17.12.2024

Day-3

Basic commands part-3

→ In recording they lost the instance and created new one.

→ we will continue with our old instance.

→ for our safety, clear everything on our instance.

Putty:

```
[root@ip-172-31-84-166 ~]# touch file1
[root@ip-172-31-84-166 ~]# cat > file1
hello
world
welcome
to
linux
class
```

[∴ CTRL + D = exit]
or
[∴ CTRL + Z = exit]

```
[root@ip-172-31-84-166 ~]# cat file1
hello
world
welcome
to
linux
class
[root@ip-172-31-84-166 ~]# nl file1
```

[∴ nl = number line]

```
1 hello
2 world
3 welcome
4 to
5 linux
6 class
```

[root@ip-172-31-84-166 ~]# grep -i file1

[:. grep - to search words]

hello
world
welcome
linux
class

[root@ip-172-31-84-166 ~]# touch file2

[root@ip-172-31-84-166 ~]# cat > file2

hello
world
to
linux
classsss

[root@ip-172-31-84-166 ~]# diff file1 file2

[:. diff - to show the difference between two files]

3d2
< welcome
6c5
< class

> classsss

[root@ip-172-31-84-166 ~]# touch file3

[root@ip-172-31-84-166 ~]# cat > file3

SRH
SRH
SRH
SRH

[root@ip-172-31-84-166 ~]# uniq file3

[:. uniq - removes consecutive duplicates]

SRH

[root@ip-172-31-84-166 ~]#

Filter commands in linux :

1. Head and Tail.
2. more and less.
3. sort.
4. cut.
5. Sed.
6. awk.
7. find.

```
[root@ip-172-31-84-166 ~]# touch headfile1
[root@ip-172-31-84-166 ~]# cat > headfile1
one
two
three
:
fourteen
fifteen

[root@ip-172-31-84-166 ~]# head headfile1
one
:
ten
```

[∴ By default "head" displays the first 10 lines of file]

```
[root@ip-172-31-84-166 ~]# head -5 headfile1
one
two
three
four
five

[root@ip-172-31-84-166 ~]# touch tailfile1
```

```
[root@ip-172-31-84-166 ~]# cat > tailfile1
one
:
fifteen

[root@ip-172-31-84-166 ~]# tail tailfile1
six
seven
:
fourteen
fifteen

[root@ip-172-31-84-166 ~]# tail -5 tailfile1
eleven
twelve
thirteen
fourteen
fifteen

[root@ip-172-31-84-166 ~]# touch morefile1
[root@ip-172-31-84-166 ~]# cat > morefile1
one
:
fifty
[root@ip-172-31-84-166 ~]# more morefile1
one
:
twenty-six
more End :/
```

[∴ By default, "tail" displays the last 10 lines of the file]

[∴ Navigating through more:

* Press spacebar - Moves to next screen (shows lines 11 to 20).
* Press Enter - scrolls one line at a time.
* Press b - Goes back one screen (only works if your terminal allows backward scroll).
* Press q - quits the more command.]

[root@ip-172-31-84-166 ~]# less makefile1

one
:
twenty-six
less:

[∴ Press spacebar → Moves to next screen.

Press b - Moves back to previous screen.

Search for a word:

Type /six → Press Enter.

The viewer jumps to the first occurence of "six".

Press n → Jump to the next occurence of "six".

Press q - Exit the "less" command.]

→ before going next delete all files and directories

[∴ Now what we gonna see is sort]

sort:

sort (used to order the value) - alphabetically.

Description:

-r = sorts data in reverse order (descending).

-n = sorts a file numerically (interprets data as numbers)

-nr = sorts a file with numeric data in reverse order. combines -n and -r

-c = checks if the file is already sorted and reports any disorder.

-u = sorts and removes duplicate lines, providing a unique sorted list.

-M = sorts by month names.

```
[root@ip-172-31-84-166 ~]# touch data.txt
[root@ip-172-31-84-166 ~]# cat > data.txt
banana
apple
grape
orange
kiwi
[root@ip-172-31-84-166 ~]# sort data.txt
apple
banana
grape
kiwi
orange
[root@ip-172-31-84-166 ~]# sort -r data.txt
orange
kiwi
grape
banana
apple
```

[∴ sorts in alphabetical order]

[∴ sorts in reverse alphabetical order]

[root@ip-172-31-84-166 ~]# touch numbers.txt

[root@ip-172-31-84-166 ~]# cat > numbers.txt

10
2
30
1
20

[root@ip-172-31-84-166 ~]# sort -n numbers.txt

[:. "-n" option ensures nu

1
2
10
20
30

are sorted in numeri

(not based on text orde

[root@ip-172-31-84-166 ~]# sort -nr numbers.txt

[:. sorts numbers in descending

30
20
10
2
1

numerical order]

[root@ip-172-31-84-166 ~]# touch dup.txt

apple
banana
apple
kiwi
banana

[root@ip-172-31-84-166 ~]# sort -u dup.txt

[:. "-u" option removes duplicate

apple
banana
kiwi

lines and sorts the output]

[root@ip-172-31-84-166 ~]# touch months.txt

[root@ip-172-31-84-166 ~]# cat months.txt

March

January

February

December

[root@ip-172-31-84-166 ~]# sort -M months.txt

[∴ "-M" option sorts months in calendar order, not alphabetical order]

January

February

March

December

[root@ip-172-31-84-166 ~]# touch sorted.txt

apple
banana
grape
kiwi
orange

[root@ip-172-31-84-166 ~]# sort -c sorted.txt

[∴ No output]

→ Since the file is already in alphabetical order, "sort -c" does not return anything.

→ This means the file is sorted.

[root@ip-172-31-84-166 ~]# touch unsorted.txt

[root@ip-172-31-84-166 ~]# cat > unsorted.txt

banana
apple
grape
kiwi
orange

[root@ip-172-31-84-166 ~]# sort -c unsorted.txt

sort: unsorted.txt: 2: disorder: apple

[∴ The error message points to line 2, where the sorting issue starts]

[∴ "apple" comes after "banana" in the file, which violates alphabetical order]

[∴ Now we gonna see "cut"]

[root@ip-172-31-84-166 ~]# touch cute

[root@ip-172-31-84-166 ~]# cat > cute

the cut command in linux (type this)

∴[CTRL + D - exit]

[root@ip-172-31-84-166 ~]# cat cute

the cut command in linux

[root@ip-172-31-84-166 ~]# cut -c 1 cute

t

[root@ip-172-31-84-166 ~]# cut -c 1,3 cute

te

[root@ip-172-31-84-166 ~]# cut -c 1,5 cute

tc

[∴ space is also considered]

```
[root@ip-172-31-84-166 ~]# cut -c 1,4 cute
t
[root@ip-172-31-84-166 ~]# cut -d " " -f 1 cute
the
[root@ip-172-31-84-166 ~]# cut -d " " -f 3 cute
command
[root@ip-172-31-84-166 ~]# cut -d " " -f 4 cute
in
[root@ip-172-31-84-166 ~]# cAt >> cute
list*of*field*number
[root@ip-172-31-84-166 ~]# cAt cute
the cut command in linux
list*of*field*number
[root@ip-172-31-84-166 ~]# cut -d " " -f 1 cute
the
list*of*field*number
[root@ip-172-31-84-166 ~]# cut -d "*" -f 1 cute
the cut command in linux
list
[root@ip-172-31-84-166 ~]# cut -d "*" -f 3 cute
the cut command in linux
field
[root@ip-172-31-84-166 ~]# cut -b 1,4 cute
t
lt
```

[∴ "cut -c" and "cut -b" are almost same]

[root@ip-172-31-84-166 ~]# cat cute

the cut command in linux

list of field number

[root@ip-172-31-84-166 ~]# cut -b 1-10 cute

the cut co

list of fi

[∴ can also give in range like this "1-10"]

[root@ip-172-31-84-166 ~]# cut -c 1-10 cute

the cut co

list of fi

[root@ip-172-31-84-166 ~]#

[∴ Now we gonna see "awk" commands]

→ used to print the things

→ pattern searching and processing

[root@ip-172-31-84-166 ~]# touch linux

[root@ip-172-31-84-166 ~]# cat >> linux

virat kholi

rohit sharma

mahendra singh dhoni

hardic pandiya

subham ghill

[root@ip-172-31-84-166 ~]# awk '{print}' linux

virat kholi

rohit sharma

mahendra singh dhoni

hardic pandiya

subham ghill

[root@ip-172-31-84-166 ~]# awk '{print $1}' linux

virat

rohit

mahendra

hardik

subham

[root@ip-172-31-84-166 ~]# awk '{print $2}' linux

kholi

sharma

singh

pandiya

dhill

[root@ip-172-31-84-166 ~]# awk '{print $1}' linux > new.txt

[root@ip-172-31-84-166 ~]# ls

. . . linux new.txt

[root@ip-172-31-84-166 ~]# cat new.txt

virat

rohit

mahendra

hardik

subham

[root@ip-172-31-84-166 ~]# awk 'NR==1 {print $1}' linux [∴ NR - for row]

virat

[root@ip-172-31-84-166 ~]# awk 'NR==1 {print}' linux

virat kholi

[root@ip-172-31-84-166 ~]# awk 'NR==2 {print}' linux

rohit sharma

SED:

Explanation:

SED - Stream Editor

-i - in place editing

's/one/hope/g'

s - substitute text

one - pattern to search for

hope - replacement text

g - "global", so every occurence of "one" in a line is replaced with "hope". If we omit "g", only the first occurence in each line will be replaced.

[root@ip-172-31-84-166 ~]# touch example.txt

[root@ip-172-31-84-166 ~]# cat > example.txt

one apple
one orange
one banana

[root@ip-172-31-84-166 ~]# sed 's/one/hope/g' example.txt

hope apple
hope orange
hope banana

[... Now it only displays with changed text as "hope" but it wont affect the original file]

[root@ip-172-31-84-166 ~]# cat example.txt

one apple
one banana
one orange

[root@ip-172-31-84-166 ~]# sed -i 's/one/hope/g' example.txt

hope apple
hope banana
hope orange

[∴ Now as we included "-i" it will change the text in original file also]

[root@ip-172-31-84-166 ~]# cat example.txt

hope apple
hope banana
hope orange

find:

[root@ip-172-31-84-166 ~]# mkdir example_dir

[root@ip-172-31-84-166 ~]# cd example_dir

[root@ip-172-31-84-166 example_dir]# mkdir sub_dir

[root@ip-172-31-84-166 example_dir]# ls

sub_dir

[root@ip-172-31-84-166 example_dir]# touch file.txt

[root@ip-172-31-84-166 example_dir]# cat > file.txt

Hello

[root@ip-172-31-84-166 example_dir]# cd sub_dir

[root@ip-172-31-84-166 sub_dir]# touch file.txt

[root@ip-172-31-84-166 sub_dir]# cat > file.txt

world

[root@ip-172-31-84-166 sub_dir]# cd

[root@ip-172-31-84-166 ~]# find . -name "file.txt" [:. find files by "name"]

- ./example_dir/sub_dir/file.txt
- ./example_dir/file.txt

[root@ip-172-31-84-166 ~]# find . -name "*.txt" [:. find all files of "specific type"]

- ./example_dir/sub_dir/file.txt
- ./example_dir/file.txt

[root@ip-172-31-84-166 ~]# cd example_dir

[root@ip-172-31-84-166 example_dir]# find . -type d [:. d = directory]
[:. to find all directories]

- .
- ./sub_dir

[root@ip-172-31-84-166 example_dir]# find . -type f [:. f = file]
[:. to find all files]

- ./sub_dir/file.txt
- ./file.txt

[root@ip-172-31-84-166 example_dir]# find . -type f -name "*.txt" [:. combine criteria]
[:. search all files with ".txt" extension]

- ./sub_dir/file.txt
- ./file.txt

[root@ip-172-31-84-166 example_dir]# cat file.txt

Hello

[root@ip-172-31-84-166 example_dir]# cd sub_dir

[root@ip-172-31-84-166 sub_dir]# cat file.txt

world

[root@ip-172-31-84-166 sub_dir]# cd

[root@ip-172-31-84-166 ~]# find . -name "*.txt" -exec cat {} \;

world
Hello

Explanation:

```
find . -name "*.txt" -exec cat {} \;
```

1. find . :

This tells the "find" command to search in the current directory (.) and its subdirectories.

2. -name "*.txt" :

find all files with ".txt" extension.

3. exec:

This option tells "find" to execute a command on each file that matches the criteria.

4. cat:

→ This is the command to be executed on each matching file.

→ "cat" reads the content of a file and prints it to the terminal

5. {}:

→ This is a placeholder. It is replaced by the full path of each file that "find" discovers.

→ For example, if "find" finds "./file1.txt"; "{}" will be replaced by "./file1.txt" when executing "cat" command.

6. \;

→ This marks the end of the "-exec" command

→ The backslash (\) escapes the semicolon (;) so that the shell doesn't interpret it as a command terminator but passes it to "find".

[∴ Filter commands over]