**75**

**Chmod Modifies File Permissions**

**https://www.howtogeek.com/437su958/how-to-use-the-chmod-command-on-linux/#:~:text=chmod%20Modifies%20File%20Permissions%20In%20Linux%2C%20who%20can,group%2C%20and%20a%20final%20set%20for%20everyone%20else.**

Text

Description automatically generated

D = Directory and – means File

--- = first 3 equals the owner permission

2nd group of --- equals group permission

3rd group of --- equals all users or other’s permissions

The letters represent for each group (Owner/Group/Others):

* *r*: Read permissions. The file can be opened, and its content viewed.
* *w*: Write permissions. The file can be edited, modified, and deleted.
* *x*: Execute permissions. If the file is a script or a program, it can be run (executed).

**For example:**

* --- means no permissions have been granted at all.
* rwx means full permissions have been granted. The read, write, and execute indicators are all present.
* In our screenshot, the first line starts with a d. This line refers to a directory called “archive.” The owner of the directory is “dave,” and the name of the group that the directory belongs to is also called “dave.”

**Understanding The Permission Syntax**

The “who” values we can use are:

* *u*: User, meaning the owner of the file.
* *g*: Group, meaning members of the group the file belongs to.
* *o*: Others, meaning people not governed by the u and g permissions.
* *a*: All, meaning all of the above.

The “what” values we can use are:

* *–*: Minus sign. Removes the permission.
* *+*: Plus sign. Grants the permission. The permission is added to the existing permissions. If you want to have this permission and only this permission set, use the = option, described below.
* *=*: Equals sign. Set a permission and remove others.

Examples:

chmod u=rw,og=r new\_file.txt

User will have read/write and other and group will have read

* Remember using the = means you are wiping out all previous permissions and assigning new permissions
* If you only want to change one permission or specific permissions you need to use the + or – appropriotly
* chmod u+wrx, g+r, o+rx new\_file.txt

**Setting Permissions for Multiple Files**

Graphical user interface, text

Description automatically generated

let’s say we want to remove the read permissions for the “other” users from files that have a “.page” extension. We can do this with the following command:

chmod o-r \*.page

**Numerical Shorthand**

Another way to use chmod is to provide the permissions you wish to give to the owner, group, and others as a three-digit number. The leftmost digit represents the permissions for the owner. The middle digit represents the permissions for the group members. The rightmost digit represents the permissions for the others.

The digits you can use and what they represent are listed here:

* 0: (000) No permission.
* 1: (001) Execute permission.
* 2: (010) Write permission.
* 3: (011) Write and execute permissions.
* 4: (100) Read permission.
* 5: (101) Read and execute permissions.
* 6: (110) Read and write permissions.
* 7: (111) Read, write, and execute permissions.

Let’s add the read permission back on the “.page” files for the others category of users. We must set the user and group permissions as well, so we need to set them to what they are already. These users already have read and write permissions, which is 6 (110). We want the “others” to have read and permissions, so they need to be set to 4 (100).

The following command will accomplish this:

chmod 664 \*.page

so after using the numerical short hand your new permissions should look like this…

first 6 = USER / second 6 = Group / and 4 = OTHERS

Graphical user interface, text

Description automatically generated