default permissions for a newly created directory are typically **rwxr-xr-x** (755 in numeric mode).

default permissions for a newly created files are typically **rwxr-x-x** (744 in numeric mode).

Umask is 022

The **umask** (user file creation mask) is a Linux command that determines the default permission settings for newly created files and directories. It essentially "masks" the default permissions to ensure that new files and directories are created with the desired permissions.

Ls -lart

**drwxr-xr-x 2 user group 4096 Feb 7 10:00 dir1**

**-rw-r--r-- 1 user group 123 Feb 7 09:00 file1.txt**

**-rw-r--r-- 1 user group 456 Feb 6 08:00 .hiddenfile**

1. **File Type and Permissions**:
   * drwxr-xr-x: Directory with read, write, and execute permissions for the owner, and read and execute permissions for the group and others.
   * -rw-r--r--: Regular file with read and write permissions for the owner, and read-only permissions for the group and others.
2. **Number of Links**:
   * 2: Number of hard links to the directory dir1.
   * 1: Number of hard links to the file file1.txt.
3. **Owner**:
   * user: The user who owns the file or directory.
4. **Group**:
   * group: The group that owns the file or directory.
5. **Size**:
   * 4096: Size of the directory dir1 in bytes.
   * 123: Size of the file file1.txt in bytes.
6. **Modification Time**:
   * Feb 7 10:00: Last modification time of dir1.
   * Feb 7 09:00: Last modification time of file1.txt.
7. **File/Directory Name**:
   * dir1: Name of the directory.
   * file1.txt: Name of the file.
   * .hiddenfile: Name of the hidden file.

Linux file system permissions control who can read, write, or execute files and directories. These permissions are crucial for system security and user privacy. Here's a breakdown:

**Permission Types**

1. **Read (r)**: Allows viewing the contents of a file or listing the contents of a directory.
2. **Write (w)**: Allows modifying the contents of a file or adding/removing files in a directory.
3. **Execute (x)**: Allows running a file as a program or entering a directory.

**Permission Categories**

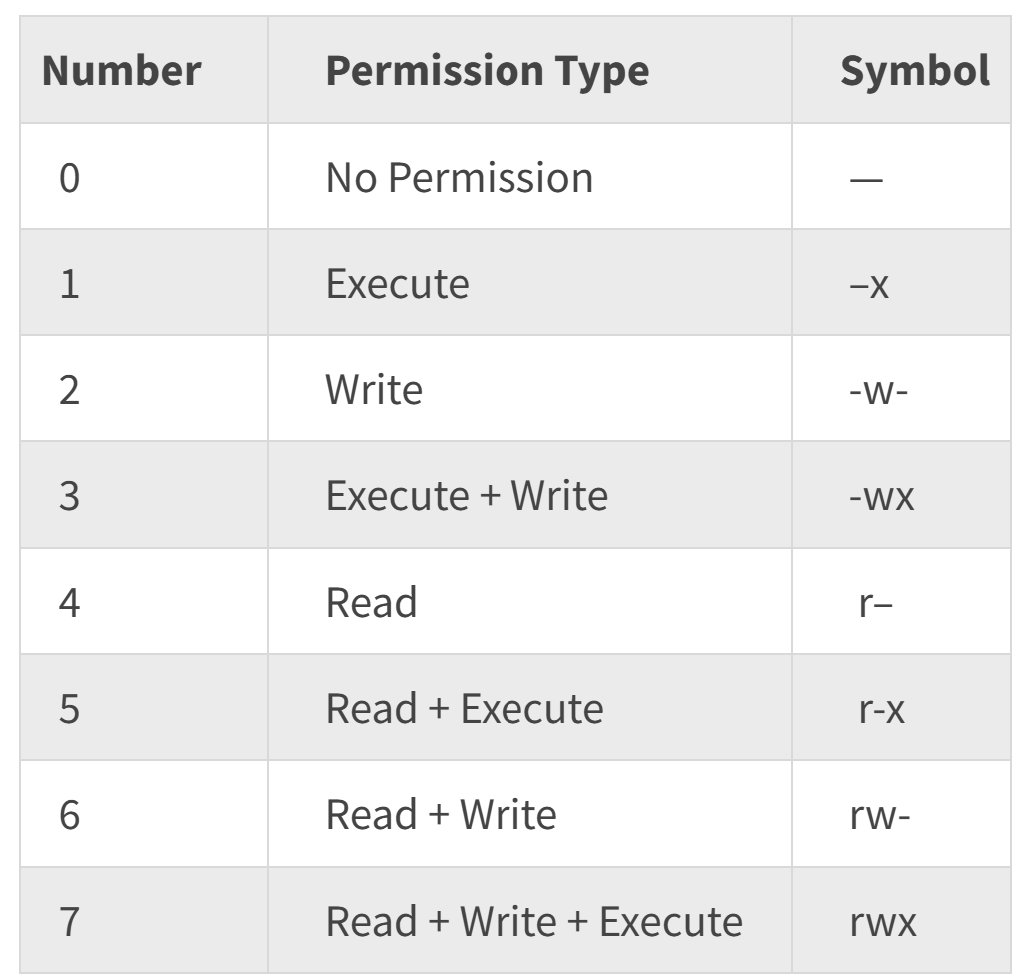
Permissions are assigned to three categories:

1. **Owner**: The user who owns the file.
2. **Group**: The group that owns the file.
3. **Others**: All other users.

**Sticky bit**

The sticky bit is a special permission in Linux that can be set on directories. When the sticky bit is set on a directory, only the owner of a file within that directory, the owner of the directory, or the root user can delete or rename the file. This is particularly useful for shared directories like /tmp, where many users have write access but should not be able to delete or rename each other's files.

The /tmp directory typically has the sticky bit set by default. This ensures that while all users can create files in /tmp, they cannot delete or rename files created by other users. This is crucial for maintaining a secure and functional shared space.



1. **chown (Change Owner)**:
   * **Purpose**: Changes the ownership of files and directories.
   * **Syntax**: chown [OPTIONS] USER[:GROUP] FILE...
   * **Example**: chown user1 file.txt changes the owner of file.txt to user1.
2. **chmod (Change Mode)**:
   * **Purpose**: Changes the permissions of files and directories.
   * **Syntax**: chmod [OPTIONS] MODE FILE...
   * **Example**: chmod 755 script.sh sets the permissions of script.sh to rwxr-xr-x.
3. **chgrp (Change Group)**:
   * **Purpose**: Changes the group ownership of files and directories.
   * **Syntax**: chgrp [OPTIONS] GROUP FILE...
   * **Example**: chgrp developers project/ changes the group of the project directory to developers.

* **Hard links** point directly to the inode of a file, allowing multiple filenames to reference the same data. Share the same inode, cannot span across different file systems, and remain valid even if the original file is deleted.
* ln original.txt hardlink.txt
* **Soft links** (or symbolic links) are special files that contain a path to another file or directory.  Have a different inode, can span across different file systems, and become invalid if the original file is deleted.

ln -n original.txt hardlink.txt

ls -li