Hard and Soft (Symbolic) Links

The **ln** program can be used to create hard links and (with the **-s** option) soft links, also known as symbolic links or symlinks.

Suppose **file1** already exists. A hard link to it is created with the command:

```
$ ln file1 file2
$ ls -li file1 file2
84 -rw-rw-r-- 2 coop coop 1551 Jun 16 16:28 file1
84 -rw-rw-r-- 2 coop coop 1551 Jun 16 16:28 file2
```

Note that two files now appear to exist. However, a closer inspection of the file listing shows that this is not quite true.

The **-i** option to **ls** prints out in the first column the inode number, which in UNIX filesystems is unique for each file object. This field is the same for both of these files; what is really going on here is that it is only one file, but it has two names. The 2 that appears later in the **ls** listing indicates that this inode has two links to it.

Let's consider another example:

```
$ ls -li /bin/g*zip
194339 -rwxr-xr-x 3 root root 62872 Jan 14 13:06 /bin/gunzip
194339 -rwxr-xr-x 3 root root 62872 Jan 14 13:06 /bin/gzip
```

which shows that **gzip** and **gunzip** are both only one program and the executable is only one file; whether it compresses or decompresses files depends on which name it is invoked with, which is always available as **argv[0]** when the program executes.

Hard links are very useful and they save space, but you have to be careful with their use, sometimes in subtle ways. For one thing, if you remove either **file1** or **file2** in the above example, the inode object (and the remaining file name) will remain, which is generally desirable.

However, if you edit one of the files, exactly what happens depends on your editor; most editors, including vi and emacs, will retain the link by default, but it is possible that modifying one of the names may break the link and result in the creation of two objects.

Symbolic (or soft) links are created with the **-s** option, as in:

```
$ ln -s file1 file2

$ ls -li file1 file2

84 -rw-rw-r-- 1 coop coop 1551 Jun 16 16:28 file1

85 lrwxrwxrwx 1 coop coop    5 Jun 16 16:43 file2 -> file1
```

Notice **file2** no longer appears to be a regular file, and it clearly points to **file1** and has a different inode number.

Symbolic links take no extra space on the filesystem (unless their names are very long), as they are stored directly in the directory inode. They are extremely convenient, as they can easily be modified to point to different places.

Unlike hard links, soft links can point to objects even on different filesystems (or partitions), which may or not be currently mounted or even exist. In the case where the link does not point to a currently mounted or existing object, one obtains a dangling link.

The **symlinks** utility can be used to examine current symbolic links, as in:

```
c7:/usr/share/gdb/auto-load>symlinks -rv .

relative: /usr/share/gdb/auto-load/lib -> usr/lib

dangling: /usr/share/gdb/auto-load/sbin -> usr/sbin

relative: /usr/share/gdb/auto-load/lib64 -> usr/lib64

dangling: /usr/share/gdb/auto-load/bin -> usr/bin
```

easily be modified to point to different places.