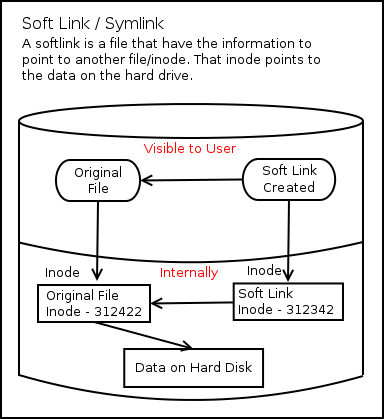
What is a Soft Link or Symbolic Link or Symlink ?

Symbolic links or Symlinks are the easiest to understand, because for sure you have used them, at least when you were using Windows. Soft links are very similar to what we say “Shortcut” in windows, is a way to link to a file or directory. Symlinks doesn’t contain any information about the destination file or contents of the file, instead of that, it simply contains the pointer to the location of the destination file. In more technical words, in soft link, a new file is created with a new inode, which have the pointer to the inode location of the original file. This can be better explained with a diagram:

[](http://www.geekride.com/?attachment_id=2013)

Symbolic links are created with the “ln” command in linux. The syntax of the command is:

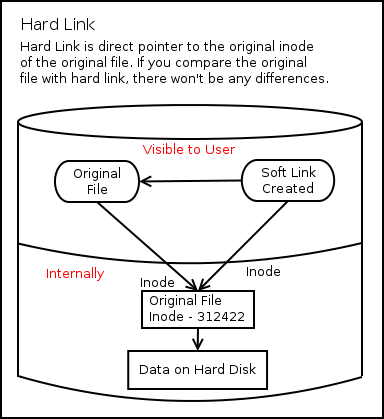
$ ln -s   
-s = This flag tells to create a symlink (if you don’t use this it will create a hard link, which we will talk about soon).

For Example, if you want to create a soft link of one fo your favorite application, like gedit, on your desktop, use the command like this:

$ ln -s /usr/bin/gedit ~/Desktop/gedit

What is a Hard Link ?

Hard link is a bit different object when compared to a symlink. In softlink a new file and a new Inode is created, but in hard link, only an entry into directory structure is created for the file, but it points to the inode location of the original file. Which means there is no new inode creation in the hard link. This can be explained like this:

[](http://www.geekride.com/?attachment_id=2033)

So, in hard link, you are referencing the inode directly on the disk, which means that there should be a way to know how many hard links exist to a file. For the same, in the inode information, you have an option for “links”, which will tell how many links exists to a file. You can find the same information by using this command:

$ stat <file name>

$ stat 01  
Size: 923383 Blocks: 1816 IO Block: 4096 regular file  
Device: 803h/2051d Inode: 12684895 **Links: 3**  
Access: (0644/-rw-r–r–) Uid: ( 0/ root) Gid: ( 0/ root)  
Access: 2012-09-07 01:46:54.000000000 -0500  
Modify: 2012-04-27 06:22:02.000000000 -0500  
Change: 2012-04-27 06:22:02.000000000 -0500

In this example, it means that the specific file have 2 hard links, which makes the count to 3.

You can create a hard link with the same command “ln” like this

# ln

So, to create a hard link of gedit program on your desktop, you will use the command like this:

# ln /usr/bin/gedit ~/Desktop/gedit

Now, the bigger question is, who will decide what is better and when to use soft link or hard link

When to use Soft Link:

1. Link across filesystems: If you want to link files across the filesystems, you can only use symlinks/soft links.
2. Links to directory: If you want to link directories, then you must be using Soft links, as you can’t create a hard link to a directory.

When to use Hard Link:

1. Storage Space: Hard links takes very negligible amount of space, as there are no new inodes created while creating hard links. In soft links we create a file which consumes space (usually 4KB, depending upon the filesystem)
2. Performance: Performance will be slightly better while accessing a hard link, as you are directly accessing the disk pointer instead of going through another file.
3. Moving file location: If you move the source file to some other location on the same filesystem, the hard link will still work, but soft link will fail.
4. Redundancy: If you want to make sure safety of your data, you should be using hard link, as in hard link, the data is safe, until all the links to the files are deleted, instead of that in soft link, you will lose the data if the master instance of the file is deleted.

The above points gives you a small idea where to use what, but doesn’t tell you that those are the only options. Everything depends on your setup.

How file is deleted having hard links:

So, as it’s pretty clear from the above article that hard links are just the reference to the main file location, and even if you delete one link, the data will still be intact. So, to remove a hard link, you need to remove all the links, which are referring to the file. Once the “link count” goes to “0”, then the inode is removed by the filesystem, and file is deleted.

FAQs:

Q. What is the one line answer to the question “What is the main difference between hard links & soft links” ?

A. A softlink will have a different Inode number than the source file, which will be having a pointer to the source file but hardlink will be using the same Inode number as the source file.

Q. How can I find all the Soft Links in my system ?

A. Use this command for the same “find /etc -type l -exec ls -li {} \;”

Q. How can I find all the files having Hard Links in my system ?

A. Use this command for the same “find / -links +2 -type f -exec ls -li {} \;”

Q. How to find whether a file is a softlink ?

A. Simply using this command “ls -l” will tell you whether a file is pointing to some other file or not.

Q. How to check whether a file have any softlink pointing to it ?

A. Till now, I am not aware of any way to do that. If I will find any, I will surely update my post.

Q. How can I find out the source file of a hard link ?

A. No, you can’t find out the source file of a hard link. Once hard link is created, there is no way to tell which was the first file created.

Q. Can I make a Soft link to a Hard link and Vice Versa ?

A. Yes, both soft links and hard links acts as normal files of the file system, so you can do both.

To understand the above article in much better way, you should go through the following article, which will give you much clear picture about the inodes.  
[Understanding Linux / Unix Filesystem Inodes](http://www.geekride.com/understanding-unix-linux-filesystem-inodes/)