What is Soft Link And Hard Link In Linux?

A symbolic or soft link is an actual link to the original file, whereas a hard link is a mirror copy of the original file. If you delete the original file, the soft link has no value, because it points to a non-existent file. But in the case of hard link, it is entirely opposite. If you delete the original file, the hard link can still has the data of the original file. Because hard link acts as a mirror copy of the original file.

**Don’t miss** – [**“15 Most Downloaded free eBooks of 2016”**](http://ostechnix.tradepub.com/?pt=main&page=home.ebookspg)

In a nutshell, a soft link

* can cross the file system,
* allows you to link between directories,
* has different inodes number and file permissions than original file,
* permissions will not be updated,
* has only the path of the original file, not the contents.

A hard Link

* can’t cross the file system boundaries,
* can’t link directories,
* has the same inodes number and permissions of original file,
* permissions will be updated if we change the permissions of source file,
* has the actual contents of original file, so that you still can view the contents, even if the original file moved or removed.

Still don’t get it? Well, allow me to show you some practical examples.

Creating Soft, or Symbolic Link

Let us create an empty directory called **“test”**.

$ mkdir test

Change to the “test” directory:

$ cd test

Now, create a new file called **source.file** with some data as shown below.

$ echo "Welcome to OSTechNix" >source.file

Let us view the data of the source.file.

**$ cat source.file**

Welcome to OSTechNix

Well, the source.file has been created.

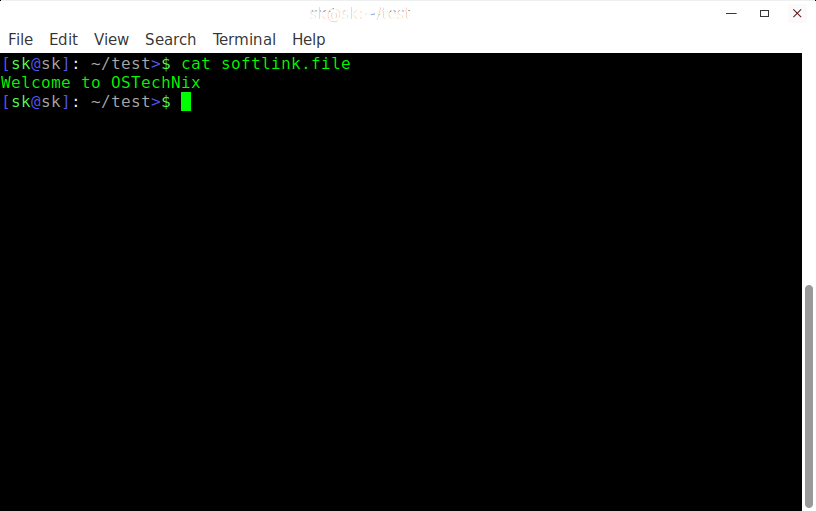
Now, create the a symbolic or soft link to the source.file.

To do so, run:

$ ln -s source.file softlink.file

Let us view the data of softlink.file.

$ cat softlink.file

[](http://www.ostechnix.com/wp-content/uploads/2016/12/sk@sk-test_008.png)

As you see in the above output, softlink.file displays the same data as source.file.

Let us check the inodes and permissions of softlink.file and source.file.

$ ls -lia

**Sample output:**

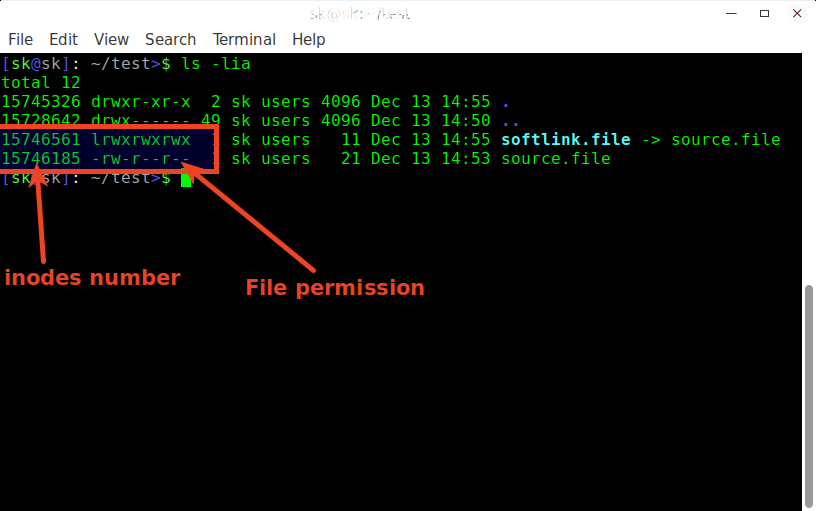
total 12

15745326 drwxr-xr-x 2 sk users 4096 Dec 13 14:55 .

15728642 drwx------ 49 sk users 4096 Dec 13 14:50 ..

**15746561 lrwxrwxrwx** 1 sk users 11 Dec 13 14:55 softlink.file -> source.file

**15746185 -rw-r--r--** 1 sk users 21 Dec 13 14:53 source.file

[](http://www.ostechnix.com/wp-content/uploads/2016/12/sk@sk-test_006.png)

As we see in the above screenshot, even though the softlink.file has same contents as source.file, the **inodes number (15746561**vs**15746185)** and **file permissions (lrwxrwxrwx**vs**lrwxrwxrwx)** are different. Hence, it is proved that soft link doesn’t share the same inodes number and permissions of original file.

Now, remove the original file (i.e source.file) and see what happens.

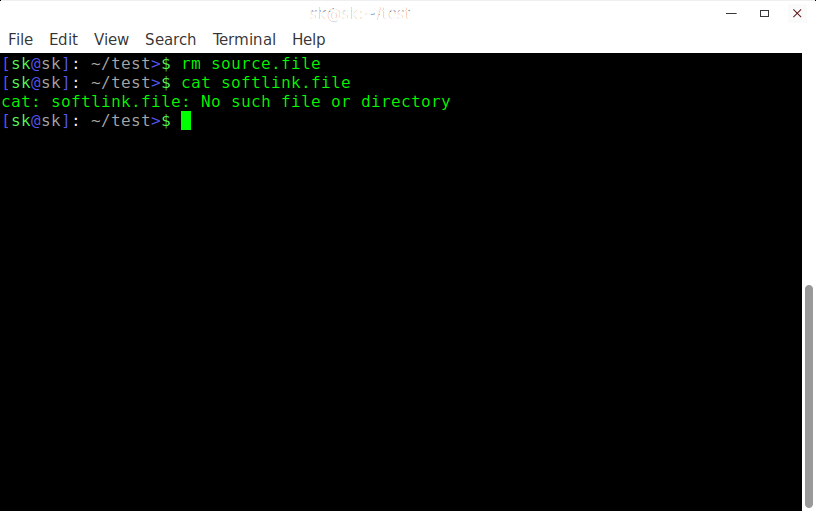
$ rm source.file

Check output of softlink.file using command:

$ cat softlink.file

**Sample output:**

cat: softlink.file: No such file or directory

[](http://www.ostechnix.com/wp-content/uploads/2016/12/sk@sk-test_009.png)

As you see above, there is no such file or directory called softlink.file after we we removed the original file (i.e source.file). So, now we understand that soft link is just a link that points to the original file. The softlink is like a shortcut to a file. If you remove the file, the shortcut is useless.

As you already know, if you remove the soft link, the original file will still present.

Creating Hard Link

Create a file called **source.file** with some contents as shown below.

$ echo "Welcome to OSTechNix" >source.file

Let us verify the contents of the file.

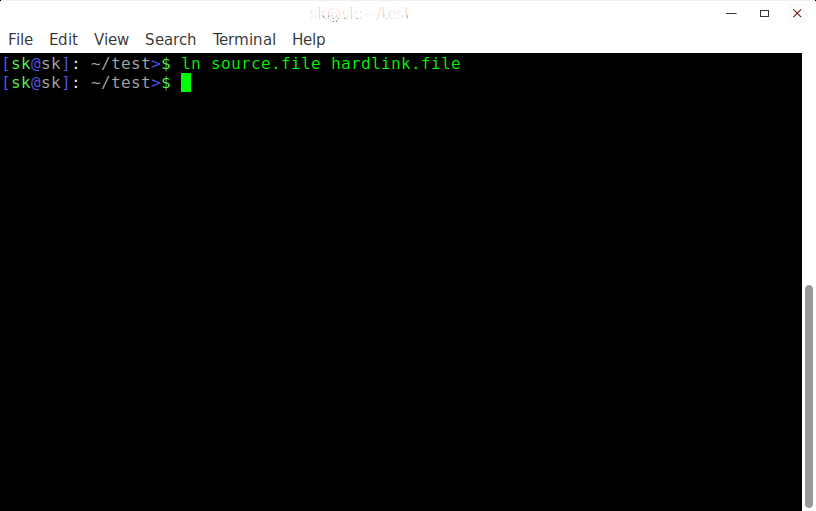
**$ cat source.file**

Welcome to OSTechNix

source.file has been created now.

Now, let us create the hard link to the source.file as shown below.

$ ln source.file hardlink.file

[](http://www.ostechnix.com/wp-content/uploads/2016/12/sk@sk-test_011.png)

Check the contents of hardlink.file.

**$ cat hardlink.file**

Welcome to OSTechNix

You see the hardlink.file displays the same data as source.file.

Let us check the inodes and permissions of softlink.file and source.file.

$ ls -lia

**Sample output:**

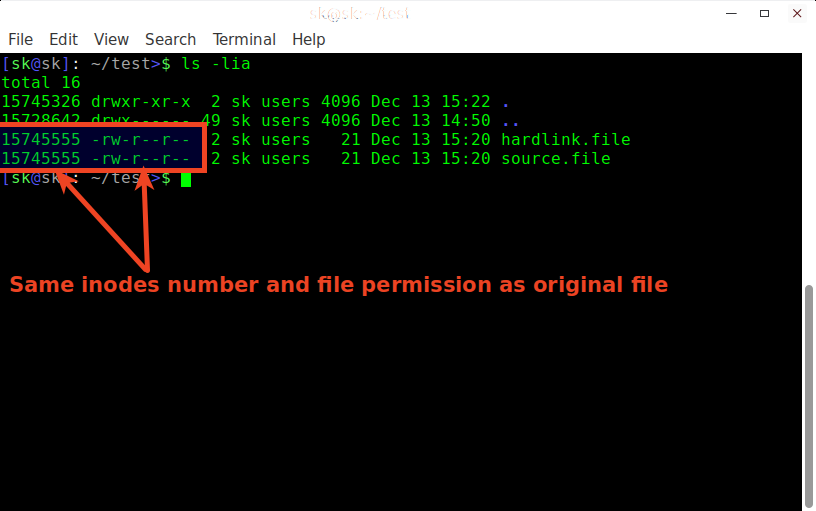
total 16

15745326 drwxr-xr-x 2 sk users 4096 Dec 13 15:22 .

15728642 drwx------ 49 sk users 4096 Dec 13 14:50 ..

**15745555 -rw-r--r--** 2 sk users 21 Dec 13 15:20 **hardlink.file**

**15745555 -rw-r--r--** 2 sk users 21 Dec 13 15:20 **source.file**

[](http://www.ostechnix.com/wp-content/uploads/2016/12/sk@sk-test_013.png)

Now, we see that both hardlink.file and source.file have the same the **inodes number (15745555)** and **file permissions (-rw-r–r–)**. Hence, it is proved that hard link file shares the same inodes number and permissions of original file.

**Note:** If we change the permissions on source.file, the same permission will be applied to the hardlink.file as well.

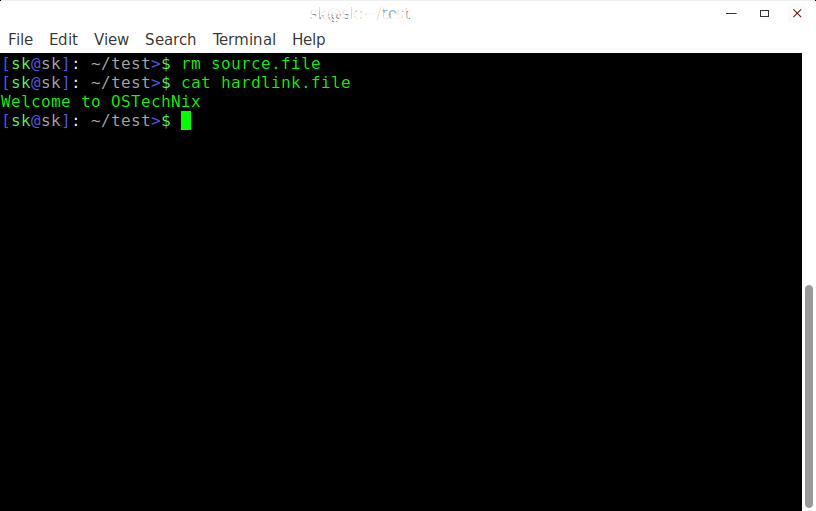
Now, remove the original file (i.e source.file) and see what happens.

$ rm source.file

Check contents of hardlink.file using command:

$ cat hardlink.file

**Sample output:**

[](http://www.ostechnix.com/wp-content/uploads/2016/12/sk@sk-test_015.png)

As you see above, even if I deleted the source file, I can view contents of the hardlink.file. Hence, it is proved that Hard link shares the same inodes number, the permissions and data of the original file.

That’s all for now. Hope you got a basic idea about symbolic or soft link and hard link. If you find this guide useful, share it on your social, professional networks and support us.

More good stuffs to come. Stay tuned!

Cheers!