Exp No: 2
Date:

LINUX BASIC COMMANDS - II

Aim:

To learn Linux commands for redirection, pipes, filters, job control, file ownership, file permissions, links and file system hierarchy.

Description:

By default, the output is displayed on the screen. Using redirection commands, it is possible to send output to file or to read input from file. A pipe is a way to connect the output of one program to the input of another program without any temporary file. A pipe is nothing but a temporary storage place where the output of one command is stored and then passed as the input for second command. A filter performs some kind of processes on the input and gives output.

REDIRECTION

PURPOSE	COMMAND SYNTAX	EXAMPLE
To output Linux commands result to file (if file already exist, it will be overwritten else new file is created)	Linux command > file name	$ ls > myfile
To output Linux commands result to END of file (appending) .If file exist data will be written to END of file without losing previous information, otherwise new file is created	Linux command >> file name	$ date >> myfile
To take input to Linux command from the file instead of key-board.	Linux command < file name	$ cat < myfile

PIPES

PURPOSE	COMMAND SYNTAX	EXAMPLE
Output of **ls** command is given as input to **more** Command so that output is printed one screen Full page at a time.	Command1| command2	$ ls | more
Output of **who** command is given as input to **sort** command so that it will print sorted list of users.	Command1| command2	$ who | sort
Output of **who** command is given as input to **wc** command so that it will number of users who logon to system.	Command1| command2	$ who | wc -l
Output of **ls** command is given as input to **wc** Command so that it will print number of files In current directory.	Command1| command2	$ ls –l | wc -l

FILTER

PURPOSE	COMMAND SYNTAX	EXAMPLE
wc command		
To see no of characters in a file	wc –c {file name}	$ wc –c
To see no of words in a file	wc –w {file name}	$ wc –w
To see no of lines in a file	wc –l {file name}	$ wc –l
To see no of lines , words, characters at a time	wc {file name}	$ wc myfile
To see the top 10 lines of a file	head {file name}	$ head myfile
To see the top 5 lines of a file	head -5 {file name}	$ head -5 myfile
To see the last 10 lines of a file	tail {file name}	$ tail myfile
To see the last 20 lines of a file	tail -20 {file name}	$ tail -20 myfile
grep command		
To search a pattern of word in a file	grep {word } {file name }	$ grep hi myfile
To search multiple words in a file	grep –E ‘word1|word2’ {f name}	$ grep –E ‘hi|no|’ myfile
To show all the lines that do not match the searched string	grep –v {word} {file name}	$ grep –v hi myfile
To display only the count of matching lines	grep –c {word} {file name}	$ grep –c my myfile
To show the matching line and its number	grep –n {word} {file name}	$ grep –n apple myfile
To match both upper and lower case	grep –i {word} {file name}	$ grep –i my myfile
sort command		
To sort out the content of the file alphabetically	sort {file name}	$ sort myfile
Reverse sorting	sort –r {file name}	$ sort –r myfile
Sort numerically	sort –n {file name}	$ sort –n myfile
Case sensitive sorting	sort –f {file name}	$ sort –f myfile
tr command		
To translate all occurrences of one Character to another	tr {character1} {character2}	$ tr ‘e’ ‘E’
To set all letters to uppercase	tr {range1} {range2}	$ tr ‘a-z’ ‘A-Z’
To translate all new lines to space	Example: $ tr ‘\n’ ‘ ‘	
To encrypt the text	Example: $ tr ‘a-z’ ‘plokmnjiuhbvgytfcrdxeszawq’	
uniq command		
To remove duplicates from a sorted list	$ sort {file name} | uniq	
To the occurrences of a word in a file	$ sort {file name} | uniq -c	

FILE PERMISSIONS AND OWNERSHIP

PURPOSE	COMMAND SYNTAX	EXAMPLE
To create a new group on the system	groupadd {group name}	$ groupadd mygroup
To show which groups you are in	groups	$ groups
To check the file permissions	ls -l	$ ls -l
To add write permission to all users	chmod a+w file name	$ chmod a+w myfile
To add rad permission to only the users In your group	chmod g+r file name	$ chmod g+r myfile
To make a file executable and runnable By any user	chmod a+x file name	$ chmod a+x myfile
To remove specific permission	chmod {u|g|o|a} - {r|w|x} {file name}	
To add and remove permissions in a Single step	chmod u+x, g+r, o-rwx {filename}	

JOB CONTROL COMMANDS

PURPOSE	COMMAND SYNTAX	EXAMPLE
To list all current running processes their corresponding pid and their status	ps	$ps
To provide a list of running process in a tree structure	pstree -p	$ pstree -p
To stop any process i.e, to kill process	kill {pid}	$ kill 1012
To stop all process except your shell	kill 0	$ kill 0
To list all jobs presently running on your system	jobs	$ jobs
For background processing	Linux command &	$ ls / -r | wc –l &

Exercise1:
Send output of **date** command to already exist file

Exercise2:
Encrypt the text in a file and display it

Exercise3:
Create a file **sample** and give 5 names init. Display only the lines that does not contain the character ‘a’, but the result should be in reverse order.

Exercise4:
Write a command to give **write** and **execute** permission on a file

Exercise5:
Write a code to list all current running processes their corresponding pid

Result:

Thus the LINUX basic commands are successfully executed and verified.

Viva voice Questions:

1. What is the use of redirection commands?
2. What is the difference between pipes and filters?
3. What is the abbreviation of grep command?
   Ans: global search for regular expression and print
4. How we set file permissions?
5. What is the various job control command?