Name :- Amit.Y.Zope
Roll No. :- 223067
Gr No. :- 21810714
Division :- C
Batch :- C3
Subject :- Operating System

Assignment No 1

Aim: Study of Important Linux Commands

Objective: To study the frequently used Linux commands

Commands:

1) man :

man - an interface to the on-line reference manuals

Discription:

man is the system's manual pager. Each page argument given to man is normally the ` name of a program, utility or function. The manual page associated with each of these arguments is then found and displayed.

Example:

man ls :- Display the manual page for the item (program) ls.
man cat :- Display the manual page for the item (program) cat.
man touch :- Display the manual page for the item (program) touch.
man grep: Display the manual page for the item (program) grep.
man mkdir :- Display the manual page for the item (program) mkdir.
man cd : Display the manual page for the item (program) cd.

File Commands

2) ls :

ls - list directory contents

SYNOPSIS

ls [OPTION]... [FILE]...

DESCRIPTION

List information about the FILEs (the current directory by default). Sort entries alphabetically if none of -cftuvSUX nor --sort is specified.

Mandatory arguments to long options are mandatory for short options too.

-a, --all
do not ignore entries starting with .

-A, --almost-all
do not list implied . and ..
etc..

Exit status:

0 if OK,

1 if minor problems (e.g., cannot access subdirectory),

2 if serious trouble (e.g., cannot access command-line argument).

Examples:

1) ls :-

ls with no option list files and directories in bare format where we won't be able to view details like file types, size, modified date and time, permission and links etc.

2) ls -l

Here, **ls -l** (**-l** is character not one) shows file or directory, size, modified date and time, file or folder name and owner of file and its permission.

3) ls -a

List all files including hidden file starting with '**.**'. it will list hidden files.

4) ls -lh

With combination of **-lh** option, shows sizes in human readable format.

5) ls -F

Using **-F** option with **ls** command, will add the **'/'** Character at the end each directory.

6) ls -ltr

With combination of **-ltr** will shows latest modification file or directory date as last.

7) ls -i

With **-i** options list file / directory with inode number.

8) ls -n

To display **UID** and **GID** of files and directories. use option **-n** with ls command.

3) pwd :

pwd – show current working directory

SYNOPSIS

pwd [OPTION]...

DESCRIPTION

Print the full filename of the current working directory.

Examples :

1) pwd -L, (logical)

use PWD from environment, even if it contains symlinks

2) pwd -P, (physical)

avoid all symlinks

--version
output version information and exit

If no option is specified, -P is assumed.

4) mkdir :

NAME
mkdir - make directories

SYNOPSIS
mkdir [OPTION]... DIRECTORY...

DESCRIPTION
Create the DIRECTORY(ies), if they do not already exist.

Mandatory arguments to long options are mandatory for short options too.

Examples:

1) mkdir –m mode directoryname, --mode=MODE
set file mode (as in chmod), not a=rwx – umask.

2) mkdir –v directory name, --verbose
print a message for each created directory.

5) cat :

NAME
cat - concatenate files and print on the standard output

SYNOPSIS
cat [OPTION]... [FILE]...

DESCRIPTION
Concatenate FILE(s) to standard output.

With no FILE, or when FILE is -, read standard input.

Examples :

1)cat -E filename, --show-ends
display $ at end of each line.

2) cat –T filename, --show-tabs
display TAB characters as ^I.

3) cat –n filename, --number
number all output lines.

4) cat -e filename
equivalent to –vE.
display $ at end of each line.

5) cat -A filename , --show-all
equivalent to -vET

6) cat –b filename, --number-nonblank
number nonempty output lines, overrides –n.

6) cp:

NAME
cp - copy files and directories

SYNOPSIS
cp [OPTION]... [-T] SOURCE DEST
cp [OPTION]... SOURCE... DIRECTORY
cp [OPTION]... -t DIRECTORY SOURCE...

DESCRIPTION
Copy SOURCE to DEST, or multiple SOURCE(s) to DIRECTORY.

Mandatory arguments to long options are mandatory for short options too.

OPTIONS

Examples :

1) -f, --force
if an existing destination file cannot be opened, remove it and try again (this option is ignored when the -n option is also used)

2) -n, --no-clobber
do not overwrite an existing file (overrides a previous -i option)

3) -v, --verbose
explain what is being done

7)mv :

NAME

mv - move (rename) files

SYNOPSIS

mv [OPTION]... [-T] SOURCE DEST
mv [OPTION]... SOURCE... DIRECTORY
mv [OPTION]... -t DIRECTORY SOURCE...

DESCRIPTION

Rename SOURCE to DEST, or move SOURCE(s) to DIRECTORY.

Mandatory arguments to long options are mandatory for short options too.

OPTIONS

Examples:

1) -f, --force
do not prompt before overwriting.

2) -v, --verbose
explain what is being done.

3) -u, --update
move only when the SOURCE file is newer than the destination file or when the destination file is missing.

8)rm :

NAME

rm - remove files or directories

SYNOPSIS

rm [OPTION]... [FILE]...

DESCRIPTION

This manual page documents the GNU version of rm. rm removes each specified file. By default, it does not remove directories.

If the -I or --interactive=once option is given, and there are more than three files or the -r, -R, or --recursive are given, then rm prompts the user for whether to proceed with the entire operation. If the response is not affirmative, the entire command is aborted.

Otherwise, if a file is unwritable, standard input is a terminal, and the -f or --force option is not given, or the -i or --interactive=always option is given, rm prompts the user for whether to remove the file. If the response is not affirmative, the file is skipped.

OPTIONS
1) -f, --force
ignore nonexistent files and arguments, never prompt.

2) -i prompt before every removal.

3) -v, --verbose
explain what is being done.

9)ln :

NAME
ln - make links between files

SYNOPSIS
ln [OPTION]... [-T] TARGET LINK_NAME (1st form)
ln [OPTION]... TARGET (2nd form)
ln [OPTION]... TARGET... DIRECTORY (3rd form)
ln [OPTION]... -t DIRECTORY TARGET... (4th form)

DESCRIPTION
In the 1st form, create a link to TARGET with the name LINK_NAME. In the 2nd form, create a link to TARGET in the current directory. In the 3rd and 4th forms, create links to each TARGET in DIRECTORY. Create hard links by default, symbolic links with --symbolic. By default, each destination (name of new link) should not already exist. When creating hard links, each TARGET must exist. Symbolic links can hold arbitrary text; if later resolved, a relative link is interpreted in relation to its parent directory.

Mandatory arguments to long options are mandatory for short options too.

OPTIONS:

1) -s, --symbolic
make symbolic links instead of hard links.

2) -t, --target-directory=DIRECTORY
specify the DIRECTORY in which to create the links.

3) -v, --verbose
print name of each linked file.

4) -i, --interactive

prompt whether to remove destinations.

10) touch :

NAME
touch - change file timestamps

SYNOPSIS
touch [OPTION]... FILE...

DESCRIPTION
Update the access and modification times of each FILE to the current time.

A FILE argument that does not exist is created empty, unless -c or -h is supplied.

A FILE argument string of - is handled specially and causes touch to change the times of the file associated with standard output.

Mandatory arguments to long options are mandatory for short options too.

OPTIONS

Examples :

1) touch -a file_name ,
change only the access time.

2) touch -m file_name ,
change only the modification time.

3)touch -c file_name,
-c, --no-create do not create any files.

11) more :

NAME
more - file perusal filter for crt viewing

SYNOPSIS
more [options] file...

DESCRIPTION

more is a filter for paging through text one screenful at a time. This version is especially primitive. Users should realize that less(1) provides more(1) emulation plus extensive enhancements.

OPTIONS

Options are also taken from the environment variable MORE (make sure to precede them with a dash (-)) but command-line options will override those.

1) more –d file_name,

Prompt with "[Press space to continue, 'q' to quit.]", and display "[Press 'h' for instructions.]" instead of ringing the bell when an illegal key is pressed.

2) more +num file_name,

Start displaying each file at line number.

3) more -s file_name ,

Squeeze multiple blank lines into one.

12) head :

NAME

head - output the first part of files

SYNOPSIS

head [OPTION]... [FILE]...

DESCRIPTION

Print the first 10 lines of each FILE to standard output. With more than one FILE, precede each with a header giving the file name.

With no FILE, or when FILE is -, read standard input.

Mandatory arguments to long options are mandatory for short options too.

OPTIONS :

Examples :

1) head -n num file_name ,

print the first NUM lines instead of the first 10; with the leading '-', print all but the last NUM lines of each file .

2) head -c num file_name ,

print the first NUM bytes of each file; with the leading '-',

print all but the last NUM bytes of each file.

3)head –q file1_name file2_name.
never print headers giving file names.

13) tail :

NAME
tail - output the last part of files

SYNOPSIS
tail [OPTION]... [FILE]...

DESCRIPTION
Print the last 10 lines of each FILE to standard output. With more than one FILE, precede each with a header giving the file name.

With no FILE, or when FILE is -, read standard input.

Mandatory arguments to long options are mandatory for short options too.

OPTIONS :

1) -c, --bytes=[+]NUM
output the last NUM bytes; or use -c +NUM to output starting with byte NUM of each file.

2) -n, --lines=[+]NUM
output the last NUM lines, instead of the last 10; or use -n +NUM to output starting with line NUM.

3) –f , --follow[={name|descriptor}]
output appended data as the file grows;

an absent option argument means 'descriptor'

Process Commands

14) ps:

NAME
ps - report a snapshot of the current processes.

SYNOPSIS

ps [options]

DESCRIPTION

ps displays information about a selection of the active processes. If you want a repetitive update of the selection and the displayed information, use top(1) instead.

This version of ps accepts several kinds of options:

1 UNIX options, which may be grouped and must be preceded by a dash.
2 BSD options, which may be grouped and must not be used with a dash.
3 GNU long options, which are preceded by two dashes.

OPTIONS

1) ps -a , Select all processes except both session leaders (see getsid(2)) and processes not associated with a terminal.

15) top :

NAME

top - display Linux processes

SYNOPSIS

top -hv|-bcHiOSs -d secs -n max -u|U user -p pid -o fld -w [cols]

The traditional switches `-' and whitespace are optional.

DESCRIPTION

The top program provides a dynamic real-time view of a running system. It can display system summary information as well as a list of processes or threads currently being managed by the Linux kernel. The types of system summary information shown and the types, order and size of information displayed for processes are all user configurable and that configuration can be made persistent across restarts.

The program provides a limited interactive interface for process manipulation as well as a much more extensive interface for personal configuration -- encompassing every aspect of its operation. And while top is referred to throughout this document, you are free to name the program anything you wish. That new name, possibly an alias, will then be reflected on top's display and used when reading and writing a configuration file.

OPTIONS:

1) -n :Number-of-iterations limit as: -n number
Specifies the maximum number of iterations, or frames, top should produce before ending.

2) -u | -U :User-filter-mode as: -u | -U number or name
Display only processes with a user id or user name matching that given. The `-u' option matches on effective user whereas the `-U' option matches on any user (real, effective, saved, or filesystem).

Prepending an exclamation point (`!') to the user id or name instructs top to display only processes with users not matching the one provided.

The `p', `u' and `U' command-line options are mutually exclusive.

16) kill pid:

NAME

kill - send a signal to a process

SYNOPSIS
kill [options] <pid> [...]

DESCRIPTION
The default signal for kill is TERM. Use -l or -L to list available signals. Particularly useful signals include HUP, INT, KILL, STOP, CONT, and 0. Alternate signals may be specified in three ways: -9, -SIGKILL or -KILL. Negative PID values may be used to choose whole process groups; see the PGID column in ps command output. A PID of -1 is special; it indicates all processes except the kill process itself and init.

OPTIONS
<pid> [...]
Send signal to every <pid> listed.

-<signal>
-s <signal>
--signal <signal>
Specify the signal to be sent. The signal can be specified by using name or number. The behavior of signals is explained in signal(7) manual page.

1) -l, --list [signal]
List signal names. This option has optional argument, which

will convert signal number to signal name, or other way round.

2) -L, --table
List signal names in a nice table.

Examples :

1) kill -9 -1
Kill all processes you can kill.

2) kill -l 11
Translate number 11 into a signal name.

3) kill -L
List the available signal choices in a nice table.

17) killall name :

NAME

killall - kill processes by name

SYNOPSIS

killall [-Z, --context pattern] [-e, --exact] [-g, --process-group] [-i, --interactive] [-n, --ns PID] [-o, --older-than TIME] [-q, --quiet] [-r, --regexp] [-s, --signal
SIGNAL, -SIGNAL] [-u, --user user] [-v, --verbose] [-w, --wait] [-y, --younger-than TIME] [-I, --ignore-case] [-V, --version] [--] name ...
killall -l
killall -V, --version

DESCRIPTION

killall sends a signal to all processes running any of the specified commands. If no signal name is specified, SIGTERM is sent.

Signals can be specified either by name (e.g. -HUP or -SIGHUP) or by number (e.g. -1) or by option -s.

If the command name is not regular expression (option -r) and contains a slash (/), processes executing that particular file will be selected for killing, independent of
their name.

killall returns a zero return code if at least one process has been killed for each listed command, or no commands were listed and at least one process matched the -u and
-Z search criteria. killall returns non-zero otherwise.

A killall process never kills itself (but may kill other killall processes).

OPTIONS

1) -I, --ignore-case

Do case insensitive process name match.

2)-i, --interactive
Interactively ask for confirmation before killing

3) -l, --list
List all known signal names.

18)pkill pattern :

NAME
pgrep, pkill - look up or signal processes based on name and other attributes

SYNOPSIS
pgrep [options] pattern
pkill [options] pattern

DESCRIPTION
pgrep looks through the currently running processes and lists the process IDs which match the selection criteria to stdout. All the criteria have to match. For example,

$ pgrep -u root sshd

will only list the processes called sshd AND owned by root. On the other hand,

$ pgrep -u root,daemon

will list the processes owned by root OR daemon.

pkill will send the specified signal (by default SIGTERM) to each process instead of listing them on stdout.

File Permission Commands

19) chmod permissions filename :

NAME
chmod - change file mode bits

SYNOPSIS
chmod [OPTION]... MODE[,MODE]... FILE...
chmod [OPTION]... OCTAL-MODE FILE...
chmod [OPTION]... --reference=RFILE FILE...

DESCRIPTION

Change the permission of file to octal,which can be found separately for user,group,world by adding,

- 4-read(r)
- 2-write(w)
- 1-execute(x)

This manual page documents the GNU version of chmod. chmod changes the file mode bits of each given file according to mode, which can be either a symbolic representation of changes to make, or an octal number representing the bit pattern for the new mode bits.

OPTIONS

Change the mode of each FILE to MODE. With --reference, change the mode of each FILE to that of RFILE.

Examples :

1) -c, --changes
   like verbose but report only when a change is made

2) -f, --silent, --quiet
   suppress most error messages

3) -v, --verbose
   output a diagnostic for every file processed

--no-preserve-root
   do not treat '/' specially (the default)

--preserve-root
   fail to operate recursively on '/'

--reference=RFILE
   use RFILE's mode instead of MODE values

4) -R, --recursive
   change files and directories recursively

Searching Commands

20) grep pattern filename :

NAME

grep, egrep, fgrep, rgrep - print lines matching a pattern

SYNOPSIS

grep [OPTIONS] PATTERN [FILE...]
grep [OPTIONS] -e PATTERN ... [FILE...]
grep [OPTIONS] -f FILE ... [FILE...]

DESCRIPTION

grep searches for PATTERN in each FILE. A FILE of "-" stands for standard input. If no FILE is given, recursive searches examine the working directory, and nonrecursive searches read standard input. By default, grep prints the matching lines.

In addition, the variant programs egrep, fgrep and rgrep are the same as grep -E, grep -F, and grep -r, respectively. These variants are deprecated, but are provided for backward compatibility.

OPTIONS

1) -E, --extended-regexp
Interpret PATTERN as an extended regular expression .

2) -G, --basic-regexp
Interpret PATTERN as a basic regular expression (BRE, see below). This is the default.

3) -n, --line-number
Prefix each line of output with the 1-based line number within its input file.

4) -v, --invert-match
Invert the sense of matching, to select non-matching lines.

21) locate:

NAME

locate - find files by name

SYNOPSIS

locate [OPTION]... PATTERN...

DESCRIPTION

locate reads one or more databases prepared by updatedb(8) and writes file names matching at least one of the PATTERNs to standard output, one per line.

If --regex is not specified, PATTERNs can contain globbing characters. If any PATTERN contains no globbing characters, locate behaves as if the pattern were *PATTERN*.

By default, locate does not check whether files found in database still exist (but it does require all parent directories to exist if the database was built with --require-visibility no). locate can never report files created after the most recent update of the relevant database.

EXIT STATUS

locate exits with status 0 if any match was found or if locate was invoked with one of the --limit 0, --help, --statistics or --version options. If no match was found or a fatal error was encountered, locate exits with status 1.

Errors encountered while reading a database are not fatal, search continues in other specified databases, if any.

OPTIONS

1) -A, --all
   Print only entries that match all PATTERNs instead of requiring only one of them to match.
2) -c, --count
   Instead of writing file names on standard output, write the number of matching entries only.
3) -p, --ignore-spaces
   Ignore punctuation and spaces when matching patterns.

22) command | grep pattern :

Example :

1) ps –a | grep firefox

The above command will display all current working processes and will match the pattern give to grep command her i.e firefox.

23) find :

NAME

find - search for files in a directory hierarchy

SYNOPSIS

find [-H] [-L] [-P] [-D debugopts] [-Olevel] [starting-point...] [expression]

DESCRIPTION

This manual page documents the GNU version of find. GNU find searches the directory tree rooted at each given starting-point by evaluating the given expression from left

to right, according to the rules of precedence (see section OPERATORS), until the outcome is known (the left hand side is false for and operations, true for or), at which

point find moves on to the next file name. If no starting-point is specified, `.' is assumed.

If you are using find in an environment where security is important (for example if you are using it to search directories that are writable by other users), you should

read the `Security Considerations' chapter of the findutils documentation, which is called Finding Files and comes with findutils. That document also includes a lot more

detail and discussion than this manual page, so you may find it a more useful source of information.

OPTIONS:

1) -name pattern ,

Base of file name (the path with the leading directories removed) matches shell pattern pattern. Because the leading directories are removed, the file names con-

sidered for a match with -name will never include a slash, so `-name a/b' will never match anything (you probably need to use -path instead). A warning is issued

if you try to do this, unless the environment variable POSIXLY_CORRECT is set. The metacharacters (`*', `?', and `[]') match a `.' at the start of the base name

(this is a change in findutils-4.2.2; see section STANDARDS CONFORMANCE below). To ignore a directory and the files under it, use -prune; see an example in the

description of -path. Braces are not recognised as being special, despite the fact that some shells including Bash imbue braces with a special meaning in shell

patterns. The filename matching is performed with the use of the fnmatch(3) library function. Don't forget to enclose the pattern in quotes in order to protect

it from expansion by the shell.

2) -empty ,

File is empty and is either a regular file or a directory.

3) -perm mode,

File's permission bits are exactly mode (octal or symbolic). Since an exact match is required, if you want to use this form for symbolic modes, you may have to specify a rather complex mode string. For example `-perm g=w' will only match files which have mode 0020 (that is, ones for which group write permission is the only permission set). It is more likely that you will want to use the `/' or `-' forms, for example `-perm -g=w', which matches any file with group write permission. See the EXAMPLES section for some illustrative examples.

24) pgrep:

NAME

pgrep, pkill - look up or signal processes based on name and other attributes

SYNOPSIS

pgrep [options] pattern
pkill [options] pattern

DESCRIPTION

pgrep looks through the currently running processes and lists the process IDs which match the selection criteria to stdout. All the criteria have to match. For example,

$ pgrep -u root sshd

will only list the processes called sshd AND owned by root. On the other hand,

$ pgrep -u root,daemon

will list the processes owned by root OR daemon.

pkill will send the specified signal (by default SIGTERM) to each process instead of listing them on stdout.

OPTIONS:

1) -i, --ignore-case

Match processes case-insensitively.

2) -l, --list-name

List the process name as well as the process ID. (pgrep only.)

3) -c, --count

Suppress normal output; instead print a count of matching processes. When count does not match anything, e.g. returns zero, the command will return non-zero value.

System Info Commands

25) date :

NAME

date - print or set the system date and time

SYNOPSIS

date [OPTION]... [+FORMAT]

date [-u|--utc|--universal] [MMDDhhmm[[CC]YY][.ss]]

DESCRIPTION

Display the current time in the given FORMAT, or set the system date.

Mandatory arguments to long options are mandatory for short options too.

OPTIONS:

1) -R, --rfc-email

output date and time in RFC 5322 format. Example: Mon, 14 Aug 2006 02:34:56 -0600

26)cal :

NAME

cal, ncal — displays a calendar and the date of Easter

SYNOPSIS

cal [-31jy] [-A number] [-B number] [-d yyyy-mm] [[month] year]

cal [-31j] [-A number] [-B number] [-d yyyy-mm] -m month [year]

ncal [-C] [-31jy] [-A number] [-B number] [-d yyyy-mm] [[month] year]

ncal [-C] [-31j] [-A number] [-B number] [-d yyyy-mm] -m month [year]

ncal [-31bhjJpwySM] [-A number] [-B number] [-H yyyy-mm-dd] [-d yyyy-mm] [-s country_code] [[month] year]
ncal [-31bhJeoSM] [-A number] [-B number] [-d yyyy-mm] [year]

DESCRIPTION

The cal utility displays a simple calendar in traditional format and ncal offers an alternative layout, more options and the date of Easter. The new format is a little cramped but it makes a year fit on a 25x80 terminal. If arguments are not specified, the current month is displayed.

OPTIONS :

1) -h Turns off highlighting of today.