Operating Systems

Section 04

Prepared by Eng. Ahmed Adel

Linux Links

INODE, Hard or symbolic links, build a Playground .

INODE(Index Node)

- Every file in the system has an inode(index node).
- Contains all file information except the file contents and name.
- Just like a personal ID or a password (with out a name !).
- They contain the following
  - Inode number
  - File size
  - Owner information
  - Permissions
  - File type
  - Number of links
  - ETC

Types Of Links

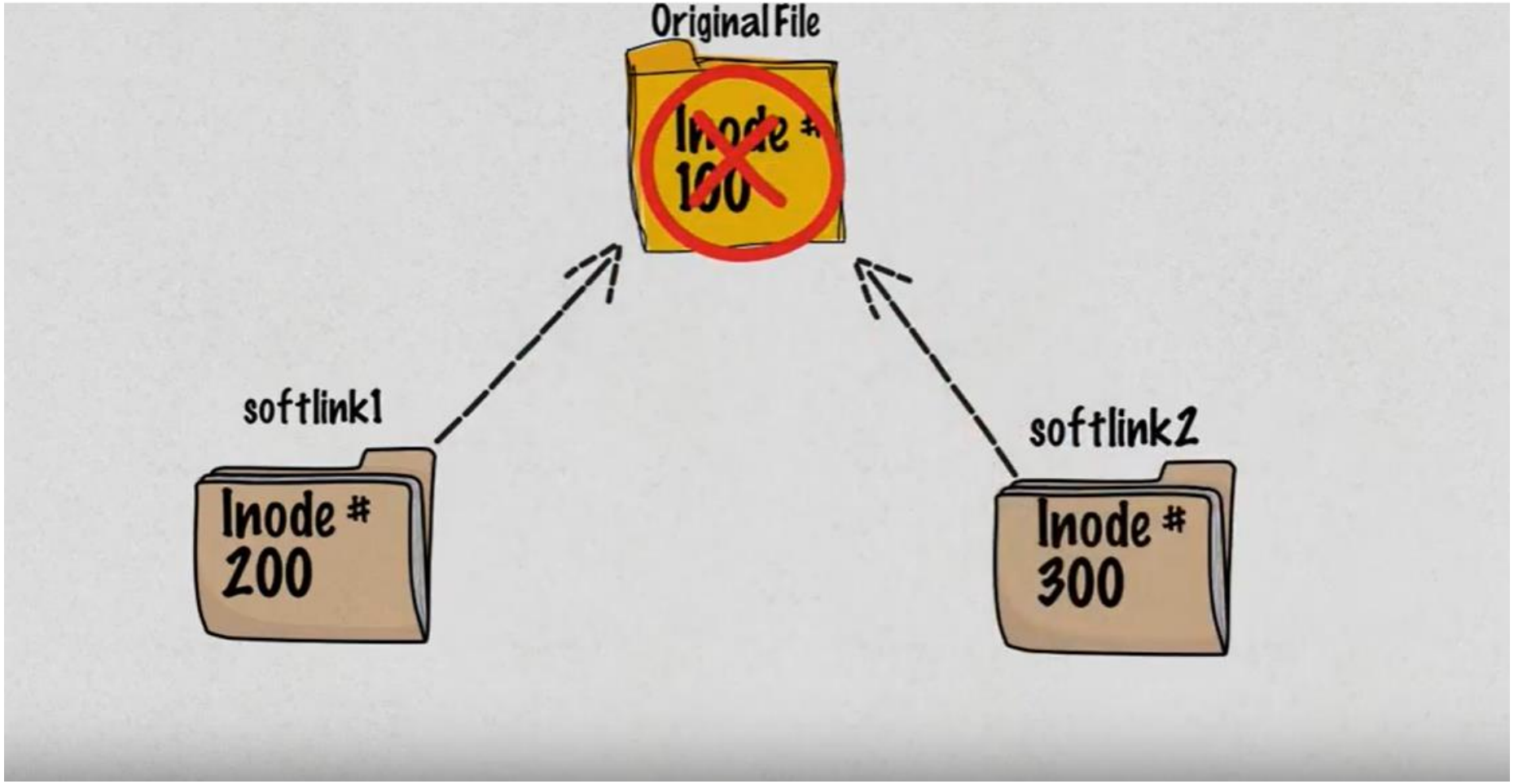
- There are Two types of links Soft (Symbolic) link or hard link.
- The following creates a hard link
  - **ln *file link***

- The following creates a Symbolic link
  - **ln -s *item link***
  - Item is either a file or directory.

Symbloic Links

- It is a pointer to the original file.
- Is also known as Soft links.
- Just like a shortcut in windows.
- The file size is smaller than the file size of the original file.
- Different inode number of the original file.
- If we delete the original file the soft links will become useless.
- If delete the symbolic link only the link is deleted not the original file.

Symbloic Links

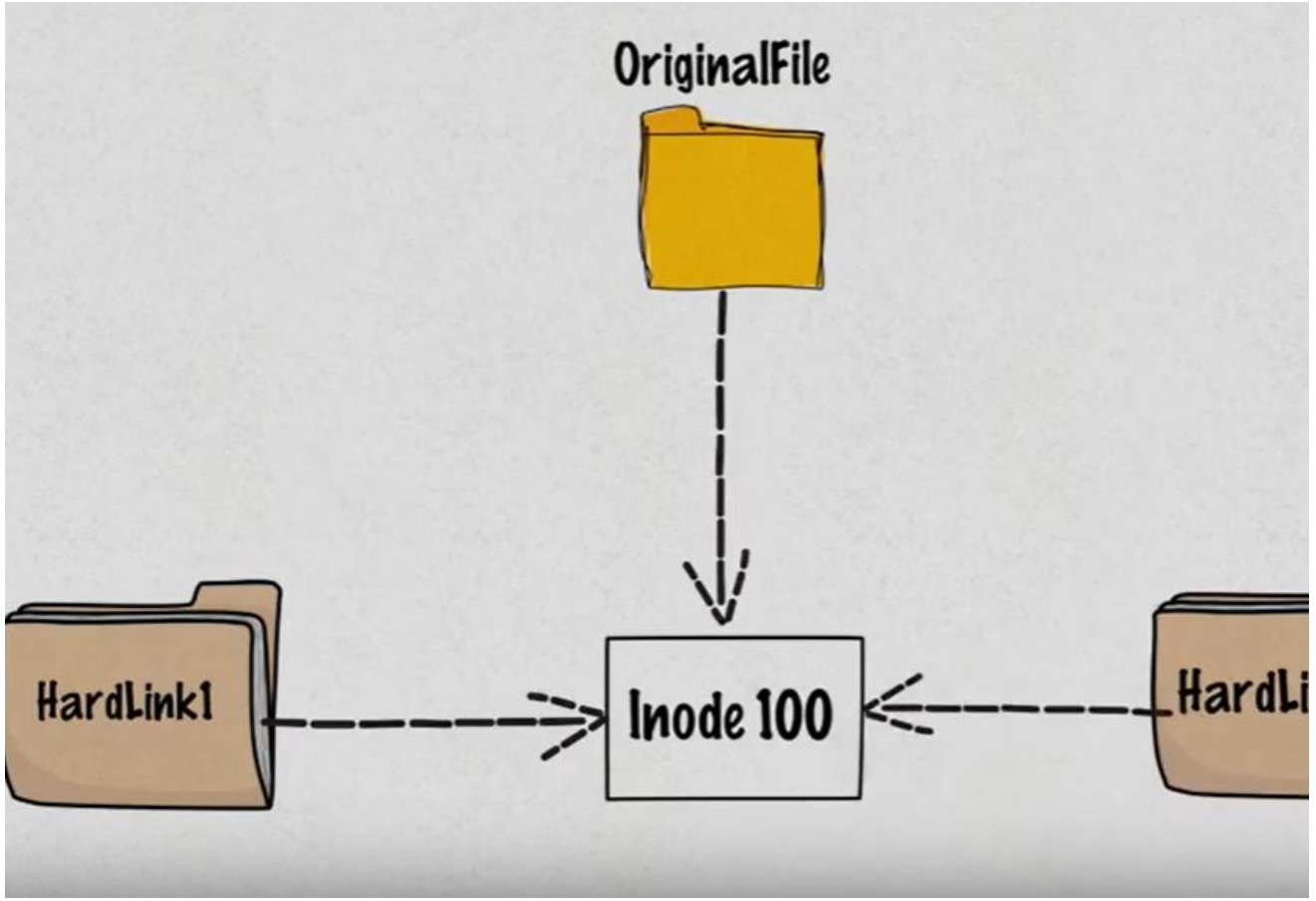


Hard Links

- The original Unix way of creating links.
- Every file has a single hard link that gives the file its name.
- When we create a hard link, we create an additional directory entry for a file.
- Have two important limitations
  - cannot reference a file outside its own file system.
  - may not reference a directory.
- Different name of the same file.
- Same file size.
- Same inode number.

Hard Links

- If we delete the original file the hard links will still contain the data that were in the original fil



Let’s build a Playground

- Creating Directories
  - **mkdir playground**
  - **cd playground**
  - **mkdir dir1 dir2**
- Copying Files
  - **cp /etc/passwd .**
  - **ls -l**
- Moving and Renaming Files
  - **mv passwd fun**
  - **mv fun dir1**
  - **mv dir1/fun dir2**
  - **mv dir2/fun .**
  - **mv fun dir1**
  - **mv dir1 dir2**
  - **mv dir2/dir1 .**
  - **mv dir1/fun .**

Let’s build a Playground

- Creating Hard Links
  - **ln fun fun-hard**
  - **ln fun dir1/fun-hard**
  - **ln fun dir2/fun-hard**
  - **ls –l**

```
[me@linuxbox playground]$ ls -l
total 16
drwxrwxr-x 2 me   me   4096 2018-01-14 16:17 dir1
drwxrwxr-x 2 me   me   4096 2018-01-14 16:17 dir2
-rw-r--r-- 4 me   me   1650 2018-01-10 16:33 fun
-rw-r--r-- 4 me   me   1650 2018-01-10 16:33 fun-hard
```

Let’s build a Playground

- we notice is that both the second fields in the listings for fun and fun-hard contain a **4** which is the number of hard links that now exist for the file.
- Remember that a file will always have **at least one link** because the file's name is created by a link.
- How do we know that fun and fun-hard are, in fact, the same file?

```
[me@linuxbox playground]$ ls -li
total 16
12353539 drwxrwxr-x 2 me   me   4096 2018-01-14 16:17 dir1
12353540 drwxrwxr-x 2 me   me   4096 2018-01-14 16:17 dir2
12353538 -rw-r--r-- 4 me   me   1650 2018-01-10 16:33 fun
12353538 -rw-r--r-- 4 me   me   1650 2018-01-10 16:33 fun-hard
```

Let’s build a Playground

- The first field is the inode number and, as we can see, both fun and fun-hard share the same inode number, which confirms they are the same file.
- Creating Symbolic Links
  - **ln -s fun fun-sym**
  - **ln -s ../fun dir1/fun-sym**
  - **ln -s ../fun dir2/fun-sym**
- Removing Files and Directories
  - **rm fun-hard**
  - **ls –l**
  - **rm -i fun**
  - **ls -l**

Working With Commands

What exactly are command, type,which,help,man,apropos,info,whatis,alias

What Exactly are commands

- An executable program
  - like all those files we saw in /usr/bin
- A command built into the shell itself
  - The cd command, for example, is a shell builtin.
- A shell function
  - Shell functions are miniature shell scripts incorporated into the *environment*
- An alias
  - Aliases are commands that we can define ourselves, built from other commands.

Type – Display a command's Type

- A shell builtin that displays the kind of command the shell will execute, given a particular command name.
- type command
  - type type
  - type ls
  - type cp

```
[me@linuxbox ~]$ type type
type is a shell builtin
[me@linuxbox ~]$ type ls
ls is aliased to `ls --color=tty'
[me@linuxbox ~]$ type cp
cp is /bin/cp
```

which – Display an Executable's Location

- To determine the exact location of a given executable.
- Only works for executable programs, not built-ins nor aliases that are substitutes for actual executable programs.
- which command
  - which ls
  - which cp

```
[me@linuxbox ~]$ which ls
/bin/ls
```

Getting a command’s Documentation

- --help – Display Usage Information
  - **mkdir --help**
- help – Get Help for Shell Builtins
  - **help cd**
- man – Display a Program's Manual Page
  - **man ls**
  - On most Linux systems, man uses less to display the manual page, so all of the familiar less commands work while displaying the page.
  - The “manual” that man displays is broken into sections and covers not only user commands but also system administration commands, programming interfaces, file formats and more.
  - See the book page 70 .

Getting a command’s Documentation

- whatis – Display One-line Manual Page Descriptions
  - **whatis ls**
  - displays the name and a one-line description of a man page matching a specified keyword:
- apropos – Display Appropriate Commands
  - **apropos partition**
  - earch the list of man pages for possible matches based on a search term.
  - The first field in each line of output is the name of the man page, and the second field shows the section.
- info – Display a Program's Info Entry

alias

- Used to create a command of our own.
- It's possible to put more than one command on a line by separating each command with a semicolon.
- *command1*; *command2*; *command3*...
- **cd /usr; ls; cd -**

```
[me@linuxbox ~]$ cd /usr; ls; cd -
bin  games  include  lib  local  sbin  share  src
/home/me
[me@linuxbox ~]$
```

alias

- alias name=“String”
  - **alias foo='cd /usr; ls; cd -’**
  - **Foo**
  - **type foo**

- To remove an alias
  - **unalias foo**
  - **type foo**

References

- The book.
- https://www.youtube.com/watch?v=4-vye3QFTFo