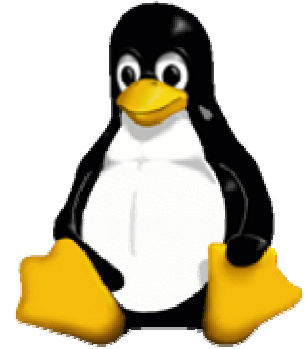


OPERATING SYSTEMS



Božo Krstajić, PhD, University of Montenegro Podgorica

bozok@cg.ac.yu

File permissions and ownership

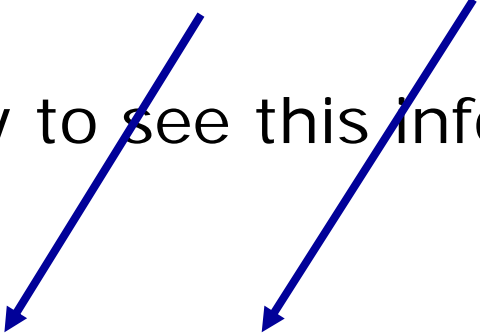
Linux is a multi-user operating system. Every aspect of the system is multi-user, even the filesystem.

The system stores information like who owns a file and who can read it, ...

The filesystem stores ownership information for each file and directory on the system.

This includes what **owner** and **group own** a particular file.

The easiest way to see this information is with the **ls -l** command



```
-rw-r--r-- 1 st21 users 9253 Dec 23 19:12 sample.ttt
```



File permissions

Moreover, permissions are the other important part of the multi-user aspects of the filesystem. With these, you can change who can read, write, and execute files.

As you might have noticed, file permissions can be shown with the **ls -l** command:

```
$ ls -l sample.txt
```

```
-rw-r--r-- 1 st21 users 9253 Dec 23 19:12 sample.txt
```



The permissions are shown in the first column.



File permissions

Permissions that can be set are:

- read (**r**),
- write (**w**),
- execute (**x**).

These permissions can we set for three classes:

- owner/user (**u**),
 - group (**g**),
 - others (**o**).
- or all (**a**)



File permissions

The permissions are visible in the second to ninth character in the first column.

These nine characters are divided in three group;

1. the first three characters represent the permissions for the owner (u ser),
2. the next three characters represent the permissions for the group,
3. finally the last three characters represent the permissions for other users.

- **rw-** **r--** **r--** 1 st21 users 9253 Dec 23 19:12 sample.ttt

The example file can be written(w) to by the owner and can be read(r) by all three classes of users.



File permissions

Each GNU/Linux system has many distinct users, and a user can be a member of certain groups.

This kind of user management provides makes it possible to manage detailed permissions for each file.

In the example shown above **st21** is the owner of the file **sample.ttt** and group permissions apply to the group **users**.

In this example groups rights do not differ from the rights of other users.



File permissions

chown

The `chown` command is used to set the file owner and to which group group permissions should apply to.

You must be logged as **root** if you want to change file ownership or group.

Suppose we want to make **st5** the owner of the file `sample..tst`, this can be done with the `chown`

```
$ chown st5 sample.tst
```

-R

The group can be changed by adding a dot after the owner, followed by the name of the group (**st5.users**).



File permissions

chmod

File permissions can be manipulated using the chmod command. *You must be a file owner if you want to change permission (or root).*

The most basic syntax of chmod is:

chmod [u,g,o,a][+/-][r,w,x] filename

Parameters determine the following elements:

1. which classes this manipulation permission applies to (u, g, o, a),
2. if the permissions should be added (+) or removed (-), and
3. which permissions should be manipulated (r,w,x).



File permissions

Suppose we want to make the file *memo* writable for the owner of the file and the groups for which the group permissions apply.

This can be done with the following command:

```
$ chmod ug+w memo
```

Just like the `chown` command it is also possible to do recursive (`-R`) operations.



File permissions

Another way for using chmod command is:

\$ chmod xyz file_name

where $0 \leq x, y, z \leq 7$ (x, y, z are integer numbers).

If you want to make this permissions: **rwX rw- r- -**
We can express that in binary form: **111 110 100**
Or in decimal form: **7 6 4**

Finally:

\$ chmod 764 file_name



Example 1:

- In your HOME directory create subdirectory **perm**.
- In **perm** create files: **my_f**, **our_f** and **shared_f**.
- Show which are default permissions on these files.
- Are permissions same for all new files and directories?
- Change permissions for **my_f** such that *owner* will have read and write, but *group* and *other* will not have any permission
- Change permissions for **our_f** such that *owner* and *group* will have read and write but *other* wont have any permission.
- Change permissions for **shared_f** such that *owner*, *group* and *other* (all users) will have read and write permission.
- Change permissions for **perm** such that *owner*, *group* and *other* (all users) can see and enter in this directory.
- Which permissions you established above?



Example 2/1:

- In your HOME directory create subdirectory **def_dir**.
- Change permissions for **def_dir** such that owner will have only **read** permission
 - Can you see **def_dir** content?
 - Can you enter in **def_dir** subdirectory ?
 - Can you rename the subdirectory?
 - Can you delete the subdirectory?
 - Can you copy the subdirectory?
- Change permissions for **def_dir** such that owner will have only **execute** permission
 - Can you see **def_dir** content?
 - Can you enter in **def_dir** subdirectory ?
 - Can you rename the subdirectory?
 - Can you delete the subdirectory?
 - Can you copy the subdirectory?



Example 2/1 - answers:

- If your directory has just **read** permission YOU:
 - Can see **def_dir** content.
 - Can **not** enter in **def_dir** subdirectory
 - Can rename the subdirectory
 - Can **not** delete the subdirectory
 - Can copy the subdirectory
- If your directory has just **execute** permission YOU
 - Can **not** see **def_dir** content?
 - Can enter in **def_dir** subdirectory ?
 - Can **not** rename the subdirectory?
 - Can **not** delete the subdirectory?
 - Can **not** copy the subdirectory?



Example 2/2:

Change permissions for **def_dir** such that owner will have only **write** permission

- Can you see **def_dir** content?
 - Can you enter in **def_dir** subdirectory ?
 - Can you rename the subdirectory?
 - Can you delete the subdirectory?
 - Can you copy the subdirectory?
- Change permissions for **def_dir** such that owner will have **read** and **execute** permission
- Can you see **def_dir** content?
 - Can you enter in **def_dir** subdirectory ?
 - Can you rename the subdirectory?
 - Can you delete the subdirectory?
 - Can you copy the subdirectory?



Example 2/2 - answers:

- If your directory has just **write** permission YOU:
 - Can **not** see **def_dir** content.
 - Can **not** enter in **def_dir** subdirectory
 - Can rename the subdirectory
 - Can **not** delete the subdirectory
 - Can **not** copy the subdirectory
- If your directory has **read** and **execute** permission YOU
 - Can see **def_dir** content?
 - Can enter in **def_dir** subdirectory ?
 - Can rename the subdirectory?
 - Can **not** delete the subdirectory?
 - Can copy the subdirectory?



Example 3/1:

- In your HOME directory create file **def_file**
- Change permissions for **def_file** such that owner will have only **read** permission
 - Can you see **def_file** content?
 - Can you add some text in **def_file**?
 - Can you rename the file?
 - Can you delete the file?
 - Can you copy the file in **def_dir** subdirectory?
- Change permissions for **def_file** such that owner will have only **write** permission
 - Can you see **def_file** content?
 - Can you add some text in **def_file**?
 - Can you rename the file?
 - Can you delete the file?
 - Can you copy the file in **def_dir** subdirectory?



Example 3/1 - answers:

- If your file has just **read** permission YOU
 - Can see **def_file** content
 - Can **not** add some text in **def_file**
 - Can rename the file
 - Can **not** delete the file
 - Can copy the file .
- If your file has just **write** permission YOU
 - Can **not** see the file content.
 - Can add some text in the file.
 - Can rename the file?
 - Can delete the file.
 - Can **not** copy the file. WR



Example 3/2:

- Change permissions for **def_file** such that owner will have **read** and **write** permission
 - Can you see **def_file** content?
 - Can you add some text in **def_file**?
 - Can you rename the file?
 - Can you delete the file?
 - Can you copy the file in **def_dir** subdirectory?
 - Can you change ownership over this file?
 - Change file group to **root**, if you can.



Example 3/2:

- If your file has **read** and **write** permission YOU
 - Can see the file content.
 - Can add some text in the file.
 - Can rename the file.
 - Can delete the file.
 - Can you copy the file in some directory.
 - Can **not** change ownership over this file. Just **root** can do that!
 - Change file group to **root**, if you can. **You can't do that !!**



Default permission

You can change default permissions (permissions for new file and directory) with command **umask**

umask p_mask

If you want `rw-rw-r--` as default permission (764) you must use `p_mask= 777-764 =013` and type

umask 013

For files execute permission not allowed as default !

Therefore, above command will give `rw-rw-r--` permission for files and `rwxrw-r--` for directories .