

Pranshu Sangwan

VIT REG.NO : 20MIC0146

COURSE : CYBERSECURITY & ETHICAL HACKING(SMARTBRIDGE EXTERNSHIP)

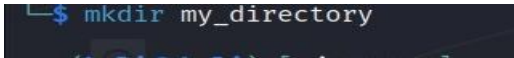
DIGITAL ASSESSMENT – 2 (27 ,28may)

TASK1 . File and directory manipulation

1. Create a directory called "my_directory".


`mkdir my_directory`

`ls`



```
$ mkdir my_directory
```

2. Navigate into the "my_directory".`cd my_directory`



```
$ cd my_directory
```

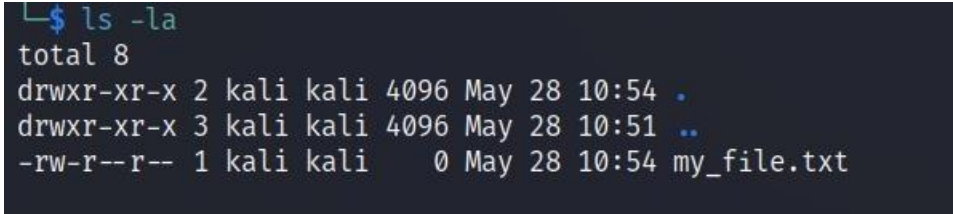
3. Create an empty file called "my_file.txt".`touch my_file.txt`

`ls`



```
$ touch my_file.txt
$ ls
my_file.txt
```

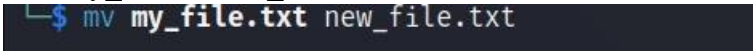
4. List all the files and directories in the current directory.`ls -la`



```
$ ls -la
total 8
drwxr-xr-x 2 kali kali 4096 May 28 10:54 .
drwxr-xr-x 3 kali kali 4096 May 28 10:51 ..
-rw-r--r-- 1 kali kali   0 May 28 10:54 my_file.txt
```

5. Rename "my_file.txt" to "new_file.txt".`mv my_file.txt new_file.txt`

`ls`



```
$ mv my_file.txt new_file.txt
```

6. Display the content of "new_file.txt" using a pager tool of your choice.`more new_file.txt`
(to display content of new_file.txt , I have added random words to it)
7. Append the text "Hello, World!" to "new_file.txt".`echo 'Hello, World!' >> new_file.txt`

```
$ cat new_file.txt
just random letters written by ABHIRUP KONWAR
ADSF
FW
EF
WE
F2

2F
2
F
2F3
Hello, World!
```

8. Create a new directory called "backup" within "my_directory".

ls

mkdir backup

ls

```
(kali㉿kali)-[~/ABHIRUP/my_directory]
$ ls
new_file.txt

(kali㉿kali)-[~/ABHIRUP/my_directory]
$ mkdir backup

(kali㉿kali)-[~/ABHIRUP/my_directory]
$ ls
backup new_file.txt
```

```
$ more new_file.txt
just random letters written by ABHIRUP KONWAR
ADSF
FW
EF
WE
F2

2F
2
F
2F3
```

9. Move "new_file.txt" to the "backup" directory.
mv new_file.txt backup

```
$ mv new_file.txt backup
```

10. Verify that "new_file.txt" is now located in the "backup" directory.

```
ls
cd backup
ls
```

11. Delete the "backup" directory and all its contents.

```
ls
rm -rf backup
ls
```

r : recursive (remove directories and their contents recursively)
f: force (ignore non-existent file , never prompt)

TASK 2 : PERMISSIONS AND SCRIPTING

1. Create a new file called "my_script.sh" touch my_script.sh
ls

```
$ touch my_script.sh
(kali@kali)-[~/ABHIRUP]
$ ls
my_script.sh
```

2. Edit my_script.sh using any text editor , add the given lines, make it executable , and run.
vim my_script.sh #!/bin/bash
echo "Welcome to my script!"
echo "Today's date is \$(date)."

```
$ vim my_script.sh
```

```
File Actions Edit View Help
#!/bin/bash
echo "Welcome to my script!"
echo "Today's date is $(date)."
```

```
File Actions Edit View Help
#!/bin/bash
echo "Welcome to my script!"
echo "Today's date is $(date)."
```

w : save changes made to the file
q : exit Vim

chmod +x my_script.sh
./my_script.sh

TASK 3 : COMMAND EXECUTION AND PIPELINES

1. List all the processes running on your system using the "ps" command. ps
aux

The ps aux command is used to display a detailed list of all running processes on a Linux or Unix system.

```

$ ps aux
USER          PID %CPU %MEM    VSZ   RSS TTY      STAT START   TIME COMMAND
root           1  0.0  0.6 168072 12284 ?        Ss   10:49   0:00 /sbin/init splash
root           2  0.0  0.0      0      0 ?        S    10:49   0:00 [kthreadd]
root           3  0.0  0.0      0      0 ?        I<   10:49   0:00 [rcu_gp]
root           4  0.0  0.0      0      0 ?        I<   10:49   0:00 [rcu_par_gp]
root           5  0.0  0.0      0      0 ?        I<   10:49   0:00 [slub_flushwq]
root           6  0.0  0.0      0      0 ?        I<   10:49   0:00 [netns]
root           8  0.0  0.0      0      0 ?        I<   10:49   0:00 [kworker/0:0H-events_highpri]
root          10  0.0  0.0      0      0 ?        I<   10:49   0:00 [mm_percpu_wq]
root          11  0.0  0.0      0      0 ?        I    10:49   0:00 [rcu_tasks_kthread]
root          12  0.0  0.0      0      0 ?        I    10:49   0:00 [rcu_tasks_rude_kthread]
root          13  0.0  0.0      0      0 ?        I    10:49   0:00 [rcu_tasks_trace_kthread]
root          14  0.0  0.0      0      0 ?        S    10:49   0:00 [ksoftirqd/0]
root          15  0.0  0.0      0      0 ?        I    10:49   0:01 [rcu_preempt]
root          16  0.0  0.0      0      0 ?        S    10:49   0:00 [migration/0]
root          18  0.0  0.0      0      0 ?        S    10:49   0:00 [cpuhp/0]
root          19  0.0  0.0      0      0 ?        S    10:49   0:00 [cpuhp/1]
root          20  0.0  0.0      0      0 ?        S    10:49   0:00 [migration/1]
root          21  0.0  0.0      0      0 ?        S    10:49   0:00 [ksoftirqd/1]
root          23  0.0  0.0      0      0 ?        I<   10:49   0:00 [kworker/1:0H-events_highpri]

```

2. Use the "grep" command to filter the processes list and display only the processes with "bash" in their name. `ps aux | grep bash`
(grep is used for matching a pattern or string)

```

$ ps aux | grep bash
kali 23094 0.0 0.1 6332 2132 pts/0 S+ 11:31 0:00 grep --color=auto bash

```

3. Use the "wc" command to count the number of lines in the filtered output.
`ps aux | grep bash | wc -l`

```

$ ps aux | grep bash | wc -l
1

```