Symbolic Link Files

[Hard link and Soft link]

Source : Devconnected

Additional Reference :
http://linux-training.be/security/ch09.html#:~:text=An%20inode%20is%20a%20data,of%20the%20file%2C%20and%20more.

Ashwini Mathur [Assistant Professor]

We are used to thinking about a directory containing files. This is really an illusion. Directories do not contain files. The data of the files is not stored in the directory.

A directory is really **just a file**. It's a special file with special rules (you can't just type "cp /dev/null directory" to erase it. It's got special bits to make sure a mere mortal can't mess it up. Because if a file system gets corrupted, then you can say goodbye to your data. On older UNIX systems, you actually could "read" the contents, using 'cat .', of a directory. But let me get back to that in a second...

A Unix **file** is "stored" in two different parts of the disk - the data blocks and the inodes. (I won't get into superblocks and other esoteric information.) The data blocks contain the "contents" of the file. The information **about** the file is stored elsewhere - in the inode.

Both the inodes and data blocks are stored in a "filesystem" which is how a disk partition is organized. But these inodes are strange and confusing. Let me give you an introduction.

What is in an inode?

inode number

Each **inode** has a unique number (the inode number). You can see the **inode** numbers with the **ls -li** command.

The inode contains the following pieces of information

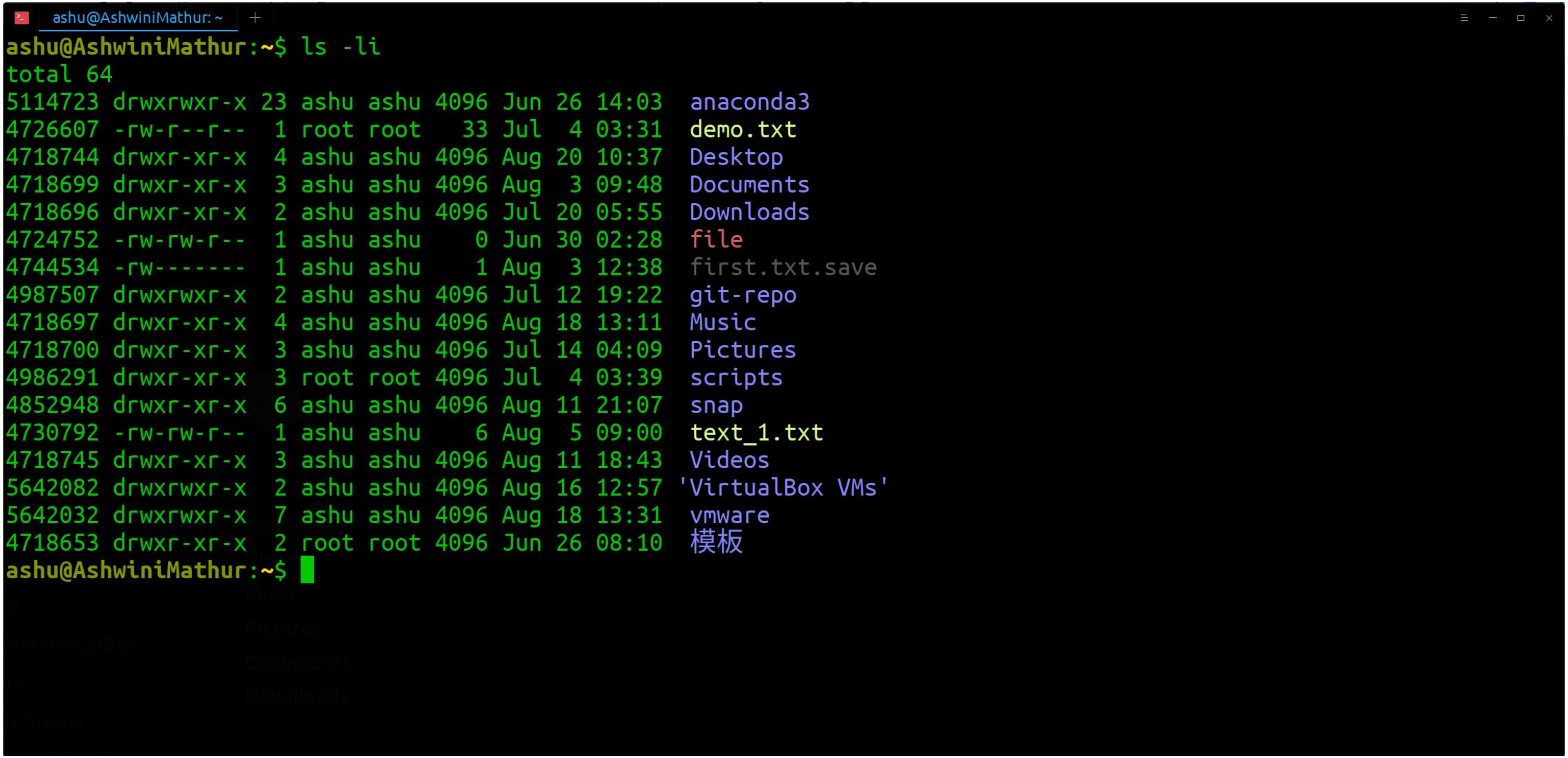
- Mode/permission (protection)
- Owner ID
- Group ID
- Size of file
- Number of hard links to the file
- Time last accessed
- Time last modified
- Time inode last modified

"ls -i" lists the inode of a file

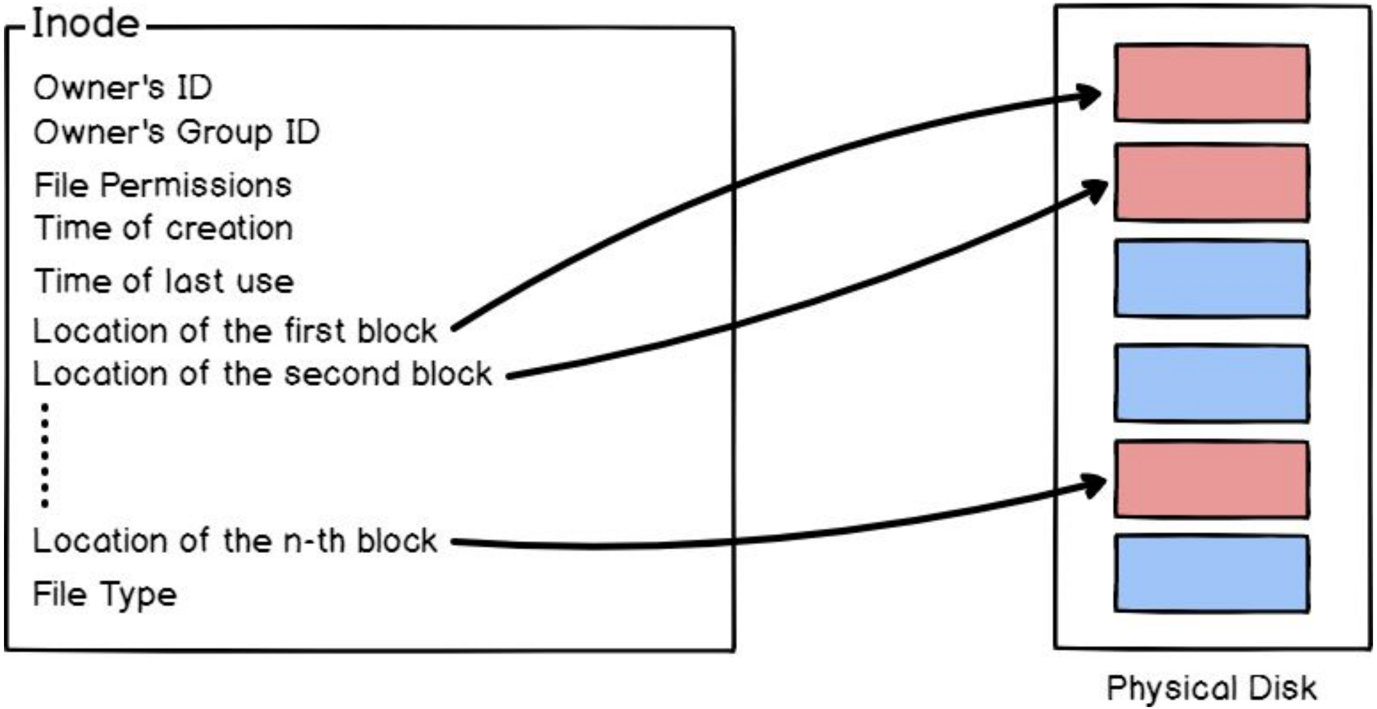
Normal Unix/Linux/MacOS users aren't even aware that inodes exist. But there's an easy way to discover them - using the "ls -i" command. Let's look at the root file system:

```
ls -lai
```

Use head or tail command with pipe to control the veiw of the files.



```
ashu@AshwiniMathur:~$ ls -li
total 64
5114723 drwxrwxr-x 23 ashu ashu 4096 Jun 26 14:03  anaconda3
4726607 -rw-r--r--  1 root root   33 Jul  4 03:31  demo.txt
4718744 drwxr-xr-x  4 ashu ashu 4096 Aug 20 10:37  Desktop
4718699 drwxr-xr-x  3 ashu ashu 4096 Aug  3 09:48  Documents
4718696 drwxr-xr-x  2 ashu ashu 4096 Jul 20 05:55  Downloads
4724752 -rw-rw-r--  1 ashu ashu    0 Jun 30 02:28  file
4744534 -rw-------  1 ashu ashu    1 Aug  3 12:38  first.txt.save
4987507 drwxrwxr-x  2 ashu ashu 4096 Jul 12 19:22  git-repo
4718697 drwxr-xr-x  4 ashu ashu 4096 Aug 18 13:11  Music
4718700 drwxr-xr-x  3 ashu ashu 4096 Jul 14 04:09  Pictures
4986291 drwxr-xr-x  3 root root 4096 Jul  4 03:39  scripts
4852948 drwxr-xr-x  6 ashu ashu 4096 Aug 11 21:07  snap
4730792 -rw-rw-r--  1 ashu ashu    6 Aug  5 09:00  text_1.txt
4718745 drwxr-xr-x  3 ashu ashu 4096 Aug 11 18:43  Videos
5642082 drwxrwxr-x  2 ashu ashu 4096 Aug 16 12:57 'VirtualBox VMs'
5642032 drwxrwxr-x  7 ashu ashu 4096 Aug 18 13:31  vmware
4718653 drwxr-xr-x  2 root root 4096 Jun 26 08:10  模板
ashu@AshwiniMathur:~$ 
```



Inode

Owner's ID

Owner's Group ID

File Permissions

Time of creation

Time of last use

Location of the first block

Location of the second block

Location of the n-th block

File Type

Physical Disk

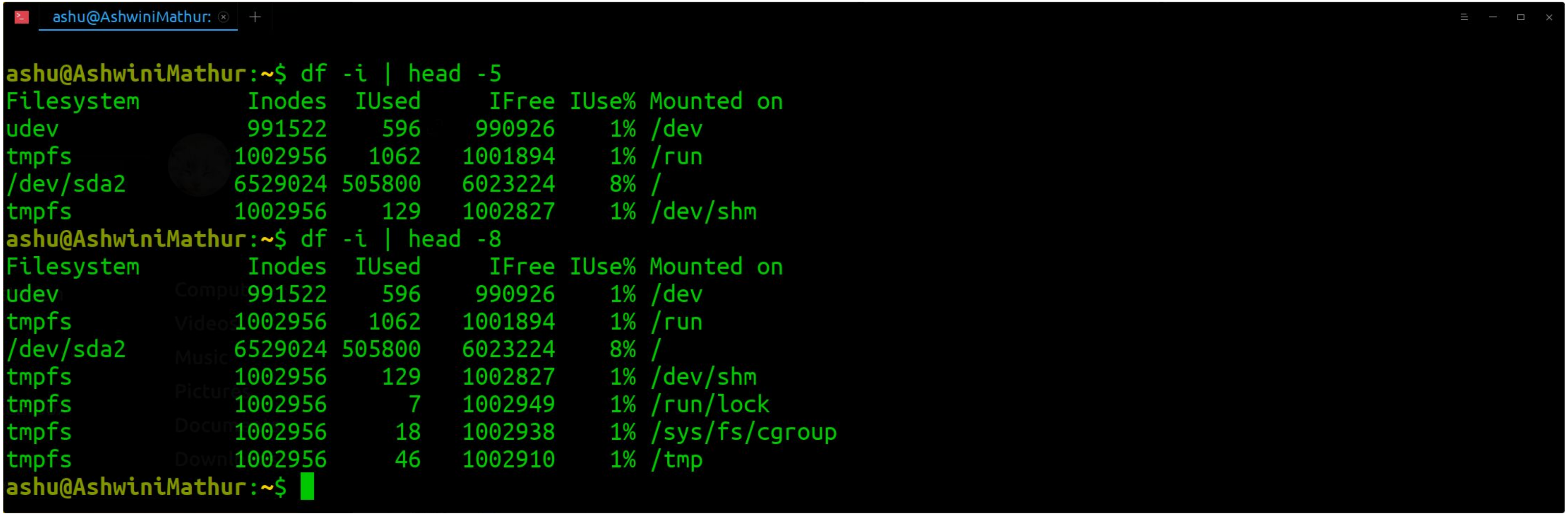
inode contents

An **inode** is a data structure that contains metadata about a file. When the file system stores a new file on the hard disk, it stores not only the contents (data) of the file, but also extra properties like the name of the file, the creation date, its permissions, the owner of the file, and more. All this information (except the name of the file and the contents of the file) is stored in the **inode** of the file.

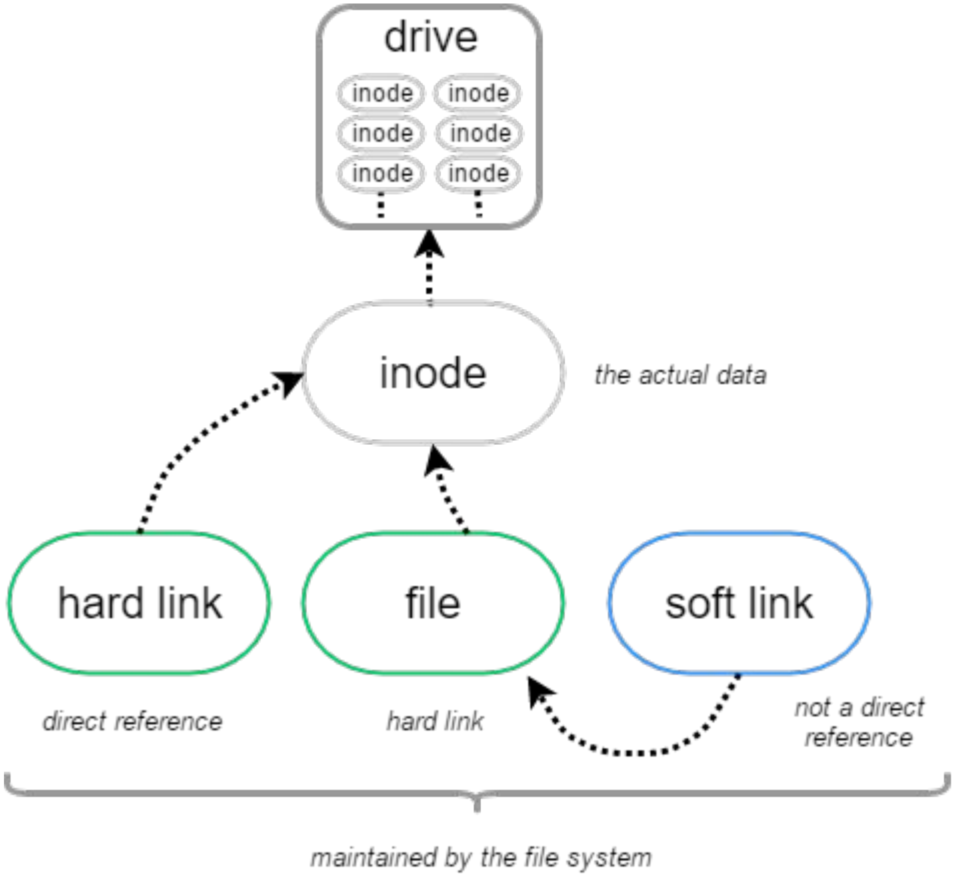
The **ls -l** command will display some of the **inode contents,**

inode table

The **inode table** contains all of the **inodes** and is created when you create the file system (with **mkfs**). You can use the **df -i** command to see how many **inodes** are used and free on mounted file systems.



```
ashu@AshwiniMathur:~$ df -i | head -5
Filesystem        Inodes  IUsed     IFree IUse% Mounted on
udev              991522    596    990926    1% /dev
tmpfs            1002956   1062   1001894    1% /run
/dev/sda2        6529024 505800   6023224    8% /
tmpfs            1002956    129   1002827    1% /dev/shm
ashu@AshwiniMathur:~$ df -i | head -8
Filesystem        Inodes  IUsed     IFree IUse% Mounted on
udev              991522    596    990926    1% /dev
tmpfs            1002956   1062   1001894    1% /run
/dev/sda2        6529024 505800   6023224    8% /
tmpfs            1002956    129   1002827    1% /dev/shm
tmpfs            1002956      7   1002949    1% /run/lock
tmpfs            1002956     18   1002938    1% /sys/fs/cgroup
tmpfs            1002956     46   1002910    1% /tmp
ashu@AshwiniMathur:~$ 
```



drive

inode

inode

inode

inode

inode

inode

inode

the actual data

hard link

file

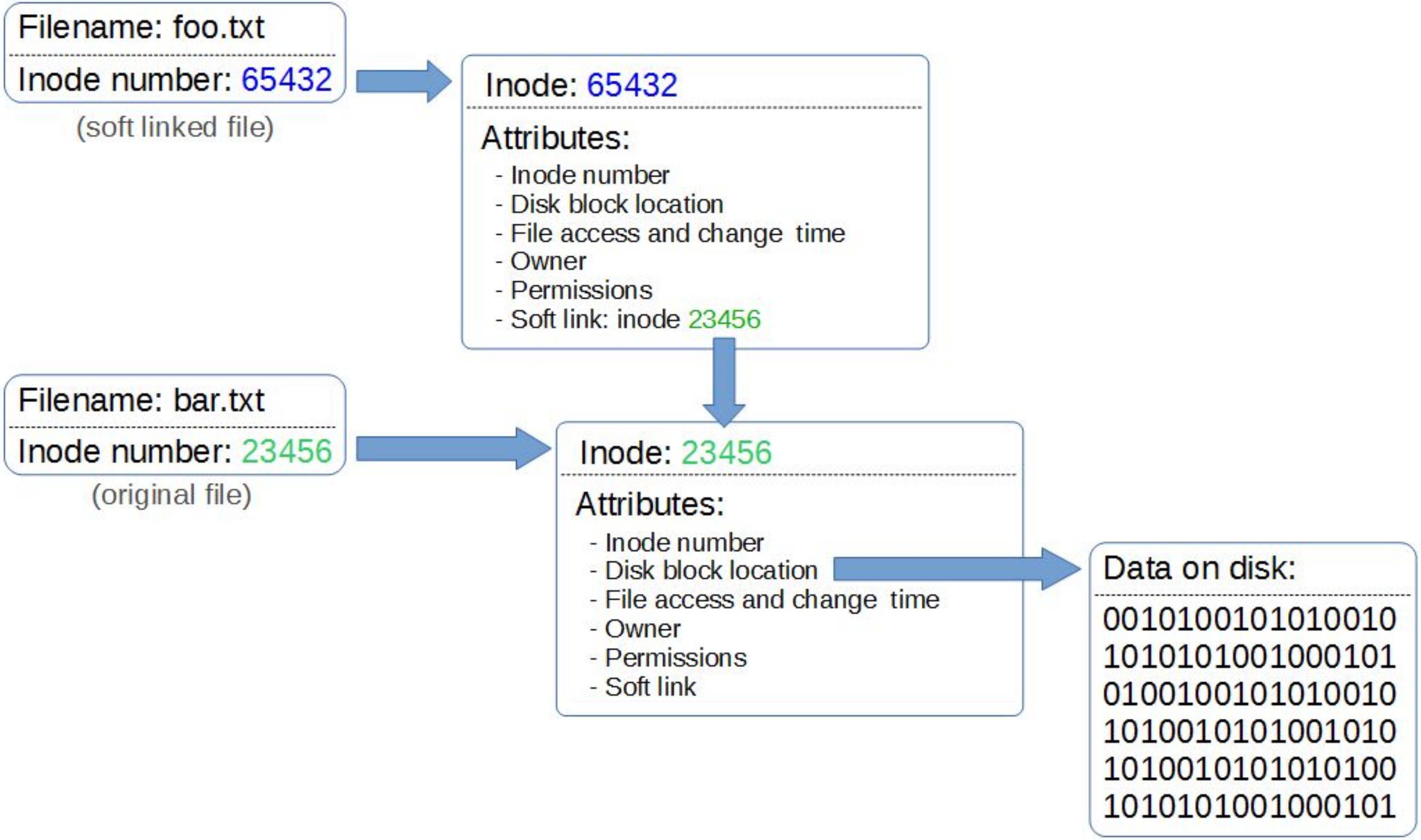
soft link

direct reference

hard link

not a direct reference

maintained by the file system



Filename: foo.txt

Inode number: 65432

(soft linked file)

Inode: 65432

Attributes:

- Inode number
- Disk block location
- File access and change time
- Owner
- Permissions
- Soft link: inode 23456

Filename: bar.txt

Inode number: 23456

(original file)

Inode: 23456

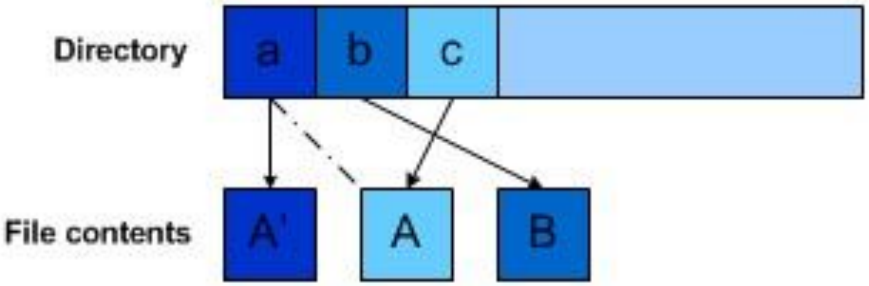
Attributes:

- Inode number
- Disk block location
- File access and change time
- Owner
- Permissions
- Soft link

Data on disk:

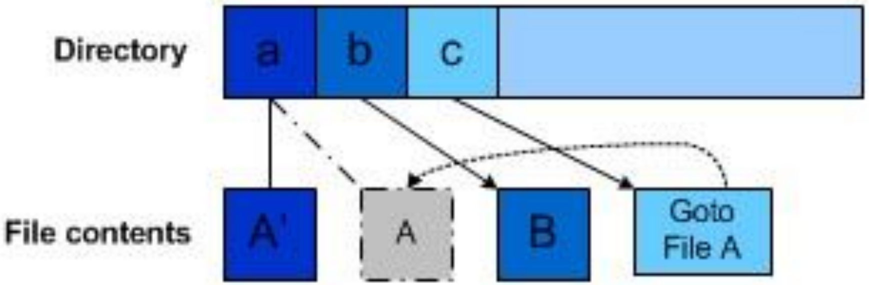
```
0010100101010010
1010101001000101
0100100101010010
1010010101001010
1010010101010100
1010101001000101
```

Hard Links



Deleting filename "a" doesn't delete file A if c is also hard-linked to it.

Symlinks

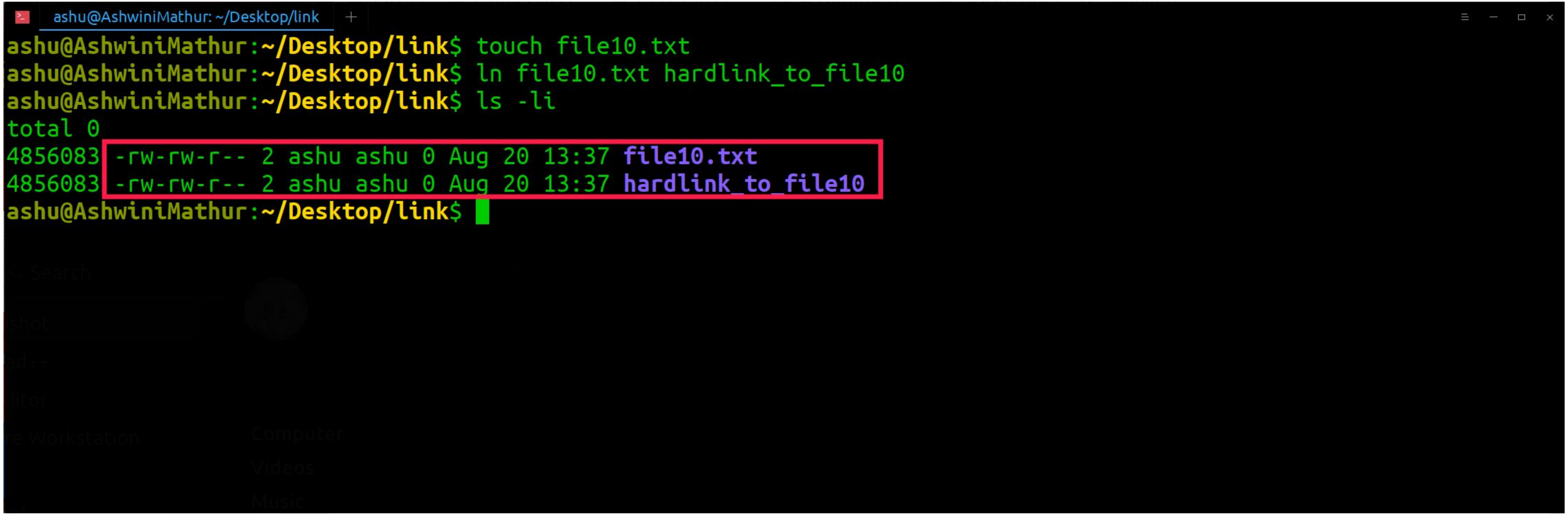


After deleting A , the symbolic link through filename "c" is no longer available --> dangling-link problem

hard links

creating hard links

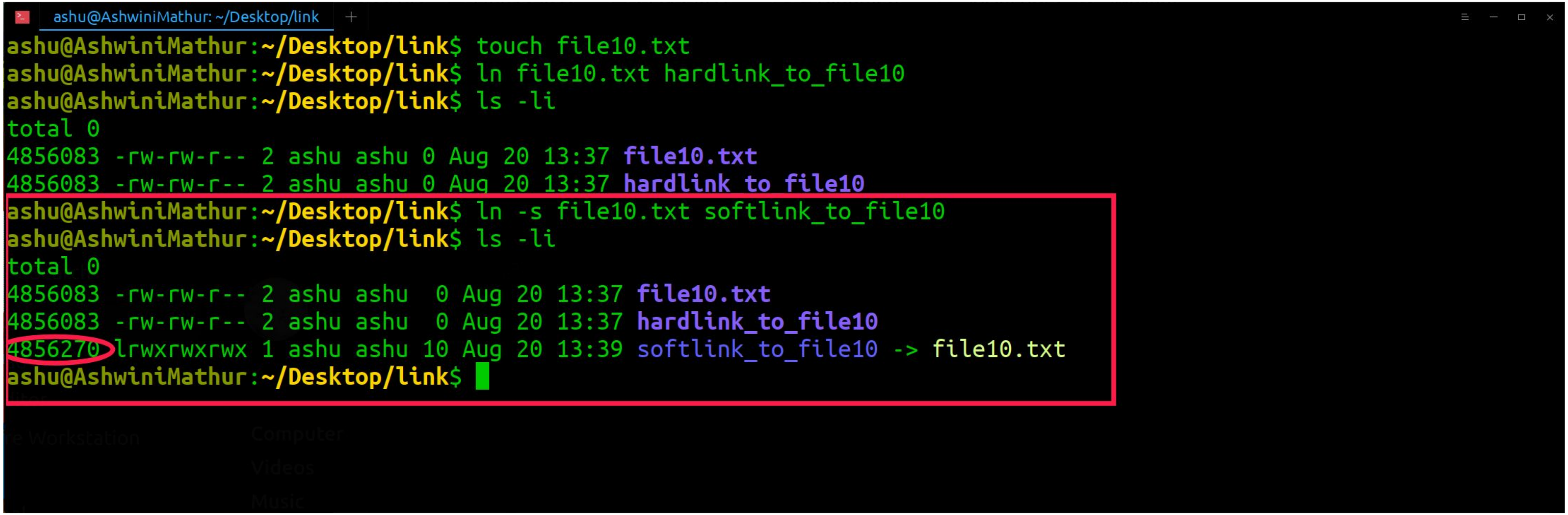
When we create a **hard link** to a file with **ln**, an extra entry is added in the directory. A new file name is mapped to an existing inode.



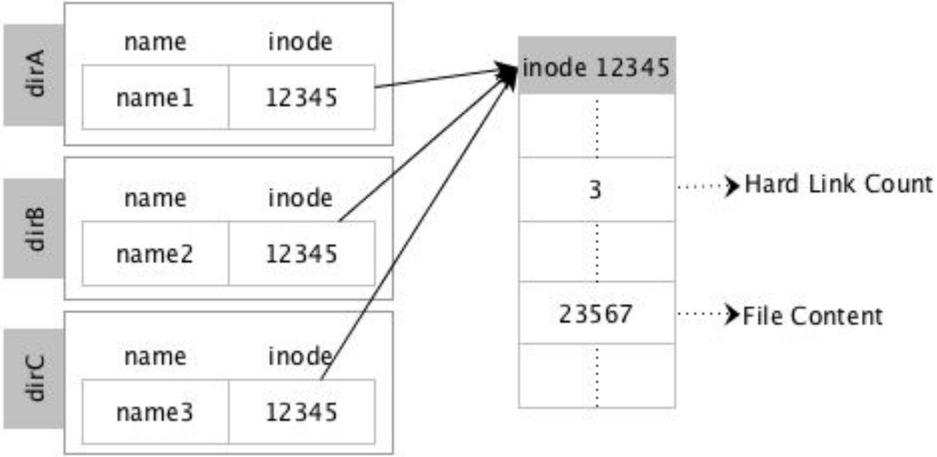
```
ashu@AshwiniMathur:~/Desktop/link$ touch file10.txt
ashu@AshwiniMathur:~/Desktop/link$ ln file10.txt hardlink_to_file10
ashu@AshwiniMathur:~/Desktop/link$ ls -li
total 0
4856083 -rw-rw-r-- 2 ashu ashu 0 Aug 20 13:37 file10.txt
4856083 -rw-rw-r-- 2 ashu ashu 0 Aug 20 13:37 hardlink_to_file10
ashu@AshwiniMathur:~/Desktop/link$ 
```

symbolic links

Symbolic links (sometimes called **soft links**) do not link to inodes, but create a name to name mapping. Symbolic links are created with **ln -s**. As you can see below, the **symbolic link** gets an inode of its own.



```
ashu@AshwiniMathur:~/Desktop/link$ touch file10.txt
ashu@AshwiniMathur:~/Desktop/link$ ln file10.txt hardlink_to_file10
ashu@AshwiniMathur:~/Desktop/link$ ls -li
total 0
4856083 -rw-rw-r-- 2 ashu ashu 0 Aug 20 13:37 file10.txt
4856083 -rw-rw-r-- 2 ashu ashu 0 Aug 20 13:37 hardlink_to_file10
ashu@AshwiniMathur:~/Desktop/link$ ln -s file10.txt softlink_to_file10
ashu@AshwiniMathur:~/Desktop/link$ ls -li
total 0
4856083 -rw-rw-r-- 2 ashu ashu  0 Aug 20 13:37 file10.txt
4856083 -rw-rw-r-- 2 ashu ashu  0 Aug 20 13:37 hardlink_to_file10
4856270 lrwxrwxrwx 1 ashu ashu 10 Aug 20 13:39 softlink_to_file10 -> file10.txt
ashu@AshwiniMathur:~/Desktop/link$ 
```



dirA

name	inode
name1	12345

dirB

name	inode
name2	12345

dirC

name	inode
name3	12345

inode 12345

3 → Hard Link Count

23567 → File Content

Practice Assignment

1. Create two files named winter.txt and summer.txt, put some text in them.

2. Create a hard link to winter.txt named hlwinter.txt.

3. Display the inode numbers of these three files, the hard links should have the same inode.

4. Use the find command to list the two hardlinked files

5. Everything about a file is in the inode, except two things : name them!

6. Create a symbolic link to summer.txt called slsummer.txt.

7. In the second column of ls -l command. What does this information tell you ?

8. Look at the directories /etc/init.d/ /etc/rc2.d/ /etc/rc3.d/ ... do you see the links ?

9. Look in /lib with ls -l...

10. Use **find** to look in your home directory for regular files that do not(!) have one hard link.