Linux Commands

Wednesday, July 19, 2023 7:54 PM

➢ Finding Files and Directories:

1. Find Command :

Find [path] expression

Used to find the files in the specific path with given expression

Search Patterns :

-name :	find files matches that pattern.	Ex: $find /home -name test.txt op: /home/test.txt
-iname :	ignores case sensitive.	Ex: $ find /home -iname test.txt op: /home/test.txt
-name "*v" or "v*" or "*v*" :	files ending or starting or has "v" in it.	Ex: $find /home -name "t*" op: /home/test.txt
-mtime +/-days :	files older/not older than days.	Ex: $ find -mtime -10 op: ./test.txt
-size +/-num :	files that are size of num. or greater (+) or lesser (-)	Ex: $ find /home -size 0 op: /home/test.txt
-newer file :	files that are newer than file.	Ex: $ find /home -newer /root op: /home/test.txt

2. Locate Command:

Locate pattern

List files that matches the pattern
Faster than file command
Results are not in real time
Looks up on index
May not be enabled on all systems
Install : *sudo apt-get install mlocate*

Search Patterns:

Locate name :	finds files having name.	Ex. $ locate uptime op: /usr/share/zsh/functions/Completion/Unix/_uptime

File types :

-ld :	extracts directory
-type f :	extracts files

Basic Commands:

whatis	short description about command

➢ Viewing Files and Nano Editor:

1. Displaying the contents of file:

1. Cat file: display contents of file.

```
$ cat test.txt
1
2
3
4
5
6
7
8
9
10
11
12
```

2. Head file: file content from beginning. (By default 10 lines) to customise head -15 file

```
$ head test.txt
1
2
3
4
5
6
7
8
9
10
```

```
$ head -2 test.txt
1
2
```

3. Tail file: file content from last. (By default 10 lines) to customise tail -15 file

$ tail test.txt
3
4
5
6
7
8
9
10
11
12

$ tail -2 test.txt
11
12

4.

More file:	browse through the file. Ex: log files
Less file:	more features than more command. Ex: log files
Tail -f file:	to get realtime changes in a file (Used instead of cat) Ex: log files

2. Nano Editor:

Nano file name
Simple editor
Easy to learn
Not as advanced as vi
If nano not available look for pico
Ex: nano test.txt
To save & exit: Ctrl+X

➢ Vi Editor:

vi [file]:	Edit file.
vim [file]:	Same as vi, but more features.
view [file]:	Starts vim in read-only mode.

1. Vi Command Mode and Navigation:

1.

k	Up one line.
j	Down one line.
h	Left one character.
l(L)	Right one character.
w	Right one word.
b	Left one word.
^	Go to the beginning of the line.
$	Go to the end of the line.

2. Vi Insert Mode:

1.

i:	Insert at the cursor position.
I(capital i):	Insert at the beginning of the line.
a:	Append after the cursor position.
A:	Append at the end of the line.

3. Vi Line Mode:

:w	Writes (saves) the file.
:w!	Forces the file to be saved.
:q	Quit.

1

:q!	Quit without saving changes.
:wq!	Write and quit.
:x	Same as :wq.

4. **Vi Searching:**

/	Start a forward search.
?	Start a reverse search.

➢ Deleting, Copying, Moving and Renaming Files:

1. Deleting Files:

Rm file	delete file
Rm -r dir	delete directory and its files
Rm -f file	force removal not asking for confirmation

2. Copying files:

cp source_file destination_file	Copy source_file to destination_file.	Ex: $ cp test2.txt test3.txt
cp src_file1 [src_fileN ...] dest_dir	Copy source_files to destination_directory	Ex: cp test.txt test2.txt test4.txt home1
cp - i(small i)	Interactive mode.	Ex: $ cp -i test.txt test2.txt | cp: overwrite 'test2.txt'? Y
cp -r dir dir2	copy contents from dir to dir2(creates new directory).	Ex: $ cp -r home1 home2

3. Moving and Renaming Files:

1.	*mv source destination*	moves the files from source to destination	Ex: $ mv home home2
2.	*mv -i*(small i)	source destination - interactive mode	

4. Sort Options:

1. | *sort -k F filename* | sort in alphabetic order for the cloumn F |

```
$ sort -u -k2 test
c : a
a : c
b : f
d : w
d : w
```

2. | *sort -r filename* | sorts in reverse order |

```
$ sort -r test
d : w
d : w
c : a
b : f
a : c
```

3. | *sort -u filename* | removes duplicates |

```
$ sort -u -k2 test
c : a
a : c
b : f
d : w
d : w
```

5. Creating a collection of Files:

tar [-] c|x|t f tarfile [pattern]	Create, extract or list contents of a tar archive using pattern, if supplied.
c	Create a tar archive.
x	Extract files from the archive.
t	Display the table of contents (list).
v	Be verbose.
z	Use compression.
f	file Use this file

6. Compressing Files:

gzip	Compress files.
gunzip	Uncompress files.
gzcat	Concatenates compressed files.
zcat	Concatenates compressed files.

7. Disk Usage:

du	Estimates file usage.
du -k	Display sizes in Kilobytes.
du -h	Display sizes in human readable format.

➢ Comparing files:

1. | diff file1 file2 | Compare two files. |
   $ diff file1 file2
   3c3
   ...
   LineNumFile1-Action-LineNumFile2
   Action = (A)dd (C)hange (D)elete

   $ diff file1 file2
   3c3
   < this is a line in a file.

   > This is a Line in a File!
   < Line from file1
   > Line from file2
2. | sdiff file1 file2 | Side-by-side comparison. |
   $ sdiff file1 file2
   line in file1 | line in file2
   > more in file2
   | Differing lines
   < Line from file1
   > Line from file2
3. | vimdiff file1 file2 | Highlight differences in vim. |
   Ctrl-w w Go to next window
   :q Quit (close current window)
   :qa Quit all (close both files)
   :qa! Force quit all

➢ Searching in Files (Using Pipes):

1. grep:

1. | *grep pattern file* | Display lines matching a pattern. |
-i	Perform a search, ignoring case.
-c	Count the number of occurrences in a file.
-n	Precede output with line numbers.
-v	Invert Match. Print lines that don't match.

2. The file command:

➢ | file file_name | Display the file type. |
   $ file sales.data s
   ales.data: ASCII text
   $ file *
   bin: directory jason.tar: POSIX tar archive

3. Pipes:

| | | Pipe symbol |
command-output | command-input
cat file | grep pattern