

Overview of Computer Workshop

Unit-3, Lecture – 3

File Access rights

- Linux is a multi-user operating system which can be accessed by many users simultaneously.
- Linux can also be used in mainframes and servers without any modifications.
- But this raises security concerns as an unsolicited or malign user can corrupt, change or remove crucial data.
- For effective security, Linux divides authorization into 2 levels.
 - Ownership
 - Permission

Ownership of files

- Every file and directory on your Linux system is assigned 3 types of owner, given below

- **User:** A user is the owner of the file. By default, the person who created a file becomes its owner. Hence, a user is also sometimes called an owner.
- **Group:** A user- group can contain multiple users. All users belonging to a group will have the same access permissions to the file.

Suppose you have a project where a number of people require access to a file, instead of manually assigning permissions to each user, you could add all users to a group, and assign group permission to file such that only this group members and no one else can read or modify the files.

- **Other:** Any other users who has access to a file. This person has neither created the file nor he belongs to the user group.

Permission or access rights

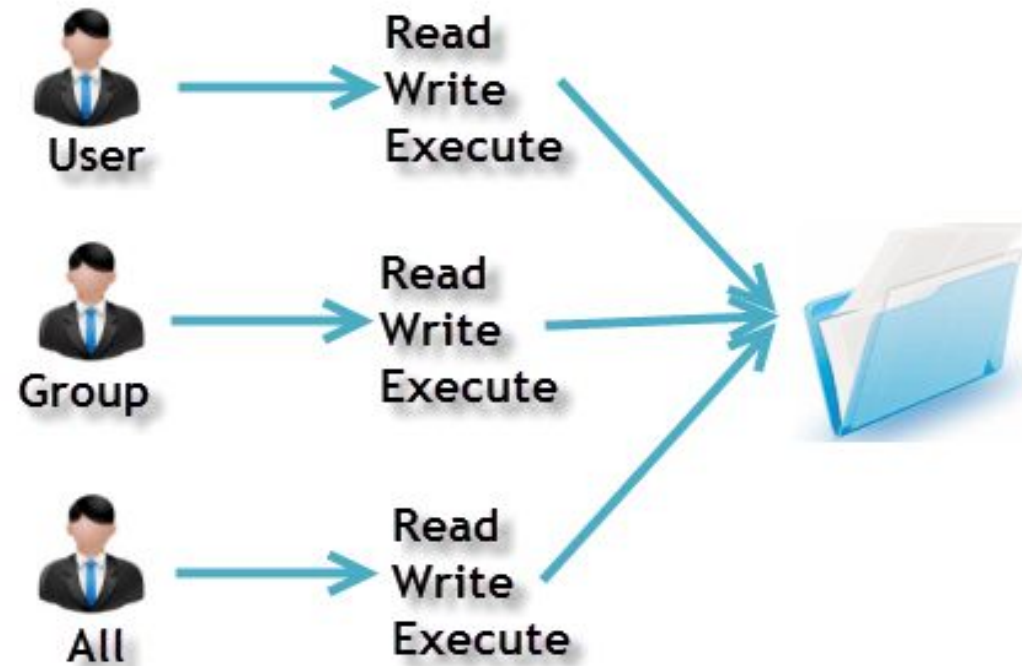
- Every file and directory in your Linux system has following 3 permission defined for all the 3 owners discussed above.
 - **Read:** This permission gives you the authority to open and read a file. Read permission on a directory gives you the ability to list its content.
 - **Write:** The write permission gives you the authority to modify the contents of a file. The write permission on a directory gives you the authority to add, remove and rename files stored in the directory.

Consider a scenario where you have to write permission on file but do not have write permission on the directory where the file is stored. You will be able to modify the file contents. But you will not be able to rename, move or remove the file from the directory.

- **Execute:** In Windows, an executable program usually has an extension ".exe" and which you can easily run. In Unix/Linux, you cannot run a program unless the execute permission is set. If the execute permission is not set, you might still be able to see/modify the program code (provided read & write permissions are set), but not run it.

File Permissions in Linux

Owners assigned Permission On Every File and Directory



- Let's see file permissions in Linux with examples:

`ls -l` on terminal gives

File type and Access Permissions.

```
home@VirtualBox: ~  
home@VirtualBox:~$ ls -l  
-rw-rw-r-- 1 home home    0 2012-08-30 19:06 My File
```

- Here, the first '-' implies that we have selected a file

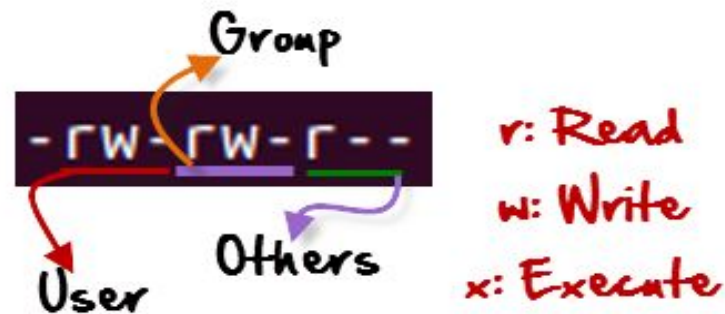
-rw-rw-r--
↓
indicates
file

- Else, if it were a directory, `d` would have been shown.

d represents directory

```
drwxr-xr-x 2 ubuntu ubuntu 80 Sep  6 07:27 Desktop
```

- The first part of the code is 'rw-'. This suggests that the owner 'Home' can:
 - Read the file
 - Write or edit the file
 - He cannot execute the file since the execute bit is set to '-'.
- The second part is 'rw-'. It for the user group 'Home' and group-members can:
 - Read the file
 - Write or edit the file
- The third part is for the world which means any user. It says 'r--'. This means the user can only:
 - Read the file



Command "chmod"

- It is used to change the access mode of a file. **chmod** stands for 'change mode'.
- Using the command, we can set permissions (read, write, execute) on a file/directory for the owner, group and the other users.

Syntax: `chmod permissions filename`

- There are 2 ways to use the command -
 - Absolute mode
 - Symbolic mode

Absolute Mode

- In this mode, file permissions are not represented as characters but as a three-digit octal number. The numeric code for different type of permissions is represented as in the table:

Number	Permission Type	Symbol
0	No Permission	---
1	Execute	--X
2	Write	-W-
3	Execute + Write	-WX
4	Read	r--
5	Read + Execute	r-X
6	Read + Write	rW-
7	Read + Write + Execute	rWX

■ Example:

Checking Current File Permissions

```
ubuntu@ubuntu:~$ ls -l sample
-rw-rw-r-- 1 ubuntu ubuntu 15 Sep  6 08:00 sample
```

chmod 764 and checking permissions again

```
ubuntu@ubuntu:~$ chmod 764 sample
ubuntu@ubuntu:~$ ls -l sample
-rwxrw-r-- 1 ubuntu ubuntu 15 Sep  6 08:00 sample
```

■ '764' absolute code says the following:

- Owner can read, write and execute
- Usergroup can read and write
- World can only read This is shown as '-rwxrw-r--'

■ This is shown as '-rwxrw-r--'

Symbolic Mode

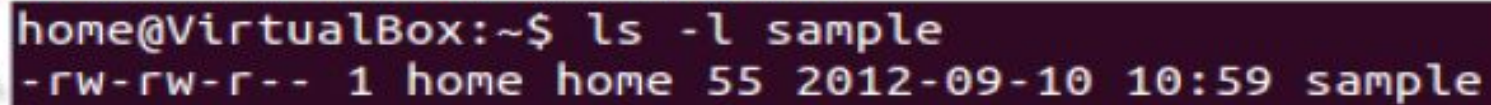
- In the symbolic mode, one can modify permissions of a specific owner. It makes use of mathematical symbols to modify the file permissions.

Operator	Description
+	Adds a permission to a file or directory
-	Removes the permission
=	Sets the permission and overrides the permissions set earlier.

User Denotations	
U	user/owner
G	Group
O	Other
A	All

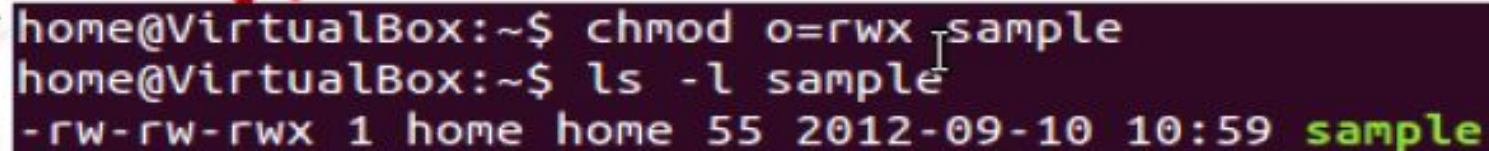
Example:

Current File Permissions



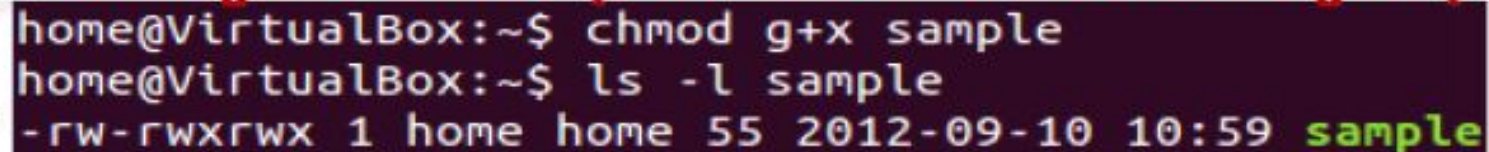
```
home@VirtualBox:~$ ls -l sample
-rw-rw-r-- 1 home home 55 2012-09-10 10:59 sample
```

Setting permissions to the 'other' users



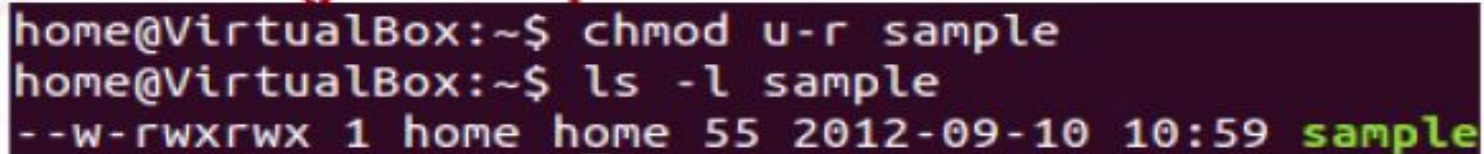
```
home@VirtualBox:~$ chmod o=rwx sample
home@VirtualBox:~$ ls -l sample
-rw-rw-rwx 1 home home 55 2012-09-10 10:59 sample
```

Adding 'execute' permission to the usergroup



```
home@VirtualBox:~$ chmod g+x sample
home@VirtualBox:~$ ls -l sample
-rw-rwxrwx 1 home home 55 2012-09-10 10:59 sample
```

Removing 'read' permission for 'user'



```
home@VirtualBox:~$ chmod u-r sample
home@VirtualBox:~$ ls -l sample
--w-rwxrwx 1 home home 55 2012-09-10 10:59 sample
```

Assignment -1 (Additional Questions)

16. Write a command to give write permission to all the users of file reginfo.
17. Write a command to discard write permission from group users group users of file reginfo.
18. Write the command to set rwx permissions for all the users of file reginfo.