172.16.8.84

Ex No: 1 b

Date: 17.2.2024

BASIC LINUX COMMANDS

1.1 GENERAL PURPOSE COMMANDS

1. The 'date' command:

The date command display the current date with day of week, month, day, time (24 hours clock) and the year.

SYNTAX: $ date

The date command can also be used with following format.

Format	Purpose	Example
+ %m	To display only month	$ date + %m
+ %h	To display month name	$ date + %h
+ %d	To display day of month	$ date + %d
+ %y	To display last two digits of the year	$ date + %y
+ %H	To display Hours	$ date + %H
+ %M	To display Minutes	$ date + %M
+ %S	To display Seconds	$ date + %S

date +%m

OUTPUT

Sat Feb 17 12:20:26 IST 2024

02
Feb
17
24
12
25
28

2. The echo'command:

The echo command is used to print the message on the screen.

SYNTAX: $ echo

EXAMPLE: $ echo "God is Great"

OUTPUT echo " Hello,World!"

Hello World!

3. The 'cal' command:

The cal command displays the specified month or year calendar.

SYNTAX: $ cal [month] [year]

EXAMPLE: $ cal Jan 2012

OUTPUT

Cal Nov 2024

November 2024

Su	Mo	Tu	We	Th	Fr	Sa
					1	2
3	4	5	6	7	8	9
10	11	12	13	14	15	16
17	18	19	20	21	22	23
24	25	26	27	28	29	30

4. The 'bc' command:

Unix offers an online calculator and can be invoked by the command bc.

SYNTAX: $ bc → integer only calculation

EXAMPLE: bc –l → floating point calculation

16/4

5/2

OUTPUT

bc

bc 1.06.95

16/4

4

bc -l

16/4

4.0000000000000000000

5. The 'who' command

The who command is used to display the data about all the users who are currently logged into the system.

SYNTAX: $ who

OUTPUT

```
osuser310 pts/0   2024-02-17 12:36 (172.16.8.104)
osuser309 pts/1   2024-02-17 12:36 (172.16.8.212)
osuser331 pts/2   2024-02-17 12:36 (172.16.8.85)
   :
```

6. The 'who am i' command

The who am i command displays data about login details of the user.

SYNTAX: $ who am i

OUTPUT

```
osuser331 pts/2 2024-02-17 12:36 (172.16.8.85)
```

7. The 'id' command

The id command displays the numerical value corresponding to your login.

SYNTAX: $ id

OUTPUT

```
uid=1708(osuser331) gid=1708(osuser331) groups=1708(osuser331)
context=unconfined_u:unconfined_r:unconfined_t:s0-c0.c1023
```

8. The 'tty' command

The tty (teletype) command is used to know the terminal name that we are using.

SYNTAX: $ tty

OUTPUT

```
/dev/pts/2
```

9. The 'clear' command

The clear command is used to clear the screen of your terminal.

SYNTAX: $ clear

OUTPUT

10. The 'man' command → gives description of the command

The man command gives you complete access to the Unix commands.

SYNTAX: $ man [command]

OUTPUT man clear

NAME

clear - clear the terminal screen

SYNOPSIS

clear

DESCRIPTION

clear clears your screen. . .

11. The 'ps' command

The ps command is used to the process currently alive in the machine with the 'ps' (process status) command, which displays information about process that are alive when you run the command. 'ps;' produces a snapshot of machine activity.

SYNTAX: $ ps

EXAMPLE: $ ps

$ ps –e

$ps –aux

OUTPUT PS

PID	TTY	TIME	CMD
1166	pts/2	00:00:00	bash
2434	pts/2	00:00:00	bc
2440	pts/2	00:00:00	bc
⋮			

12. The 'uname' command

The uname command is used to display relevant details about the operating system on the standard output.

-m -> Displays the machine id (i.e., name of the system hardware)

-n -> Displays the name of the network node. (host name)

-r -> Displays the release number of the operating system.

-s -> Displays the name of the operating system (i.e.. system name)

-v -> Displays the version of the operating system.

-a -> Displays the details of all the above five options.

SYNTAX: $ uname [option]

EXAMPLE: $ uname –a

OUTPUT

uname-a

Linux localhost.localdomain 4.11.8-300.fc26.i686+PAE #1 SMP Thu Jun 29 20:38:21 UTC 2017 i686 i686 i386 GNU/Linux

1.2 DIRECTORY COMMANDS

1. The 'pwd' command:

The pwd (print working directory) command displays the current working directory.

SYNTAX: $ pwd

OUTPUT

/home/osusex331

2. The 'mkdir' command:

The mkdir is used to create an empty directory in a disk.

SYNTAX: $ mkdir dirname

EXAMPLE: $ mkdir receee

OUTPUT mkdir shiiv

3. The 'rmdir' command:

The rmdir is used to remove a directory from the disk. Before removing a directory, the directory must be empty (no files and directories).

SYNTAX: $ rmdir dirname

EXAMPLE: $ rmdir receee

OUTPUT

rmdir shiiv

4. The 'cd' command:

The cd command is used to move from one directory to another.

SYNTAX: $ cd dirname

EXAMPLE: $ cd receee

OUTPUT

cd shiiv

[osuser331@local host shiiv] $

5. The 'ls' command:

The ls command displays the list of files in the current working directory.

SYNTAX: $ ls

EXAMPLE: $ ls

$ ls –l

$ ls –a

OUTPUT

ls

hello

To append file

cat >> filename.

1.3 FILE HANDLING COMMANDS

1. The 'cat' command:

The cat command is used to create a file.

SYNTAX: $ cat > filename

EXAMPLE: $ cat > rec

OUTPUT

cat > hello

Hello, world!

My name is shiiv!

Bye :)

2. The 'Display contents of a file' command:

The cat command is also used to view the contents of a specified file.

SYNTAX: $ cat filename

OUTPUT

cat hello
Hello, world!
My name is Shiiv!
Bye :)

3. The 'cp' command:

The cp command is used to copy the contents of one file to another and copies the file from one place to another.

SYNTAX: $ cp oldfile newfile

EXAMPLE: $ cp cse ece

OUTPUT

cp hello hi

4. The 'rm' command:

The rm command is used to remove or erase an existing file

SYNTAX: $ rm filename

rmdir dirname (TO delete directory)

EXAMPLE: $ rm rec

$ rm –f rec

Use option –fr to delete recursively the contents of the directory and its subdirectories.

OUTPUT

rm hi

5. The 'mv' command:

The mv command is used to move a file from one place to another. It removes a specified file from its original location and places it in specified location.

SYNTAX: $ mv oldfile newfile

EXAMPLE: $ mv cse eee

OUTPUT mv hello hi

6. The 'file' command:

The file command is used to determine the type of file.

SYNTAX: $ file filename

EXAMPLE: $ file receee

OUTPUT file hi
hi : ASCII text

7. The 'wc' command: → word count

The wc command is used to count the number of words, lines and characters in a file. lines, words & characters

SYNTAX: $ wc filename

EXAMPLE: $ wc receee

wc -l → no. of line
wc -w → " " words
wc -c " " chars.

OUTPUT wc hello
3 9 39 hello

8. The 'Directing output to a file' command:

The ls command lists the files on the terminal (screen). Using the redirection operator '>' we can send the output to file instead of showing it on the screen.

SYNTAX: $ ls > filename

EXAMPLE: $ ls > cseeee

OUTPUT ls > hello
cat hello
hello
hi

9. The 'pipes' command:

The Unix allows us to connect two commands together using these pipes. A pipe (|) is an mechanism by which the output of one command can be channeled into the input of another command.

SYNTAX: $ command1 | command2

EXAMPLE: $ who | wc –l → doing no. of lines in who command

35

ls | wc -w

2.

10. The 'tee' command:

While using pipes, we have not seen any output from a command that gets piped into another command. To save the output, which is produced in the middle of a pipe, the tee command is very useful.

SYNTAX: $ command | tee filename

EXAMPLE: $ who | tee sample | wc -l → save output of who in new file called sample.

ls| tee list| wc -l

cat list

hello hi list mr_rogers sample

5

The 'Metacharacters of unix' command:

Metacharacters are special characters that are at higher and abstract level compared to most of other characters in Unix. The shell understands and interprets these metacharacters in a special way.

* - Specifies number of characters

?- Specifies a single character

[ ]- used to match a whole set of file names at a command line.

! – Used to Specify

EXAMPLE:

ls h**
hello hi

$ ls r** - Displays all the files whose name begins with 'r'

$ ls ?kkk - Displays the files which are having 'kkk', from the second characters irrespective of the first character.

ls ??llo
hello

$ ls [a-m]* – Lists the files whose names begins alphabets from 'a' to 'm'

$ ls [!a-m]* – Lists all files other than files whose names begins alphabets from 'a' to 'm'

ls [a-m]*

11. The 'File permissions' command:

File permission is the way of controlling the accessibility of file for each of three users namely Users, Groups and Others.

There are three types of file permissions are available, they are

r-read
w-write
x-execute

The permissions for each file can be divided into three parts of three bits each.

First three bits	Owner of the file
Next three bits	Group to which owner of the file belongs
Last three bits	Others

EXAMPLE: $ ls college

-rwxr-xr-- 1 Lak std 1525 jan10 12:10 college

Where,

-rwx The file is readable, writable and executable by the owner of the file.

Lak Specifies Owner of the file.

r-x Indicates the absence of the write permission by the Group owner of the file.

Std Is the Group Owner of the file.

r-- Indicates read permissions for others.

12. The 'chmod' command:

The chmod command is used to set the read, write and execute permissions for all categories of users for file.

To remove permission to access certain files & directories

SYNTAX: $ chmod category operation permission file

Category	Operation	permission
u-users	+ assign	r-read
g-group	-Remove	w-write
o-others	= assign absolutely	x-execute
a-all		

EXAMPLE:

no space

$ chmod u –wx college

chmod u-w sample.

Removes write & execute permission for users for 'college' file.

$ chmod u +rw, g+rw college

Assigns read & write permission for users and groups for 'college' file.

$ chmod g=wx college

Assigns absolute permission for groups of all read, write and execute permissions for 'college' file.

13. The 'Octal Notations' command:

The file permissions can be changed using octal notations also. The octal notations for file permission are

Read permission	4
Write permission	2
Execute permission	1

all three = 7
only rw = 6

EXAMPLE:

$ chmod 761 college

Assigns all permission to the owner, read and write permissions to the group and only executable permission to the others for 'college' file.

1.4 GROUPING COMMANDS

1. The 'semicolon' command:

The semicolon(;) command is used to separate multiple commands at the command line.

print both outputs

SYNTAX: $ command1;command2;command3… ..... ;commandn

EXAMPLE: $ who;date who am i ;date

op: OSUSCL331 pts/1 2024-2-27 13:28 (172.16.8.85)
Tue Feb 27 13:52:22 IST 2024

2. The '&&' operator:

The '&&' operator signifies the logical AND operation in between two or more valid Unix commands.It means that only if the first command is successfully executed, then the next command will executed.

SYNTAX: $ command1 && command && command3…….....&&commandn

EXAMPLE: $ who && date date && whoami
Tue Feb 27 13:53:57 IST 2024
OSUSCL331

3. The '||' operator:

The '||' operator signifies the logical OR operation in between two or more valid Unix commands.It means, that only if the first command will happen to be un successfully,it will continue to execute next commands. otherwise stop after 1st command

SYNTAX: $ command1 || command || command3..... ||commandn

EXAMPLE: $ who || date date || who
Tue Feb 27 13:57:42 IST 2024

1.5 FILTERS

only 1st 10

1. The head filter

It displays the first ten lines of a file.

LIKE PYTHON.

SYNTAX: $ head filename

EXAMPLE: $ head college — Display the top ten lines.

$ head -5 college — Display the top five lines.

head -2 hi
op: Hello World,
my name is Shiiv !

last 10

2. The tail filter

It displays ten lines of a file from the end of the file.

SYNTAX: $ tail filename

EXAMPLE: $ tail college — Display the last ten lines.

$tail -5 college — Display the last five lines.

tail -2 hi
op : My name is Shiiv!
Byc :)

3. The more filter:

The pg command shows the file page by page.

SYNTAX: $ ls –l | more → do outside directory.

total 4

drwxr--r--. 2 osuser331 osuser331 4096 Feb 24 13:19 shiiv

4. The 'grep' command: → search for pattern in a file

This command is used to search for a particular pattern from a file or from the standard input and display those lines on the standard output. "Grep" stands for "global search for regular expression."

SYNTAX: $ grep [pattern] [file_name]

grep world hi

EXAMPLE: $ cat > student

o/P: Hello world,

Arun cse

Ram ece

Kani cse

$ grep "cse" student

Arun cse

Kani cse

5. The 'sort' command:

The sort command is used to sort the contents of a file. The sort command reports only to the screen, the actual file remains unchanged.

SYNTAX: $ sort filename

sort -r hello

OP: sample
mr_rogers
list
hi
hello

EXAMPLE: $ sort college

OPTIONS:

Command	Purpose
Sort –r college	Sorts and displays the file contents in reverse order
Sort –c college	Check if the file is sorted
Sort –n college	Sorts numerically
Sort –m college	Sorts numerically in reverse order
Sort –u college	Remove duplicate records
Sort –l college	Skip the column with +1 (one) option.Sorts according to second column

6. The 'nl' command: → prints the file content w/ no.s but doesn't save the numbering.

The nl filter adds lines numbers to a file and it displays the file and not provides access to edit but simply displays the contents on the screen.

nl hello

SYNTAX: $ nl filename

O/P
1 hello
2 hi
3 list
4 mr_rogers
5 sample

EXAMPLE: $ nl college

7. The 'cut' command: ?

We can select specified fields from a line of text using cut command.

SYNTAX: $ cut -c filename

cut -c -3 hello
sam
hel
hi
lis
mr-

EXAMPLE: $ cut -c college

OPTION:

-c – Option cut on the specified character position from each line.

1.5 OTHER ESSENTIAL COMMANDS

1. **free**

Display amount of free and used physical and swapped memory system.

synopsis- free [options]

example

[root@localhost ~]# free -t → total

	total	used	free	shared	buff/cache	available
Mem:	4044380	605464	2045080	148820	1393836	3226708
Swap:	2621436	0	2621436			
Total:	6665816	605464	4666516			

- c N → count
- s N → seconds } repeat after every N secs
- l → lohi (low & hi)
- v → version
- b → bytes
- k, -m, -g → giga, tera →
  kilo mega
- w → wide
- t → total

2. **top**

It provides a dynamic real-time view of processes in the system.

synopsis- top [options]

example top

O/P PID	USER	PR	NI	VIRT	RES	SHR	S	% CPU	% MEM	TIME +	COMMAND
776	root	20	0	550292	30580	24980	S	0.3	0.4	0:00.74	rsyslogd
3819	root	20	0	420020	10240	9168	S	0.3	0.1	0:00.07	sshd
8055	osuser3+	20	0	17032	4552	3132	R	0.3	0.1	0:00.74	top

⋮

[root@localhost ~]# top

top - 08:07:28 up 24 min, 2 users, load average: 0.01, 0.06, 0.23

Tasks: 211 total, 1 running, 210 sleeping, 0 stopped, 0 zombie

%Cpu(s): 0.8 us, 0.3 sy, 0.0 ni, 98.9 id, 0.0 wa, 0.0 hi, 0.0 si, 0.0 st

KiB Mem : 4044380 total, 2052960 free, 600452 used, 1390968 buff/cache

KiB Swap: 2621436 total, 2621436 free, 0 used. 3234820 avail Mem

PID	USER	PR	NI	VIRT	RES	SHR	S	%CPU	%MEM	TIME+	COMMAND
1105	root	20	0	175008	75700	51264	S	1.7	1.9	0:20.46	Xorg
2529	root	20	0	80444	32640	24796	S	1.0	0.8	0:02.47	gnome-term

3. **ps**

It reports the snapshot of current processes

synopsis- ps [options]

<u>example</u>

[root@localhost ~]# ps -e

PID	TTY	TIME	CMD
1	?	00:00:03	systemd
2	?	00:00:00	kthreadd
3	?	00:00:00	ksoftirqd/0

PS

PID	TTY	TIME	CMD
3705	pts/1	00:00:00	bash
6179	pts/1	00:00:00	cat
6197	pts/1	00:00:00	cat

⋮

4. **vmstat**

It reports virtual memory statistics

synopsis- vmstat [options]

<u>example</u>

[root@localhost ~]# vmstat

procs ----------memory---------- --swap- - io----- -system- -------cpu------

r b swpd free buff cache si so bi bo in cs us sy id wa st

0 0 0 1879368 1604 1487116 0 0 64 7 72 140 1 0 97 1 0

vmstat

procs ------- memory ---- ---swap---- io ---system--- cpu---

r b swpd free buff cache si so bi bo in cs us sy id wa st

0 0 0 6327196 101728 1053324 0 0 10 3 62 93 96 0 0 99 0 0

5. **df**

It displays the amount of disk space available in file-system.

Synopsis- df [options]

example

[root@localhost ~]# df

Filesystem	1K-blocks	Used	Available	Use%	Mounted on
devtmpfs	2010800	0	2010800	0%	/dev
tmpfs	2022188	148	2022040	1%	/dev/shm
tmpfs	2022188	1404	2020784	1%	/run
/dev/sda6	487652	168276	289680	37%	/boot

6. ping

It is used verify that a device can communicate with another on network. PING stands for Packet Internet Groper.

synopsis- ping [options]

[root@localhost ~]# ping 172.16.4.1

PING 172.16.4.1 (172.16.4.1) 56(84) bytes of data.
64 bytes from 172.16.4.1: icmp_seq=1 ttl=64 time=0.328 ms
64 bytes from 172.16.4.1: icmp_seq=2 ttl=64 time=0.228 ms
64 bytes from 172.16.4.1: icmp_seq=3 ttl=64 time=0.264 ms
64 bytes from 172.16.4.1: icmp_seq=4 ttl=64 time=0.312 ms
^C
--- 172.16.4.1 ping statistics ---
4 packets transmitted, 4 received, 0% packet loss, time 3000ms
rtt min/avg/max/mdev = 0.228/0.283/0.328/0.039 ms

ping 172.16.8.84
64 bytes from 172.16.8.84 : icmp_seq=1 ttl=64 time=0.054 ms
or
.
.
.

7. ifconfig

It is used configure network interface.

synopsis- ifconfig [options]

example

```
[root@localhost ~]# ifconfig

enp2s0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
        inet 172.16.6.102  netmask 255.255.252.0  broadcast 172.16.7.255
        inet6 fe80::4a0f:cfff:fe6d:6057  prefixlen 64  scopeid 0x20<link>
        ether 48:0f:cf:6d:60:57  txqueuelen 1000  (Ethernet)
        RX packets 23216  bytes 2483338 (2.3 MiB)
        RX errors 0  dropped 5  overruns 0  frame 0
        TX packets 1077  bytes 107740 (105.2 KiB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0
```

8. traceroute

It tracks the route the packet takes to reach the destination.

synopsis- traceroute [options]

example

```
[root@localhost ~]# traceroute www.rajalakshmi.org
traceroute to www.rajalakshmi.org (220.227.30.51), 30 hops max, 60 byte packets
1  gateway (172.16.4.1)  0.299 ms  0.297 ms  0.327 ms
2  220.225.219.38 (220.225.219.38)  6.185 ms  6.203 ms  6.189 ms
```

traceroute chat.openai.com
(104.18.37.228)
1. gateway (172.16.8.1) 0.264 ms
0.221 ms 0.337 ms

RESULT

The above commands have been executed and the output has been verified successfully.

27/2/24