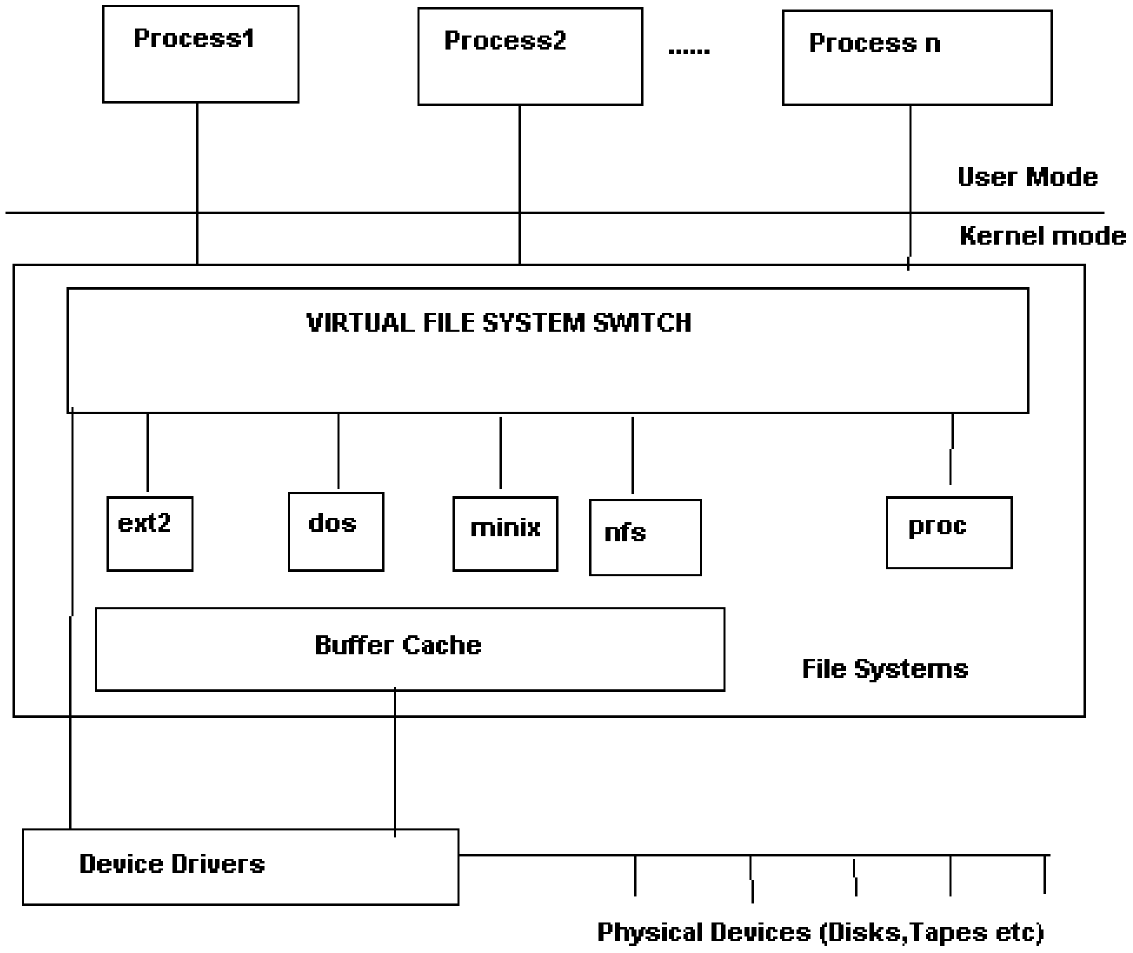
Linux file system

Contents

Logic file system



Structure of file system

One/Many hierrachical tree(s) with directories and files

- File: group of bits
- Directory: group of files and directories

Root directory (/) is the root for the whole hierrachical tree

Files are leaves

Popular Linux directories

/ (root directory)

- /bin : essential user binaries
- /boot : static boot files the OS needs in order to boot
- /etc : contains all configuration files of the system or in its sub-directories
- /dev : where all your devices live: keyboard, mouse, printer, disk, partition.
- /home : contains a home folder for each user
- /lib : contains libraries needed by the essential binaries in the /bin and /sbin
- /usr : contains applications and files used by users, as opposed to applications and files used by the system
- /var : is the writable counterpart to the /usr directory
- /proc: special files that represent system and process information

Linux files vs. Windows files

Similar

- Maximum length of file names is 255
- Accept most characters to name files except some special characters such as * ? [ ] & as they are used for special purposes

Linux files only

- A single hierrachical structure for the file system, unlike Windows
- No definition of extension part of file name (character '.' is treated the same as other characters).
- No logic disks the hierrachical file system
- '/' is used instead of '\'

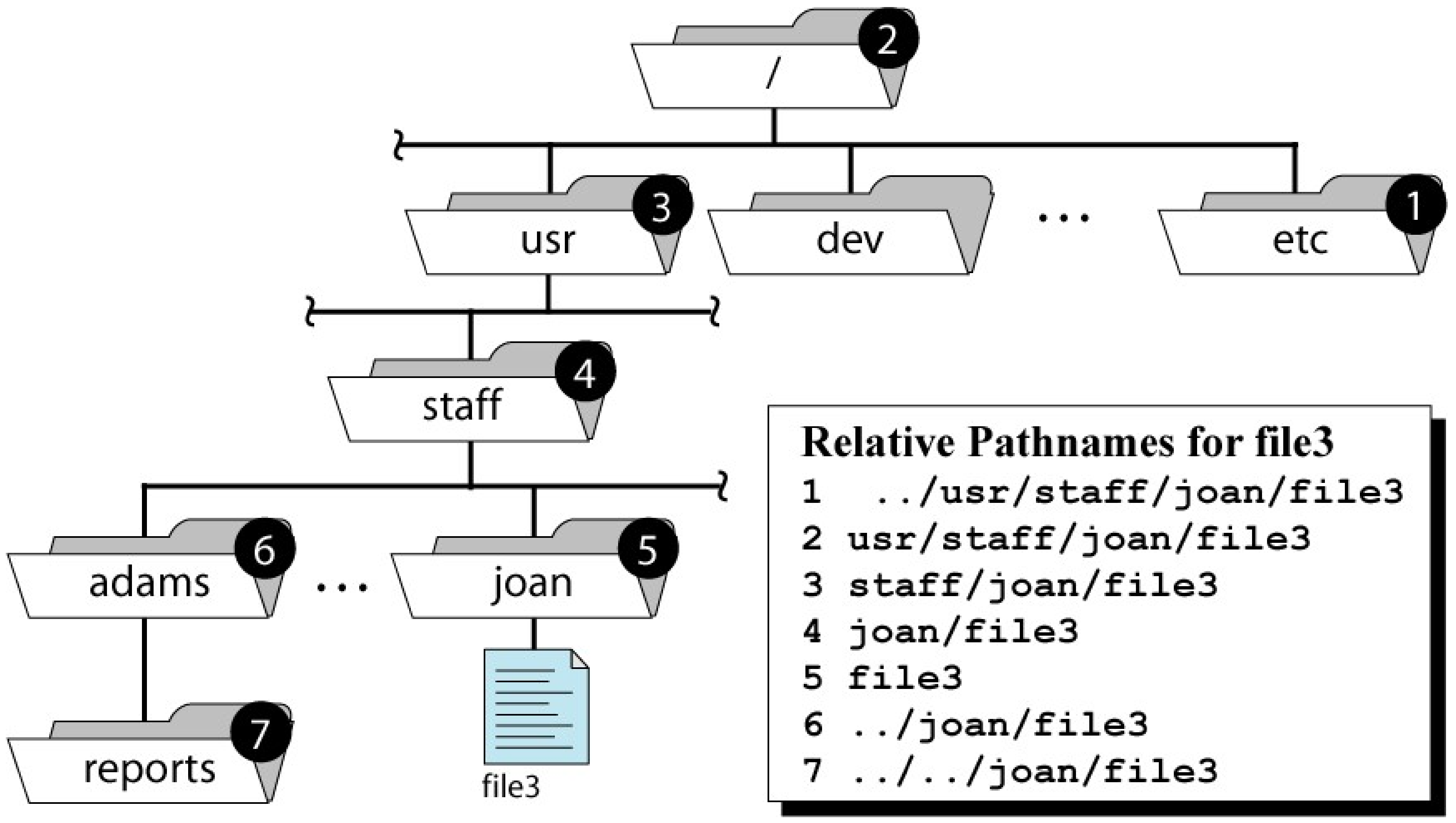
Special paths and directories

To access files and directories, we need to use paths (absolute or relative)

Path can be started from special directories

- / : root directory
- ~/ : home directory
- . : current directory
- .. : parent directory of the current one

Absolute vs relative paths



Relative Pathnames for file3

```
1  ../usr/staff/joan/file3
2 usr/staff/joan/file3
3 staff/joan/file3
4 joan/file3
5 file3
6 ../joan/file3
7 ../../joan/file3
```

Basic command to manage directories

pwd

cd

ls –la [new dir]

mkdir [-p] [new dir]

rmdir [empty dir]

Manage directory

pwd: absolute path of the current/working directory

cd: change the working directory

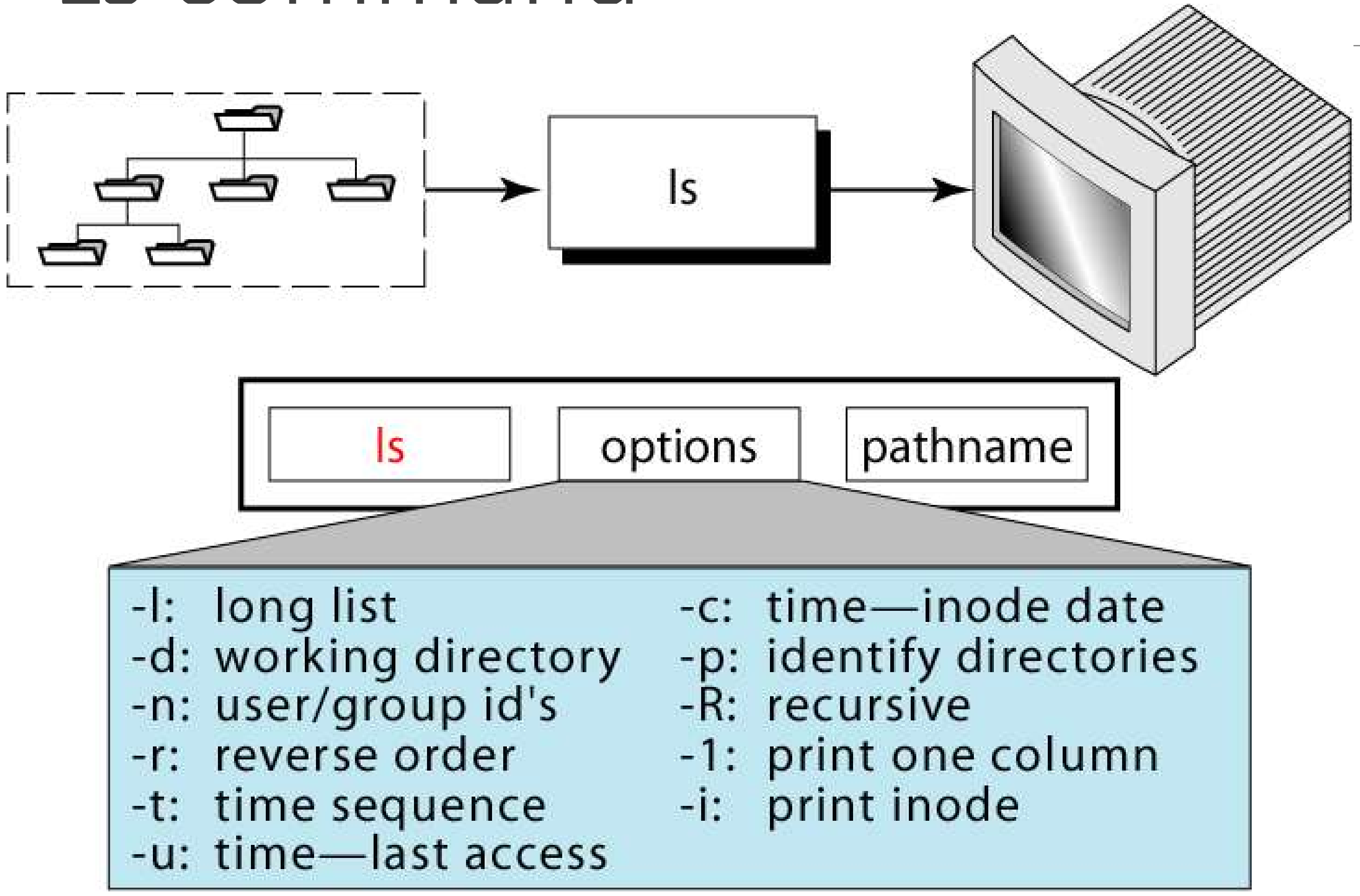
- $ cd /home/tuananh ↵
- $ cd tuananh ↵

ls: list files inside a directory

- $ ls ↵
- $ ls /home/tuananh
- $ ls –la tuananh
  - Option –a to list hidden files
  - Option -l to list all information, not just names mkdir: tạo một thư mục rỗng

rmdir: delete an empty directory

Ls command


ls
ls
options
pathname
-l: long list
-d: working directory
-n: user/group id's
-r: reverse order
-t: time sequence
-u: time—last access
-c: time—inode date
-p: identify directories
-R: recursive
-1: print one column
-i: print inode

File types

Following characters represent file types of Linux

- : normal file

d : directory

b : block device file (special)

c : character device file (special)

l : symbolic link

m : share memory

p : named pipe

Special names: hidden files start as « . » (Ex: /home/tuananh/.bashrc)

Example

```
$ cd ~
$ pwd
/home/tuananh
$ ls -la
-rw-r--r-- 1 tuananh  user1 2451 Feb  7 07:30 .bashrc
-rw-r--r-- 1 tuananh  user1 4025 Feb 10 19:12 linux.ppt
drwxr-xr-- 2 tuananh  user1  512 Feb 10 19:12 linux
$ mkdir vanban
$ cd vanban
$ pwd
/home/tuananh/vanban
$ cd ..
$ pwd
$ rmdir vanban
```

Wildcard characters

- **Asterisk (*)** matches one or more occurrences of any character, including no character
- **Question mark (?)** represents or matches a single occurrence of any character
- **Bracketed characters [ ]** matches any occurrence of character enclosed in the square brackets
- **Bracketed characters with exclamation mark [!]** matches any occurrence of character not in the square brackets

Examples

```
$ ls -l *.[c,h]
-rw-r--r-- 1 tuananh  user1 2451 Feb  7 07:30 myprog.c
-rw-r--r-- 1 tuananh  user1 2451 Feb  7 07:30 myprog.h
$ ls -l *prog
drwxr-xr-- 2 tuananh  user1  512 Feb 10 19:12 c_prog
drwxr-xr-- 2 tuananh  user1  512 Feb 10 19:12 java_prog
$ ls -l .*
-rw-r--r-- 1 tuananh user1 451 Feb 7 07:30 .bashrc
-rw-r--r-- 1 tuananh user1 225 Feb 7 07:30 .bash_profile
-rw-r--r-- 1 tuananh user1 351 Feb 7 07:30 .bash_logout
```

Manage files

$cp file1 […] dir

- Copy one or more files to a directory

$mv file1 […] dir

- Move one or more files to a directory
- And/or change names

$rm file1 […]

- Remove one or more files

option -R (recursive)

- Allow to copy/move/remove a whole directory including child directories and files

Manage files (cont.)

cat: quick look of a file

more: view each line of a file

tail: view the end of a file

head: view the beginning of a file

touch: create a new file, update an old one

echo content > [file]

Example

```
$ ls -l
-rw-r--r-- 1 tuananh  user1   16 Feb 10 19:12 test.txt
drwxr-xr-- 2 tuananh  user1  512 Feb 10 19:14 vanban
$ cp test.txt vanban
$ ls -l vanban
-rw-r--r-- 1 tuananh  user1   16 Feb 12 20:03 test.txt
$ rm –R vanban
$ ls -l
-rw-r--r-- 1 tuananh  user1   16 Feb 10 19:12 test.txt
$ rm test.txt
$ ls -l
```

inode

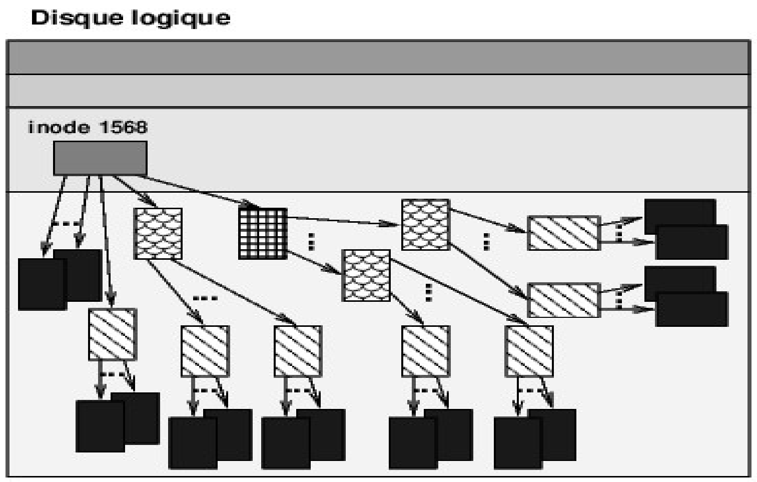
The inode (index node) is a data structure that describes a file-system object such as a file or a directory

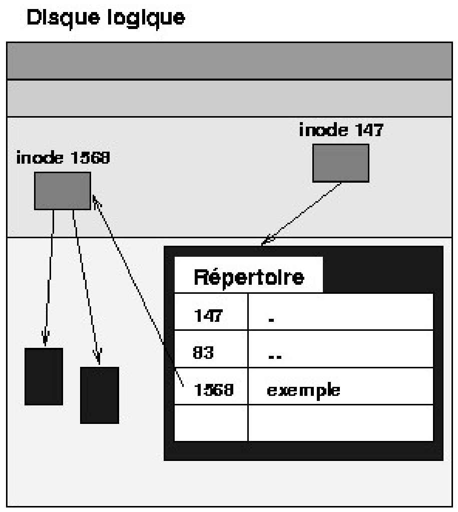
Content of a file is stored in data blocks

- Blank file = inode without data blocks

A directory is a link with the content of a reference table

- A link attaches a file name with an inode





Hard link (1)

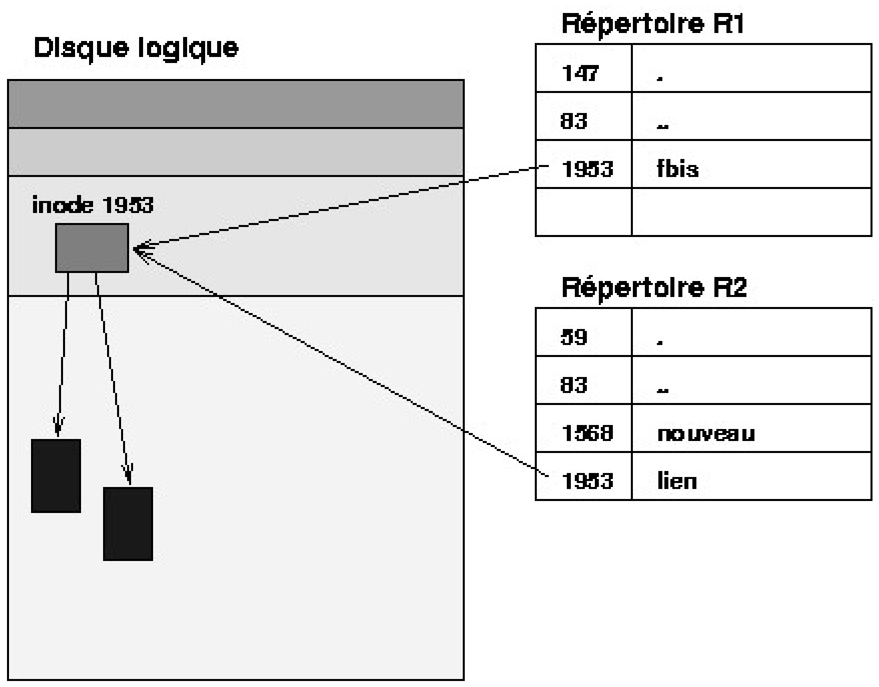
A hard link is a direct reference to a file via its inode

There might be multiple hard links to a same inode

Command ln allows to create a new hard link to an existing inode

- The new file share the same inode with the original fiel
- Syntax: ln <old_file> <new_file>

$ln fbis lien



Hard link (2)

The number of hard link(s) to an inode can be show by using ls –l

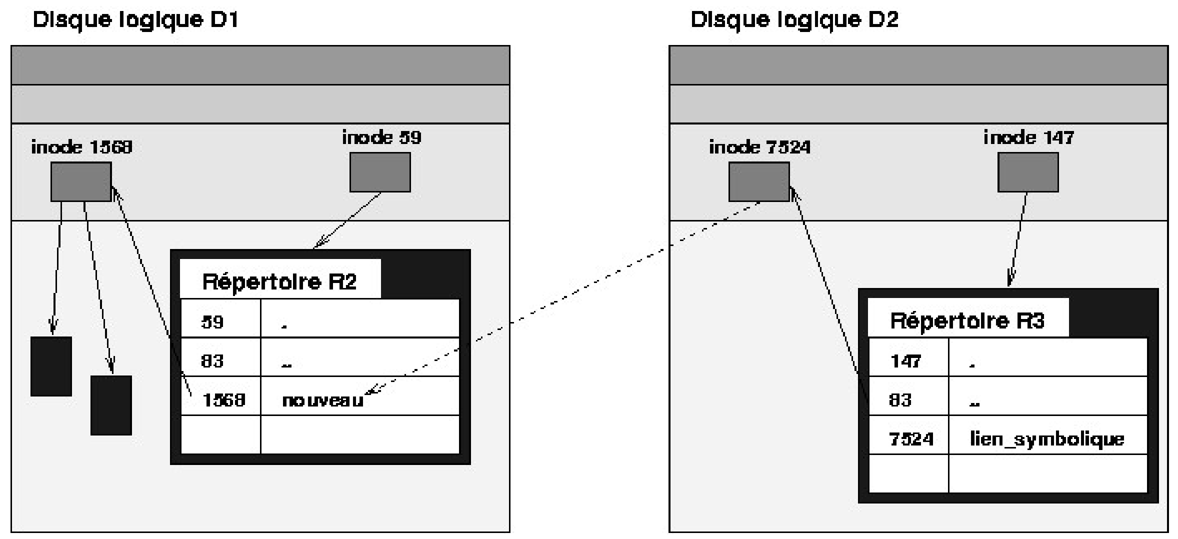
```
$ ls -l
-rw-rw-r-- 1 tuananh user1    0 Nov 12 15:19 file
drwxr-xr-x 2 tuananh user1 4096 Dec 14 17:50 dir
```

Question: Why does a directory alway have at least 2 hard links

Removing a file means deleting a hard link to the inode

- If the deleted file is the last link to the inode, the inode will be removed as well

Symbolic link



ln -s R2/nouveau R3/lien_symbolique

- While creating a symbolic link (option –s), a new inode is created
- Inode contains the path (relative or absolute) of reference object

Hard link vs symbolic link

Symbolic can be used to overcome the limitation of storage

- A hard link need more storage than a symbolic link

How could we distinguish a symbolic file and the original file of a symbolic link?

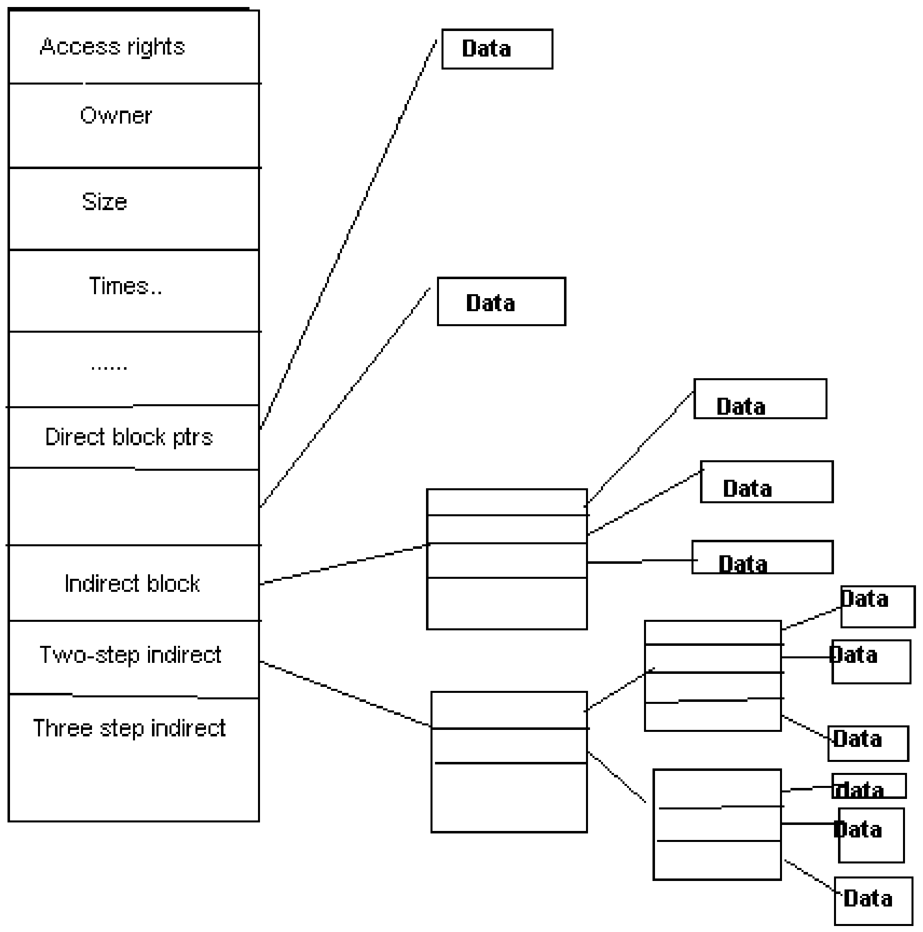
- What will happend if we delete the original file?

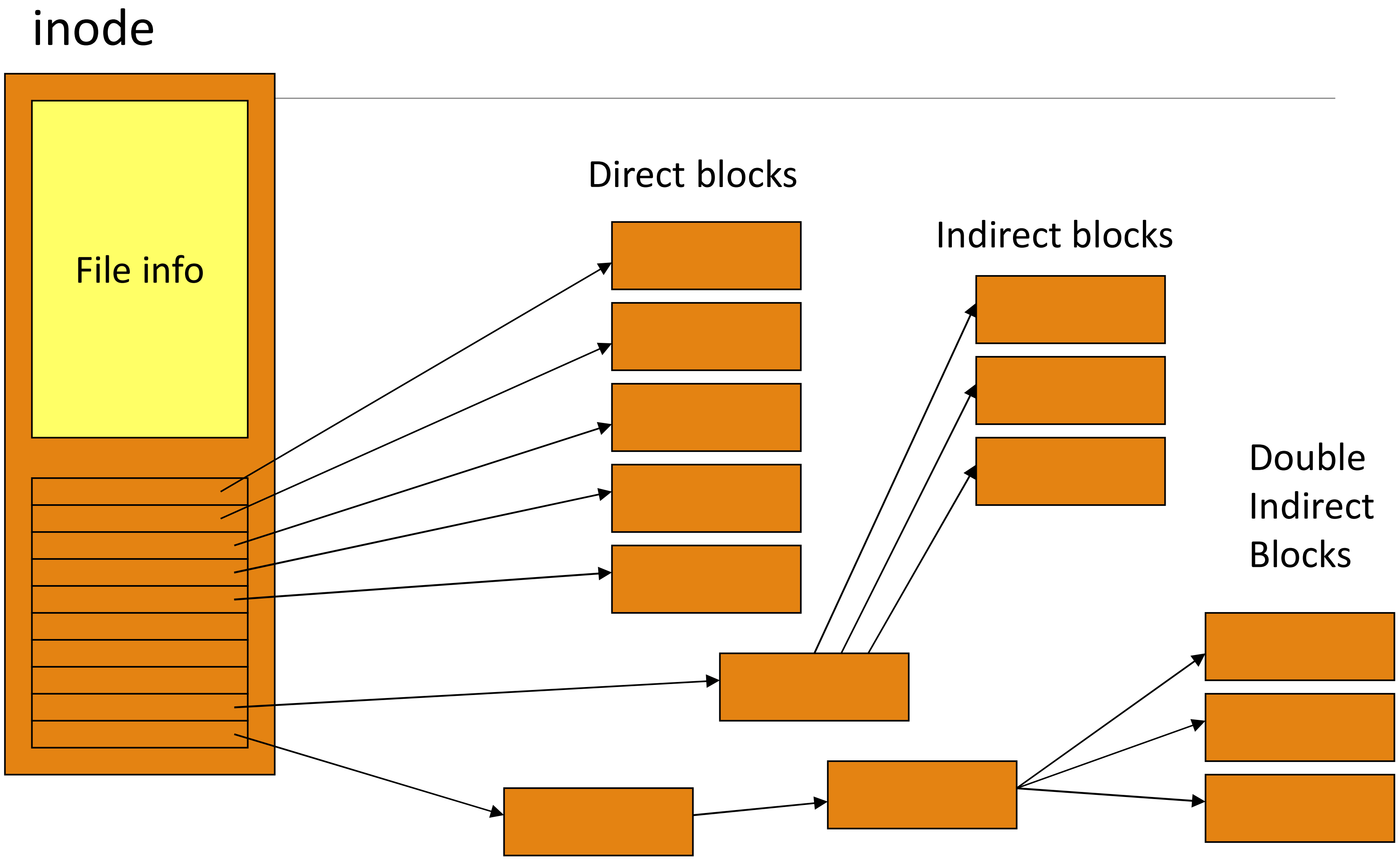
Symbolic link is similar to shortcut in Windows OS

Examples

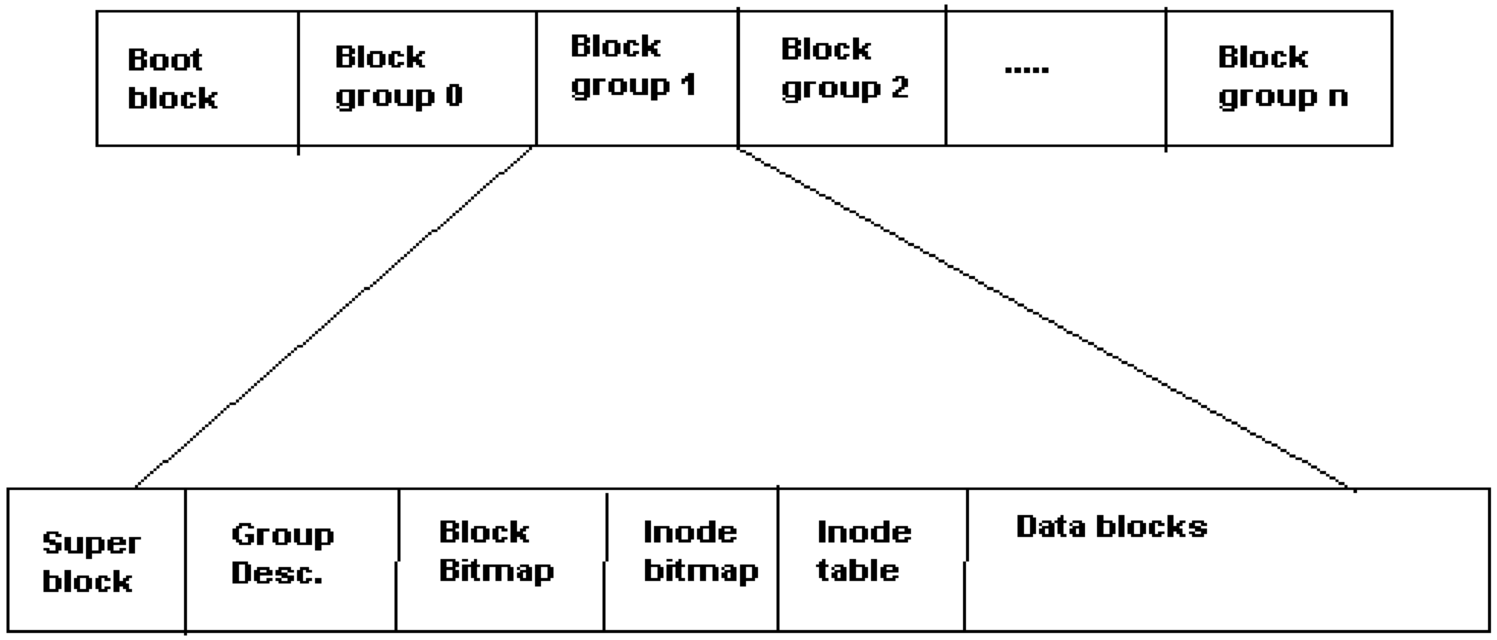
```
$ ls -l
-rw-r--r-- 1 tuananh user1 8 Feb 10 1:12 test.txt
$ ln test.txt link1
$ ln -s test.txt link2
$ ls -l link*
-rw-r--r-- 2 tuananh user1 16 Feb 10 1:12 link1
lrw-r--r-- 1 tuananh user1 16 Feb 10 1:13 link2-
 >test.txt
```

Inode structure




inode
File info
Direct blocks
Indirect blocks
Double
Indirect
Blocks

Organisation on hard drive(s)



EXT2FS STRUCTURE

Search files

$ find <name of directory> expressions

- Allow to search files inside a directory (default is the working directory) with some conditions or commands to be executed on found files.

Conditions

- Name : -name <name>
- Permission: -perm <permission>
- Type : -type d/f/...
- Size: -size N
- Thời gian : -atime N, -mtime N, -ctime N

Executable commands on found files

- -print
- -exec command

Examples

$find /usr -name toto
- Find files named toto inside the directory /usr (including child directories of /usr)

$find /usr -name " *.c »
- List all files ending as « .c »

$find / -mtime 3
- Find all files been modified last 3 days

$find / -size 2000
- Find all files with the size of 1 MB (= 2000 block 512 B)

$find / -size +20M
- Find all files with the size over 20 MB

$find / -size -2GB
- Find all files with the size under 2GB

$find / -type f -user olivier -perm 755
- Find all files belonging to the user oliver and having the permission of 755