3. Linux File Commands

Linux File Commands

Linux Files

In Linux system, everything is a file and if it is not a file, it is a process. A file doesn’t include only text files, images and compiled programs but also include partitions, hardware device drivers and directories. Linux consider everything as as file.

Files are always case sensitive.

Linux File Commands

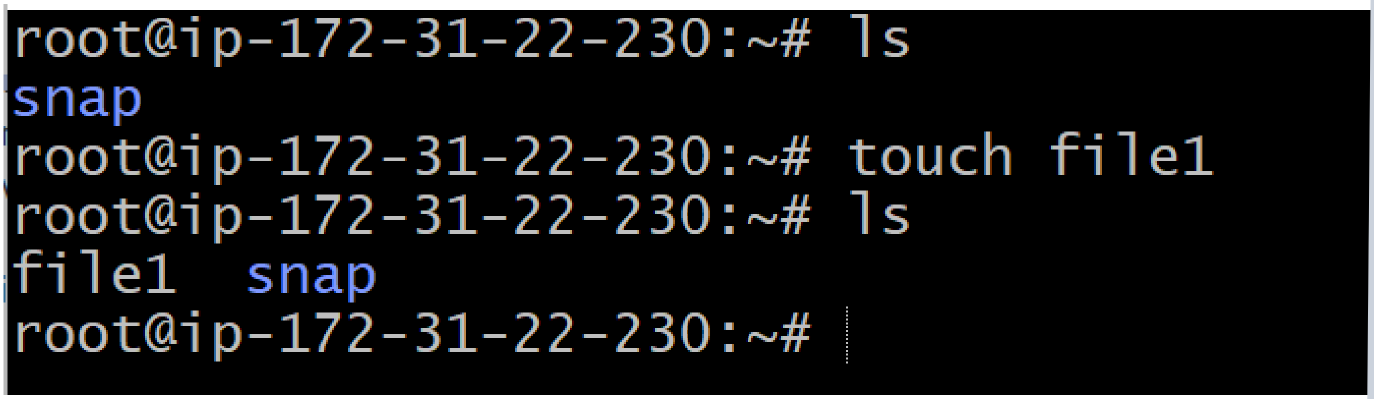
Command	Description
file	Determines file type.
touch	Used to create a file.
rm	To remove a file.
cp	To copy a file.
mv	To rename or to move a file.
rename	To rename file.

When we install the Linux operating system, Linux offers many file systems such as **Ext, Ext2, Ext3, Ext4, JFS, ReiserFS, XFS, btrfs,** and **swap**.

The file system Ext stands for **Extended File System**.

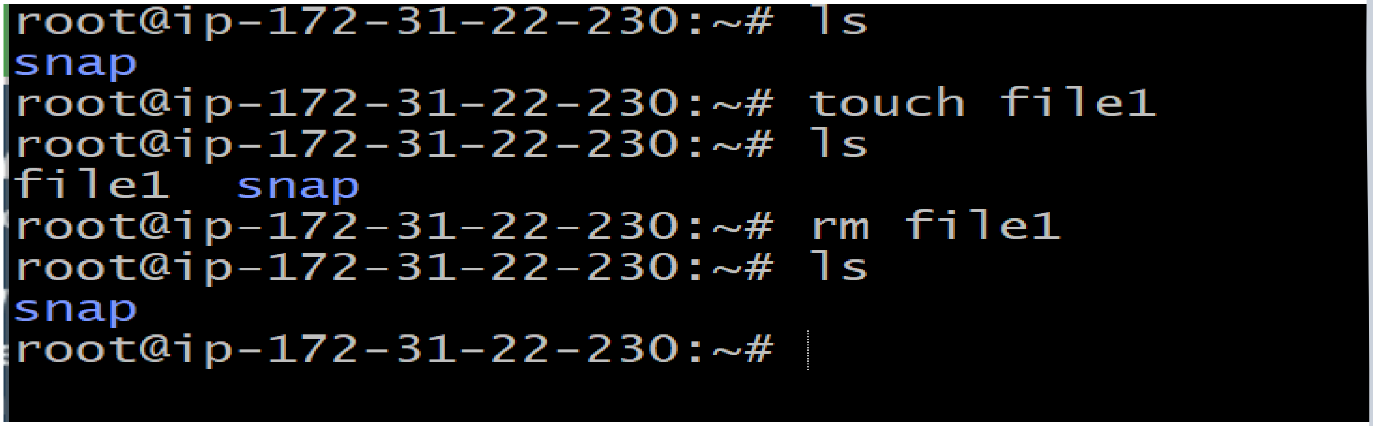
Examples-

1. **touch file1**



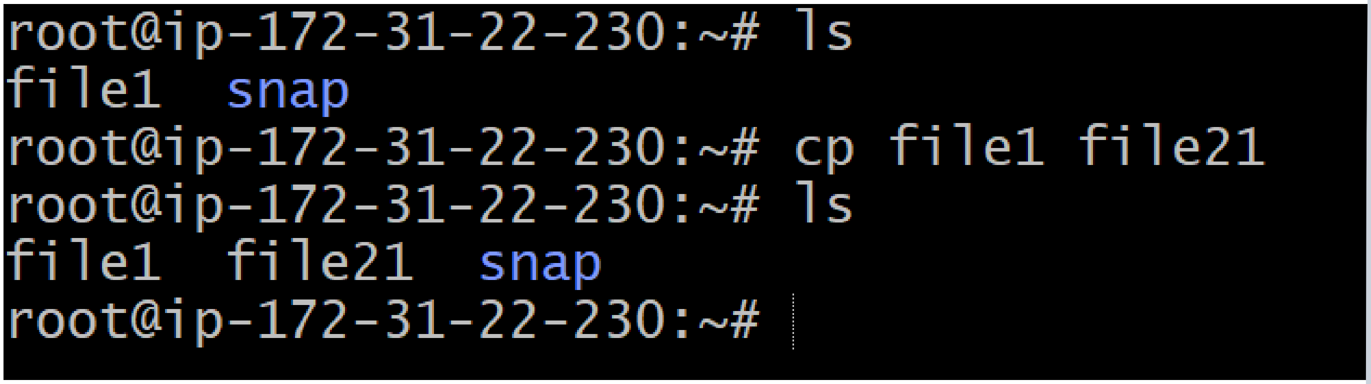
```
root@ip-172-31-22-230:~# ls
snap
root@ip-172-31-22-230:~# touch file1
root@ip-172-31-22-230:~# ls
file1  snap
root@ip-172-31-22-230:~#
```

1. rm file1



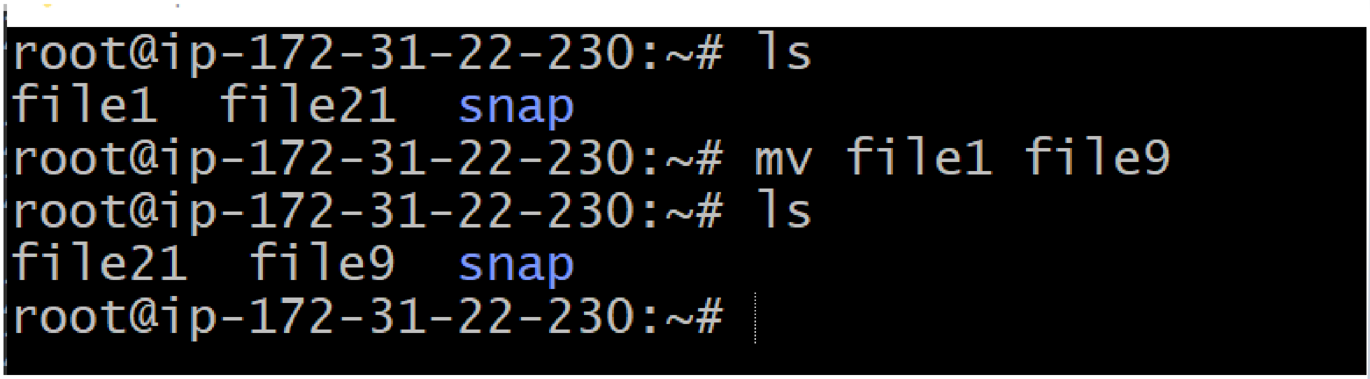
```
root@ip-172-31-22-230:~# ls
snap
root@ip-172-31-22-230:~# touch file1
root@ip-172-31-22-230:~# ls
file1  snap
root@ip-172-31-22-230:~# rm file1
root@ip-172-31-22-230:~# ls
snap
root@ip-172-31-22-230:~#
```

1. cp <existing file name> <new file name>



```
root@ip-172-31-22-230:~# ls
file1  snap
root@ip-172-31-22-230:~# cp file1 file21
root@ip-172-31-22-230:~# ls
file1  file21  snap
root@ip-172-31-22-230:~#
```

1. mv <existing file name> <new file name>



```
root@ip-172-31-22-230:~# ls
file1  file21  snap
root@ip-172-31-22-230:~# mv file1 file9
root@ip-172-31-22-230:~# ls
file21  file9  snap
root@ip-172-31-22-230:~#
```

==

Linux File Ownership–

Every Linux system have three types of owner:

1. **User:** A user is the one who created the file. By default, whosoever, creates the file becomes the owner of the file. A user can create, delete, or modify the file.
2. **Group:** A group can contain multiple users. All the users belonging to a group have same access permission for a file.
3. **Other:** Any one who has access to the file other than **user** and **group** comes in the category of **other**. Other has neither created the file nor is a group member.

File Permissions-

All the three owners (user owner, group, others) in the Linux system have three types of permissions defined. Nine characters denotes the three types of permissions.

1. **Read (r) :** The read permission allows you to open and read the content of a file. But you can't do any editing or modification in the file.
2. **Write (w) :** The write permission allows you to edit, remove or rename a file. For instance, if a file is present in a directory, and write permission is set on the file but not on the directory, then you can edit the content of the file but can't remove, or rename it.
3. **Execute (x):** In Unix type system, you can't run or execute a program unless execute permission is set.But in Windows, there is no such permission available.

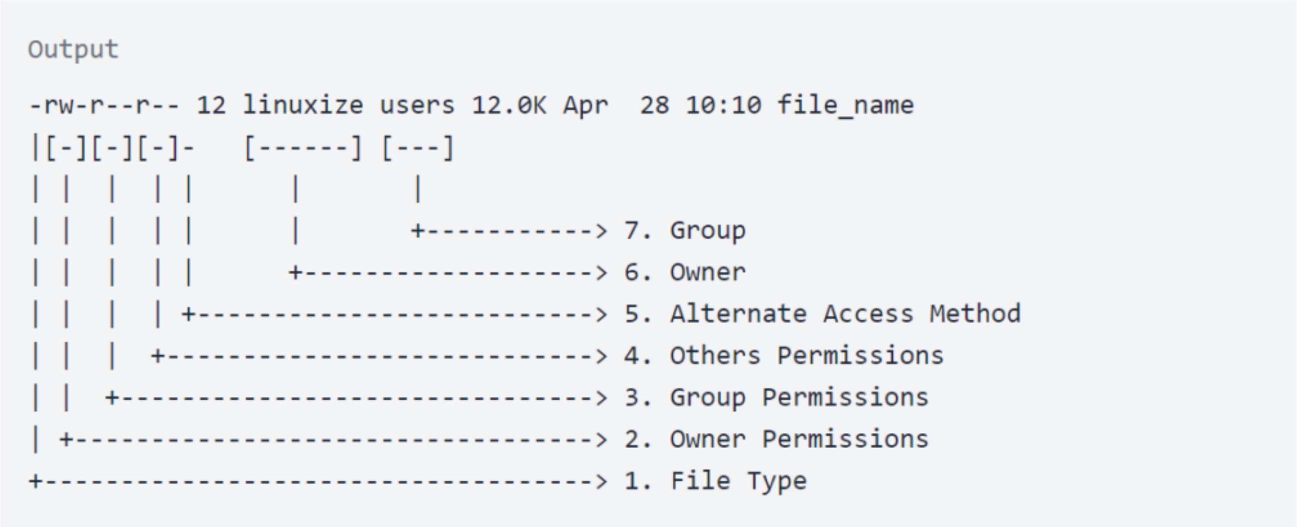
permission	**on a file**	**on a directory**
r (read)	read file content (cat)	read directory content (ls)

w (write)	change file content (vi)	create file in directory (touch)
x (execute)	execute the file	enter the directory (cd)

Permissions are listed below-

Lets see output of below cmd-

ls -l file_name



```
Output
-rw-r--r-- 12 linuxize users 12.0K Apr  28 10:10 file_name
|[-][-][-]-   [------] [---]
| |  |  | |      |       |
| |  |  | |      |       +-----------> 7. Group
| |  |  | |      +-------------------> 6. Owner
| |  |  | +--------------------------> 5. Alternate Access Method
| |  |  +----------------------------> 4. Others Permissions
| |  +-------------------------------> 3. Group Permissions
| +----------------------------------> 2. Owner Permissions
+------------------------------------> 1. File Type
```

Changing File permissions-

The File permissions can be changed using the chmod command. Only root, the file owner, or user with sudo privileges can change the permissions of a file. Be extra careful when using chmod, especially when recursively changing the permissions. The command can accept one or more files and/or directories separated by space as arguments.

Numeric Method-

- r (read) = 4
- w (write) = 2
- x (execute) = 1
- no permissions = 0

The permissions number of a specific user class is represented by the sum of the values of the permissions for that group.

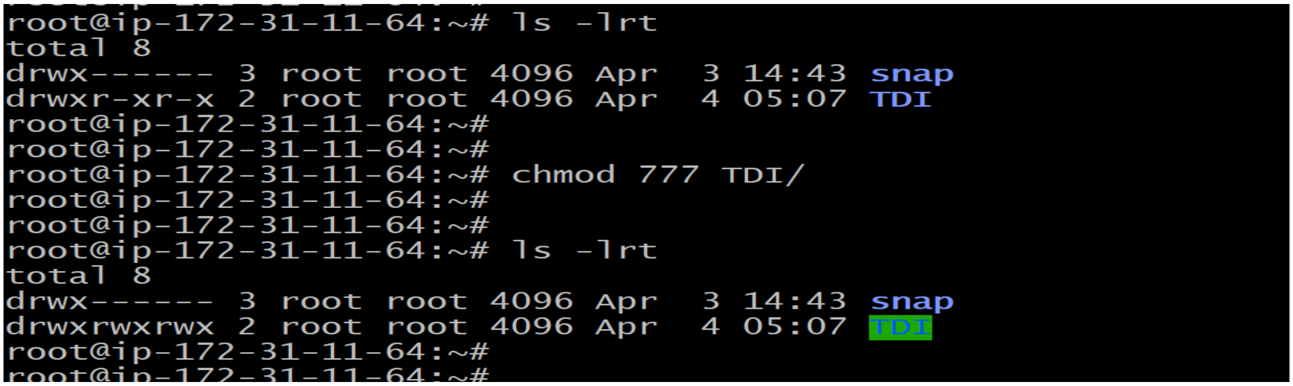
To find out the file's permissions in numeric mode, simply calculate the totals for all users' classes. For example, to give read, write and execute permission to the file's owner, read and execute permissions to the file’s group and only read permissions to all other users, you would do the following:

- Owner: rwx=4+2+1=7
- Group: r-x=4+0+1=5
- Others: r-x=4+0+1=5

Ex. chmod 644 dirname

Or

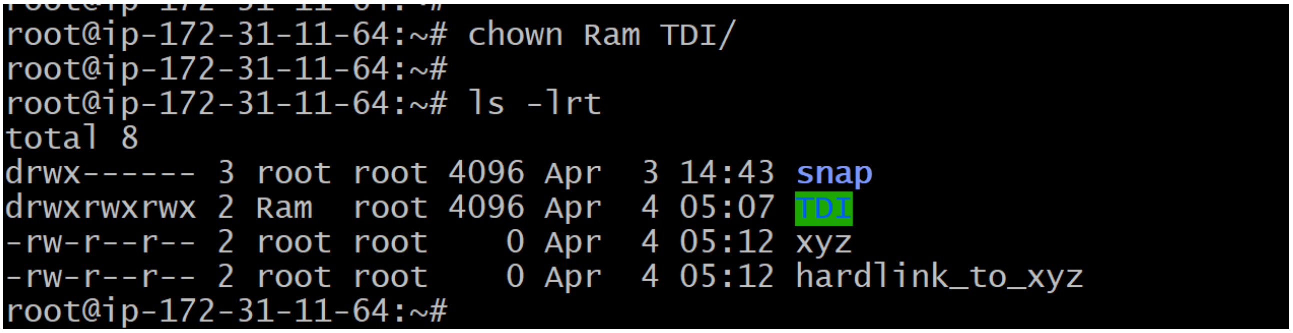
chmod 777 filename



```
root@ip-172-31-11-64:~#
root@ip-172-31-11-64:~# ls -lrt
total 8
drwx------ 3 root root 4096 Apr  3 14:43 snap
drwxr-xr-x 2 root root 4096 Apr  4 05:07 TDI
root@ip-172-31-11-64:~#
root@ip-172-31-11-64:~#
root@ip-172-31-11-64:~# chmod 777 TDI/
root@ip-172-31-11-64:~#
root@ip-172-31-11-64:~#
root@ip-172-31-11-64:~# ls -lrt
total 8
drwx------ 3 root root 4096 Apr  3 14:43 snap
drwxrwxrwx 2 root root 4096 Apr  4 05:07 TDI
root@ip-172-31-11-64:~#
root@ip-172-31-11-64:~#
```

chown owner_name file_name

chown Ram TDI



```
root@ip-172-31-11-64:~#
root@ip-172-31-11-64:~# chown Ram TDI/
root@ip-172-31-11-64:~#
root@ip-172-31-11-64:~# ls -lrt
total 8
drwx------ 3 root root 4096 Apr  3 14:43 snap
drwxrwxrwx 2 Ram  root 4096 Apr  4 05:07 TDI
-rw-r--r-- 2 root root    0 Apr  4 05:12 xyz
-rw-r--r-- 2 root root    0 Apr  4 05:12 hardlink_to_xyz
root@ip-172-31-11-64:~#
```

===

Linux File Links –

A Linux filesystem has many hard links and symbolic links. A link is a connectivity between the filename and the actual data byte in the di-c sk space. More than one filename can **link** to the same data.

There are two types of links in Linux OS:

1. Hard Links
2. Soft Links

1) Hard Links

They are the low-level links. It links more than one filename with the same Inode and it represents the physical location of a file.

When hard link is created for a file, it directly points to the Inode of the original file in the disk space, which means no new Inode is created. Directories are not created using hard links and they can not cross filesystem boundaries. When the source file is removed or moved, then hard links are not affected.

2) Soft Links (Symbolic Links)

Soft links are very common. It represents a virtual or abstract location of the file. It is just like the shortcuts created in Windows. A soft link doesn't contain any information or content of the linked file, instead it has a pointer to the location of the linked file. In other words, a new file is created with new Inode, having a pointer to the Inode location of the original file.

It is used to create link between directories and can cross filesystem boundaries. When the source file is removed or moved, then soft links are not updated.

Linux Inodes –

An Inode number is a uniquely existing number for all the files in Linux and all Unix type systems.

When a file is created on a system, a file name and Inode number is assigned to it.

Generally, to access a file, a user uses the file name but internally file name is first mapped with respective Inode number stored in a table.

Note: Inode doesn't contain the file name. Reason for this is to maintain hard-links for the files. When all the other information is separated from the file name then only we can have various file names pointing to the same Inode.

Inode Contents –

An Inode is a data structure containing metadata about the files.

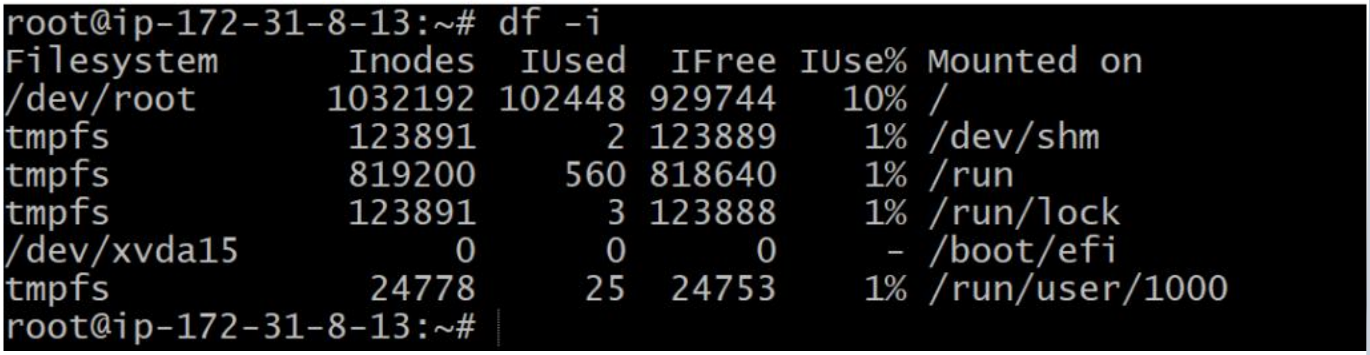
Following contents are stored in the Inode from a file:

- User ID of file
- Group ID of file
- Device ID
- File size
- Date of creation

- Permission
- Owner of the file
- File protection flag
- Link counter to determine number of hard links

Inode Table-

The Inode table contains all the Inodes and is created when file system is created. The **df -i** command can be used to check how many inodes are free and left unused in the filesystem.

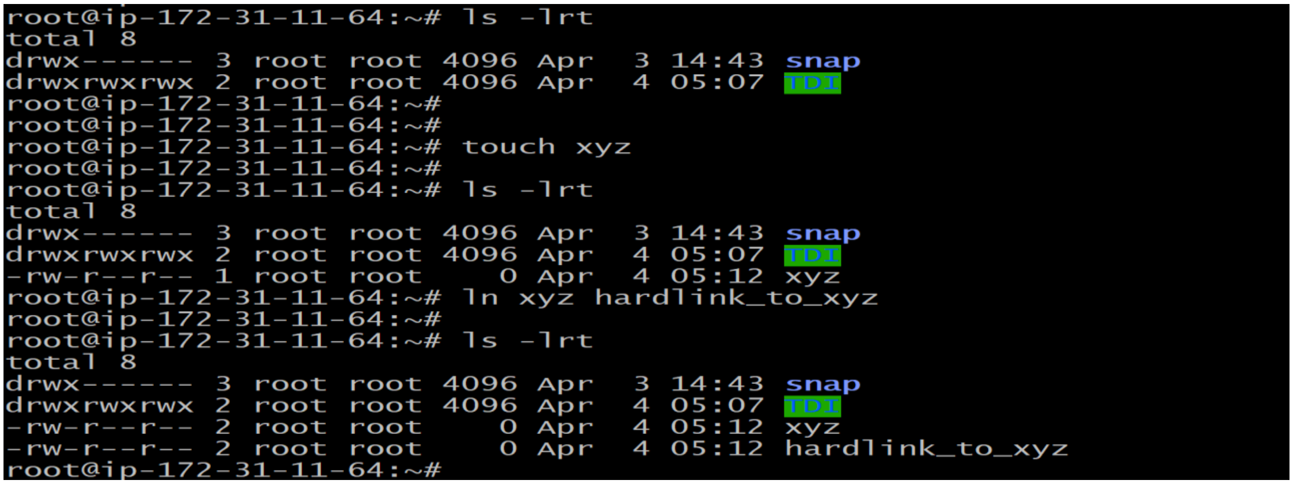


```
root@ip-172-31-8-13:~# df -i
Filesystem       Inodes  IUsed  IFree IUse% Mounted on
/dev/root       1032192 102448 929744   10% /
tmpfs            123891      2 123889    1% /dev/shm
tmpfs            819200    560 818640    1% /run
tmpfs            123891      3 123888    1% /run/lock
/dev/xvda15           0      0      0     - /boot/efi
tmpfs             24778     25  24753    1% /run/user/1000
root@ip-172-31-8-13:~#
```

Hard Links –

Creating Hard Links

Hard links for any file can be created with command ln. One extra hard link file will be created in the respective directory.



```
root@ip-172-31-11-64:~# ls -lrt
total 8
drwx------ 3 root root 4096 Apr  3 14:43 snap
drwxrwxrwx 2 root root 4096 Apr  4 05:07 TDI
root@ip-172-31-11-64:~#
root@ip-172-31-11-64:~#
root@ip-172-31-11-64:~# touch xyz
root@ip-172-31-11-64:~#
root@ip-172-31-11-64:~# ls -lrt
total 8
drwx------ 3 root root 4096 Apr  3 14:43 snap
drwxrwxrwx 2 root root 4096 Apr  4 05:07 TDI
-rw-r--r-- 1 root root    0 Apr  4 05:12 xyz
root@ip-172-31-11-64:~# ln xyz hardlink_to_xyz
root@ip-172-31-11-64:~#
root@ip-172-31-11-64:~# ls -lrt
total 8
drwx------ 3 root root 4096 Apr  3 14:43 snap
drwxrwxrwx 2 root root 4096 Apr  4 05:07 TDI
-rw-r--r-- 2 root root    0 Apr  4 05:12 xyz
-rw-r--r-- 2 root root    0 Apr  4 05:12 hardlink_to_xyz
root@ip-172-31-11-64:~#
```

Look at the above snapshot, we have created a hard link for the file **xyz** in the directory **new1.**

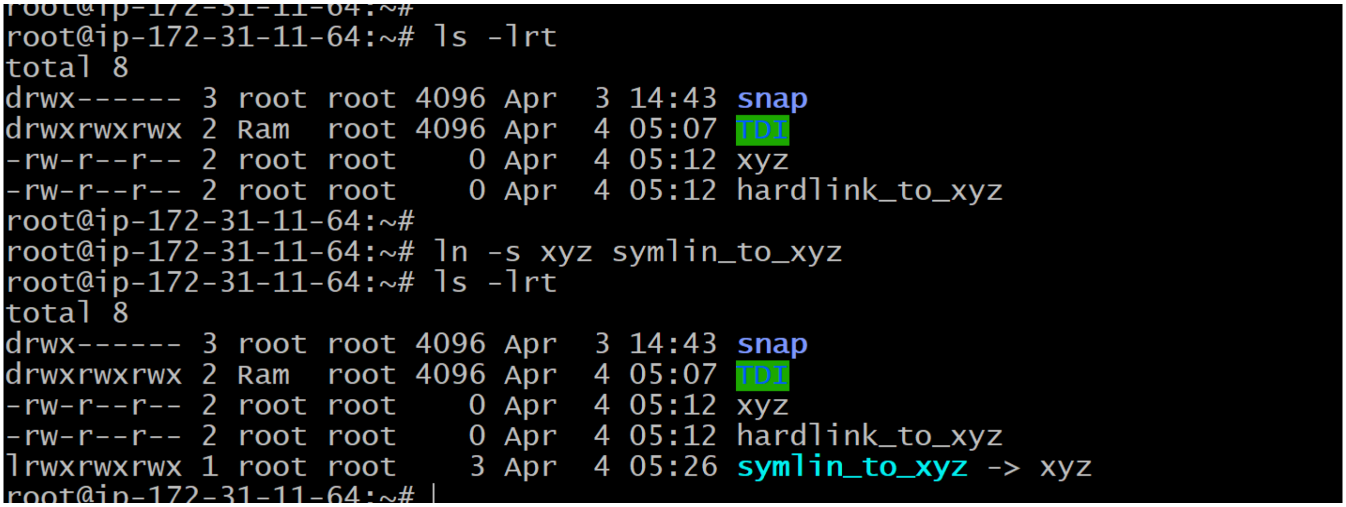
The original file and hard linked file both contain the same Inode number and hence, they have the same permissions and same owners. Content will also be the same for both the files. In short, both the files are equal now, but if original file will be removed then hard link file will not be affected.

Symbolic Links-

Symbolic links are also called **soft links.** Command **ln -s** is used to create soft link. It doesn’t link to Inodes but create a name to mapping. It create its own Inode number.

ln -s xyz symlink_to_xyz

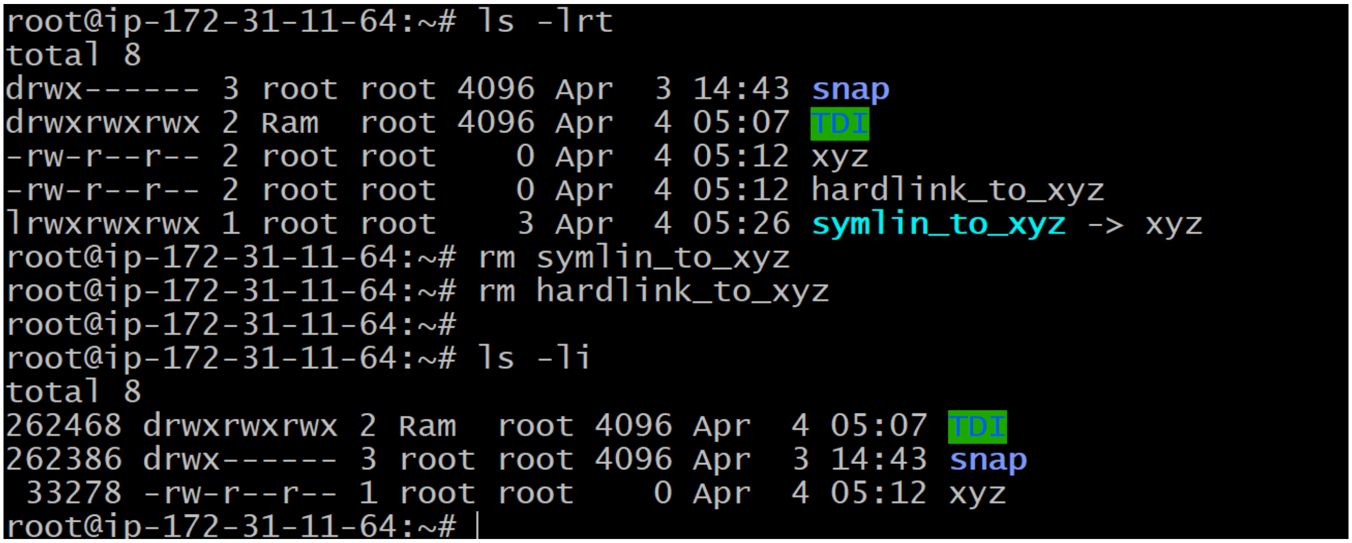
Look at the above snapshot, we have created a symbolic link for file **xyz** with command **“ln -s xyz symlink_to_xyz”**. Symbolic link Inode is different from the original file Inode number. Target permissions are applied on the symlink file. Hard links are limited to their own partition, but symbolic links can be linked anywhere.



```
root@ip-172-31-11-64:~#
root@ip-172-31-11-64:~# ls -lrt
total 8
drwx------ 3 root root 4096 Apr  3 14:43 snap
drwxrwxrwx 2 Ram  root 4096 Apr  4 05:07 TDI
-rw-r--r-- 2 root root    0 Apr  4 05:12 xyz
-rw-r--r-- 2 root root    0 Apr  4 05:12 hardlink_to_xyz
root@ip-172-31-11-64:~#
root@ip-172-31-11-64:~# ln -s xyz symlin_to_xyz
root@ip-172-31-11-64:~# ls -lrt
total 8
drwx------ 3 root root 4096 Apr  3 14:43 snap
drwxrwxrwx 2 Ram  root 4096 Apr  4 05:07 TDI
-rw-r--r-- 2 root root    0 Apr  4 05:12 xyz
-rw-r--r-- 2 root root    0 Apr  4 05:12 hardlink_to_xyz
lrwxrwxrwx 1 root root    3 Apr  4 05:26 symlin_to_xyz -> xyz
root@ip-172-31-11-64:~# |
```

Removing Links-

With **rm** command links can be removed.



```
root@ip-172-31-11-64:~# ls -lrt
total 8
drwx------ 3 root root 4096 Apr  3 14:43 snap
drwxrwxrwx 2 Ram  root 4096 Apr  4 05:07 TDI
-rw-r--r-- 2 root root    0 Apr  4 05:12 xyz
-rw-r--r-- 2 root root    0 Apr  4 05:12 hardlink_to_xyz
lrwxrwxrwx 1 root root    3 Apr  4 05:26 symlin_to_xyz -> xyz
root@ip-172-31-11-64:~# rm symlin_to_xyz
root@ip-172-31-11-64:~# rm hardlink_to_xyz
root@ip-172-31-11-64:~#
root@ip-172-31-11-64:~# ls -li
total 8
262468 drwxrwxrwx 2 Ram  root 4096 Apr  4 05:07 TDI
262386 drwx------ 3 root root 4096 Apr  3 14:43 snap
 33278 -rw-r--r-- 1 root root    0 Apr  4 05:12 xyz
root@ip-172-31-11-64:~# |
```

Look at the above snapshot, directory **link** contains both hard link and soft link. With the command **rm** we have removed both the links.

==

Tar command

tar command in Linux with examples –

The Linux ‘tar’ stands for tape archive, is used to create Archive and extract the Archive files. tar command in Linux is one of the important command which provides archiving functionality in Linux. We can use Linux tar command to create compressed or uncompressed Archive files and also maintain and modify them.

Cmd-

tar [options] [archive-file] [file or directory to be archived]

Options:

-c : Creates Archive

-x : Extract the archive

-f : creates archive with given filename

-t : displays or lists files in archived file

-u : archives and adds to an existing archive file

-v : Displays Verbose Information

-A : Concatenates the archive files

-z : zip, tells tar command that creates tar file using gzip

-j : filter archive tar file using tbzip

-W : Verify a archive file

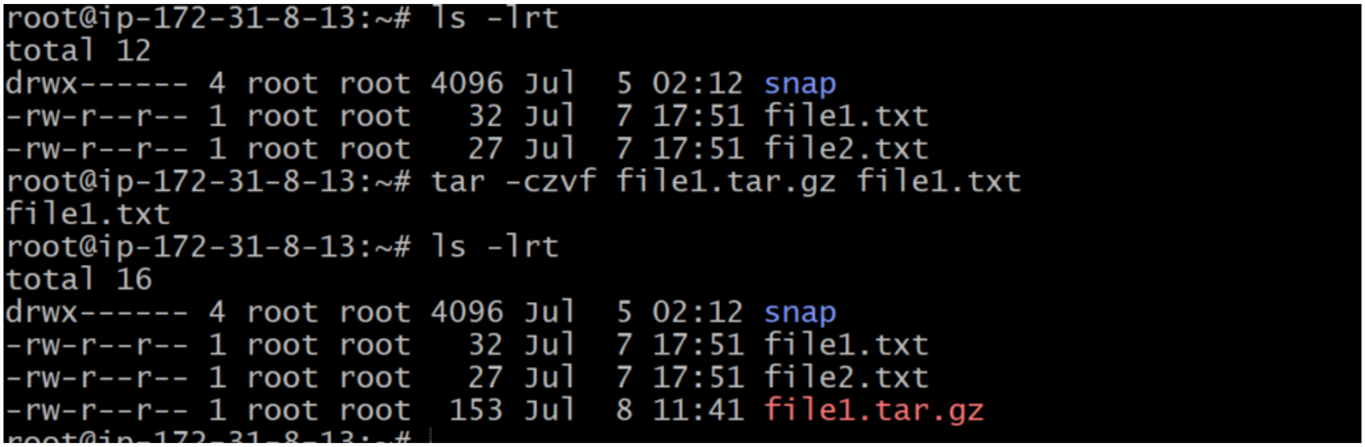
-r : update or add file or directory in already existed .tar file

What is an Archive file?

An Archive file is a file that is composed of one or more files along with metadata. Archive files are used to collect multiple data files together into a single file for easier portability and storage, or simply to compress files to use less storage space.

Ex.

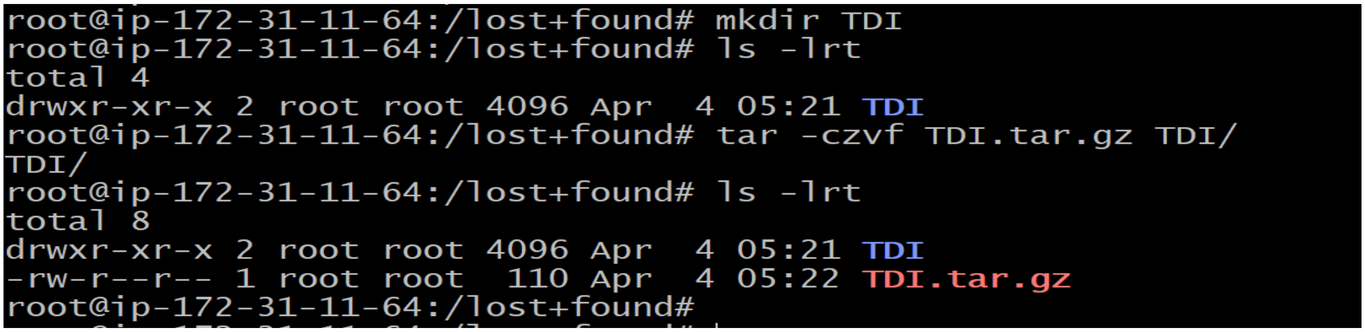
tar -czvf file1.tar.gz file1.txt



```
root@ip-172-31-8-13:~# ls -lrt
total 12
drwx------ 4 root root 4096 Jul  5 02:12 snap
-rw-r--r-- 1 root root   32 Jul  7 17:51 file1.txt
-rw-r--r-- 1 root root   27 Jul  7 17:51 file2.txt
root@ip-172-31-8-13:~# tar -czvf file1.tar.gz file1.txt
file1.txt
root@ip-172-31-8-13:~# ls -lrt
total 16
drwx------ 4 root root 4096 Jul  5 02:12 snap
-rw-r--r-- 1 root root   32 Jul  7 17:51 file1.txt
-rw-r--r-- 1 root root   27 Jul  7 17:51 file2.txt
-rw-r--r-- 1 root root  153 Jul  8 11:41 file1.tar.gz
root@ip-172-31-8-13:~#
```

Compress directory using the tar command:

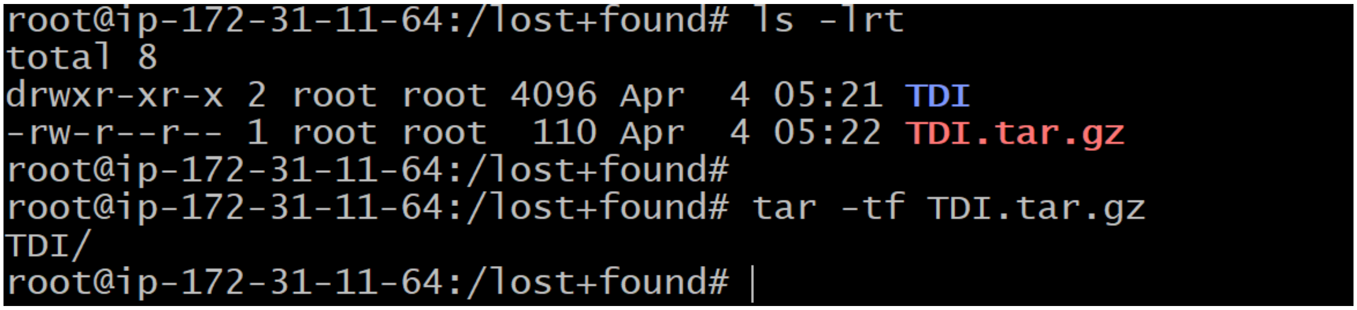
tar -czvf TDI.tar.gz TDI/



```
root@ip-172-31-11-64:/lost+found# mkdir TDI
root@ip-172-31-11-64:/lost+found# ls -lrt
total 4
drwxr-xr-x 2 root root 4096 Apr  4 05:21 TDI
root@ip-172-31-11-64:/lost+found# tar -czvf TDI.tar.gz TDI/
TDI/
root@ip-172-31-11-64:/lost+found# ls -lrt
total 8
drwxr-xr-x 2 root root 4096 Apr  4 05:21 TDI
-rw-r--r-- 1 root root  110 Apr  4 05:22 TDI.tar.gz
root@ip-172-31-11-64:/lost+found#
root@ip-172-31-11-64:/lost+found# |
```

Show the archive content:

tar -tf archive.tar.gz



```
root@ip-172-31-11-64:/lost+found# ls -lrt
total 8
drwxr-xr-x 2 root root 4096 Apr  4 05:21 TDI
-rw-r--r-- 1 root root  110 Apr  4 05:22 TDI.tar.gz
root@ip-172-31-11-64:/lost+found#
root@ip-172-31-11-64:/lost+found# tar -tf TDI.tar.gz
TDI/
root@ip-172-31-11-64:/lost+found# |
```