Linux Management Permissions

Ana Cifuentes Romero

DW1E – 18/01/2022

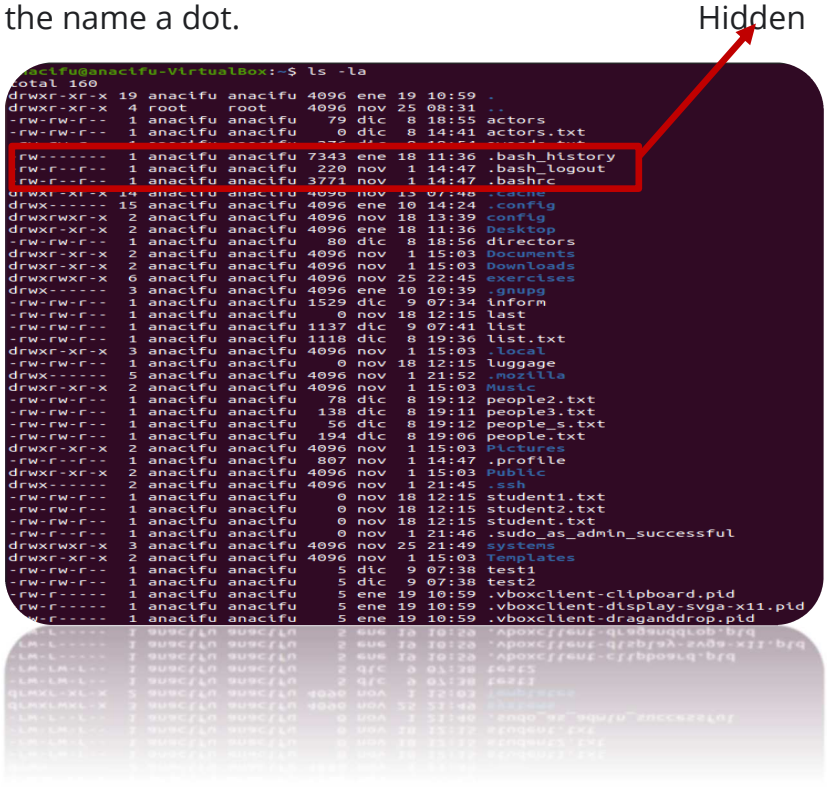
Exercises about file and directory permission

1.List the permissions in your current directory, including hidden files.

$ ls -la

The simple command of **ls -l** means, **to list files and directories**. It has an option of **-l**, which lists the contents in a long format like the picture below. It allows you to look through the file system.

-a: which list the hidden files. Hidden files and directories are those that bear in front of the name a dot. Hidden



2. Create a file called perm1. Now, check the default permissions and user and group
Ownership.

$ touch perm1
$ ls -l perm1

The **touch** command is used to create an empty file and using **ls-l perm1** we can see this file created and its permissions.

```
anacifu@anacifu-VirtualBox:~$ touch perm1
anacifu@anacifu-VirtualBox:~$ ls -l perm1
-rw-rw-r-- 1 anacifu anacifu 0 ene 19 11:37 perm1
```

```
anacifu@anacifu-VirtualBox:~$ find perm1
perm1
```

3. Change permissions of perm1 so that everyone can read and only the owner user can write.
Specify the command in all possible ways.

The **chmod** command accepts options in two forms: **symbolic and octal**.

SIMBOLIC: Permissions may be specified symbolically, using the symbols u (user), g (group), o (other), a (all), r (read), w (write), x (execute), + (add permission), - (take away permission) and = (assign permission).

OCTAL: Permissions may also be specified as a sequence of 3 octal digits. Each octal digit represents the access permissions for the user/owner, group and others respectively. The mappings of permissions onto their corresponding octal digits where 0 means no permissions and 7 all permissions.

$ chmod u=rw,go=r perm1

This means that user (u) can read(r) and write (w), groups and others only can read (r) in the perm1 file.

```
anacifu@anacifu-VirtualBox:~$ chmod u=rw,go=r perm1
anacifu@anacifu-VirtualBox:~$ ls -l perm1
-rw-r--r-- 1 anacifu anacifu 0 ene 19 11:37 perm1
```

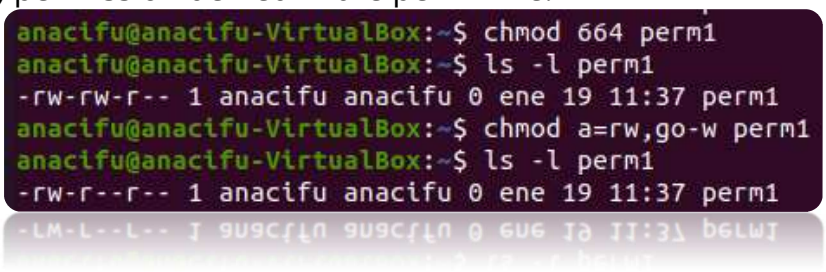
$ chmod a=r,u+w perm1

This means that all (a) can read and user can also write in the perm1 file.

```
anacifu@anacifu-VirtualBox:~$ chmod 664 perm1
anacifu@anacifu-VirtualBox:~$ ls -l perm1
-rw-rw-r-- 1 anacifu anacifu 0 ene 19 11:37 perm1
anacifu@anacifu-VirtualBox:~$ chmod a=r,u+w perm1
anacifu@anacifu-VirtualBox:~$ ls -l perm1
-rw-r--r-- 1 anacifu anacifu 0 ene 19 11:37 perm1
```

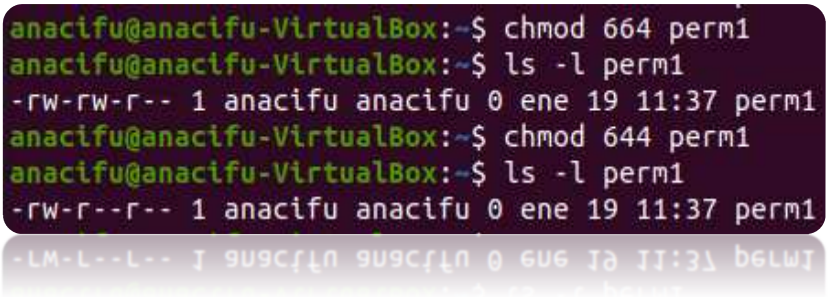
$ chmod a=rw,go-w perm1

This means that all (a) can read (r) and write (w) and groups and others have written (w) permission denied in the perm1 file.



$ chmod 644 perm1

This means that user can read and write, groups can only read like others in the perm1 file.



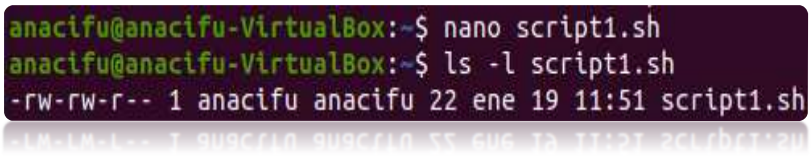
4. Create a file called script1.sh, including the content below. List the default permissions.
#!/bin/bash
clear
who

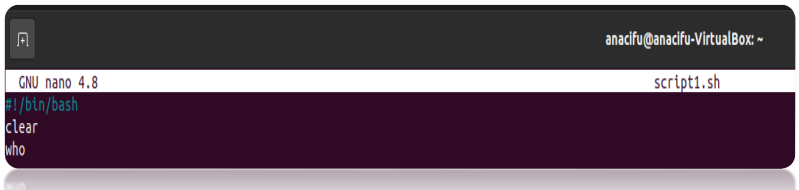
$ nano script1.sh

.sh: A file with .sh extensión is a scripting language commands file that contains computer program to be run by Unix shell. It can contain a series of commands that run sequentially to carry out operations such as files processing, execution of programs and other such tasks.

The **nano** command is a simple text editor. This create and open the file and you can add content inside.

#!: its function is to indicate to the system that a set of instructions will then be presented and thus processed. The second part, /bin/bash, indicates the shell that will be used to execute the commands.
-rw-r—r--: is the default configuration when the file is created.

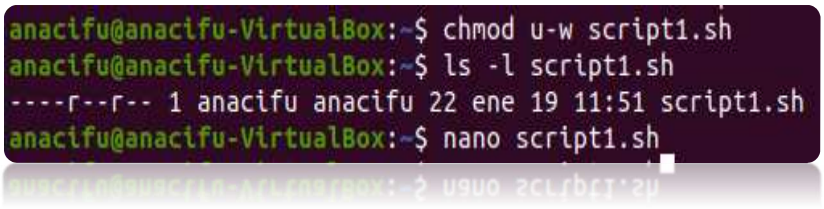


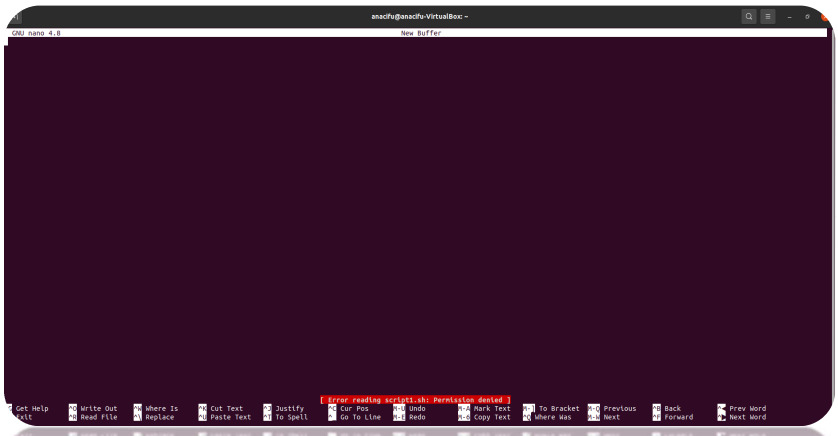


5. Remove the read permission from the owner and try to open the file.

$ chmod u-r script1.sh

This means that user have read permission denied and for that reason, you cannot open the file.



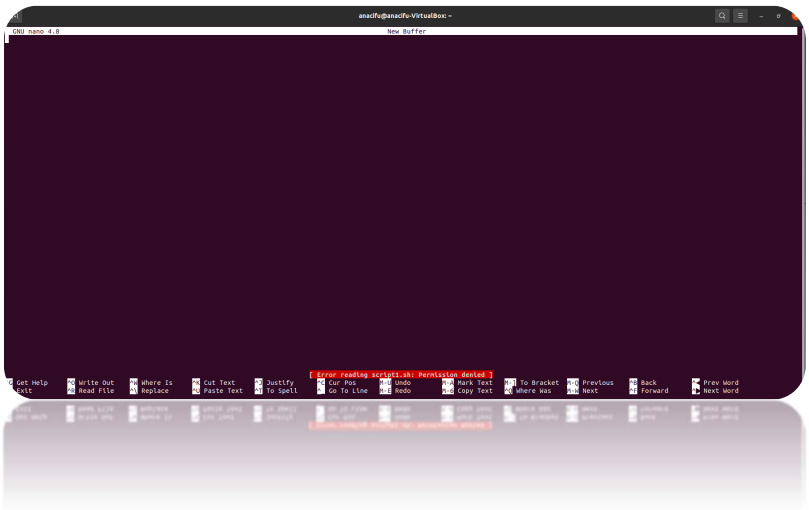


6. Remove the write permission from the owner on the file script1.sh. Add the line below. Is it possible? Why?
new line

$ chmod u-w script1.sh

It is not possible to add anything in this file because you do not even have the possibility to read the text, the file is not accessible.

```
student@student-VirtualBox:~$ sudo chmod u-w script1.sh
student@student-VirtualBox:~$ nano script1.sh
```



7. Change the permissions on the file script1.sh so that the owner can read, write and execute, but you deny all the permissions from the group and others.

$ chmod 700 script1.sh

The file color is displayed in green because you have permission to run it.

```
anacifu@anacifu-VirtualBox:~$ chmod 700 script1.sh
anacifu@anacifu-VirtualBox:~$ ls -l script1.sh
-rwx------ 1 anacifu anacifu 22 ene 19 11:51 script1.sh
```

8. Add the line indicated in exercise 6, in case it was not possible. Try to run the file like a command.

$ nano script1.sh

$ ./script1.sh -> This is one of the ways to run a file like a command from the current directory. In this case it is not possible.

```
anacifu@anacifu-VirtualBox:~$ cat script1.sh
#!/bin/bash
clear
who
new line
```

```
anacifu@anacifu-VirtualBox:~$ ./script1.sh
/bin/bash: ./script1.sh: Permission denied
```

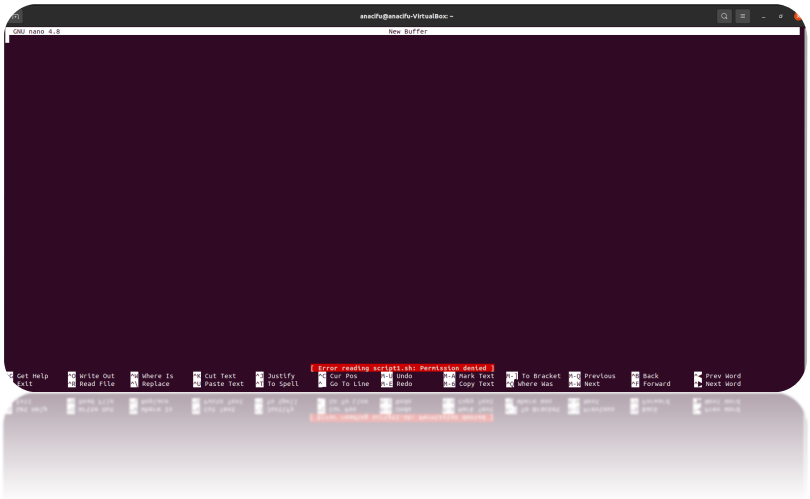
$ bash script1.sh -> This is another option to run the file. In this case it is not possible.

```
anacifu@anacifu-VirtualBox:~$ bash script1.sh
bash: script1.sh: Permission denied
```

9. Remove the read permission from the owner on the file script1.sh. Try to run the file. Is it possible?

$ chmod u-r script1.sh -> This is one way to deny read permission in one file.
$ ls -l script1.sh -> This permit to see the permissions.

```
anacifu@anacifu-VirtualBox:~$ chmod u-r script1.sh
anacifu@anacifu-VirtualBox:~$ ls -l script1.sh
--wx------ 1 anacifu anacifu 31 ene 19 12:08 script1.sh
anacifu@anacifu-VirtualBox:~$ nano script1.sh
```



10. Create a directory called “systems”. Remove the write permission from it and try to copy script1.sh inside.

$ mkdir systems1 -> The mkdir command is used to create a directory.
$ chmod u-w systems1 -> This is used to remove the user’s write permission
$ cp script1.sh systems1 -> The **cp** command is used to copy a file or directory.
In this case we do not have read permission so, it is not possible to copy.

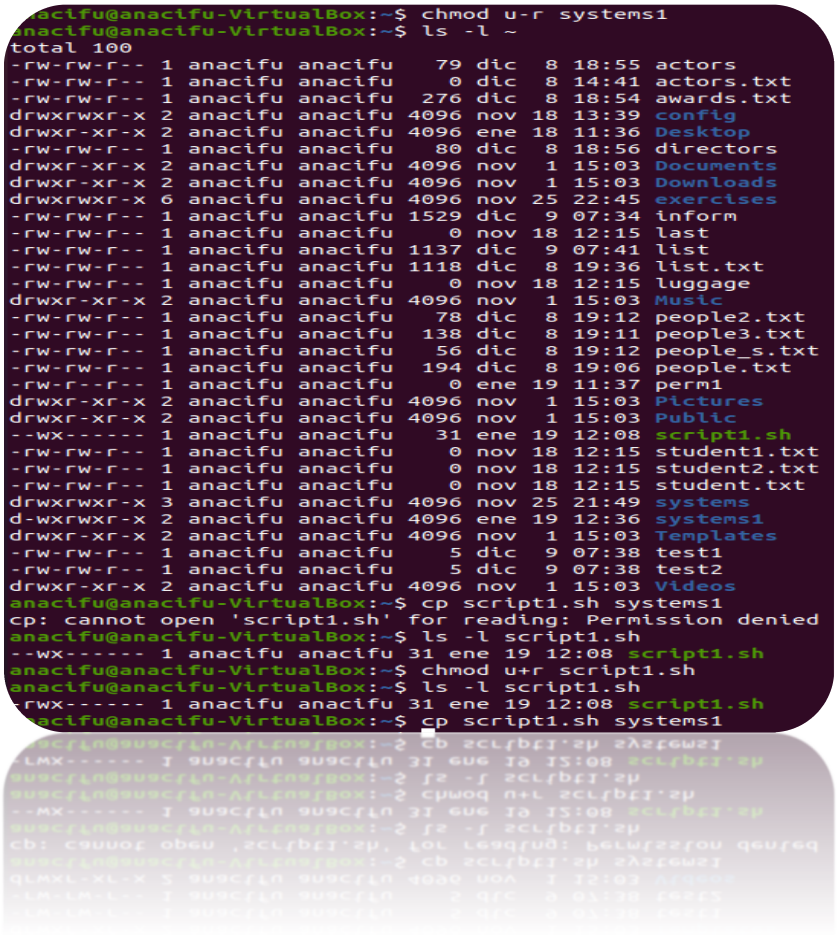
```
anacifu@anacifu-VirtualBox:~$ mkdir systems
mkdir: cannot create directory ‘systems’: File exists
anacifu@anacifu-VirtualBox:~$ mkdir systems1
anacifu@anacifu-VirtualBox:~$ chmod u-r systems1
anacifu@anacifu-VirtualBox:~$ ls -l systems1
ls: cannot open directory 'systems1': Permission denied
anacifu@anacifu-VirtualBox:~$ cp script1.sh systems1
cp: cannot open 'script1.sh' for reading: Permission denied
```

I made a mistake when I wrote the command so I correct it.

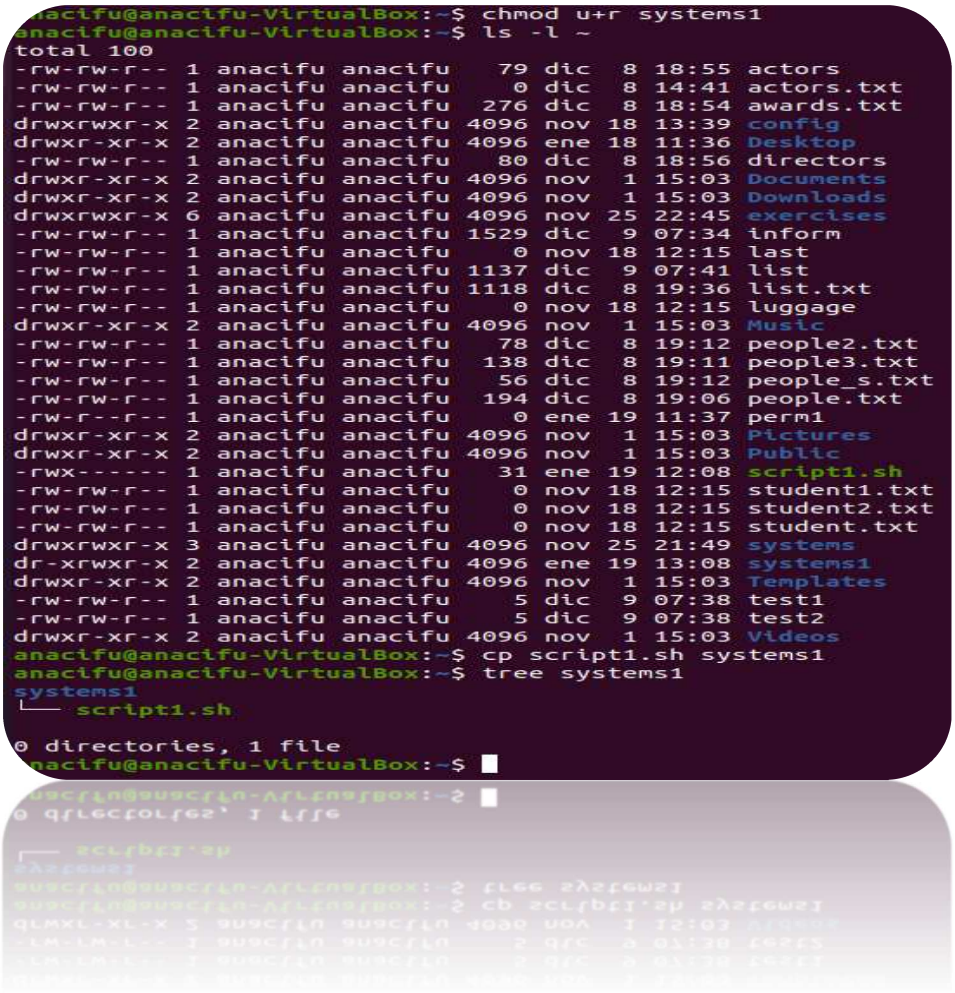
```
anacifu@anacifu-VirtualBox:~$ chmod u-w systems1
anacifu@anacifu-VirtualBox:~$ cp script1.sh systems1
```

11. If you were not able to copy the file, add the write permission again and copy the file inside.

$ chmod u-w systems1 -> This is used to deny write permission in a file or directory.
$ cp script1.sh systems1 -> The **cp** command is used to copy a file or directory.
I am not able to copy the file because I made a mistake when I wrote the command and now it is fixed.



This is the result after correction:

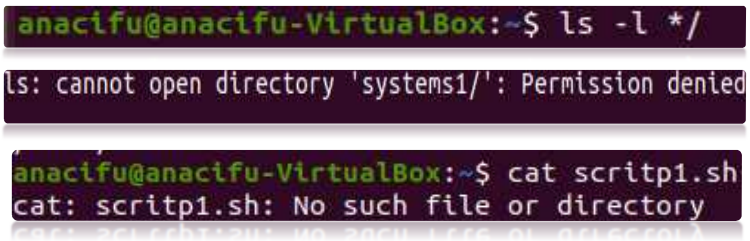


12. Remove the read permission from the user on the directory “systems” and try to list its contents.

$ chmod u-r systems1

$ cat systems1 -> The **cat** command is commonly used to view the contents of a file, to redirect, create and concatenate files. The access in this case is denied.

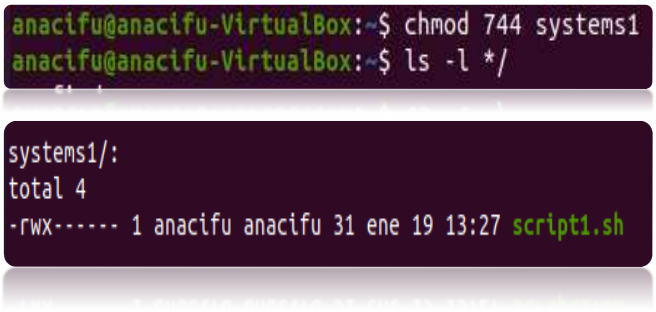
$ ls -l */ -> This is a command to list all the directories. The access in this case is denied.



13. Change the permissions from “systems” so that the owner can read, write and execute, but the group and others can only read.

$ chmod 744 systems1

$ ls -l */ -> This is a command to list all the directories. The access is now authorized.

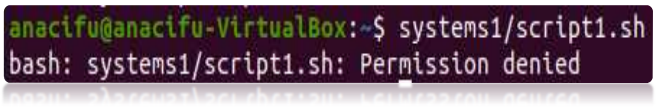


14. Remove the execute permission from "systems". Can you execute systems/script1.sh? Is it possible to access the directory to execute the file?

$ chmod u-x systems1

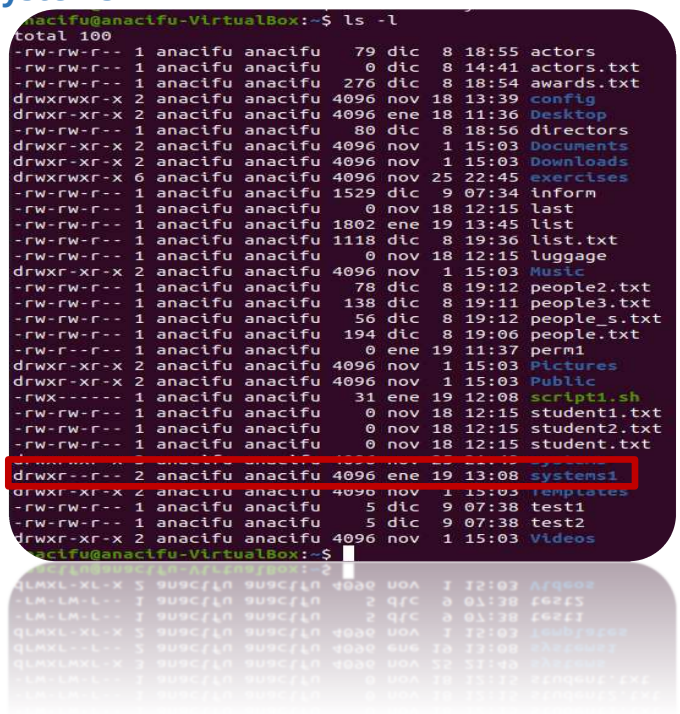
$ systems1/script1.sh

It is not possible to execute the file



15. Assign the execute permission to the directory again

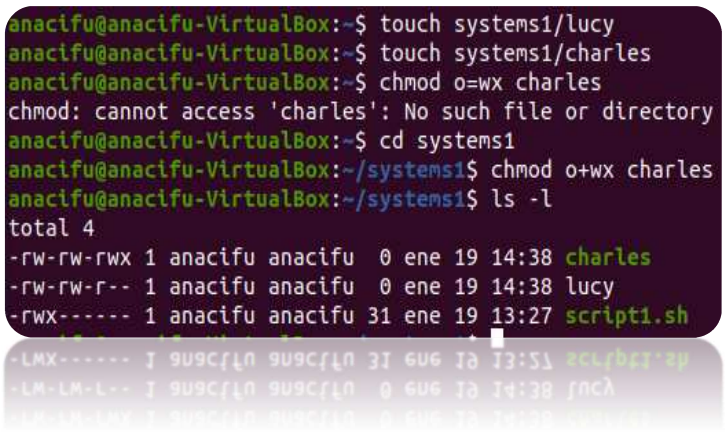
$ chmod u+x systems1



16. Create two files called “lucy” and “charles” into “systems”. Change permissions of “charles”, so that others can write and execute.

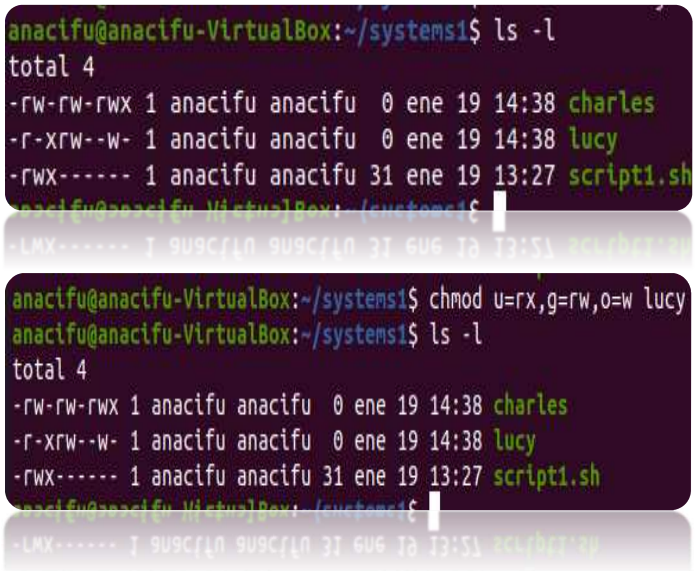
$ touch systems1/lucy
$ touch systems1/Charles
$ cd systems1
$ chmod o+wx charles

We use the **touch** command to create empty files inside systems1. Then, you must access inside systems1 directory **(cd systems1)** in order to change the permissions of Charles file.



17. Change permissions of “lucy” so that the owner can read and execute, the group can read and write and others can only write. Specify the command in all possible ways.

$chmod u+rw,g+rw,o+w lucy
$chmod 562 lucy



Linux Management Permissions 2

18. Log in as root. Change the ownership of “charles” to “root”. Exit the root session. Now, try to change the permission so that others cannot read and execute. Is it possible? Why?
19. Change the permissions of “charles” so that everybody can do everything
20. Change the permissions of “lucy” so that the group can read and write, but the owner and others cannot do anything. Can you open the file?
21. Create a group called “newgroup”. Set the group as the owner of the file “lucy” and “root” as the owner user.
22. Add your user to the secondary group “newgroup”. Try to open the file “lucy” now. Is it possible?
23. Change permissions of “lucy” so that everybody can read.
24. Do exercise 13 again, but this time granting permissions to the folder “systems” including
files and subfolders
25. Change the group owner of “systems” to “root” including files and subfolders