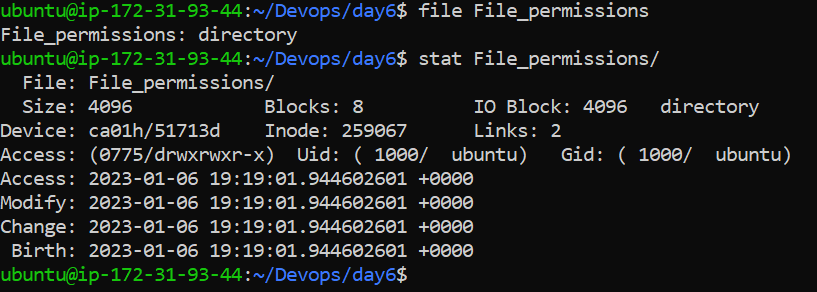
**File Permissions:**

**Types of Files:**

1. **Regular files (-):** It contain programs, executable files and text files.
2. **Directory files (d):** It is shown in blue color. It contain list of files.
3. **Special files**
   * **Block file (b)**
   * **Character device file (c)**
   * **Named pipe file (p)**
   * **Symbolic link file (l)**
   * **Socket file (s)**

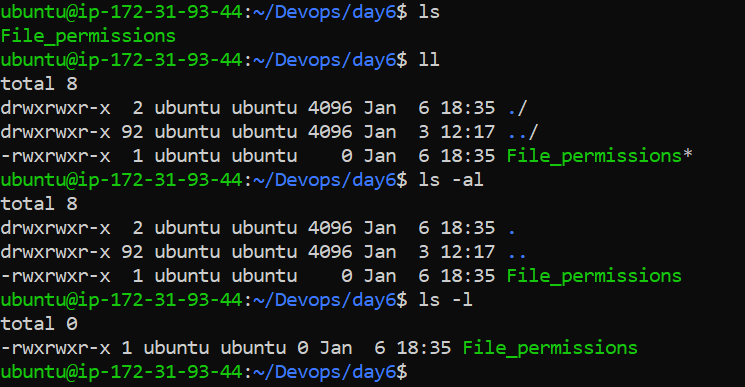
Basic Commands for Files:

|  |  |  |
| --- | --- | --- |
| **Command** | **Description** | **Formats** |
| [file](https://www.javatpoint.com/linux-file) | Determines file type. | file <Filename> |
| [stat](https://www.javatpoint.com/linux-file) | to view details (Access, Modify,change time, size, inode, block ... ) |  |
| [touch](https://www.javatpoint.com/linux-touch) | Used to create a file. | touch <Filename> |
| [rm](https://www.javatpoint.com/linux-rm) | To remove a file. | rm <Filename> |
| [cp](https://www.javatpoint.com/linux-cp) | To copy a file. | cp <Source File> <Destination> |
| [mv](https://www.javatpoint.com/linux-mv) | To rename or to move a file. | rm filename1 filename2 filename3 |
| [rename](https://www.javatpoint.com/linux-rename) | To rename file. | mv <original name> |



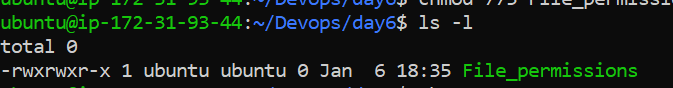
Different ways to list the files:

There are multiple way to list a file and you will see the difference between all of them. This is something you didn’t find anywhere, below is simple yet effective image to compare and understand :



Let’s take below example and understand each column in detail:

Example1:



|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| File Permissions | Number of memory blocks | Owner | Group | File Size in KB | Month, Date  Modified Time | Directory name /File name |
| 1 | 2 | 3 | 4 | 5 | 6 | 7 |

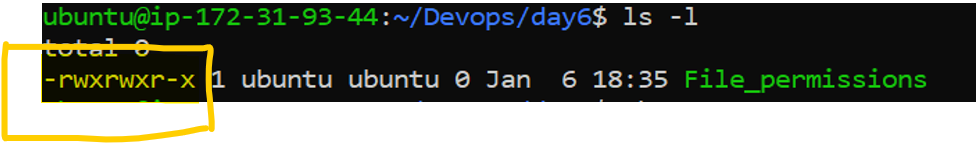
* **First Column** − Represents the file type and the permission given on the file. Below is the description of all type of files.
* **Second Column** − Represents the number of memory blocks taken by the file or directory.
* **Third Column** − Represents the owner of the file. This is the user who created this file.
* **Fourth Column** − Represents the group of the owner. Every user will have an associated group.
* **Fifth Column** − Represents the file size in bytes.
* **Sixth Column** − Represents the date and the time when this file was created or modified for the last time.
* **Seventh Column** − Represents the file or the directory name.

**File Type:**

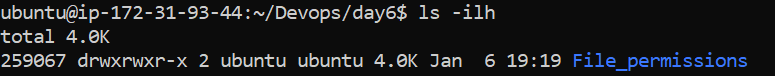
|  |  |
| --- | --- |
| **Sr.No.** | **Prefix & Description** |
| **-** | Regular file, such as an ASCII text file, binary executable, or hard link. |
| **b** | Block special file. Block input/output device file such as a physical hard drive. |
| **c** | Character special file. Raw input/output device file such as a physical hard drive. |
| **d** | Directory file that contains a listing of other files and directories. |
| **l** | Symbolic link file. Links on any regular file. |
| **p** | Named pipe. A mechanism for inter-process communications. |
| **s** | Socket used for inter-process communication. |

File Permissions:

|  |  |  |  |
| --- | --- | --- | --- |
| **Number** | **Permission Representation** | **Ref** | **Octal representation** |
| **0** | No permission | --- | 0 |
| **1** | Execute permission | --x | 1 |
| **2** | Write permission | -w- | 2 |
| **3** | Execute and write permission: 1 (execute) + 2 (write) | -wx | 3 |
| **4** | Read permission | r-- | 4 |
| **5** | Read and execute permission: 4 (read) + 1 (execute) | r-x | 5 |
| **6** | Read and write permission: 4 (read) + 2 (write) | rw- | 6 |
| **7** | All permissions: 4 (read) + 2 (write) + 1 (execute) | rwx | 7 |



|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 |
| - | r | w | x | r | w | x | r | w | x |
| File type | User permission | | | Group Permission | | | Others | | |



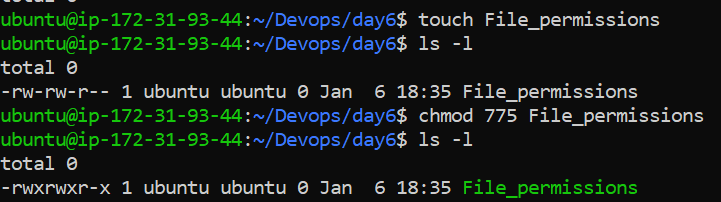
|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Inode Number | File Permissions | Number of Symbolic Link | Owner | Group | Size in KB | Month Date | Modified Time | Directory name /File name |
| 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 |

Modifying permission via **chmod**:

This is used to modify the file permission for Users, Groups and Others

Syntax:

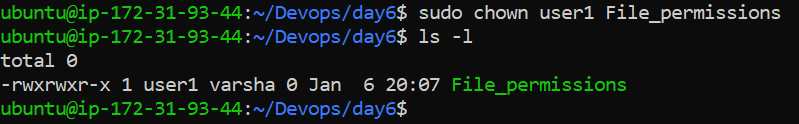
chmod <permission> <File/Directory Name>



Changing Owner via **chown:**

This is used to modify the owner of a file

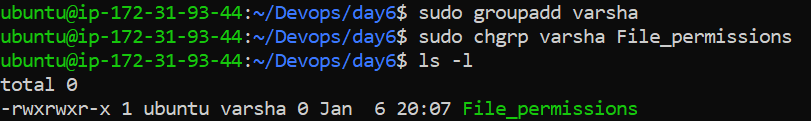
Syntax: chown <owner name> <file>

****

Changing Group via **chgrp:**

This is used to modify the group of the file.

Syntax: chgrp <group name> file



**Controlling access to files with ACLs (Access control list)**

**ACLs are used on Linux filesystems to set custom and more personalized permissions on files and folders. ACLs allow file owners or privileged users to grant rights to specific users or to specific groups.**

In Linux, as you probably know, the permissions are divided into three categories : one for the owner of the file, one for the group and one for the others. However, in some cases, you may want to grant access to a directory (the execute permission for example) to a specific user without having to put this user into the group of the file.

*This is exactly why access control lists were invented in the first place.*

On Linux, access control lists are not enabled when you create a new file or directory on your host (except if a parent directory has some ACLs predefined).

ACL is not pre installed over the machine. We can use Below Syntax:

sudo apt-get install acl -y

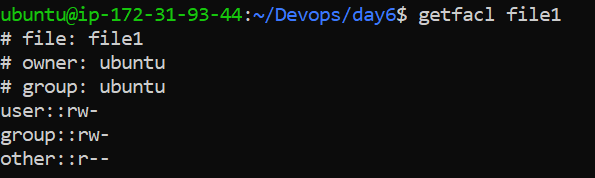
To see if access control lists are defined for a file or directory, run the ls command and look for a “**+**” character at the end of the permission line.

****

**Checking access control lists using getfacl:**

**The getfacl command is used on Linux to print a complete listing of all regular permissions and access control lists permissions on a file or directory.**

getfacl <file, directory>

****

**Setting access control lists using setfacl:**

The setfacl command is used on Linux to create, modify and remove access control lists on a file or directory.

setfacl {-m, -x} {u, g}:<name>:[r, w, x] <file, directory>

Where curly brackets mean one of the following options and regular brackets mean one or several items.

* **-m** : means that you want to **modify**one or several ACL entries on the file or directory.
* **-x** : means that you want to **remove**one or several ACL entries on a file or directory.
* **{u, g}** : if you want to modify the ACL for a user or for a group.
* **name**: this is an optional parameter, it can be omitted if you want to set the ACL entries for every user or for every group on your host.
* **[r, w, x]** : in order to set read, write or execute permissions on the file or directory.

For example, in order to set specific write permissions for a user on a file, you would write the following command

$ setfacl -m u:user:w <file, directory>

In order to set execute permissions for all users on your host, you would write the following command

$ setfacl -m u::x <file, directory>

To set full permissions for a specific group on your host, you would write the setfacl this way

$ setfacl -m g:group:rwx <file, directory>

Now let’s say that you want to remove an ACL entry from a file.

In order to remove a user specific entry from a file, **you would specify the x option.**

*Note : you cannot specific rights from a single ACL entry, meaning that you can’t remove write permissions, keeping the ACL read permissions active.*

$ setfacl -x u:<user> <file, directory>

Similarly, to remove ACL related to groups on your host, you would write the following command

$ setfacl -x g:<group> <file, directory>

Example1: Adding Write permission for another user:

