Linux - File Links

What are File Links in Linux?

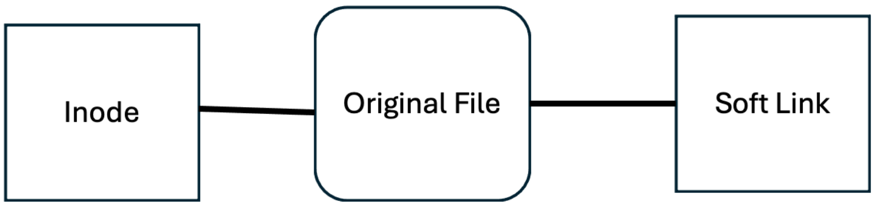
Linux links provide a mechanism to create a shortcut or alternative name for an existing file or directory, which means you can create multiple names for the same file or directory in Linux to make their access and usage very easy based on your requirements.

Linux/Unix provides two types of links:

- Soft links or Symbolic Links
- Hard Links

Symbolic Links

A symbolic in Linux is also called Soft Link or Symlink. This is a special file in Linux which points to another file or a directory. This original file or directory could be available on either the same file system or different file system. Symlink is like a shortcut in Windows which contains the path of the original file and not the contents.



You can access original file data using any of the available Symbolic link on this file and removing the symlink does not affect the target file. But If you delete the original file, then all the Symbolic links on this file will be broken and your data will not be accessible any more.

How to Create a Symbolic Link

Following is the syntax to create a symbolic link on an existing file:

```
$ ln -s  file/directory symlink
```

Here file or directory can be a full file path or directory and symlink will be the name of symbolic link which will point to the file or directory.

To start with, lets create a simple file **file.txt** as follows:

```
$ echo 'Hello, World!' > file.txt
```

Now we can create a soft link on this file using the following command:

```
$ ln -s file.txt soft-link
$ ls -l
total 4
-rw-r--r-- 1 root root 14 May  1 20:11 file.txt
lrwxrwxrwx 1 root root  8 May  1 20:11 soft-link -> file.txt
```

Here notice the first character of permission string **lrwxrwxrwx**, it is **l** which means this file is a symbolic link. Now you can access the content of file **file.txt** using **soft-link** and update it's content which will reflect in original file.txt.

How to remove a Symbolic Link

You can use either unlink or rm command to remove an existing symbolic link. As long as your are removing symbolic link, its not going to impact the original file but if you delete the original file, then it will create broken symbolic links available on that file.

Following command will delete soft-link which we had create in previous section:

```
$ unlink soft-link

$
```

Finding Broken Symbolic Links

You can use the following find command to find all the broken symbolic links:

```
$ find /path/to/directory -xtype l
```

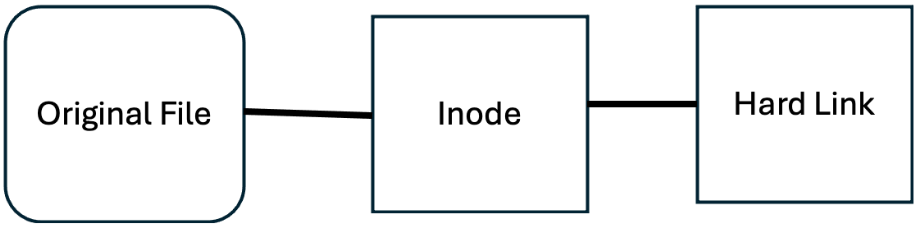
This will list all broken symlinks in the /path/to/directory directory. You can use **-delete** option to delete the found broken links

```
$ find /path/to/directory -xtype l -delete
```

~~Learn **Linux/Unix** in-depth with real-world projects through our **Linux/Unix certification course**. Enroll and become a certified expert to boost your career.~~

Hard Links

Hard links are also shortcuts to files but a hard link cannot be created for a folder or file available on a different file system.



Hard link is a mirror copy of the original file. Deleting the original file will not impact anything, because the hard link file, will act as a mirror copy of the original file.

How to Create a Hard Link

Following is the syntax to create a hard link on an existing file:

```
$ ln  filepath hardlink
```

Here **filepath** can be a full file path of the original file and **hardlink** will be the name of hard link which will point to the file. Let's create a hard link on **file.txt** file using the following command:

```
$ ln file.txt hard-link
$ ls -li
total 8
72744984 -rw-r--r-- 2 root root 14 May  1 20:11 file.txt
72744984 -rw-r--r-- 2 root root 14 May  1 20:11 hard-link
```

Here notice the inode number **72744984** which is same for both the files. Number after permission string is 2 which means, there are two copies of the same file. If you will create one more hard link on this file file.txt, then this number will become 3.

Now you can access the content of file **file.txt** using **hard-link** and update its content which will reflect in file.txt.

How to remove a Hard Link

You can use either **unlink** or **rm** command to remove an existing hard link. Removing hard link or original link will not impact another file because hard link creates a mirror copy of the original file.

Following command will delete soft-link which we had create in previous section:

```
$ unlink hard-link
$ ls -li
total 4
72744984 -rw-r--r-- 1 root root 14 May  1 20:11 file.txt
```

Symbolic Links Vs Hard Links

The two types of Linux filesystem links are hard and soft. The difference between these two types of links is significant which we have listed down here:

- Symbolic links can be created on directories as well as files where as hard links can be created only on files and not on directories.
- Symbolic links can be created on files or directories even if they are available on difference file systems where as hard links can be created on the same file system.
- Symbolic links don't inherit the original permissions from the original file. That means any permission changes made in the original file aren't reflected in the symbolic link. Files that are hard-linked together share the same inode number. When changes are made to one file, the other reflects those changes and the permissions, link count, ownership, timestamps. Though file content remain the exact same across all the linked files in both the cases.
- Symbolic links are very small in size because they are just pointer to the original content and not being a mirror of the original file, its size is only the number of bytes necessary to compose the name of the file or directory. Where as hard links consume more space because they create a mirror copy of the original file.