

Experiment 1

Exploring basic commands of Unix/Linux

1. General Purpose Utilities (who,whoami,which,uname,ttty,cal, date,bc)
2. Working With Directories (mkdir,rmdir,ls,cd,rm)
3. File Handling Commands (touch,cat,cp,rm,mv,wc,cmp,diff,comm,ln,cut)
4. Changing File Attributes (chmod,umask)

1.**who**The who command gives the information about the users logged on to the system.

Syntax:who

2.whoami

It tells you about the system's username.

Syntax: whoami

3. which command

The Linux **which** command is used to **locate the executable files** or location of a program from the file system. It displays the path where the specified file or command is stored.

If you are curious to know where the specified program is stored, **which** command will help you to identify the path, it is quite straightforward to use.

Syntax:which <program name>

which node

4.uname

The command '*uname*' displays the information about the system.

Syntax:

uname [OPTION]

Options and Examples

1. **-a option:** It prints all the system information in the following order: *Kernel name, network node hostname, kernel release date, kernel version, machine hardware name, hardware platform, operating system*
2. **-s option:** It prints the kernel name
3. **-n option:** It prints the hostname of the network node(current computer)
4. **-r option:** It prints the kernel release date.
5. **-v option:** It prints the version of the current kernel
6. **-m option:** It prints the machine hardware name
7. **-p option:** It prints the type of the processor.
8. **-i option:** It prints the platform of the hardware.
9. **-o option:** It prints the name of the operating system

5. **TTY** The **tty** command of terminal basically prints the file name of the terminal connected to standard input. **tty** is short of teletype, but popularly known as a terminal it allows you to interact with the system by passing on the data (you input) to the system, and displaying the output produced by the system.

tty [OPTION]....-s, —silent, —quiet : Prints nothing, only returns an exit status.

- **—help** : It will display the help message and exit.
- **—version** : Prints the version information and exits

6. **cal** command is a calendar command in Linux which is used to see the calendar of a specific month or a whole year.

Syntax:

cal [[month] year]

7. **date** command is used to display the system date and time. **date** command is also used to set date and time of the system. By default the **date** command displays the date in the time zone on which unix/linux operating system is configured. You must be the super-user (root) to change the date and time.

Syntax:

date

8. **bc** command is used for command line calculator. It is similar to basic calculator by using which we can do basic mathematical calculations.

Input : `$ echo "12+5" | bc`

Output : 17

Input : `$ echo "10^2" | bc`

Output : 100

Input: `$ echo "var=10;var" | bc`

Output: 10

9. **pwd** Command

The **pwd** command is used to display the location of the current working directory.

Syntax:

pwd

10. **mkdir** Command

The **mkdir** command is used to create a new directory under any directory.

Syntax: mkdir <directory name>

11. rmdir Command

The **rmdir** command is used to delete a directory.

Syntax: rmdir <directory name>

12. ls Command

The **ls** command is used to display a list of content of a directory.

Syntax: ls

5.The **cd** command is used to change the current directory.

Syntax:

cd <directory name>

13. touch Command

The **touch** command is used to create empty files. We can create multiple empty files by executing it once.

Syntax: touch <file name>

touch <file1> <file2>

14. cat Command

The **cat** command is a multi-purpose utility in the Linux system. It can be used to create a file, display content of the file, copy the content of one file to another file, and more.

Syntax: cat [OPTION]... [FILE]..

cat > <file name>
// Enter file content

Press "**CTRL+ D**" keys to save the file. To display the content of the file, execute it as follows:

cat <file name>

cat is also used for merging of files
Syntax :cat file1.txt file2.txt>filemerge.txt

15. rm Command

The **rm** command is used to remove a file.

Syntax: rm <file name>

16.. cp Command

The **cp** command is used to copy a file or directory.

Syntax:To copy in the same directory:

cp <existing file name> <new file name>

17. mv Command

The **mv** command is used to move a file or a directory from one location to another location.

Syntax:mv <file name> <directory path>

18.wc

wc 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 arguments.
- 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.
- **wc [OPTION]... [FILE]...**

```
$ cat state.txt
```

```
Andhra Pradesh
Arunachal Pradesh
Assam
Bihar
Chhattisgarh
```

```
$ wc state.txt
```

```
5 7 58 state.txt
```

Options: 1. **-l:** This option prints the **number of lines** present in a file.

-w: This option prints the **number of words** present in a file

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

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

```
5 state.txt
```

```
$ wc -w state.txt
```

```
7 state.txt
```

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

19. cmp command in Linux/UNIX is used to compare the two files byte by byte and helps you to find out whether the two files are identical or not.

- When cmp is used for comparison between two files, it reports the location of the first mismatch to the screen if difference is found and if no difference is found *i.e* the files compared are identical.
- cmp displays no message and simply returns the prompt if the files compared are identical.
 - **\$cmp file1.txt file2.txt**
 - **\$cmp file1.txt file2.txt**
file1.txt file2.txt differ: byte 9, line 2

/*indicating that the first mismatch found in
two files at byte 20 in second line*/

20. diff stands for **difference**. This command is used to display the differences in the files by comparing the files line by line. Unlike its fellow members, [cmp](#) and [comm](#), it tells us which lines in one file have to be changed to make the two files identical.

diff [options] File1 File2

```
$ diff a.txt b.txt
```

```
0a1
```

```
> Tamil Nadu
```

```
2,3c3
```

```
< Uttar Pradesh
```

```
Andhra Pradesh
```

```
5c5
```

```
Uttar prades
```

21. comm Command The '[comm](#)' command is used to compare two files or streams. By default, it displays three columns, first displays non-matching items of the first file, second indicates the non-matching item of the second file, and the third column displays the matching items of both files.

Syntax:

```
comm <file1> <file2>
```

22.Chmod Linux chmod command is used to change the access permissions of files and directories. It stands for **change mode**. It can not change the permission of symbolic links. Even, it ignores the symbolic links come across recursive directory traversal.

In the [Linux](#) file system, each file is associated with a particular owner and have permission access for different users. The user classes may be:

- o owner
- o group member

- o Others (Everybody else)

The **file permissions** in Linux are the following three types:

- o read (r)
- o write (w)
- o execute (x)

Syntax:

The basic syntax of chmod command is as follows:

1. `chmod <options> <permissions> <file name>`
2. The chmod command supports the following command-line options:
3. **-c, --changes:** It is similar to the verbose option, but the difference is that it is reported if a change has been made.
4. **-f, --silent, --quiet:** It is used to suppress the error messages.
5. **-v, --verbose:** It is used to display a diagnostic for every processed file.
6. **--no-preserve-root:** It is used for not treating the backslash symbol ('/'), especially (the default).
7. **--preserve-root:** If this option is used, it will fail to operate recursively on backslash ('/').
8. **--reference=RFILE:** It is used to specify the RFILE's mode alternatively MODE values.
9. **-R, --recursive:** It is used to change files and directories recursively.
10. **--help:** It is used to display the help manual having a brief description of usage and support options.
11. **--version:** It is used to display the version information.

Setting and Updating the Permissions

To set the permission of a file, execute a permission statement with the chmod command. For example, we want to set the read and write permission for all users and groups of file 'Demo.txt.' We have to pass the "u=rw,go=rw Demo.txt" permission statement with chmod command. To display the file permission, execute the below command:

1. `ls -l Demo.txt`
The above command will display the file's current file permission of the 'Demo.txt' file.

To change the permission, execute the below command:

1. `chmod u=rw,go=rw Demo.txt`

Consider the below output:

```
javatpoint@javatpoint-Inspiron-3542:~$ groups
javatpoint adm cdrom sudo dip plugdev lpadmin sambashare
```

From the above output, the access permission of 'Demo.txt' has changed.

Setting Permissions for Multiple Files

We can set permission for multiple files at once by using the chmod command. To change the file permission of multiple files, specify the file pattern with the chmod command. For example, if we want to set read and write permission for all text files, specify the *.txt pattern with chmod command.

To view the permission of all text file from the current working directory, execute the below command:

1. `ls -l *.txt`

It will list all the text files with their permission mode. Consider the below output:

```
javatpoint@javatpoint-Inspiron-3542:~$ ls -l *.txt
-rw-r--r-- 1 javatpoint javatpoint 21 Jun  3 01:47 Demo1.txt
-rw-rw-rw- 1 javatpoint javatpoint  0 Jul  8 23:43 Demo.txt
-rw-r--r-- 1 javatpoint javatpoint 37 Jun 23 01:01 lookups.txt
-rw-r--r-- 1 javatpoint javatpoint 37 Jul  2 20:50 pings.txt
-rw-r--r-- 1 javatpoint javatpoint  0 Jun 11 21:21 ref.txt
-rw-r--r-- 1 javatpoint javatpoint 131 Jun  8 22:46 time.txt
-rw-r--r-- 1 javatpoint javatpoint 132 Jun  8 22:32 timme.txt
```

From the above output, many files have only read permission for other users.

To set the read and write permission for other users, execute the below command:

1. `chmod o+w *.txt`

It will set the read and write permission for other users of the text files. Consider the below output:

```
javatpoint@javatpoint-Inspiron-3542:~$ chmod o+w *.txt
javatpoint@javatpoint-Inspiron-3542:~$ ls -l *.txt
-rw-r--rw- 1 javatpoint javatpoint 21 Jun  3 01:47 Demo1.txt
-rw-rw-rw- 1 javatpoint javatpoint  0 Jul  8 23:43 Demo.txt
-rw-r--rw- 1 javatpoint javatpoint 37 Jun 23 01:01 lookups.txt
-rw-r--rw- 1 javatpoint javatpoint 37 Jul  2 20:50 pings.txt
-rw-r--rw- 1 javatpoint javatpoint  0 Jun 11 21:21 ref.txt
-rw-r--rw- 1 javatpoint javatpoint 131 Jun  8 22:46 time.txt
-rw-r--rw- 1 javatpoint javatpoint 132 Jun  8 22:32 timme.txt
```

23. The umask command in Linux is used to set default permissions for files or directories the user creates.

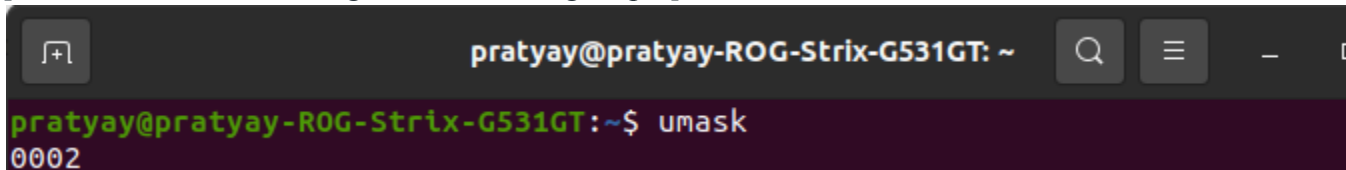
How does the umask command work?

- The umask command specifies the permissions that the user does not want to be given out to the newly created file or directory.
- umask works by doing a Bitwise AND with the bitwise complement (where the bits are inverted, i.e. 1 becomes 0 and 0 becomes 1) of the umask.
- The bits which are set in the umask value, refer to the permissions, which are not assigned by default, as these values are subtracted from the maximum permission for files/directories.

- **Syntax:**

- \$umask

- [The above command will give the following output]

A terminal window with a dark background. The prompt is 'pratyay@pratyay-ROG-Strix-G531GT: ~'. The command 'umask' has been entered, and the output '0002' is displayed on the next line.

```
pratyay@pratyay-ROG-Strix-G531GT: ~  
pratyay@pratyay-ROG-Strix-G531GT:~$ umask  
0002
```

- Here, the first digit, 0 is called the sticky bit, it is a special security feature.
- The next three digits represent the octal values of the umask for a file or directory.

For a better understanding of umask working, we need to understand octal mode security settings. The three rwx permissions (Read-Write-Execute) values are converted into three-bit binary values and represented by a single octal value as shown in the following table:

Permissions	Octal Value	Binary Value	Description
—	0	000	No permission
—x	1	001	only permission to execute
-w-	2	010	only permission to write
-wx	3	011	permission to write and execute
r—	4	100	only permission to read
r-x	5	101	permission to read and execute
rw-	6	110	permission to read and write

Permission s	Octal Value	Binary Value	Description
rwX	7	111	permission to do all three, i.e. read, write and execute

24. head Command

The **head** command is used to display the content of a file. It displays the first 10 lines of a file.

Syntax:

head <file name>

Output:

```
javatpoint@javatpoint-Inspiron-3542:~$ head Demo.txt
1
2
3
4
5
6
7
8
9
10
```

25. tail Command

The **tail** command is similar to the head command. The difference between both commands is that it displays the last ten lines of the file content. It is useful for reading the error message.

Syntax:

1. tail <file name>

26.grep Command

The **grep** is the most powerful and used filter in a Linux system. The 'grep' stands for "**global regular expression print.**" It is useful for searching the content from a file. Generally, it is used with the pipe.

Syntax:

1. command | grep <searchWord>

Output:

```
javatpoint@javatpoint-Inspiron-3542:~$ cat marks.txt | grep 9
celena-90
```

```
$grep -i "and" doc1.txt
```

-i is case insensitive search

```
$grep -i -c "and" doc1.txt
```

Count of how many times “Anjali” appeared in doc

```
$grep -v “and” doc1.txt
```

Inverting pattern match

Options Description

- c : This prints only a count of the lines that match a pattern
- h : Display the matched lines, but do not display the filenames.
- i : Ignores, case for matching
- l : Displays list of a filenames only.
- n : Display the matched lines and their line numbers.
- v : This prints out all the lines that do not matches the pattern
- e **exp** : Specifies expression with this option. Can use multiple times.
- f **file** : Takes patterns from file, one per line.
- E : Treats pattern as an extended regular expression (ERE)
- w : Match whole word
- o : Print only the matched parts of a matching line,

27.sort Command

The **sort** command is used to sort files in alphabetical order.

Syntax:

1. sort <file name>

Output:

```
javatpoint@javatpoint-Inspiron-3542:~$ sort marks.txt
alen-70
alex-50
carry-85
celena-90
jon-75
justin-80
```

28.AWKThe **awk** command is used for **text processing** in Linux. Although, the sed command is also used for text processing, but it has some limitations, so the awk command becomes a handy option for text processing. It provides powerful control to the data.

The Awk is a powerful scripting language used for **text scripting**. It searches and replaces the texts and sorts, validates, and indexes the database.

It is one of the most widely used tools for the programmer, as they write the scaled-down effective program in the form of a statement to define the text patterns and designs.

It acts as a **filter in Linux**. It is also referred as **gawk (GNU awk)** In [Linux](#).

The Awk command is used as follows:

1. awk options 'selection_criteria {action }' input-file > output-file

The options can be:

- o **-f program files:** It reads the source code of the script written on the awk command
- o **-F fs:** It is used as the input field separator.

```
awk '{ print "Welcome to Awk command"}'
```

```
javatpoint@javatpoint-GB-BXBT-2807:~$ awk '{ print "Welcome to Awk command"}'
Welcome to Awk command
Welcome to Awk command
Welcome to Awk command
```

```
awk '{print $1,$2}' student.txt
```

```
javatpoint@javatpoint-GB-BXBT-2807:~$ awk '{print $1,$2}' student.txt
Sam CS
Daniel CS
John IT
Arya IT
Mike ECE
Helena ECE
```

28.Nroff command

The **nroff** command reads one or more files for printing on typewriter-like devices and line printers. If no file is specified or the - (minus sign) flag is specified as the last parameter, standard input is read by default. The *File* variable specifies files to be printed on a typewriter-like device by the **nroff** command.

nroff [[-e](#)] [[-h](#)] [[-i](#)] [[-q](#)] [[-z](#)] [[-o List](#)] [[-n Number](#)] [[-s Number](#)]
[[-r ANumber](#)] [[-u Number](#)] [[-T Name](#)] [[-man](#)] [[-me](#)] [[-mm](#)] [[-mptx](#)] [[-ms](#)]
[*File ...* | [-](#)]

-e Produces equally spaced words in adjusted lines, using the full resolution of a particular terminal.

- h** Uses output tabs during horizontal spacing to speed output and reduce the output character count. Tab settings are assumed to be every eight nominal character widths.
- i** Reads standard input after reading all specified files.
- man** Selects the [man](#) macro processing package.
- me** Selects the [me](#) macro processing package.
- mm** Selects the [mm](#) macro processing package.
- mptx** Selects the [mptx](#) macro processing package.
- ms** Selects the [ms](#) macro processing package.
- n *Number*** Assigns the specified number to the first printed page.
- o *List*** Prints only those pages specified by the *List* variable, which consists of a comma-separated list of page numbers and ranges, as follows:
 - A range of *Start-Stop* means print pages *Start* through *Stop*. For example, 9-15
 -
- u *Number*** Sets the bold factor (number of character overstrikes) for the third font position (bold) to the specified number, or to 0 if the *Number* variable is missing.
- z** Prints only messages generated by [.tm](#) (workstation message) requests.

- prints pages 9 through 15.
- An initial *-Stop* means print from the beginning to page *Stop*.
- A final *Start-* means print from page *Start* to the end.
- A combination of page numbers and ranges prints the specified pages. For example, -3, 6-8,10,12- prints the beginning through page 3, pages 6 through 8, page 10, and page 12 to the end.

Note: When the **-oList** flag is used in a pipeline (as with one or more of the **eqn** or **tbl** commands) you may receive a `broken pipe` message if the last

page in the document is not specified in the *List* parameter. This broken pipe message is not an indication of any problem and can be ignored.

- q** Calls the simultaneous input/output mode of the [.rd](#) request.

- r ANumber** Sets register *A* to the specified number. The value specified by the *A* variable must have a one-character ASCII name.

- s Number** Stops every specified number of pages (the default is 1). The **nroff** command halts every specified number of pages to allow paper loading or changing, then resumes upon receipt of a linefeed or new-line character. This flag does not work in pipelines (for example, with the **mm** command). When the **nroff** command halts between pages, an ASCII BEL character is sent to the workstation.

- T Name** Prepares the output for the specified printing device. Typewriter-like devices and line printers use the following *Name* variables for AIX international extended character sets, as well as English-language character sets, digits, and symbols:

29. troff - the troff processor of the groff text formatting system

troff [**-abcivzCERU**] [**-d**] [**-f**] [**-F**] [**-m**] [**-M**] [**-n**] [**-o**] [**-r**] [**-T**] [**-w**] [**-W**] [*files...*]

It is possible to have whitespace between a command line option and its parameter.

-a	Generate an ASCII approximation of the typeset output.
-b	Print a backtrace with each warning or error message. This backtrace should help track down the cause of the error. The line numbers given in the backtrace may not always be correct, for troff 's idea of line numbers gets confused by as or am requests.
-c	Disable color output (always disabled in compatibility mode).
-C	Enable compatibility mode.
-dcs	
-dname=s	
	Define <i>c</i> or <i>name</i> to be a string <i>s</i> ; <i>c</i> must be a one letter name.

-E	Inhibit all error messages of troff . Note that this doesn't affect messages output to standard error by macro packages using the tm or tm1 requests.
-ffam	Use <i>fam</i> as the default font family.
-Fdir	Search in directory (or directory path) <i>dir</i> for subdirectories devname (<i>name</i> is the name of the device) and there for the DESC file and font files. <i>dir</i> is scanned before all other font directories.
-i	Read the standard input after all the named input files have been processed.
-mname	Read in the file <i>name.tmac</i> . If it isn't found, try tmac.name instead. It will be first searched for in directories given with the -M command line option, then in directories given in the GROFF_TMAC_PATH environment variable, then in the current directory (only if in unsafe mode), the home directory, <i>/usr/lib/groff/site-tmac</i> , <i>/usr/share/groff/site-tmac</i> , and <i>/usr/share/groff/1.18.1.1/tmac</i> .
-Mdir	Search directory (or directory path) <i>dir</i> for macro files. This is scanned before all other macro directories.
-num	Number the first page <i>num</i> .
-olist	Output only pages in <i>list</i> , which is a comma-separated list of page ranges; <i>n</i> means print page <i>n</i> , <i>m-n</i> means print every page between <i>m</i> and <i>n</i> , <i>-n</i> means print every page up to <i>n</i> , <i>n-</i> means print every page from <i>n</i> . troff will exit after printing the last page in the list.
-rcn	
-rname=n	
	Set number register <i>c</i> or <i>name</i> to <i>n</i> ; <i>c</i> must be a one character name; <i>n</i> can be any troff numeric expression.
-R	Don't load troffrc and troffrc-end .
-Tname	Prepare output for device <i>name</i> , rather than the default ps .

-U	Unsafe mode. This will enable the following requests: open , opena , pso , sy , and pi . For security reasons, these potentially dangerous requests are disabled otherwise. It will also add the current directory to the macro search path.
-v	Print the version number.
-wname	Enable warning <i>name</i> . Available warnings are described in the section <u>WARNINGS</u> below. For example, to enable all warnings, use -w all . Multiple -w options are allowed.
-Wname	Inhibit warning <i>name</i> . Multiple -W options are allowed.
-z	Suppress formatted output.

30.groupadd

created a group with the name as provided. The group while creation gets a group ID and we can get to know everything about the group as its name, ID, and the users present in it in the file “/etc/group”.

```
groupadd group_name
```

Example:

```
groupadd Group1
```

```
[root@localhost ~]#
[root@localhost ~]# groupadd Group1
[root@localhost ~]#
[root@localhost ~]# tail -3 /etc/group
pesign:x:991:
screen:x:84:
Group1:x:1000:
[root@localhost ~]#
```

31.usermod

Below command is used to add a user to an existing group. The users which may be present in any primary or secondary group will exit the other groups and will become the part of this group.

```
usermod -G group_name username
```

```
usermod -G group1 John_Doe
```

```
[root@localhost ~]#  
[root@localhost ~]# usermod -G Group1 John_Wick  
[root@localhost ~]#  
[root@localhost ~]#  
[root@localhost ~]# tail -3 /etc/group  
Group1:x:1000:John_Doe,John_Wick  
John_Doe:x:1001:  
John_Wick:x:1002:  
[root@localhost ~]#
```