geeks for Geeks"
Geeks for Geeks

-e → enables interpretation of backslash escapes

\b → It removes all spaces b/w text.

Eg. echo -e "Geeks \b for \b Geeks"
GeeksforGeeks

\n → this options creates new line from where it is used.

Eg. echo -e "Geeks \n for \n Geeks"
Geeks
for
Geeks.

\t → used to create horizontal tab spaces

eg.. echo -e "Geeks \t for \t Geeks"
Geeks for Geeks

\v → used to create vertical tab spaces

Eg. echo -e "Geeks \v for \v Geeks"
Geeks
for
Geeks

Echo * → will print all files, similar to ls

\a → alert returns with backspace interpretation '-e' to have sound alert

eg. echo -e "\a Geeks for Geeks"

when this executed, it will produce an alert sound or bell.

USEFUL CHAINING OPERATORS IN LINUX

& → The function of '&' is to make the command run in background

eg to run one command in background

ping cs www.teamjd.com &

to run two commands in background.

apt-get update & apt-get upgrade &

; → The semi-colon operator makes it possible to run several commands in a single go and execution occurs sequentially.

Date / /
Page No.

eg. apt-get update ; apt get upgrade ; mkdir test

&& → The AND operator would execute the second command only, if the execution of first command succeeds. This command is very useful in checking the execution status of last command i.e exit status is 0.

foreg. I want to visit website tecmint.com using links command in terminal but before that I need to check if the host is live or not.

ping -C3 www.tecmint.com && links www.tecmint.com

|| → The OR operator is like else statement. It allows you to execute second command only if the execution of first command fails i.e. exit status of first command is 1

eg. apt-get update || links tecmint.com

what if the first command is executed successfully with exit status 0?, obviously second command won't work.

! → The NOT operator is much like an except statement. The command will execute all except conditions provided.

To understand

create a directory 'immy' in your home directory and 'cd' to it

```
$ mkdir immy
$ cd immy
```

Now, create several types of file in 'immy'.

```
$ touch a.doc b.doc a.pdf b.pdf a.html
```

See we've created all the files in immy

```
$ ls
a.doc b.doc a.pdf b.pdf a.html
```

Now delete all the files except 'html' at once.

```
$ rm -r !(*.html)
```

Date:
Page No.

&& -|| → AND-OR operator
The above operator is actually a combination of 'AND' and 'OR' operator. It is much like an 'if-else' statement.

for eg. let's do ping to youtube.com, if success echo 'verified' else echo 'Host Down'.

$ ping -C3 www.youtube.com && echo "verified" || echo "Host down".

| → The Pipe operator is useful when the output of first command act as input of second command.

eg. ls -l | less

{} → COMMAND COMBINATION OPERATOR
Combine two or more commands, the second command depends upon the execution of first command.

for eg. check if a directory 'bin' is available or not, and output corresponding output.

[-d bin] || {echo directory does not exist, creating directory now; mkdir bin;} && echo Directory exists

( ) → PRECEDENCE OPERATOR

This operator makes it possible to execute command in precedence order

Command_x1 && Command_x2 || command_x3 && command_x4

In above pseudo command, what if command_x1 fails? Neither of the command_x2, Command_x3, command_x4 would be executed. for this we use precedence operator - , as:

(command_x1 && command_x2) || (command_x3 && command_x4)

FILTERING OUTPUT and FINDING THING

#-1 CUT

The CUT command is used for cutting out the sections from each line of files and writing the result to standard output.

Let us consider file having name state.txt contains 5 names of the Indian states.

```
$ cat state.txt
Andhra Pradesh
Arunachal Pradesh
Assam
Bihar
Chhattisgarh
```

-b (byte) : To extract the specific bytes, you need to follow -b option with the list of byte number separated by comma. Range of bytes can also be specified using the hyphen (-).
Tabs and backspaces are treated like as a character of 1 byte.

List without ranges.

```
$ cut -b 1,2,3 state.txt
And
Aru
Ass
Bih
Chh.
```

List with ranges

```
$ cut -b 1-3,5-7 state.txt
Andra
Aruach
Assm
Bihr
Chhtti
```

It uses a special form for selecting bytes from beginning upto the end of the line.

In this, 1- indicates from 1st byte to end byte of a line

```
$ cut -b 1- state.txt
Andhra Pradesh
Arunachal Pradesh
Assam
Bihar
Chhattisgarh
```

In this, -3 indicates from 1st byte to 3rd byte of a line.

```
$ cut -b -3 state.txt
And
Aru
Ass
Bih
Chh.
```

-c (column) : To cut by character use -c option.
This selects the characters given to -c option. This can be a list of numbers separated by comma or a range of numbers separated by hyphen (-). Tabs and backspaces are treated as a character

$ cut -c 2,5,7 state.txt
nr
rah
sm
ir
hti

$ cut -c 1-7 state.txt
Andhra
Arunach
Assam
Bihar
Chhatti

$ cut -c -5 state.txt
Andhr
Arung
Assam
Bihar
Chatt

$ cut -c 1- state.txt
Andhra Pradesh
Arunachal Pradesh
Assam
Bihar
Chhattisgarh

-f (field): -c option is used for fixed length lines. Most unix files doesn't have fixed length lines To extract the useful information you need to cut by fields rather than columns. List of the fields number specified must be separated by comma. Ranges are described with -f option. Cut uses Tab as a default field delimiter but can also work with other delimiters by using -d option.

Like in file state.txt fields are separated by space if -d is not used then it print the whole line

```
$ cut -f 1 state.txt
Andhra Pradesh
Arunachal Pradesh
Assam
Bihar
Chhattisgarh
```

if -d option is used then it considers space as a field separator or delimiter

$ cut -d " " -f 1 state.txt
Andhra
Arunachal
Assam
Bihar
Chhattisgarh

#.2 SORT

Sort command is used to sort a file, arranging the records in a particular order.

The sort command follows these features as stated below:

→ Lines starting with a number will appear before lines starting with a letter.

⇒ Lines starting with a letter that appears earlier in the alphabet will appear before lines starting with a letter that appears later in the alphabet.

→ Lines starting with a lowercase letter will appear before lines starting with the same letter in uppercase.

-o : unix also provides us with special facilites like if you want to write the output to a new file, output.txt, redirects the output like this or you can also use the built in sort option -o, which allows you to specify an output file.

Eg.

$ sort inputfile.txt > filename.txt
$ sort -o filename.txt inputfile.txt

both does the same work.

-r : Sorting in reverse Order.

you can perform a reverse order sort using the -r flag. the -r flag is an option of the sort command which sorts the input file in reverse order i.e descending order by default

Eg.

$ sort -r inputfile.txt

-n : To sort a file numerically -n option is used. This option is used to sort the file with numeric data present inside. The output will be in ascending order.

Eg.

```
$ Sort -n file1.txt
```

-nr : To sort a file with numeric data in reverse order we use the combination -n & -r.

Eg.

```
$ Sort -nr file1.txt
```

-k : Unix also provide the feature of sorting a table on the basis of any column number by using -k option.
Use the -k option to sort on a certain column. For eg. use "-k 2" to sort on the second column.

Eg.

```
$ Sort -k 2n employee.txt
```

-C : This is used to check if the file given is already sorted or not and if a file is already sorted pass the -C option to sort. This will write to standard output if there are lines that are out of order.

This tool can be used to understand if this file is sorted and which lines are out of order.

Eg.

$ Sort -c filename.txt

-U : To sort and remove duplicates. This will write a sorted list to standard output and remove duplicates

Eg.

$ Sort -u filename.txt

-M : To sort by Month This will write a sorted list to standard output ordered by month name

Eg.

$ Sort -M file name.txt

#.3 UNIQ

Uniq is the tool that helps to detect the adjacent duplicate line and also deletes the duplicate lines

Uniq filters out the adjacent matching lines from the input file and writes the filtered data as to the output file.

// displaying the contents of Kt.txt //

$ cat Kt.txt
I love music
I love music
I love music

I love music of Kartik
I love music of Kartik.

Eg.

$ Uniq Kt.txt
I love music

I love music of Kartik

-c : It tells how many times a line was repeated by displaying a number as a prefix with the line.

Eg.

```
$ uniq -c Kt.txt
   3 I love music
   2 I love music of Kartik
```

-d : It only print the repeated lines and not the lines which aren't repeated.

Eg.

```
$ uniq -d Kt.txt
I love music
I love music of Kartik
```

-u : It allows you to print only unique lines.

Eg.

```
$ uniq -u Kt.txt
```

#4 W/C

W/C stands for word count. As the name implies it is mainly used for counting purpose

→ It is used to find out number of lines, word count, byte and characters count in the files specified in the file argument.

→ By default it displays four-columnar output.

→ First column shows number of lines present in a file specified, second column shows number of words present in the file, Third column shows number of characters present in file and fourth column itself is the file name which are given as argument.

```
$ cat state.txt
Andhra Pradesh
Arunachal Pradesh
Assam
Bihar
Chhattisgarh

$ capital.txt
Hyderabad
Dispur
```

Dispur
Patna
Raipur.

$ wc state.txt
5 7 63 state.txt

$ wc capital.txt
5 5 45 capital.txt

$ wc state.txt capital.txt

5 7 63 state.txt
5 5 45 capital.txt
10 12 108 total

-w : This option prints the number of words present in a file. In this this option wc command displays two-columnar output, 1st column shows number of words present in a file and 2nd is the file name.

Eg.
With one file name
$ wc -w state.txt
7 state.txt

With more than one file name
$ wc -w state.txt capital.txt

7 state.txt
5 capital.txt
12 total

-l : This option ~~present~~ print number of lines present in a file.

Eg.

```
$ wc -l state.txt
5 state.txt

$ wc -l state.txt capital.txt
5 state.txt
5 capital.txt
10 total
```

-c : This option displays count of bytes present in a file.

Eg.

```
$ wc -c state.txt
63 state.txt

$ wc -c state.txt capital.txt
63 state.txt
45 capital.txt
108 total
```

-m : This option display count of characters from a file.

Eg.

```
$ wc -m state.txt
63 state.txt

$ wc -m state.txt capital.txt
63 state.txt
45 capital.txt
108 total
```

-L : It can be used to print out the length of longest (no. of characters) line in a file.

#5 GREP

The grep filter searches a file for a particular pattern of character and displays all lines that contain that pattern. The pattern that is searched in the file is referred to as the regular expression (grep stands for global search for regular expression and print out).

-i : this option enables to search for a string case insensitively in the given file.

Eg.

$ grep -i "UNix" geekfile.txt

-c : This print only a count of the lines that matches a pattern

Eg.

$ grep -c "unix" geekfile.txt

-l : This display the file names that contains the given String/pattern

Eg.

$ grep -l "unix" *

-w : This option to grep makes it match only the whole word.

Eg.

$ grep -w "unix" geekfile.txt

-o : By default, grep displays the entire line which has the matched string. We can make the grep to display only the matched string by using the -o option.

Eg.

$ grep -o "unix" geekfile.txt

-n : To show the line number of file with the line matched.

Eg.

$ grep -n "unix" geekfile.txt

-v : you can display the lines that are not matched with the specified search string pattern.

Eg.

$ grep -v "unix" geekfile.txt

^ : the "regular expression pattern specifies the start of a line. This can be used in grep to match the lines which start with the given string or pattern.

Eg

$ grep "^unix" geekfile.txt

$: The $ regular expression pattern specified the end of a line. This can be used in grep to match the lines which end with the given string or pattern.

Eg.

$ grep "os$" geekfile.txt

-e : specifies expression with -e option. Can use multiple times

$ Eg. grep -e "Agarwal" -e "Aggarwal" geekfile.txt

PACKAGE MANAGEMENT with apt-get

#1 apt-get : It is a command line tool which helps in handling packages in Linux. Its main task is to retrieve the information and packages from the authenticated sources for installation, upgrade or removal of packages along with their dependencies. Here API stands for advanced Packaging tool

Most used command

Update : This command is used to synchronize the package index files from their sources again. You need to perform an update before you upgrade or dist-upgrade.

Eg.
apt-get update

Upgrade : This command is used to install the latest versions of the packages currently installed on the user's system from the sources

Eg.
apt-get upgrade

dselect-upgrade : This is used alongwith the Debian packaging tool, dselect. It follows the changes made by dselect to the status field of available packages, and performs any actions necessary to realize that state.

Eg.
apt-get dselect-upgrade

Date : / /
Page No.

dist-upgrade : This command performs the function of upgrade, and also handles changing dependencies with new versions of packages. If necessary, the apt-get command will try to upgrade important packages at the expense of less important ones. It may also remove some packages in this process.

Eg.
apt-get dist-upgrade

install : This command is used to install or upgrade packages. It is followed by one or more package names the user wishes to install.

Eg.
apt-get install [..... packages]

remove : This is similar to install, with the difference being that it removes the packages instead of installing. It does not remove any configuration files created by the package.

Eg.
apt-get remove [..... packages]

purge : This command removes the packages, and also removes any configuration files related to the packages.

Eg.
apt-get purge [..... packages].

Check : This command is used to update the package cache and checks for broken dependencies

Eg.
apt-get check

download : This command is used to download the given binary package in the current directory

Eg.
apt-get download [.... packages]

clean : This command is used to clear out the local repository of retrieved package files.

Eg.
apt-get clean

Date: / /
Page No.

autoremove : Sometimes the packages which are automatically installed to satisfy the dependencies by other packages, are no longer needed then autoremove command is used to remove these kind of packages.

Eg.
apt-get autoremove

USER ACCOUNT MANAGEMENT

~~#1. USERADD~~

#1. **Useradd** : It is a command in linux that is used to add user accounts to your system. It is just a symbolic link to adduser command in Linux and the difference between both of them is that useradd is a native binary compiled with system whereas adduser is a Perl script which uses useradd in background. It makes changes to the following files

- /etc/passwd
- /etc/shadow
- /etc/group
- /etc/gshadow
- creates a directory for new user in /home.

✡ our new user are automatically stored in /etc/passwd/

→ To add a simple user

Eg.

$ Sudo useradd test_user

This command will add the user named "test_user"

→ To give a different home directory path for new user.

$ Sudo useradd -d /home/test_user test_user

→ To create a user with specific user id

$ sudo useradd -u 1234 test_user.

This will create a new user with the user id "1234" and the name "test_user".

→ To create a user with specific group id.

$ Sudo useradd -g 1000 test_user.

This will create a new user with groupid "1000" and name "test_user".

Date : / /
Page No.

→ To create a user without home directory

$ Sudo useradd -M test_user.

→ To create a user with expiry date

$ Sudo useradd -e 2020-05-30 test_user

→ To create a user with a comment or short description

$ Sudo useradd -c "This is a test user" test_user

→ To create a user with changed login shell

$ Sudo useradd -s /bin/sh test user

This will create a user named "test_user" with the default shell /bin/sh

→ To set an unencrypted password for the user

$ Sudo useradd -p test_password test_user

#2 **passwd** : To set a password for a new created user.

$ passwd test_user

#3 **Usermod** : It is a command in linux that is used to change the properties of a user in linux through the command line.

→ To add a comment for a user.

$ Sudo usermod -c "This is test user" test_user

→ To change home directory of a user

$ Sudo usermod -d /home/manav test_user

→ To change the expiry date of a user.

$ Sudo usermod -e 2020-05-29 test_user

→ To change the group of a user

$ sudo usermod -g manav test_user.

→ To change user login name

$ sudo usermod -l test-account test_user

→ To lock a user

$ sudo usermod -L test_user

→ To unlock a user

$ sudo usermod -U test_user

→ To set an unencrypted password

$ sudo usermod -p test_password test_user

→ To create a shell for the user

$ sudo usermod -s /bin/sh test_user

→ To change user id of a user

$ sudo usermod -u 1439 test_user

PROCESSES OVERVIEW

init is the mother of all processes

all processes have parent processes

PID of init is always 1.

PROCESS SIGNALS

eg-

$ Kill -l

It will show the table of all the signals which can be used with it.

$ Kill [Signal number] [PID]

to kill a process or to send a signal to a process

$ Sudo Killall [Process name]

It will all the processes by names.

$ Sudo pkill [username]

It will kill all the process of the specific user.

Date: / /
Page No.

State, Niceness & How To Monitor Processes

Ques What is proc filesystem?

It is process filesystem.
- virtual filesystem the linux kernel mounts to post information about its processes so that applications have easy access to them.

Ques Every process wants what?

CPU time

Ques What is responsible for: scheduling what process get CPU time when?

The Kernel
- Determines this based on what state the process is in.

Ques What are the 4 states a process can be in?

i) Run-able - eligible to be scheduled for CPU time, has all the info it needs.

ii) Sleeping - waiting for something

iii) Zombie - finished with what it was doing, waiting to give back info it has come up with and be killed by kernel.

iv) Stopped - May have been in middle of doing something But received stop signal and is now waiting for the continue signal to resume.

Ques If you have a lot of Zombie processes causing an issue, what would be useful to check for?

Parent PID (PPID)
- the issue could be that the parent hasn't collected finished output of the processes and the children can't die.

Ques The most common tool/command to monitor processes is.....

top
~~htop~~ (~~better version~~)

(command: automatically sort % of cpu use, # of things running, states of processes, etc)

Ques Besides top 'top' command, what command can show you MORE detailed and granular info about processes, resources use and process states?
(hint: feature scrolling)

Date: / /
Page No.

htop
(command, you may need to install)
- nicer interface
- allows you to scroll
- can scroll to a process and send a signal to it directly
- more features, read man page.

Ques What should be your first step in exploring any new program?

Look at the man page
- familiarize yourself with the options available
- see if there are any examples at the end.

Ques What is Niceness (NI)?

A value that determines how 'nice' a process is to other processes
(the priority of the process)

Ques Within Niceness, the HIGHER the number...

the LOWER the priority
(the #s go from -20 [SUPER HIGH] to 19 [lowest priority])

Eg.
Low priority command:
nice -n 15 /backup/not/important/at/all/task

Date: / /
Page No.

☆ Niceness and customer facing systems:
with anything like this, you want to be careful not to run things that will run things that will grind everything else to a halt. (because you might want to set a lower niceness level)

- too great a level might make your system feel like it is freezing up.
- generally, you don't want to set anything to the minimum niceness / highest priority because it will block other stuff.

Ques How do you renice (change priority) of a task?

renice -5, 2744
↓ ↓
new niceness Process ID

(make sure that you have appropriate privileges)

- you can also change it with htop with F7 or F8

Ques Example problem: you are on a new system and it has slowed to a crawl. What do you do?

Date : / /
Page No.

ii) open up top (or htop)

THE /PROC FILESYSTEM

FILESYSTEM & ABSOLUTE/RELATIVE PATHNAMES

Ques What is an API?

Application Programming Interface
- The language you use to deal with the underlying filesystem.

Ques True or False: When Linux the only thing runs through the filesystem are just files?

False.
Linux runs almost everything through the filesystem.

Eg.
- all of your hardware (serial through memory through cd-rom).
- (you have 2 files each: where device is mounted & device drivers).

sockets (internal traffic, inter-process communication channels)

Date: / /
Page No.

Ques What is: Udevd ?

Set up drivers and mount locations for new devices

- it is a system daemon (process that runs in background, listening)
- allows for automatic management of devices

Ques What will this command do: df ?

It will list what is mounted
(sys filesystem, proc filesystem, etc)

syntax:
df -ah

Ques What are absolute paths?

The actual address of the file/directory/etc, the same no matter where you are (as opposed to relative paths)

Syntax example:

ls /home/dave/Desktop/

Date: / /
Page No.

Ques When would it be useful to utilize absolute paths?

- for scripting (static location)
- when you can't leave your directory / not efficient to
- for security (don't wanna give pw via an argument in a relative path)

FILESYSTEM LAYOUT

/etc
a place for all of the configuration data of all the applications. When you install an application, it will create a directory and all the configuration will be saved there.

Ques Everything begin at?

Root (/)
top of the filesystem tree

Date : / /
Page No.

Ques What is /bin?

It is where your base operating commands are
UNIX = just base commands (i.e. in free BSD)
LINUX = Linux added lots when you install programs.

In LINUX - base system put together from disparate sources.

In UNIX - Tightly integrated system (at least when it comes to /bin).

Ques What is /boot?

Here you have kernel, kernel files, & boot loader (i.e. grub)
- this is Linux (its beatiful)

Ques What is /dev?

All of your devices

HDD is here (under 'sd' moniker)
CD-ROM, etc.

Ques What is /etc?

i) System configuration
ii) all the extra stuff you have installed with your package manager.

Date : / /
Page No.

Ques What is /home?

The home for all users that are not root.
(have all their files here)

Ques What is /lib?
(and is /lib64)

Houses libraries that programs on your system use.
(they are called DLL on windows, code that any software can use)

Each can be used by multiple programs

Ques What is /media?
(on ubuntu)

Mount directory where automounted items go (on Ubuntu)
- Usually only root can mount things, but can mount a CD-ROM even this without root privileges.

Ques What is /mnt?

Mount directory where automounted items go.
(Used to mount items like CD-ROMs, USB, HDD on most systems)

Date : / /
Page No.

Ques What is /opt ?

Optional software
(you can throw extra software here if you aren't installing through the package manager)
- not used too much.

Ques What is /proc ?

Has directory and configuration files for every program on your system.
(virtual filesystem that the kernel uses to communicate with you and any other programs on your machine about the state of processes that are running)

Ques What is /root ?

Root's home

Ques What is /sbin ?

Critical files that your machine needs to get up and running
- DO NOT TOUCH.

Date : / /
Page No.

Qus What is /tmp?

temporary files live, temporary space for holding stuff.
(emptied upon reboot)

Qus What is /usr?

A subdivided list of non-super-essential files and commands

includes
(under /usr)

file headers, for writing/compiling code
i.e. like <includes> in C++.

Qus What would you put in the /usr/share?

Software/programs that you would want when you re-image.

Qus What /var/log?

This has all of your system logs in it

Date : / /
Page No.

LINUX FILE TYPES

-rwx

means that this is a regular text file.

lrwx

'l' means that this is a link.

drwx

'd' means that this is a directory
- contain references to other directories and files

brwx

'b' means that these are block device file.
(block device = anything that can store information
IE : hard disk)

crwx

'c' means character device files.
(character device = mouse or keyboard)

Date: / /
Page No.

Symbolic link (soft link)

(it is not a reference, that would be a hard link)
like the way a parent directory looks at a file.)

Ques **What is a device file?**

The intersection point between the kernel/driver and the firmware for the device (ex: /dev sda 1 is the place where our driver communicates with the hdd)

Sockets (Quick description)

A communication channel that opens. only visible to the two processes using it (a private file).

Ques **Which are used more: Named Pipes vs Network Sockets**

Network Sockets - open a high level port (like TCP port) and do their thing there
(Very seldom use named pipes)

Date : / /
Page No.

Ques Officially, How many file types are there?

7

(i) Normal files
(ii) directories
(iii) references to files
(iv) character device files
(v) block device files
(vi) Symbolic links
(vii) local domain sockets / named pipes)

SCHEDULING TASKS WITH CRON

Ques What is Cron used for?

Scheduling Tasks

Ques What is a Cron Table?

A table of scheduled processes that you want to schedule in the future.

Ques How do you list your cron tab?

Crontab -l

Date : / /
Page No.

Ques What does this syntax allow you to do to a crontab:

crontab -e

It lets you edit a crontab

Ques There are 6 fields in a crontab:

e.g. #15 23 * * 1-5 echo "$(date): checking in" >> /var/log/myscript.log
(m) (h) (dom) (moy) (wd) (command)

1-2) (m) (h) - the minute and hours that a job should run at (minute 15 of hour 23)

3) day of the month (dom) - (* = every day of the month)

4) month of the year (moy) - (* = every month of the year)

5) Weekday (wd) - (1 = monday, 7 = Sunday, 3-5 means wed, thurs & fri)

6) Command - (interpreted by sh shell, you can specify shell to run)

comments this out - you would have to be root to run this

Date : / /
Page No.

Ques. True or False: user crontabs are kept in /var/spool/cron/crontabs?

True
they are kept in /var/spool/cron/crontabs
but
there is another place:
~~/etc/cron~~ /etc/cron.d

Ques In addition to being another spot where crontabs are kept, what else is housed in /etc/cron.d

for packages to install package specific scheduled jobs.
(so that they don't contaminate user's files & keep it together.

Ques The system wide cron tab is in:

/etc/crontab

Ques The system wide crontab has an extra field (7th field) what is ---

User
Allows us to create jobs that run as specific users
(useful when certain users own certain files and directories)

Date: / /
Page No.

Ans **Advanced Syntax Breakdown:**

15 10 1-10/2 *5 root echo "$(date) - Everything looking good" >> /var/log/cronCheck.log

15 = minute
10 = hour
1-10/2 *5 = Range (Days of the month 1 through 10 every 2 days AND on Fridays)
root = run as root, enables us to write into /var/log
echo .... = command (echo)

/ implies a recurring thing
Day of the month and Day of the week go together, when one of them is true it activates task.

Ans **Can you have lists in Cron?**

Yes
15 10 1,2,3,4 *5 root echo ....
will run on 1, 2, 3, 4 days of the month (and on Friday)

Date : / /
Page No.

Ques What is this?
/etc/cron.allow

Cron whitelist
(if users aren't in here, they won't be able to create a crontab)

This is not the default, used to lock down servers.

Ques If you want to edit a specific user crontab, what can you do?

Syntax (you need to be root to do this):
crontab -e -u dave

This allows you to go in and edit other user's crontab.

8 BASH SHORTCUTS EVERY LINUX USER SHOULD KNOW

Ques What does control-C do?

Terminate a running command that's attached to your shell session.

Date : / /
Page No.

Qus what does ctrl-d do?

Exit shell session

Qus What is Tab completion?

Using tab to complete your command.
(auto-completion)

Qus What does ctrl-L do?

Clear your Terminal

Qus What do ctrl-a and ctrl-e do?

move to the beginning or end of line
(respectively)

Qus What do alt-f and alt-b do?
(or ctrl + arrows left/right)

Move forward or backward one word
(respectively)

Qus How do you search through shell history?
(key combo)

ctrl-R

Ques What do ctrl-u and ctrl-k do?

Delete everything to the left of the cursor and delete everything to the right of the cursor (respectively)

TMUX (TERMINAL MULTIPLEXER)

1) WINDOWS, PANES AND SESSIONS OVER SSH

Ques What does tmux allow you to do?

- share shell sessions
- multiple tabs/panes open (lets you work on many sessions)
- lets you work on remote machines
- can detach/re-attach from sessions (won't kill process when you exit)

Ques When you start tmux and type ctrl-b <command>, what does that tell the shell?

This command is not for the shell, it is for tmux.
ctrl-b is the prefix

Eg. ctrl-b c means to open a new window.

Date: / /
Page No.

Ques What does this do in tmux?
ctrl-b ,

This will allow you to rename the window

Ques In tmux, how do you switch from one window to the next/previous?

ctrl-b n for the next window
ctrl-b p " " previous window

Ques How do you create a window in tmux?

ctrl-b c creates a new window.

Ques How do you list the open windows in tmux?

ctrl-b w list the open windows
(scroll-able select-able list)

Ques What are panes in tmux?

You can split each window that you are in vertically or horizontally

Ques What is the default tmux command to split a window vertically?

ctrl-b % will let you split the window vertically.

Ques What do you do to horizontally split the window you are in?

ctrl-b :
then type
split-window

Ques How do you create a named tmux session?

tmux new -s sessionname
(that is how you create a new session called sessionname)

Ques How do you detach from a session?

ctrl-b d
this will detach you and keep the session running

Ques What will this command do on tmux:
tmux list-sessions

It will show you open sessions on a server.

Date: / /
Page No.

Ques What would this command do:
tmux attach -t sessionname.

It would re-attach you to a running session in tmux (on a server)

ARCHIVING & COMPRESSION ON LINUX - BASIC TAR COMMANDS

Ques What is the tar command?

It stands for tap archive. (It is an archiving utility)
- from the 1970s, like disco.

Ques What is archiving?

The process of taking all the files & directories on a system and making a single file that contains them all.
- creating one logical file out of all.

Ques The process of taking all the files

Ques What is compression?

using algorithms to look for repeating patterns inside files & save the space
- can be then expanded back to those files on other sites
- making files smaller.
(more simple a pattern, the more space you can save)

Date: / /
Page No.

Ques What would this command do:
tmux attach -t sessionname.

It would re-attach you to a running session in tmux (on a server)

ARCHIVING & COMPRESSION ON LINUX
- BASIC TAR COMMANDS

Ques What is the tar command?

It stands for tape archive. (It is an archiving utility)
- from the 1970s, like disco.

Ques What is archiving?

The process of taking all the files & directories on a system and making a single file that contains them all.
- creating one logical file out of all.

Ques The process of taking all the files

Ques What is compression?

using algorithms to look for repeating patterns inside files & save the space
- can be then expanded back to those files on other sim
- making files smaller.
(more simple a pattern, the more space you can save)

Date : / /
Page No.

Ques What is this command doing?
tar -z cvf docs.tar.gz Documents/

Archiving and compressing with tar
'f' will specify the filename for the directory (docs.tar.gz in this case)

'v' is verbose, means that we will get one line of output for each file we are compressing

'c' is for create, we are creating a new archive

'z' is for zipping (we are using the g zip program) used to compress archive as well as archiving it

Ques What is this command doing:
tar -z x f docs.tar.gz

Unzips a tar directory here (in this location)

'z' shows that it is g zipped

'x' means to extract it

'f' defines the archive filename (in this case docs.tar.gz)

Date : / /
Page No.

Ques. What is a tar bomb?

Nothing is contained
A directory that is compressed as if you were sitting in the directory. It blows up in your face and scatters files/ directories all over.

Try to compress files from outside a directory (one directory above)

Ques When should you create a tar bomb?

Never ever ever ever ever ever create a tar bomb.
(or when you don't like your new junior sysadmin)

Ques When you extract a directory, what should you do before you extract it?

Create a new directory and move inside that before extraction.
(if you extract a tar bomb, it will be contained inside the folder)

Date : / /
Page No.

BASH SCRIPTING

Ques Where does bash scripting shine?

Light task automation.

#1 VARIABLES & QUOTING

Ques What is this!
$varname

A Variable

Ques Why is it standard practice to put environment variable in all caps?

they are outside the programs
(so you know not to mess with them)
ie: $HOME (home directory)

Ques What is this doing!
myvar = "This is so wonderful"

Setting a variable
(remember no space)

Date: / /
Page No.

Ques What is this doing with the variable:
echo "This is my ${some_number}th test"

Delimiting it
(so that it can be inserted into a string

Ques What is this doing with the information inside `` characters
echo "there are `wc -l < /etc/group` lines in the /etc/shadow file"

It is telling the shell that there is a command inside this string
(and running it)
- after, results are subbed in

#2 How Bash Scripts Work

Ques What command do you use to find out where the Bash binary is?
(hint: the first thing you declare in every script)

which bash
(will output /bin/bash or other location it is stored in)

Date: / /
Page No.

Ques When you create a script, why should you use the file ending .sh?
(even though the file endings won't matter to the shell!)

Because then it will be easier for the people who come after you to understand what it is.

★ Return values can be anywhere between 0 through 255

Ques When you use 'exit' at the end by itself, what will the shell do?

It will look for the last item that was called and will return that as the program's return value
(as long as it returns 0/has no errors)

Ques When you use 'exit $?', what are you doing in your program?

You are showing that you are intentionally exiting and telling the program to output the last item that was called as the output.

Date : / /
Page No.

Ques When you execute a file, what permissions do you need to make sure that you have?

execute
(use Chmod on the file to allow execution of it)

Ques Running a program in the shell:
Why should you use: 'source helloworld.sh' rather than 'bash helloworld.sh'?

Because you will have access to the variables and information stored inside the script in your current bash session (makes it easier because you won't have to run things over and over again to modify certain elements.)

Ques What is the difference between these two:
source helloworld.sh
. helloworld.sh

There is no difference, both are interchangable.

#3 ~~VARIABLES~~ ARGUMENTS

Ques When the program is started, the filename is in what variable? (Bash Scripting)

The 0 variable
(that will be the name of our script)

Ques If you try to echo an argument that does not exist, the terminal will give you ---
(Bash Scripting)

a blank line back
(bash will forgive and not throw an error)

Ques Does bash use zero-indexing (does it start counting from 0, 1, 2. or does it start from 1, 2, 3)

Or does not use zero-indexing, it starts from 1, 2, 3

Eg. ./helloworld.sh Argument
↓
name, anything, etc.

* $1 means first arguments

Date: / /
Page No.

#4 IF ELSE STATEMENT

Eg.
```
if [ "$1" = "like" ]
then
    echo "Hey Please like this video"
else
    echo "okay, then Please subscribe :)"
fi
```

#5 IF ELIF

→ If Else If

this is one level advance of If Else statement and is used when is multiple conditions and we have to get outcome from multiple conditions

Eg.
```
#! /bin/bash
a = 100
b = 200
c = 300
```

* whenever we use strings like &&, greater than etc, we have to use [[ ]]
* -gt means greater than
* spacing is important in if brackets

```
if [[ $a -gt $b && $a -gt $c ]]
then
# a is greater than b and c
echo "A is biggest"
elif [[ $b -gt $a && $b -gt $c ]]
```

Date : / /
Page No.

b is greater than a and c
echo " B is biggest"

elif
echo " C is biggest"
fi

#6 Loops

There are total 3 looping statements which can be used in bash scripting

(i) ~~while statement~~ For statement
(ii) While statement
(iii) Until statement.

To alter the flow of loop statements, two commands are used they are

(i) break
(ii) Continue.

(i) For Loop

The for loop operate on lists of items. It repeats a set of commands for every item in a list. Here var is the name of a variable and word 1 to word N are sequences of characters separated by spaces. Each time the for loop executes, the value of the variable var is set to the next word in the list of words, word 1 to word N.

Date : / /
Page No.

Eg.
```
#!/bin/bash
for ((i=0; i<10; i++))
do
echo "$i"
done
```

It will print all ~~the~~ no. from 0 to 9.

Now we will use loop to iterate files

for that we will create 10 files

* advance touch command to create multiples files at the same time

~~touch file-{0}~~

```
touch file-{1..10}.txt
```

It will create 10 files.

file-1.txt, file-2.txt --- till file-10.txt.

Eg.
```
#!/bin/bash
for FILE in *
do
echo $FILE
done
```

* means all files in this folder.

eg.

```
#! /bin/bash

for FILE in *.txt
do
echo $FILE
done
```

this will only show txt files.

#7 FUNCTIONS

Eg. Function for adding user in the system

```
#! /bin/bash

add-user ()
{
  USER = $1
PASS = $2

useradd -m -P $PASS $USER && echo "successfully add user"

}

# MAIN

add_user Imran@ing test@123
```

Date : / /
Page No.

#8 ACTUAL SCRIPT

AUTOMATING BACKUPS WITH BASH SCRIPTS.

first we make a directory to all backups.

```
$ mkdir backups

$ nano backup.sh

#! /bin/bash

src_dir=/home/ubuntu/scripts
tgt_dir=/home/ubuntu/backups

curr_timestamp=$(date "+%Y-%m-%d-%H-%M-%S")

backup_file=$tgt_dir/$curr_timestamp.tgz

echo "Taking backup on $curr_timestamp"

tar czf $backup_file --absolute-names $src_dir

echo "Backup complete"*
```

Date : / /
Page No.

6(iii) WHILE Loop

Here command is evaluated and based on the result loop will executed, if command raise to false then loop will be terminated.

Eg,
```
#! /bin/bash
alert=90
df -H | awk '{print $5 " " $1}' | while read output;
do
  #echo " Disk Detail : $output"
  usage=$(echo $output | awk '{print $1}' | cut -d'%' -f1)
file-sys=$(echo $output | awk '{print $2}')
#echo $usage
if [ $usage -ge $alert ]
 then
      echo " CRITICAL for $file-sys"
fi
done
```

8 BASIC LSOF COMMANDS EVERY SYSADMIN SHOULD KNOW

lsof → It stands for list open files
↳ This command lists out all the files that are opened by any process in the system.

Date : / /
Page No.

Here, you observe there are details of files which are opened. Process id, the user associated with the process, FD, user, its size etc

- FD represents file descriptor
- cwd : current working directory
- txt : text file
- mem : Memory file
- mmap : Memmory mapped device

(i) lsof -u username

List of all files opened by a user.

- There are several users of a system and each user have different requirements and accordingly they use files and devices. To find a list of files that are opened by a specific user this command is useful.
Along with that we can see the type of file here and they are :-

- DIR : Directory
- REG : Regular file
- CHR : Character special file.

(ii) lsof -u ^username

List all files which are opened by everyone except a specific user

(iii) lsof -c processname

List all open files by a particular process.

Eg. lsof -c Mysql

(iv) lsof -p process ID

List all open files that are opened by a particular process using process ID.

(v) lsof -p ^process ID

List all files open by all other PID.

(vi) lsof -R

List parent process ID's

(vii) lsof -D directory path

List files opened by a directory.

Date: / /
Page No.

(viii) lsof -i

List all files opened by network connections.

Eg. lsof -i tcp

MONITORING WITH MONIT

Configured in /etc/monitrc
Stats to be monitored

- CPU
- Port 80 http
- Port 3306 mysql → it is TCP port.
  - memory usage
- php-fpm and nginx processes
- Site-specific monitoring config files in /etc/monit.d/sitename.cfg
- Web interface (local forwarding with SSH)
- Email alerts

Monit is a utility tool for monitoring & managing processes, files, directories, and filesystems on a unix system. It superpower or a super ability to start a process if its not running, restart a process if it isn't responding, and stop a process if it uses high resources that's what Monit does in laymen terms.

This open-source tool can be downloaded with simple apt-get command.

$ Sudo apt-get install monit

(i) monit -- version

To check the version of monit

(ii) Sudo apt-get install monit

To install monit on system

(iii) Systemctl start monit
Systemctl enable monit

We can start and enable the monit services with these commands

(iv) Systemctl status monit

To check the status of the Monit

#1. CONFIGURATION FOR WEB-INTERFACE

Monit has its own user-friendly web-interface. By default, it is disabled to enable it to follow the following section. With it, you can view the system status & manage the properties through the web-browser.

* monitrc → configuration file

Date : / /
Page No.

we will start by editing its configuration file /etc/monit/monitrc

sudo nano /etc/monit/monitrc

Un-comment the lines of http local host from monit. This will allow the interface to run on localhost port 2812.

Save and restart the Monit service

systemctl restart monit

Go to localhost : 2812 in browser and when prompted type Username as "admin" and password as "monit".

v) sudo tail -f /var/log/monit.log

To verify Monit logs

#2. CONFIGURE MONIT FOR APACHE, SSH & FTP

Install Apache and vsftpd to the system

Install Apache and vsftpd to the system

sudo apt-get install apache2 vsftpd

Date : / /
Page No.

Now, create a configuration file for vsftpd.

sudo nano /etc/monit/conf-available/vsftpd

Add these lines to vsftpd!

check process vsftpd with pidfile /var/run/vsftpd/vsftpd.pid
start program = "/etc/init.d/vsftpd start"
stop program = "/etc/init.d/vsftpd stop"
if failed port 21 protocol ftp then restart

Save and close nano and enable the configuration file by creating a symbolic link

sudo ln -s /etc/monit/conf-available/vsftpd /etc/monit/conf-enabled/

Also, enable Apache and SSH configuration file in similar way.

sudo ln -s /etc/monit/conf-available/apache2 /etc/monit/conf-enabled
sudo ln -s /etc/monit/conf-available/openssh-server /etc/monit/conf-enabled

and verify monit status

monit -t

then restart monit

systemctl reload monit

Date : / /
Page No.

To check the status of services like CPU, file system and Network

sudo monit status

MANAGING SERVICES : SYSTEMD

Systemd is a system that is designed specifically for the Linux Kernel. It replaces the sysvinit process to become the first process with PID = 1, which gets executed in user space during the Linux start-up process.

We'll mostly use two utility programs to manage services with systemd :

#.1 Systemctl : manages services

#.2 journalctl : manages logs

#.1 SYSTEMCTL COMMANDS

i) Systemctl enable service name

It enables the service and makes sure the service always start at boot

Eg. systemctl enable mysql nginx php5.5-fpm monit
(different services)

Date : / /
Page No.

(iii) Systemctl disable servicename

It does opposite of enable

(iii) Systemctl start servicename

It starts the service now (will not automatically start at boot)

Eg.
Systemctl start nginx php5-fpm monit

(iv) Systemctl stop servicename

It stops the service now (will not prevent starting at boot, if enabled)

(v) Systemctl reload servicename

re-read the program configuration files (config updates)

(vi) Systemctl restart servicename

kill the process and start it again, re-reading configuration files.

(vii) **Systemctl status servicename**

Check the status of service, shows last few lines of log output.

Eg.
systemctl status mysql

(viii) **Systemctl halt**
Systemctl poweroff
Systemctl reboot

To reboot the system, these commands are used.

TEE COMMAND IN LINUX

Tee command reads the standard input and writes it to both the standard output and one or more files. The command is named after the T-splitter used in plumbing.

Eg. date | tee -a mylog.log
(mylog.log: file name)

-a → It basically do not overwrite the file but append to the given file.

Date : / /
Page No.

How To Backup And Restore a MySQL/MariaDB Database

Database Backups

Store mysql root password in /root/.my.cnf so our script (running a root) can access it. Chmod 600.

```
[client]
user = root
password = yourmysql rootpass
```

Backup One User Database?

```
mysqldump --add-drop-table --databases yourdatabasename >
/home/<username>/backups/db/$(/bin/date +\%Y-\%m-\%d).sql.bak
```

Backup All Databases

```
mysqldump --all-databases --all-routines > /path/to/fulldump.sql
```

Restoring a Dabase Backup

Creating post, backups, delete post, restore database.

Date : / /
Page No.

To restore a single Database.

mysql -u root -p [database name] < backupname.sql

To restore all databases (tables need to exist, or backup needs to contain (CREATE TABLE statements):

mysql -u root -p < alldatabases.sql

* Schedule this backup command with the help of crontab

VI/VIM BASICS

(i) :q
to quit out of the file

(iii) :w
Write the current state to the file or save the file.

(iii) :wq
to save and exit/quit the file

Date : / /
Page No.

(iv) :q!

to exit / quit without saving

(v) i

hit i to enter into insert mode.

(vi) Esc

to get out of the insert mode back to command mode.

(vii) dd (in command mode)

it will delete the line

(viii) any no. dd (in command mode)

it will delete that no. of lines.

(ix) u (in command mode)

to undo the last action.

(x) cntrl r (in cmd mode)

to redo the last action

(xi) /yourtext (in cmd mode)

it is used to search in file

→ n to go steady forward
→ shift+n to go u backword

(xii) :%s/yourtext/replacetext/gc (in cmd file)

to search and replace text in file

10 LINUX FOR INTERVIEW QUESTIONS

Ques How can you see which kernel version a system is currently running?

Ans→ uname -a

Everything it got, every info about the system

uname -v

for kernel version

uname -r

for kernel release

Date : / /
Page No.

Ques How can you check a systems current IP address?

Ans ifconfig

will show everything

ip addr show

will give IP address of all devices

ip addr show specific device

to show IP address of specific device.

Ques How do you check for free disk space?

Ans df -th

to check free space for all filesystem in human readable format

Ques How do you manage services on a system?

systemctl status servicename

Date : / /
Page No.

Ques How would you check the size of a directory's contents and disk?

```
du -sh directory name
```

Ques How would you check for open ports / on a ~~Linux~~ listening sockets on a Linux system.

```
sudo netstat -tulpn
```

using sudo will show PID/program name as well

Ques How do you check CPU usage for a process?

```
ps aux | grep nginx
```
process name

or

```
top
```

or

```
htop
```

Date : / /
Page No.

Q. How do you mount a new volume/ drive in a system?

ls /mnt → mount directory on root

mount [/dev/sda2/mnt]
absolute path to the volume
command to mount the drive

mount → to check for existing mounts

less /etc/fstab → to mount the file at boot

How To Safely Delete Linux System Logs

journalctl --vacuum-time=2d or 2h or 2m
can be anything

It will delete the logs older than the specified time.

journalctl --vacuum-size=100M

It will delete the logs larger than the specified size.

Date : / /
Page No.

How To Tail (Follow) Linux Service Logs

journalctl -fu process/unit name

Eg. journalctl -fu nginx

-u = unit
-f = follow