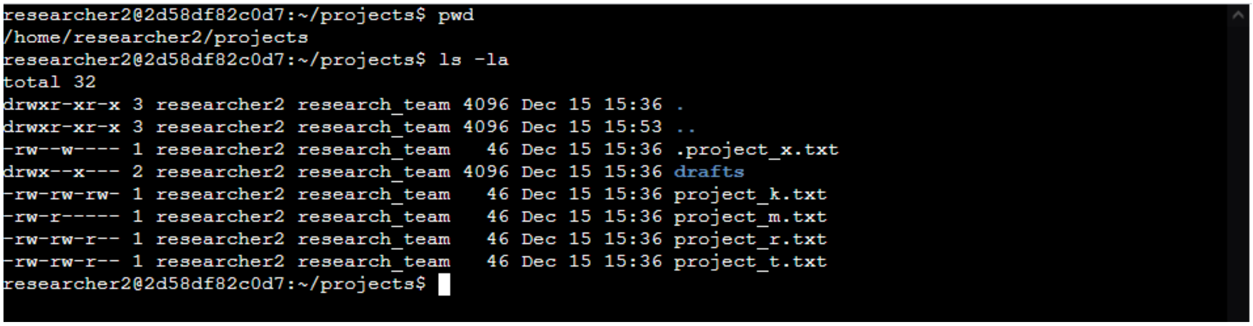
File permissions in Linux

Project description

The research team concluded that the organization needs an update on the files and directories within the *projects* directory. My task is to put into practice the principle of least privilege by giving only the necessary permissions to the employees to do their job.

Check file and directory details

ls command is used to list the files and the subdirectory of a directory. To check permissions we must add options to the command. *ls -l* will show the permissions of files and subdirectories, *ls -a* will show the hidden files. Combining *-l* with *-a* will give us the permissions of the files, hidden files and subdirectories.



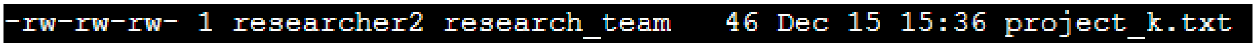
```
researcher2@2d58df82c0d7:~/projects$ pwd
/home/researcher2/projects
researcher2@2d58df82c0d7:~/projects$ ls -la
total 32
drwxr-xr-x 3 researcher2 research_team 4096 Dec 15 15:36 .
drwxr-xr-x 3 researcher2 research_team 4096 Dec 15 15:53 ..
-rw--w---- 1 researcher2 research_team   46 Dec 15 15:36 .project_x.txt
drwx--x--- 2 researcher2 research_team 4096 Dec 15 15:36 drafts
-rw-rw-rw- 1 researcher2 research_team   46 Dec 15 15:36 project_k.txt
-rw-r----- 1 researcher2 research_team   46 Dec 15 15:36 project_m.txt
-rw-rw-r-- 1 researcher2 research_team   46 Dec 15 15:36 project_r.txt
-rw-rw-r-- 1 researcher2 research_team   46 Dec 15 15:36 project_t.txt
researcher2@2d58df82c0d7:~/projects$
```

Note: *pwd* is a command that gives as output the location we are currently on.

Describe the permissions string

Each output from the previous screenshot starts with a 10-character string, that is the permission for users, groups and everyone else (aka other).
The 1st character can be represented by “d” or “-”, “d” indicates it’s a directory and “-” indicates it’s a file.
The 2nd, 3rd and 4th characters indicate the “ r “ (read), “ w “ (write) and “ x “ (execute) permissions for the user. When one of those characters is replaced by a “ - “, it indicates that this permission is not granted.
The 5th, 6th and 7th characters indicate the same type of permissions for the user, but this block is the permissions for the group. When one of those characters is replaced by a “ - “, it indicates that this permission is not granted

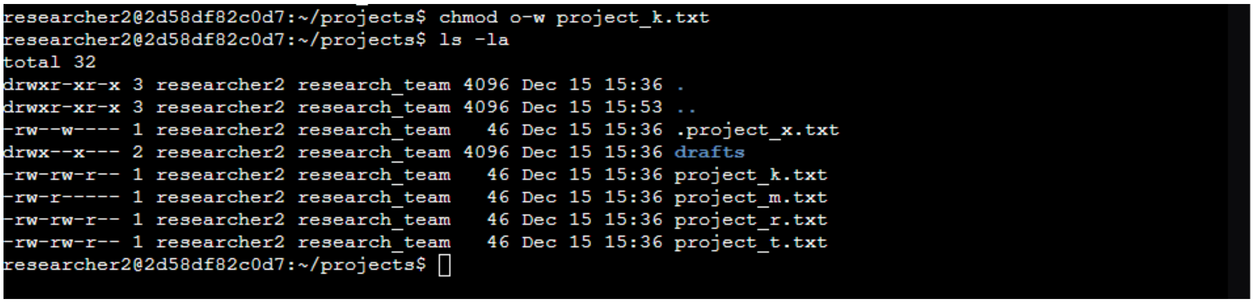
The 8th, 9th and 10th characters indicate the same type of permissions for the user and group, but this block is the permissions for the other. When one of those characters is replaced by a “ - “, it indicates that this permission is not granted.
The next screenshot shows that it is a file and the *user, group and other* have read and write permissions.



```
-rw-rw-rw- 1 researcher2 research_team   46 Dec 15 15:36 project_k.txt
```

Change file permissions

The organization doesn't want *other* to have any write permission on any files. As we can see the file *project_k.txt* as write permission for *other*. The following command will remove the write permissions for the other.

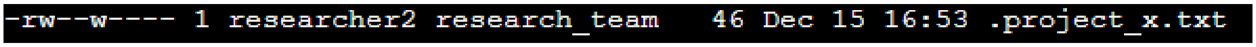


```
researcher2@2d58df82c0d7:~/projects$ chmod o-w project_k.txt
researcher2@2d58df82c0d7:~/projects$ ls -la
total 32
drwxr-xr-x 3 researcher2 research_team 4096 Dec 15 15:36 .
drwxr-xr-x 3 researcher2 research_team 4096 Dec 15 15:53 ..
-rw--w---- 1 researcher2 research_team   46 Dec 15 15:36 .project_x.txt
drwx--x--- 2 researcher2 research_team 4096 Dec 15 15:36 drafts
-rw-rw-r-- 1 researcher2 research_team   46 Dec 15 15:36 project_k.txt
-rw-r----- 1 researcher2 research_team   46 Dec 15 15:36 project_m.txt
-rw-rw-r-- 1 researcher2 research_team   46 Dec 15 15:36 project_r.txt
-rw-rw-r-- 1 researcher2 research_team   46 Dec 15 15:36 project_t.txt
researcher2@2d58df82c0d7:~/projects$ 
```

As we can see the file *project_k.txt* no longer has write permission for *other*. The *chmod* command changes permissions on files and directories.
The *chmod* command requires two arguments. The first argument indicates how to change permissions, and the second argument indicates the file or directory we want to change permissions. The 1st character of the first argument it’s either “ *u* “ (user), “ *g* “ (group) or “ *o* “ (other). The 2nd character of the first argument it’s either to add (+), remove (-) and assign as specified (=). The 3rd and up to 5th character it’s either “ r “ (read), “ w “ (write) and “ x “ (execute). So “ o-w “ will remove write permission to *other.* This could be accomplished too by the following command: *chmod o=r project_k.txt.*

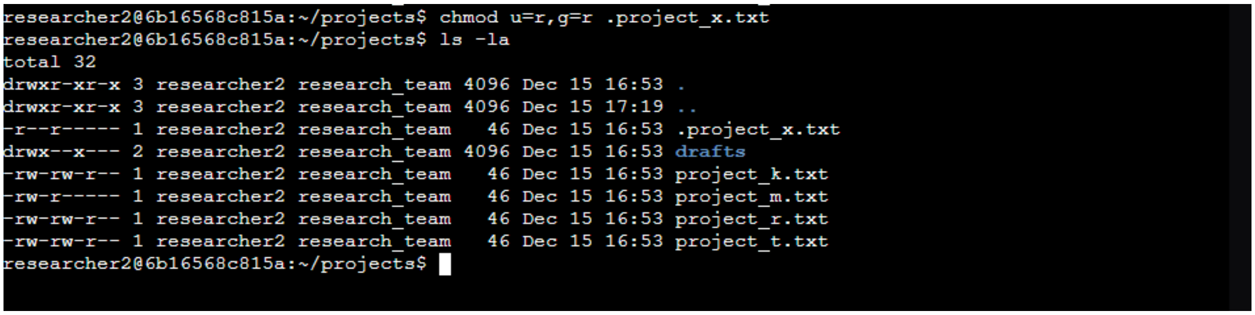
Change file permissions on a hidden file

The team recently archived *project_x.txt*. They don’t want anyone to have write access to this project, but the user and group should have read permission.
The following screenshots shows *project_x.txt* is a hidden file and that user and group have write permission to it:



```
-rw--w---- 1 researcher2 research_team   46 Dec 15 16:53 .project_x.txt
```

Hidden files always start with a “ . “.
The following command with give read only permission to user and group:

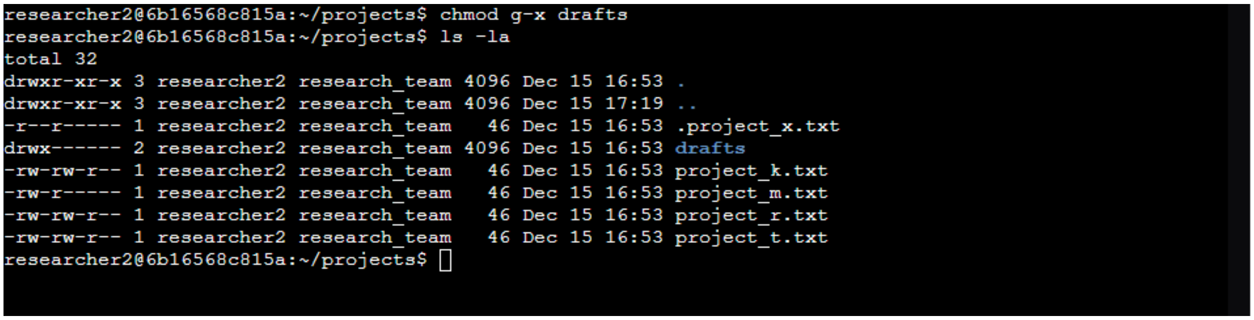


```
researcher2@6b16568c815a:~/projects$ chmod u=r,g=r .project_x.txt
researcher2@6b16568c815a:~/projects$ ls -la
total 32
drwxr-xr-x 3 researcher2 research_team 4096 Dec 15 16:53 .
drwxr-xr-x 3 researcher2 research_team 4096 Dec 15 17:19 ..
-r--r----- 1 researcher2 research_team   46 Dec 15 16:53 .project_x.txt
drwx--x--- 2 researcher2 research_team 4096 Dec 15 16:53 drafts
-rw-rw-r-- 1 researcher2 research_team   46 Dec 15 16:53 project_k.txt
-rw-r----- 1 researcher2 research_team   46 Dec 15 16:53 project_m.txt
-rw-rw-r-- 1 researcher2 research_team   46 Dec 15 16:53 project_r.txt
-rw-rw-r-- 1 researcher2 research_team   46 Dec 15 16:53 project_t.txt
researcher2@6b16568c815a:~/projects$ 
```

In this case we managed to change the permission for *user* and *group* in one command, just by adding commas (,) between them.

Change directory permissions

The organization only wants the user *researcher2* to have access to the directory *drafts*. This means that only the user must have execute permission. From the screenshot provided we can see that the group has execute permission to the *drafts* directory.
The following example will show how to remove the execute permission for the *group*, giving access only to *user*:



```
researcher2@6b16568c815a:~/projects$ chmod g-x drafts
researcher2@6b16568c815a:~/projects$ ls -la
total 32
drwxr-xr-x 3 researcher2 research_team 4096 Dec 15 16:53 .
drwxr-xr-x 3 researcher2 research_team 4096 Dec 15 17:19 ..
-r--r----- 1 researcher2 research_team   46 Dec 15 16:53 .project_x.txt
drwx------ 2 researcher2 research_team 4096 Dec 15 16:53 drafts
-rw-rw-r-- 1 researcher2 research_team   46 Dec 15 16:53 project_k.txt
-rw-r----- 1 researcher2 research_team   46 Dec 15 16:53 project_m.txt
-rw-rw-r-- 1 researcher2 research_team   46 Dec 15 16:53 project_r.txt
-rw-rw-r-- 1 researcher2 research_team   46 Dec 15 16:53 project_t.txt
researcher2@6b16568c815a:~/projects$ 
```

In this case we managed to remove the permission by adding the *g-x* argument into the *chmod* command.

Summary

I changed some permission on the files and directories within the *projects* directory. The first step to achieve this was to look at the current permissions by using *ls -la*.
To change the permissions i used multiple times the *chmod* command.