Linux
System Administration

[File handling Commands]
[ST.1]

Prepared by: Amitabh Srivastava

Top-level directories

Directories	Description
/bin	binary or executable programs.
/etc	system configuration files.
/home	home directory. It is the default current directory.
/opt	optional or third-party software.
/tmp	temporary space, typically cleared on reboot.
/usr	User related programs.
/var	log files.

touch command

touch command is used to create empty files and also to modify the timestamp of a file.

Options-

touch file1.txt
It will create an empty file called 'file1.txt'.

touch file1.txt file2.txt file3.txt
It will create three empty files in current directory called 'file1.txt', 'file2.txt' and 'file3.txt'.

touch -- -file.txt
It will create an empty file as **'-file.txt'** whose first character is hyphen (-). in current directory.

touch –a file1.txt
It will change the access time of file called 'file1.txt' and set it as current time.
(use **'stat <filename>'** *command to check the access time of file)*

touch –m file1.txt
It will change only the modification date & time of file 'file1.txt' to current date and time.

touch –t 201210280605 file1.txt (format used- YYYYMMDDHHMM)
It will change the access & modification date & time of file 'file1.txt' to 2012-10-28 06:05.

touch –r oldfile newfile
It will change the modification date & time of newfile as per oldfile.

Prepared by: Amitabh Srivastava

rm command

rm command is used to remove files, directories and links from a specific directory.
By default, it does not remove directories. It works silently so we should be careful while deleting any file/directory using ‘rm’ command.

Example-
rm file1.txt -> this command will delete file called ‘file1.txt’.

Options-
rm file1 file2 file3
This will delete all three mentioned files.

rm –i <filename(s)>
It will interactively confirm the deletion of file from the user.

rm –f <filename>
If the file is write-protected, it will not ask for the confirmation and forcefully deletes the file.

rm –r <filename> or rm –R <filename>
It performs a tree-walk and delete all the files and sub-directories recursively from the current directory.

rm -- <-filename>
It will delete the file whose first character is hyphen (-).

Prepared by: Amitabh Srivastava

rmdir command

rmdir command is used to remove empty directories.
By default, it does not remove directories. It works silently so we should be careful while deleting any file/directory using ‘rm’ command.

Options-

rmdir dir1
This will delete single directory ‘dir1’ only if it is empty.

rmdir dir1 dir2 dir3
This will delete directories ‘dir1’, ‘dir2’ and ‘dir3’ only if they are empty.

rmdir –v dir1 dir2 dir3 (verbose mode)
This will delete directories ‘dir1’, ‘dir2’ and ‘dir3’ only if they are empty and also displays the message of removal.

rmdir –p dir1/d1/d2/d3
This will delete the entire given directory structured only if it is empty(no files/subdirectories should be inside).

Merging Files

- **cat** command is used to merge two or more files in Linux.

Examples-

cat file1 file2

It will display the contents of file1 and file2 but not save them in any other file.

cat file1 file2 > file3

It will send/redirect the output of cat command to file ‘file3’.

cat file1 file2 >> file3

It will send/redirect the output of cat command to file ‘file3’ and appended the output to file3.

Merging Commands

- We can also execute two or multiple commands in a single line.
- To do so separate the commands with semicolon (;) symbol.

Some Useful Commands

- **mv file1 file2**

  It will change the name (rename) of ‘file1’ to ‘file2’.

- **mv file1 dir1/file2**

  It will move the file ‘file1’ into the directory ‘dir1’ and set the filename as ‘file2’.

- **pwd**

  It will display the absolute path of current directory. (**P**resent **W**orking **D**irectory)

- **whoami**

  It will print the logged in username.

- **man <command>**

  It will display the manual/help of the given command. **‘spacebar’** to move screen forward, **‘b’** to move screen backword and **‘q’** to quit from manual.

- **file <filename>**

  It will display the file type information along with filename.

- **file –b <filename>**

  Will display the file type information in brief mode. Removes the filename from output.

Prepared by: Amitabh Srivastava

find command

find command is used to find files and directories in a Linux file system.

Options-

-name <filename>
Search files with specific filename.

-inum <inode>
Search file having specified i-node number.

-empty
Search only empty files and directories.

-perm <permission_in_number>
Search files and directories having specified permissions.

-user <username>
Search files owned by specified username.

-type f/d <filename>
Search files or directories only having specific name.

Prepared by: Amitabh Srivastava

locate command

locate is an utility command used to search files and directories quickly. It is more convenient and effective than ‘find’ command.

locate command doesn’t search the entire file system, but it looks through a regularly updated file database called **‘mlocate’** in the system. To update the locate database, **‘updated’** command is used.

If any file created after updating the locate database, it will not include in the search result.

Example-
locate file1 -> this command will search and display all files containing the ‘file1’ pattern in the filename.

Options-
locate file1 | less
This will be convenient way to see the output if there are large number of files in the output. **‘Spacebar’** is used to move forward, **‘b’** key to move backward and **‘q’** key to quit.

locate –c file1 or locate --count file1
Display only the **count** of files containing ‘file1’ pattern.

locate file1 –n 10 or locate file1 –l 10 or locate file1 –limit 10
Limit the number of search results.

locate command...cont.

Options-

locate –i <filename> or locate –ignore-case <filename>
Ignores the case of filename.

locate -0 <filename> or locate --null <filename>
Displays the result containing the specified pattern but omits newline character.

locate –h or locate --help
Displays the usage of locate command with its available options.

more command

- **more** command is used to display the contents of text files on command window (terminal window).

- It facilitates to display the contents one screen/page at a time in case of large files along with the display ratio (in %age).

Options-

more <filename>
It will display the contents one screen/page at a time.

more –d <filename>
It will display the help to the user to regarding navigation controls at the end of every screen.
"[Press space to continue, 'q' to quit.]"

more -5 <filename>
It will display 5 number of lines per screen of the given filename.

more + 10 <filename>
It will start displaying the contents after leaving 10 lines of the given filename.

more –p <filename>
It will first clear the screen and then start displaying the contents of given filename.

Prepared by: Amitabh Srivastava

less command

- **less** is used to display the contents of a text file one screen/page at a time.
- It is fast because for large files, it does not access the entire file at a time but access the contents page-by-page.
- G- Moves to the end of the file
- g- Moves to the beginning of the file.
- **/pattern** is used to search the pattern in a file.
- PageUp, PageDown, Up Arrow and Down Arrow keys are working as usual.
- To quit from 'less' utility, we must need to press the **'q'** key.

Options-

less <filename>
It will display the contents one screen/page at a time.

less –E <filename>
It will automatically exit when reaching at the end of the file.

less -F <filename>
Automatically exits if the complete file can be displayed on the first screen.

less -N <filename>
It will display the content of the given filename along with the line numbers.

history command

- **history** command is used to view previously executed commands.
- Supported by BASH and Korn shells and not available under Bourne Shell.
- Stored under **.bash_history** file.

history
It will display the list of previously used commands.

history 5
It will display the last/recent 5 commands used.

!!
It will execute the last command used.

!<command_number>
It will display and excutes the the command from history at given command_number.
Example- !1201

!<command>
It will display and execute the most recent use of given command.

^<last_command_executed>^<new_command>
It will change the last executed command with new command.

history –d <command_number>
It will delete the command from history at given command number position.

Prepared by: Amitabh Srivastava

head command

- **head** command is used to display the top N number of lines from the given input.
- By default, it will print first 10 lines from the given file/input.

Options-

head myfile
It will display the first 10 lines of file called ‘myfile’.

head –n 7 <filename> or head –7 <filename>
It will display the first 7 lines from the given filename.

head –c <num_of_chars> <filename>
It will display mentioned number of characters/bytes from the beginning of the given filename.
Example- head –c 8 myfile.txt

head –q <file 1> <file 2>.....<file N>
This option is used with multiple files. If used, it would not show the filename. For long files, user may confuse that where first file contents are finished and contents of next file starts.

head –v <filename>
It will always displays the filename before the contents of the file.

tail command

- **tail** command is used to display the last N number of lines from the given input.
- By default, it will print last 10 lines from the given file/input.

Options-

tail myfile
It will display the last 10 lines of file called ‘myfile’.

tail –n 7 <filename> or tail –7 <filename>
It will display the last 7 lines from the given filename.

Extra feature not in ‘head’ command-

tail +5 <filename>
It will display the contents starting from line number 5 up to end of the file.

tail –c <num_of_chars> <filename>
It will display mentioned number of characters/bytes from the end of the given filename.
Example- tail –c 8 myfile.txt

tail –q <file 1> <file 2>.....<file N>
This option is used with multiple files. If used, it would not show the filename. For long files, user may confuse that where first file contents are finished and contents of next file starts.

tail –v <filename>
It will always displays the filename before the contents of the file.

Prepared by: Amitabh Srivastava

head & tail Applications

Q.1

- Print lines between 3 and 7 from file ‘file1’. Means the actual lines need to be display are 4, 5 and 6.
- To achieve this we can use the pipe symbol to redirect the output of one command to another command.
- **Solution-**

  tail +4 file1 | head -3

  Or

  head -6 file1 | tail -3

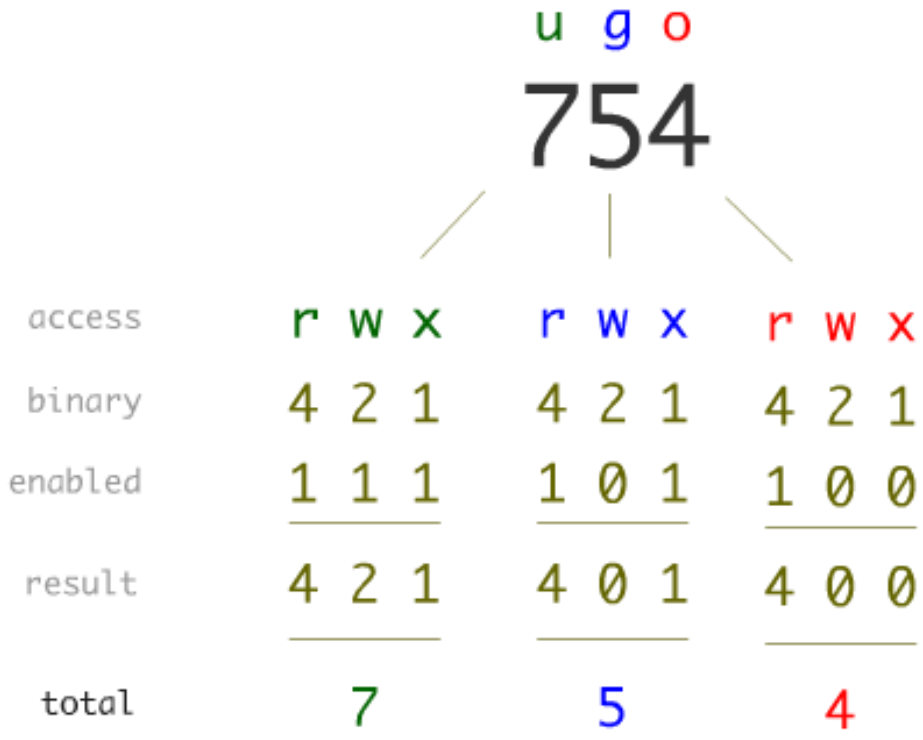
Q.2

- Print the most recently modified files from the current directory.
- **Solution-**

  ls –t | head -3

chmod command

- **chmod** (**C**hange **M**ode) command is used to change the access mode of a file.
- It actually allows to set the permissions on a file/directory like who can read the file, who can edit the file contents and who can execute the file (if file is executable).
- Symbols **"r"**, **"w"** and **"x"** are used to represent the read, write and execute permissions, respectively.
- These letters are collectively form a tri-group.
- There are 3 tri-groups or triads as-
  - "User/Owner" **(u)**
  - "Group" **(g)**
  - "Other" **(o)**
- All the three permissions (r, w and x) can be set on any triad.
- Numbers used to represent these permissions are-
  Read **(4)** Write **(2)** Execute **(1)**



Example:
To give permissions r, w and x to the owner and r and x to the group
And r to others on file **file1.txt**-

$chmod 754 file1.txt OR $chmod +u+rwx,g+rx,o+r

Prepared by: Amitabh Srivastava

chown command

- **chown** command is used to change the owner or group of a file/directory.

Options-

chown <username> <filename/directory>
It will change the current user/owner of the file/directory to the given username.

chown <username>:<groupname> <filename/directory>
It will change the current owner and group of the file/directory to the given username and groupname.

chown :<groupname> <filename/directory>
It will change only the current group of the file/directory to the given groupname.

chown -c <username>:<groupname> <filename/directory>
This option **(-c)** will also display a confirmation message after changing the ownership.

chown -v <username>:<groupname> <filename/directory>
The verbose option **(-v)** will also display a message on the screen after changing the ownership.

chgrp command

- **chgrp** command is used to change the group ownership of a file/directory.

Options-

chgrp <groupname> <filename>
It will change the current group of the file to the given groupname. The groupname must be exist and can be checked within **/etc/group** file.

chgrp <groupname> <file...1> <file...2> <file...3> <file...N>
It will change the current group of all the files to the given groupname.

chgrp <groupname> <directory>
It will change the current group of the directory to the given groupname but does not change the group of files and directories inside it.

chgrp -R <groupname> <directory>
It will change the current group of the directory to the given groupname and also **recursively** change the group ownership of all folders and files inside it.

chgrp -c <groupname> <file/directory>
It will display a confirmation message after changing the group ownership.

Prepared by: Amitabh Srivastava

df command

- **D**isk **F**ree utility provides valuable information on disk space utilization.
- Displays information about file system's disk space usage on the mounted file system.

Options-

df
It displays information about all the mounted file systems like total size, used space, usage percentage, and the mount point.

df <filename>
It will display the mount information of the given filename.

df -h
It will display the disk space usage in human readable form (in KB/MB/GB).

df --total
It will display the grand of total disk space usage at the end.

df --help
It will display the help on 'df' command.

du command

- **D**isk **U**sage utility provides information of a directory or file space usage.
- Displays information about the storage consumption of files and directories.

Options-

du
It displays the disk usage information of current directory.

du <filename>
It will display the storage information of the given filename.

du -h
It will display the disk space usage in human readable form (in KB/MB/GB).

du --time <filename>
It will display the disk space usage along with the last modification time of the given filename.

du -c <dirname> or du --total <dirname>
It will display the disk space usage along with the total disk usage of the given directory.

du --help
It will display the help on ‘du’ command.

mount/umount command

- The files and directories under Linux filesystem are arranged in a root directory called **/**.
- **mount** command is used to attach the filesystem of an external device to the filesystem of an Operating System.
- It mounts (attaches) the external storage devices like hard disks, USB drives, etc. to the OS filesystem.

Examples-

mount –t type <device> <directory>
It will mount the filesystem of device at the specified directory. If the directory is not given, it will look for the mount-point in **/etc/fstab** file. The **/etc/fstab** file contains information about which device should be mounted where.

mount
It will display all currently mounted filesystems on a system.

mount –V or mount --version
It will display the version information of mount utility.

Umount <devicefile>
It will unmount (dettach) the filesystem of device.

Prepared by: Amitabh Srivastava

Linux
System Administration

[ST.2]

Prepared by: Amitabh Srivastava

User Management

- A user is an entity in Linux OS that can manipulate files and perform several other operations.
- Each user has a unique id.
- ID of ‘root’ user (Superuser) is always 0.
- System user IDs starts from 1 to 999.
- Local user IDs starts from 1000 onwards.
- **User Management Commands (‘root’ user)-**
  - passwd
  - useradd
  - usermod
  - groupadd
  - groupdel
  - userdel
  - id

passwd command

- It is used to change/modify the password of an existing user account through command line.
- The ‘root’ user can change the password of any user account but a normal user can change the password only for his/her account.

Options-

passwd
If given without any options or username, it will ask to change the account password of currently logged user.

passwd <username>
It will change the password of the specified username.

passwd –l <username>
It will lock the password of specified user so that user cannot login to his/her account.

passwd –u <username>
It will unlock the password of specified user so that user can login to his/her account.

passwd –d <username>
It will delete the password of the specified user. Means, user can login without password.

passwd –e <username>
It will enforce the user to change the account password on next login attempt.

passwd command...(cntd.)

- All the password information is stored in a file **/etc/shadow**.
- User password expiry information can be viewed using command '**chage –l <username>**'

Options-

passwd –n <no. of days> <username>
It will change the minimum number of days between password change. Means, if the no. of days is set to 2, and user changes its own password one time then he/she cannot change its own password until 2 days have passed. By default, this value is set to 'zero' (0) means, user can change its own password at any time.

passwd –S <username>
It will show the password status of the user in 7 fields.
First field- User's login name
Second field- Indicates if user has a locked password **(L)**, has no Password **(NP)**, or has a usable password **(P)**.
Third field- Date of the last password change.
Next four fields are the minimum age, maximum age (enforce to change the password), warning period, and inactivity period for the password. These ages are expressed in days.

useradd command

- It is used to add/create new user accounts through command line.

Options-

useradd –c <comment> <username>
It will add a new user and add the specified comment in /etc/passwd file.

useradd –d <homedirectorypath> <username>
It will add a new user and set the home directory as specified.

useradd –m <username>
It will add a new user and creates the home directory as specified.

useradd –u <userid> <username>
It will create new user with specified user id.

useradd –e <yyyy-mm-dd> <username>
It will create new user and set the account expiry date as mentioned.

useradd --help
It will display the help section of useradd command.

usermod command

- It is used to change/modify the properties of an existing user through command line.
- After creating a user, if you want to change its home directory or group information, etc. then you can use ‘usermod’ command.

Options-

usermod –c “Local user” <username>
It will add a comment “Local user” for the given username. Comments are stored in 5th field of /etc/passwd file.

usermod –d /home/ram ram001
It will change the default home directory of user ram001 to /home/ram. Home directories are stored in 6th field of /etc/passwd file.

usermod –e 2023-12-25 <username>
It will set the expiry date for the given username. Expiry dates are stored inside /etc/shadow file.
Check expiry information using command- **‘chage –l <username>’**

usermod –g <newgroup> <username>
It will change the group for the given username to <newgroup>. Group-id is stored in 4th field of /etc/passwd file. All group names are stored in /etc/group file.

usermod –aG sudo <username>
It will add the given username to the ‘sudo’ group. **‘id’** command is used to check the group information.

groupadd command

- It is used to add a new group.
- Using groups, we can group together a number of users, and set privileges and permissions for the entire group.

Options-

groupadd <grpname>
It will create the specified group.

groupadd –g <grpid> <grpname>
It will create the specified group with mentioned group id.

groupadd –r <grpname>
It will create the specified system group. The IDs of system groups are chosen from a range 1 to 999 defined for system groups in the configuration file.

groupdel command

- It is used to delete an existing group.

Options-

groupdel <grpname>
It will delete the specified group.

groupdel –f <grpname>
It will delete the specified group even if it is the primary group of a user.

groupdel –r <grpname> or groupdel --system <grpname>
It will delete a specified system group. The IDs of system groups are chosen from a range 1 to 999 defined for system groups in the configuration file.

groupdel –h or groupdel --help
It will display the help of groupdel command with all its options.

userdel command

- It is used to delete an existing user account.

Options-

userdel <username>
It will delete the specified user account.

userdel –f <username>
It will delete the specified user even if the user still logged in.

userdel –r <username>
It will also delete the user's home directory.

userdel –h or userdel --help
It will display the help of userdel command with all options.

Vi or Vim Editor

- Vi editor is elaborated as **vi**sual editor.
- Available to every Unix system.
- Available to all Linux distros.
- Improved version of Vi is known as V**im**.
- Used to create simple text files in Linux using command line interface.
- Two vi modes- *Command* mode and *Insert* mode.
- To switch from command mode to insert mode press "**i**" key.
- To switch from insert mode to command mode press '**Esc**' key.
- By default, Vi editor starts in command mode.

vi Editor Quick Reference Chart

Execute in Command Mode

Save Files and Exit

:w	write buffer to disk
:w file	write buffer to file
:w! file	write Absolutely
:wq	write buffer and quit
:q	quit
:q!	quit and discard buffer
:c!	reedit and discard changes

Move and Insert Text

:3,8d	delete lines 3-8
:4,9m 12	move lines 4-9 to 12
:2,9m 13	copy lines 2-5 to 13
:59w file	write lines 5-9 to file

Cancel Edit Function

u	undo last change
*	do last change again

Search Functions

/exp	go forward to expression
?exp	go backward to expression
n	repeat previous search
N	reverse previous search

Cursor Movement

h	left
j	down
k	up
l	right
O	go to beginning of line
S	go to end of line
%	go to match
G	go to last line
3G	go to line 3
W	go forward 1 word
3W	go forward 3 words
B	go back 1 word
3B	go back 3 words

Delete Text

x	delete 1 character
dw	delete 1 word
dd	delete 1 line
D	delete to end of line
d0	delete to beginning of line
dG	delete to end of file
4dd	delete 4 lines

Add/Append Text

a	append after cursor
A	append to end of line
i	insert before cursor
5i	insert 5 times
I	insert at beginning of line

Add New Lines

o	new line below cursor
O	new line above cursor

Change Text

cw	change word
3cw	change 3 words
C	change line
r	replace 1 character
R	replace line

Copy and Insert Text

yy	copy a line
3yy	copy 3 lines
p	put below cursor
P	put above cursor

Prepared by: Amitabh Srivastava

Pico/Nano Editor

- **Nano** is a user-friendly, simple and WYSIWYG(**W**hat **Y**ou **S**ee **I**s **W**hat **Y**ou **G**et) text editor.
- Unlike vim editor or any other command-line editor, it doesn't have any mode.
- Available as PICO editor in modern Linux distros.

- To create a file using nano editor, give the following command-
  - nano <filename>

- To save a file- **Ctrl + o**

- To cut the selection- **Ctrl + k**

- To paste- **Ctrl + u**

- To search and replace- **Ctrl + **

- To read/insert another file text- **Ctrl + r**

- To copy and paste- **Ctrl + 6, Alt + 6 and Ctrl + u**

- To quit from nano editor- **Ctrl + x**

Prepared by: Amitabh Srivastava

Vi or Vim Editor

- Vi editor is elaborated as **vi**sual editor.
- Available to every Unix system.
- Available to all Linux distros.
- Improved version of Vi is known as V**im**.
- Used to create simple text files in Linux using command line interface.
- Two vi modes- *Command* mode and *Insert* mode.
- To switch from command mode to insert mode press "**i**" key.
- To switch from insert mode to command mode press '**Esc**' key.
- By default, Vi editor starts in command mode.

grep command

- It is a filter command that searches a file for a given pattern of characters.
- It displays all the lines containing that given pattern.

Options and Examples-

grep linux about.txt
It searches the given pattern “linux” in about.txt file.

grep –i linux about.txt
It will displays all the lines containing the given pattern by ignoring the case of pattern.

```
amitabh123@Amitabh-laptop:~$ cat about.txt
Linux is great Operating System. It was developed by Linus Torvalds.
It has many many different distros as it has open source license.
Linux is easy to learn. It is a multiuser OS.
It is very user friendly OS.
```

grep –c Linux about.txt
Will display only the count of matches.

grep –l “linux” *
It will display the filenames that matches that pattern.

grep –w “user” about.txt
It will display the lines having match pattern as a whole word and not as a substring (default behaviour).

grep command...contd.

Options and Examples-

grep -o “Linux” about.txt
It will display all the matched patterns only and not the entire line containing matched pattern.

grep –n “Linux” about.txt
It will displays all the lines containing the given pattern along with the line numbers.

grep –v “Linux” about.txt
It will displays all the lines **NOT containing** the given pattern.

grep “^Linux” about.txt
It will displays all the lines containing the given pattern at the beginning of line.

grep “OS.$” about.txt
It will displays all the lines containing the given pattern at the end of line.

grep –e “Linux” –e “Linus” about.txt OR grep –E “Linux|Linus” about.txt
Used to search multiple patterns in a file.

grep command...contd.

Options and Examples-

grep -f patterns about.txt
It will use take the pattern from the given file and display all the lines containing that pattern.

grep –A1 “Linus” about.txt
It will displays all the lines containing the given pattern along with **one line below** the matched one.

grep –B1 “Linus” about.txt
It will displays all the lines containing the given pattern along with **one line above** the matched one.

grep –C1 “source” about.txt
It will displays all the lines containing the given pattern along with **one line above** and **one line below** the matched one.

grep –R “Linux” <directory>
It will look for the given pattern in the given directory **recursively** in all files and displays all the lines containing matched pattern.

grep command with pipe

Examples-

ls –l | grep –v <pattern>
It will display all the files whose owner or group is not matched with given pattern.

ls –l /etc | grep “Nov”
It will display all the files from /etc directory having November in the month field.

ls –l | grep “.txt$”
It will display all the files having extension .txt.

ls –l | grep “d”
It will display all the directories only.

ls –l | gre “d” | wc -l
It will display total no. of directories.

cat /etc/passwd | grep <username>
It will display the user information of the given username as matched pattern.

cat /etc/group | grep “sudo”
It will display the group information named “sudo”.

Prepared by: Amitabh Srivastava

egrep command

- It is a filter command that searches a file for a given pattern of characters.
- It is faster than grep.
- Difference between grep and egrep is that egrep uses extended regular expressions.
- All the options are same as grep command.

Example-

$ cat q1

```
Hello how are you?
I am fine.
How about you
I am fine too.
The End
```

$ egrep "^[a-z A-Z]+$" q1

```
How about you
The End
```

sort command

- It is used to sort a file and arrange the records/lines in a particular order.
- By default, files are sorted by assuming the contents are ASCII (characters).

Features-

- Sorts the contents of a text file line-by-line.
- Supports sorting alphabetically, by number, by months, in reverse order.
- It can also remove duplicate records.
- Supports sorting on columns/fields. (default field separator is blank space)

Options & Examples-

```
amitabh123@Amitabh-laptop:~$ cat names
Deepak
amit
bhaskar
Harish
Alok
Chandra
Om
Lalit
mahesh
```

$ sort names

```
Alok
Chandra
Deepak
Harish
Lalit
Om
amit
bhaskar
mahesh
```

sort command...contd.

Options & Examples-

$ sort –r names

Sort in reverse order.

```
mahesh
bhaskar
amit
Om
Lalit
Harish
Deepak
Chandra
Alok
```

File having numbers:

$ cat > numbers

```
12
23
5
100
67
32
16
4
28
45
^C
```

Sorting is not correct:

$ sort numbers

```
100
12
16
23
28
32
4
45
5
67
```

Sorting is now correct:

$ sort –n numbers

```
4
5
12
16
23
28
32
45
67
100
```

sort command...contd.

- **Sorting on Columns-**

$ cat items

```
12      Laptop
3       Mouse
100     Cables
68      Usb drives
45      Monitors
55      Watches
```

$ sort –k2 items

```
100     Cables
12      Laptop
45      Monitors
3       Mouse
68      Usb drives
55      Watches
```

$ sort –nk1 items

```
3       Mouse
12      Laptop
45      Monitors
55      Watches
68      Usb drives
100     Cables
```

- **Storing sort result in a file-**

$ sort –o <output filename> <input filename>

OR

$ sort <filename> > <output filename>

sort command...contd.

- **Check whether a file is sorted-**

$ cat fruits
```
Apple
Banana
Cherry
Orange
```

$ sort –c fruits
no output means file is sorted!

$ cat fruits
```
Apple
Cherry
Banana
Orange
```

$ sort –c fruits
sort: fruits:3: disorder: Banana

- **Remove duplicates-**

$ cat fruits
```
Apple
Banana
Cherry
Orange
Banana
Papaya
Cherry
Mango
```

$ sort –u fruits
```
Apple
Banana
Cherry
Mango
Orange
Papaya
```

sort command...contd.

- **Sort by Month names-**

$ cat months

```
February
March
April
September
May
January
```

$ sort –M months

```
January
February
March
April
May
September
```

wc command

- Stands for **Word Count** and used mainly for counting of characters in a file.
- By default, it produces 4 columnar output- no. of lines, word count, char(byte) count and filename, respectively.

Options-

wc <filename>
It will produce 4 columnar output- no. of lines, word count, char(byte) count and filename, respectively.

wc <filename1> <filename2> <filename..N>
It will produce 4 columnar output for all the filename mentioned.

wc –l <filename1>
It will display no. of lines for the mentioned filename.

wc –w <filename1>
It will display no. of words for the mentioned filename.

wc –c <filename1> OR wc –m <filename>
It will display no. of characters (bytes) for the mentioned filename.

wc command...contd.

Options-

wc –L <filename>
It will display the length of longest line (characters count) of mentioned filename.

wc --version
It will display the version, author and copyright information of wc command.

Examples-

ls | wc –l
To count the no. of files and directories in the current directory.

cat /etc/passwd | wc –l
To count the total no. of users.

cut command

- Used to cut out the specific sections from each line of files and display the result to standard output.
- It slices the line by character(byte) or column/field position and extracts the text.

Options and Examples-

$ cat Employees

```
SN,Ecode,Name,Dept,Salary
1,emp001,Alok,Sales,25000
2,emp002,Deepak,Accounts,45000
3,emp003,Firoz,Sales,28000
4,emp004,Ram,IT,78000
5,emp005,Gopal,IT,86000
```

$ cut –c 1 Employees

```
S
1
2
3
4
5
```

$ cut –c 1,4 Employees

```
SE
1m
2m
3m
4m
5m
```

$ cut –c 3-8 Employees

```
,Ecode
emp001
emp002
emp003
emp004
emp005
```

cut command...contd.

Options and Examples-

$ cut –c 3- Employees

```
,Ecode,Name,Dept,Salary
emp001,Alok,Sales,25000
emp002,Deepak,Accounts,45000
emp003,Firoz,Sales,28000
emp004,Ram,IT,78000
emp005,Gopal,IT,86000
```

$ cut –c -3 Employees

```
SN,
1,e
2,e
3,e
4,e
5,e
```

$ cut –d “,” –f 3 Employees

```
Name
Alok
Deepak
Firoz
Ram
Gopal
```

$ cut –d “,” -f3,5 Employees

```
Name,Salary
Alok,25000
Deepak,45000
Firoz,28000
Ram,78000
Gopal,86000
```

cut command...contd.

Options and Examples-

$ cut –d “,” -f3- Employees

```
Name,Dept,Salary
Alok,Sales,25000
Deepak,Accounts,45000
Firoz,Sales,28000
Ram,IT,78000
Gopal,IT,86000
```

$ cut –d “,” –f 3 --output-delimiter=“|” Employees

```
Name|Dept|Salary
Alok|Sales|25000
Deepak|Accounts|45000
Firoz|Sales|28000
Ram|IT|78000
Gopal|IT|86000
```

$ cut --complement –d “,” -f3- Employees

```
SN,Ecode
1,emp001
2,emp002
3,emp003
4,emp004
5,emp005
```

$ cut --version

Displays the version, author and copyright information.

$ cat /etc/passwd | cut –d “:” –f1

Display the names of all the user accounts.

sed command

- **SED** is a **S**tream **Ed**itor to perform inserting, deleting, searching and replacing text in a text file without using any editor.

Options and Examples-

```
amitabh123@Amitabh-laptop:~$ cat about.txt
Linux is great Operating System. It was developed by Linus Torvalds.
It has many many different distros as it has open source license.
Linux is easy to learn. It is a multiuser OS.
It is very user friendly OS.
```

$ sed 's/\bis/\bwas/' about.txt
It will replace the 1st occurrence of "is" as a whole word with "was" in each line. **\b** is used for word boundary.

$ sed 's/Linux/LINUX/' about.txt
It will replace first occurrence of word "Linux" with "LINUX" in each line in the file called about.txt. Here, 's' specifies the substitution operation. The "/" is the delimiter symbol.

Note- By default, the sed command replaces the first occurrence of the pattern/word in each line and it wouldn't replace the second or other occurrences in the same line.

$ sed 's/is/was/g' about.txt
It will replace all the occurrence of pattern "is" with "was" in each line in the file called about.txt. Here, "g" specifies the "global" replacement.

sed command...contd.

Options and Examples-

$ sed ‘s/is/was/2’ about.txt
It will replace second occurrence of word “is” with “was” in each line in the file called about.txt. Here, ‘2’ specifies the 2nd occurrence of the pattern.

$ sed ‘s/is/was/2g’ about.txt
It will replace the pattern starting from second occurrence to all the occurrences in each line.

$ sed ‘3 s/is/was/’ about.txt
It will replace the first occurrence of the matching pattern in 3rd line only.

$ sed ‘3 s/is/was/2’ about.txt
It will replace the 2nd occurrence of the matching pattern in 3rd line only.

$ sed ‘3 s/is/was/g’ about.txt
It will replace all the occurrences of the matching pattern in 3rd line only.

$ sed ‘s/is/was/p’ about.txt
It will replace the 1st occurrence of the matching pattern in each line and prints the replaced line twice on the terminal. If a line doesn’t have any matching pattern then it prints only once.

sed command...contd.

Options and Examples-

$ sed '1,3 s/is/was/' about.txt
It will replace 1st occurrence of word "is" with "was" in lines 1 to 3.

$ sed '3,$ s/is/was/' about.txt
It will replace 1st occurrence of word "is" with "was" from 3rd line to last line ($) in the file about.txt.

$ sed '5d' about.txt
It will delete 5th line from the file about.txt.

$ sed '2d;4d' about.txt
It will delete only 2nd and 4th line from the file about.txt

$ sed '$d' about.txt
It will delete last line from the file about.txt.

$ sed '2, 4d' about.txt
It will delete lines from 2nd to 4th from the file about.txt.

$ sed '2, $d' about.txt
It will delete 2nd to last line from the file about.txt.

$ sed '/Linus/d' about.txt
It will delete all the lines in which the matching pattern 'Linus' is found.

sed command...contd.

Options and Examples-

$ sed G about.txt
It will insert a blank line after each line.

$ sed 2G about.txt
It will insert a blank line after 2nd line.

$ sed ‘G;G’ about.txt
It will insert 2 blank lines after each line.

$ sed ‘/Linux/G’ about.txt
It will insert a blank line after each line in which matching pattern ‘Linux’ found.

$ sed '/^$/d' about.txt
Deletes all blank lines.

$ sed 's/^/   /' about.txt
It will insert 5 spaces at the beginning of every line.

$ sed = about.txt | sed 'N;s/\n/\t/’
Inserts line number at the beginning of each line. \t is used for a tab between number and sentence.

awk command

- **Awk** is a scripting language used for manipulating data and generating reports. The awk command programming language requires no compiling and allows the user to use variables, numeric functions, string functions, and logical operators.

- Awk operations include-
  - Scans a file line by line
  - Splits each input line into fields
  - Compares input line/fields to pattern
  - Performs action(s) on matched lines

Options and Examples-

$ cat Employees

```
1 emp001 Alok Sales 25000
2 emp002 Deepak Accounts 45000
3 emp003 Firoz Sales 28000
4 emp004 Ram IT 78000
5 emp005 Gopal IT 86000
```

$ awk '{print}' Employees

```
1 emp001 Alok Sales 25000
2 emp002 Deepak Accounts 45000
3 emp003 Firoz Sales 28000
4 emp004 Ram IT 78000
5 emp005 Gopal IT 86000
```

awk command...contd.

Options and Examples-

$ awk '/Sales/ {print}' Employees

```
1 emp001 Alok Sales 25000
3 emp003 Firoz Sales 28000
```

$ awk '{print NR, $2, $5}' Employees

```
1 emp001 25000
2 emp002 45000
3 emp003 28000
4 emp004 78000
5 emp005 86000
```
(NR- Record No.)

$ awk '{print $3, $5}' Employees

```
Alok 25000
Deepak 45000
Firoz 28000
Ram 78000
Gopal 86000
```

$ awk '{print $2, $NF}' Employees

```
emp001 25000
emp002 45000
emp003 28000
emp004 78000
emp005 86000
```
(NF- Last Field)

$ awk 'END { print NR }' Employees

5 (displays the line count)

$ awk 'length($0) > 28' Employees

Displays the line whose length is greater than 28 chars.

$ awk '{ if($4 == "Sales") print $0;}' Employees

Displays only those records where value of 4th field is equal to “Sales”.

date command

- **date** command is used to display the current system date and time information.

Options and Examples-

date
It will display the current system date and time. The format is:

```
Tue Nov 28 10:35:59 IST 2023
```

date –u
It will display the system date and time in GMT(Greenwich Mean Time)/UTC(Coordinated Universal Time).

```
Tue Nov 28 05:14:16 UTC 2023
```

date –d “02/03/2010” or date --date “02/03/2010”
It will display the given date in the date format.

```
Wed Feb  3 00:00:00 IST 2010
```

date command...contd.

Options and Examples-

Date --date “2 years ago”
It will display the date and time 2 years ago from the current system date and time.

```
Sun Nov 28 10:54:23 IST 2021
```

date --date “5 minutes ago”
It will display the date and time 5 minutes ago from the current system date and time.

date --date “10 seconds ago”
It will display the date and time 10 seconds ago from the current system date and time.

date --date “2 months ago”
It will display the date and time 2 months ago from the current system date and time.

date --date “next tuesday” or date --date “tomorrow”
It will display the date on next Tuesday or of tomorrow.

date --date “5 days”
It will display the date 5 days after the current system date and time.

date command...contd.

Options and Examples-

date --set “Tue Nov 28 16:15:35 IST 2023”
It will set the current system date and time as specified.

$ cat datefile.txt
Sep 23 2010
Nov 27 2022
Oct 28 2009

$ date --file=“datefile.txt”

```
Thu Sep 23 00:00:00 IST 2010
Sun Nov 27 00:00:00 IST 2022
Wed Oct 28 00:00:00 IST 2009
```

Date: Format Specifiers

Following are the format specifiers used to display the date and time information in a specific format-

```
%D: Display date as mm/dd/yy.
%d: Display the day of the month (01 to 31).
%a: Displays the abbreviated name for weekdays (Sun to Sat).
%A: Displays full weekdays (Sunday to Saturday).
%h: Displays abbreviated month name (Jan to Dec).
%b: Displays abbreviated month name (Jan to Dec).
%B: Displays full month name(January to December).
%m: Displays the month of year (01 to 12).
%y: Displays last two digits of the year(00 to 99).
%Y: Display four-digit year.
%T: Display the time in 24 hour format as HH:MM:SS.
%H: Display the hour.
%M: Display the minute.
%S: Display the seconds.
```

To use these specifiers, follow the syntax-

$ date +format-specifier

Examples-

$ date +%D
11/28/23

$ date +"Today is: %B %d %Y"
Today is: November 28 2023

cal command

It is used to display the calendar on Linux terminal.

Options and examples-

cal
It will display the calendar of current month.

cal <mm> <yyyy>
It will display the calendar of given/selected month and year.

cal <yyyy>
It will display all the months of given year.

cal –j <yyyy>
It will display the calendar of given year in Julian calendar format. In Julian calendar date is not reset to 1st after every month.

bc command

- Stands for **Basic Calculator** used for command line calculations.
- It can perform simple arithmetic operations to complex calculations including mathematical functions.

Options and examples-

$ echo ‘20+5’ | bc

$ echo ‘scale=2; 3/2’ | bc

$ echo ‘a=5; b=12; a+b’ | bc

$ echo 'a[1]=10; a[2]=20; a[1]+a[2]' | bc

$ echo 'a=0; while(a<=5) { print a; a+=1; print "\n";}' | bc

$ echo 'x=45; sqrt(x)' | bc

$ echo 'x=5674; length(x)' | bc
Returns the no. of digits in x.

$ bc -i
Starts interactive mode. To quit from interactive mode type ‘quit’.

Prepared by: Amitabh Srivastava

uname command

- Prints the system information.

Options-

$ uname -s
Prints the kernel name.

$ uname –n
Prints the host name.

$ uname –r
Prints the kernel release.

$ uname -v
Prints the kernel version with release date.

$ uname -m
Prints the machine's hardware type.

$ uname -o
Prints the operating system name.

$ uname -a
Prints all the system information.

$ uname -p
Prints the processor type.

uptime command

- It is used to find out how long the system is active (running).

Options-

$ uptime

```
13:36:51 up 58 min,  1 user,  load average: 0.00, 0.00, 0.00
```

$ uptime -p

Returns the total no. of hours and minutes of system active.

$ uptime -s

Returns the starting time by when the system is running.

```
2023-11-29 12:38:03
```

$ uptime –V or uptime --version

Returns the version information.

```
uptime from procps-ng 3.3.17
```

hostname command

- *hostname* command in Linux is used to obtain the DNS (Domain Name System) name and set the system's hostname.
- Its main purpose is to uniquely identify over a network.

Options-

$ hostname
Displays the system's hostname.

$ hostname -a or --alias
Displays all the alias names of the host system. Aliases are stored inside the file **/etc/hosts**.

$ hostname -f or --fqdn
Displays the system's Fully Qualified Domain Name (FQDN).

$ hostname -F myhost.txt
Set the hostname as specified under the given filename. (must be 'root' or 'sudo' user).

$ hostname -h
Displays the help of hostname command.

ping command

- PING **(Packet Internet Groper)** command is used to check the network connectivity between two hosts in a network.
- It takes IP address or the URL as input and sends a data packet to the specified address with the message “PING” and get a response from the server/host.
- The time is recorded which is called *‘latency’*. Low latency(less time) means faster connection.
- Ping uses **ICMP(Internet Control Message Protocol)** to send an **ICMP echo message** to the specified host if that host is available then it sends **ICMP reply message**.

Options and Examples-

$ ping www.google.com
Sends packets with echo message to the specified host URL.

$ ping -c 2 www.google.com
Sends 2 packets with echo message to the specified host URL.

$ ping -s 40 www.google.com
Sends packets of 40 bytes (+8 bytes) with echo message to the specified host URL.

$ ping -i 2 www.google.com
Sends each packet with the interval of 2 seconds. By default, ping waits for 1 sec. to send next packet.

$ ping –q
Display only the summary information.

$ ping -V
Display the ping version information.

Prepared by: Amitabh Srivastava

ifconfig command

- **ifconfig**(interface configuration) command is used to configure the network interfaces.
- It is used at the boot time to set up the interfaces as necessary.
- This command is used to assign the IP address and netmask to the network interface or to enable or disable a network interface.

Options and Examples-

$ ifconfig -a
Display the information of all the interfaces available.

$ ifconfig -s
Display the short information of all the interfaces available.

$ ifconfig <interface> down
Deactivates/disables the driver for the given interface.

$ ifconfig <interface> up
Activates/enables the driver for the given interface.

$ ifconfig <interface> <ipaddress> netmask <subnetmask>
Set the specified ip-address and netmask to the given interface.

$ ifconfig --help
Displays help of ifconfig command.

Prepared by: Amitabh Srivastava

traceroute command

- **traceroute** command prints the route that a packet takes to reach the host.
- This command is useful to know about the route and about all the hops (in between nodes) that a packet takes.
- It sends three packets to each hop.

Options and Examples-

$ traceroute google.com
Displays the rout information in different columns.
- First column is the hop count.
- Second column is the ip-address of that hop.
- Then the three times for three packets, respectively.

$ traceroute -f 10 google.com
Instead of first hop, it starts tracing route from the 10^{th} hop.

$ traceroute -n google.com
Do not resolve the IP-addresses to their domain names.

$ traceroute google.com 100
Change the default packet length from 60 bytes to 100 bytes.