

Assignment -2: Bash Shell Basics



VIT[®]
Vellore Institute of Technology
(Deemed to be University under section 3 of UGC Act, 1956)



SMARTBRIDGE
Let's Bridge the Gap

Name: Gokul R

Reg. No.: 20MIS0332

Course: Cyber Security and Ethical Hacking

Campus: VIT Vellore

Task 1: File and Directory Manipulation

1. Create a directory called "my_directory".

Mkdir my_directory

2. Navigate into the "my_directory".

Cd mkdir

```
gokul@kali: ~/my_directory
(gokul@kali)~$ mkdir my_directory
(gokul@kali)~$ ls my_directory
(gokul@kali)~$ touch my_file.txt
(gokul@kali)~$ ls
backup Desktop Documents Downloads Music my_directory my_file.txt Pictures Public Templates Videos
(gokul@kali)~$ cd my_directory
(gokul@kali)~/my_directory$ touch my_file.txt
(gokul@kali)~/my_directory$ ls
my_file.txt
(gokul@kali)~/my_directory$
```

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

Touch my_file.txt

4. List all the files and directories in the current directory.

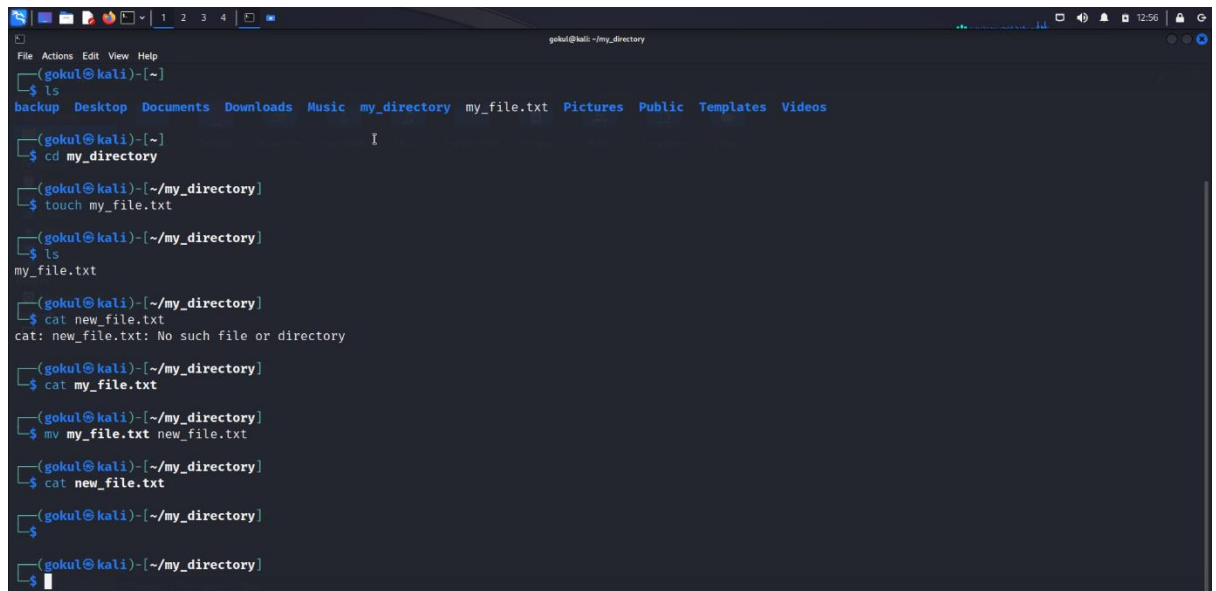
ls

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

Mv my_file.txt new_file.txt

6. Display the content of "new_file.txt" using a pager tool of your choice.

Cat new_file.txt



```
gokul@kali: ~/my_directory
$ ls
backup Desktop Documents Downloads Music my_directory my_file.txt Pictures Public Templates Videos

(gokul@kali)~$ cd my_directory

(gokul@kali)~/my_directory$ touch my_file.txt

(gokul@kali)~/my_directory$ ls
my_file.txt

(gokul@kali)~/my_directory$ cat new_file.txt
cat: new_file.txt: No such file or directory

(gokul@kali)~/my_directory$ cat my_file.txt

(gokul@kali)~/my_directory$ mv my_file.txt new_file.txt

(gokul@kali)~/my_directory$ cat new_file.txt

(gokul@kali)~/my_directory$

(gokul@kali)~/my_directory$
```

7. Append the text "Hello, World!" to "new_file.txt".

Nano new_file.txt

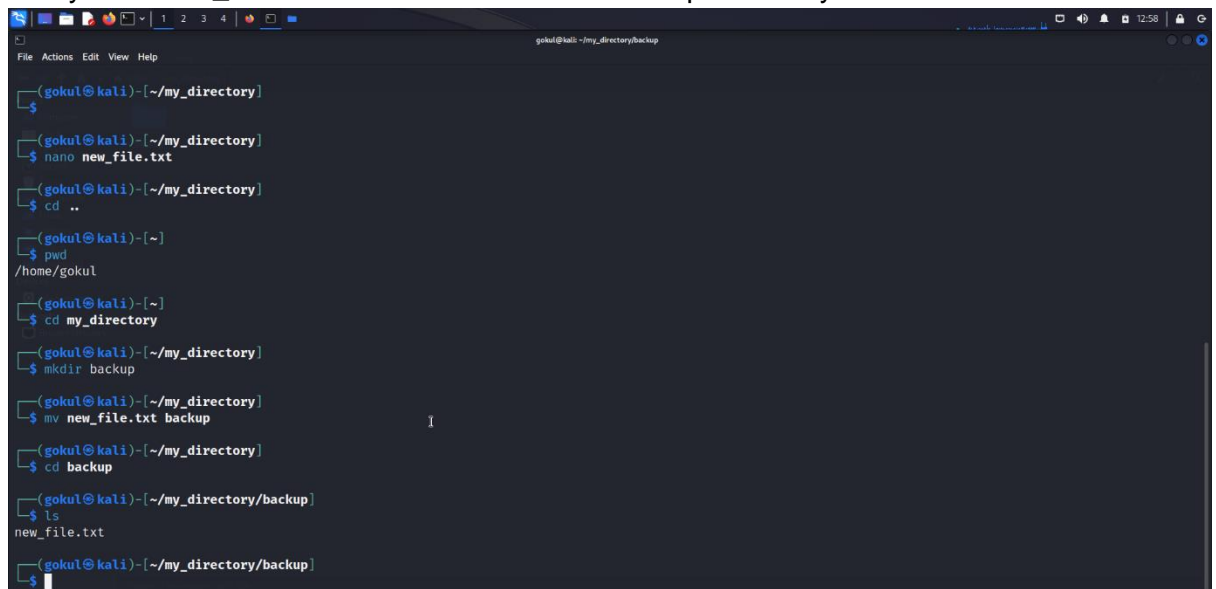
8. Create a new directory called "backup" within "my_directory".

Mkdir backup

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

Mv new_file.txt backup

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



```
gokul@kali: ~/my_directory/backup

(gokul@kali)~/my_directory$ nano new_file.txt

(gokul@kali)~/my_directory$ cd ..

(gokul@kali)~$ pwd
/home/gokul

(gokul@kali)~$ cd my_directory

(gokul@kali)~/my_directory$ mkdir backup

(gokul@kali)~/my_directory$ mv new_file.txt backup

(gokul@kali)~/my_directory$ cd backup

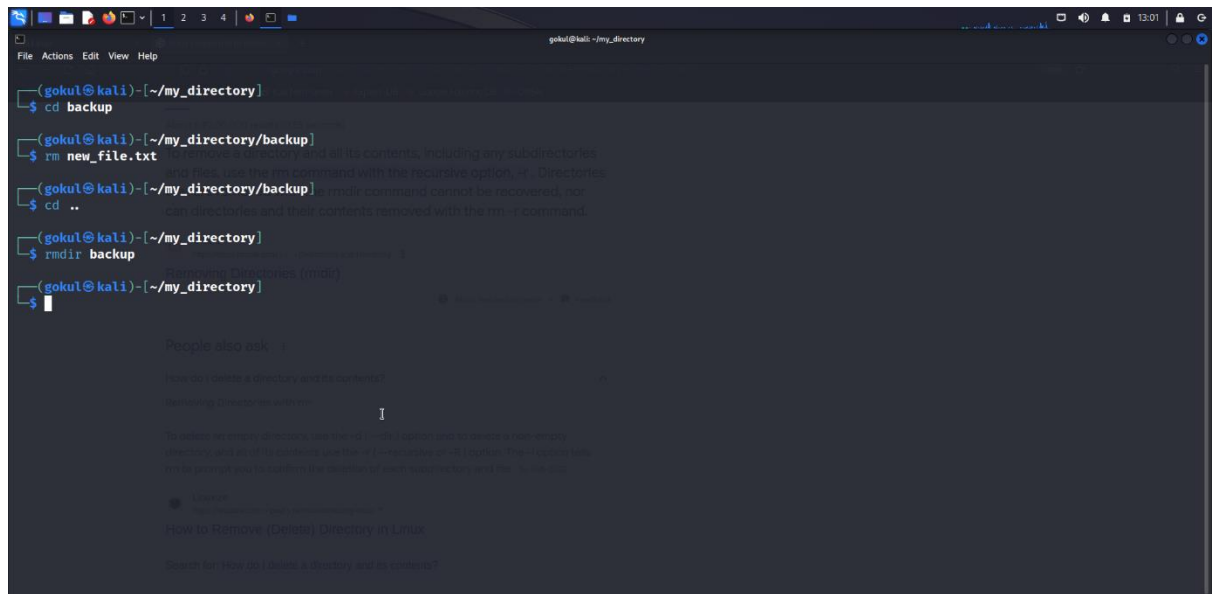
(gokul@kali)~/my_directory/backup$ ls
new_file.txt

(gokul@kali)~/my_directory/backup$
```

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

Rm new_file.txt

Rmdir backup



```
(gokul@kali)~[/my_directory]
$ cd backup
(gokul@kali)~[/my_directory/backup]
$ rm new_file.txt
(gokul@kali)~[/my_directory/backup]
$ cd ..
(gokul@kali)~[/my_directory]
$ rmdir backup
(gokul@kali)~[/my_directory]
$
```

The screenshot shows a terminal window with the following commands and output:

- `(gokul@kali)~[/my_directory]`
- `$ cd backup`
- `(gokul@kali)~[/my_directory/backup]`
- `$ rm new_file.txt`
- `(gokul@kali)~[/my_directory/backup]`
- `$ cd ..`
- `(gokul@kali)~[/my_directory]`
- `$ rmdir backup`
- `(gokul@kali)~[/my_directory]`
- `$`

Below the terminal output, there is a section titled "People also ask" with a link to "How to Remove (Delete) Directory in Linux".

Task 2: Permissions and Scripting

- Create a new file called "my_script.sh".
- Edit "my_script.sh" using a text editor of your choice and add the following lines:

bash

#!/bin/bash

echo "Welcome to my script!"

echo "Today's date is \$(date)."

Save and exit the file.

➔ **touch my_script.sh**

➔ **nano my_script.sh**

- Make "my_script.sh" executable.

➔ **Chmod u+x my_script.exe**

```
gokul@kali ~  
$ touch my_script.sh  
$ nano my_script.sh  
#  
$ cat my_script.sh  
#!/bin/bash  
echo "Welcome to my script"  
echo "Today's date is $date "  
$ chmod u+x my_script.sh
```

- Run "my_script.sh" and verify that the output matches the expected result.

➔ **Sh my_script.sh**

```
gokul@kali ~  
$ ./my_script.sh  
zsh: ./my_script.sh: bad interpreter: bin/bash: no such file or directory  
$ cat my_script.sh  
#!/bin/bash  
echo "Welcome to my script"  
echo "Today's date is $date "  
$ sh my_script.sh  
Welcome to my script  
Today's date is  
$ nano my_script.sh  
$ sh my_script.sh  
Welcome to my script  
Today's date is Mon May 29 12:04:45 AM EDT 2023  
$
```

Task 3: Command Execution and Pipelines

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

Ps -ef

```
(gokul@kali)-[~]
$ ps -ef
UID          PID    PPID  C  STIME TTY          TIME CMD
root         1      0  0  May28 ?        00:00:01 /sbin/init splash
root         2      0  0  May28 ?        00:00:00 [kthreadd]
root         3      0  0  May28 ?        00:00:00 [rcu_gp]
root         4      2  0  May28 ?        00:00:00 [rcu_par_gp]
root         5      2  0  May28 ?        00:00:00 [slub_flushwq]
root         6      2  0  May28 ?        00:00:00 [netns]
root        10      2  0  May28 ?        00:00:00 [mm_percpu_wq]
root        11      2  0  May28 ?        00:00:00 [rcu_tasks_kthread]
root        12      2  0  May28 ?        00:00:00 [rcu_tasks_rude_kthread]
root        13      2  0  May28 ?        00:00:00 [rcu_tasks_trace_kthread]
root        14      2  0  May28 ?        00:00:00 [ksoftirqd/0]
root        15      2  0  May28 ?        00:00:00 [rcu_preempt]
root        16      2  0  May28 ?        00:00:00 [migration/0]
root        18      2  0  May28 ?        00:00:00 [cpuhp/0]
root        19      2  0  May28 ?        00:00:00 [cpuhp/1]
root        20      2  0  May28 ?        00:00:00 [migration/1]
root        21      2  0  May28 ?        00:00:00 [ksoftirqd/1]
root        23      2  0  May28 ?        00:00:00 [kworker/1:0H-events_highpri]
root        24      2  0  May28 ?        00:00:00 [cpuhp/2]
root        25      2  0  May28 ?        00:00:00 [migration/2]
root        26      2  0  May28 ?        00:00:00 [ksoftirqd/2]
root        28      2  0  May28 ?        00:00:00 [kworker/2:0H-events_highpri]
root        30      2  0  May28 ?        00:00:00 [kworker/u6:1-events_unbound]
```

- Use the "grep" command to filter the processes list and display only the processes with "bash" in their name.

Ps -ef | grep bash

Use the "wc" command to count the number of lines in the filtered output

Ps -ef | grep bash | wc -l

```
(gokul@kali)-[~]
$ ps -ef
gokul        2887      741  0 00:01 ?        00:00:00 /usr/libexec/xdg-desktop-portal-gtk
root         4649      2  0 00:03 ?        00:00:00 [kworker/2:0-ata_sff]
root         5389      2  0 00:04 ?        00:00:00 [kworker/1:0-cgwb_release]
root         5390      2  0 00:04 ?        00:00:00 [kworker/0:0-events]
gokul        5483     919  0 00:05 ?        00:00:00 /usr/libexec/gvfsd-network --spawner :1.15 /org/gtk/gvfs/exec_spaw/1
gokul        5512     919  0 00:05 ?        00:00:00 /usr/libexec/gvfsd-dnssd --spawner :1.15 /org/gtk/gvfs/exec_spaw/3
gokul        6436     952  2 00:06 ?        00:00:07 /usr/lib/firefox-esr/firefox-esr
gokul        6503     6436  0 00:06 ?        00:00:00 /usr/lib/firefox-esr/firefox-esr -contentproc -parentBuildID 20230214011352 -prefsLen
gokul        6542     6436  0 00:06 ?        00:00:00 /usr/lib/firefox-esr/firefox-esr -contentproc -childID 1 -isForBrowser -prefsLen 30
gokul        6577     6436  0 00:06 ?        00:00:00 /usr/lib/firefox-esr/firefox-esr -contentproc -childID 2 -isForBrowser -prefsLen 34
gokul        6611     6436  0 00:06 ?        00:00:00 /usr/lib/firefox-esr/firefox-esr -contentproc -childID 3 -isForBrowser -prefsLen 34
gokul        6661     6436  0 00:06 ?        00:00:00 /usr/lib/firefox-esr/firefox-esr -contentproc -childID 4 -isForBrowser -prefsLen 35
gokul        6665     6436  0 00:06 ?        00:00:00 /usr/lib/firefox-esr/firefox-esr -contentproc -childID 5 -isForBrowser -prefsLen 35
gokul        6704     6436  0 00:06 ?        00:00:00 /usr/lib/firefox-esr/firefox-esr -contentproc -childID 6 -isForBrowser -prefsLen 35
root         7748      2  0 00:08 ?        00:00:00 [kworker/2:1-ata_sff]
gokul        9547    1386  99 00:12 pts/0    00:00:00 ps -ef

(gokul@kali)-[~]
$ ps -ef | grep bash
gokul        9857    1386  0 00:13 pts/0    00:00:00 grep --color=auto bash

(gokul@kali)-[~]
$ ps -ef | grep bash | wc -l
1

(gokul@kali)-[~]
$
```