INSTITUT
POLYTECHNIQUE
DE PARIS

BasicOS

File Systems

Ludovic Apvrille ludovic.apvrille@telecom-paris.fr
Eurecom, office 470

https://perso.telecom-paris.fr/apvrille/BasicOS/

TELECOM Paris

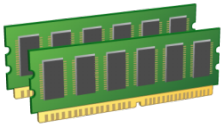
IP PARIS

Outline

Data Storage

DRAM, cache memory

- Data are lost when power is cut (e.g., after a shutdown)
  - Non persistent



HDDs, SSDs, CDs, Blu-rays, USB keys, SD Cards,...

- Data remain in the storage once written, even when power is cut
  - Persistent

TELECOM Paris
IP PARIS

Data Storage: Challenges

Storage structure

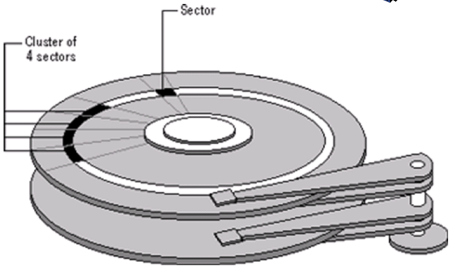
- Programmers don't want to care with the diversity of physical storage devices
  - Uniform access to files and directories, whatever the hardware support
  - *open()*, *read()*, *write()*, *close()*
  - And not: move disk head to cylinder 123, wait for sector 321, ...

Various file needs

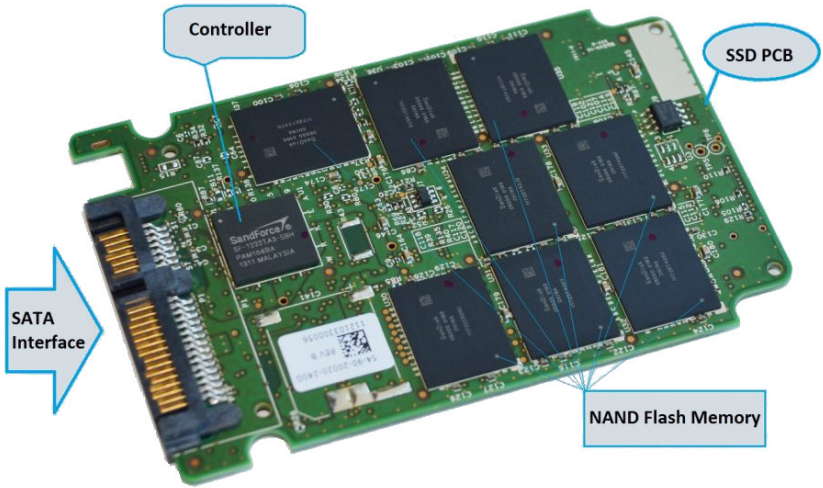
- Tiny files to huge files (from a few bytes to GB)
- More read operations than write operations
- Starting an application requires many read operations
- Quick file browsing (using terminal or graphical file explorers)

Architecture of HDDs

- Low-level formatting
  - Organization in sectors
- Logical formatting
  - File-system data structure is stored on the disk
    - FAT, FAT32, NTFS, ZFS, etc.



Architecture of SDDs



credit: mycloudwiki

Outline

TELECOM Paris

IP PARIS

File Systems

Definition

Logical organization of files and directories (in a physical device), and ways to perform operations on files and directories

Files

- Storage location with an identity (file name)
- Must belong to one directory
- Text files contain readable characters
- Binary files contain any binary values

Directories

- A.k.a. "folder"
- Collection of files and directories
- The "/" directory represents the root of a filesystem
- ".": current directory
- "..": parent directory

Inodes and Data Blocks

File system = a superblock + a collection of inodes (="index nodes") and of data blocks

```
$ man inode
Each  file  has an inode containing metadata about the file.
An application can retrieve this metadata using stat(2) (or related calls)

$ stat slidesFileSystems.tex
  File: slidesFileSystems.tex
  Size: 8811            Blocks: 24         IO Block: 4096   regular file
Device: fd01h/64769d    Inode: 22814078    Links: 1
Access: (0644/-rw-r--r--)  Uid: ( 8003/apvrille)   Gid: (  105/soc_staff)
Access: 2022-09-16 18:10:22.641433749 +0200
Modify: 2022-09-16 18:10:22.625433571 +0200
Change: 2022-09-16 18:10:22.625433571 +0200
 Birth: -
```

The file uses 24 blocks of (probably) 512 bytes, and read and write operations for this file are performed with 4096 bytes at a time.

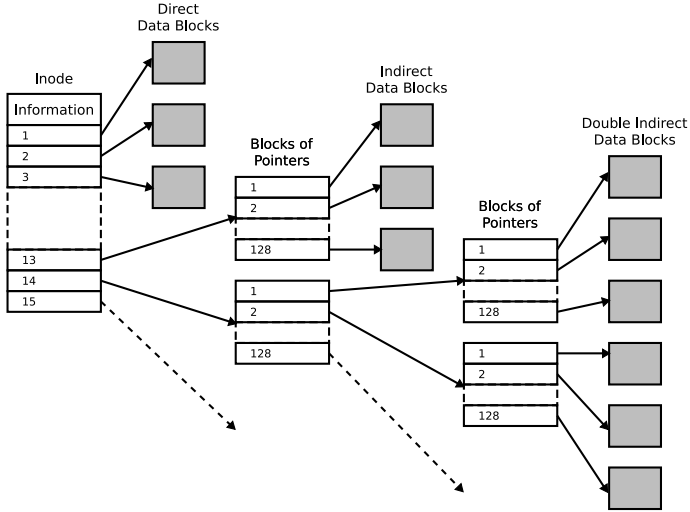
TELECOM Paris

IP PARIS

Inode Structure

Inode
Inode number, device number
Type (file, directory, link)
Owner and access rights
Timestamps for access (read), modify (write), change of metadata (permission)
Information on blocks (size, number of blocks, . . .)
Pointer to block #1 . . . Pointer to block #12
Pointer to indirection table Pointer to double indirection table Pointer to triple indirection table

Inode: Use of indirections



Ext2 inode indirections. Credit: wikipedia

Superblock

- First block of the file system

Superblock
File system type
Name of partition
Number of data blocks and inodes
Pointer to the inode table (inode of "/")
Pointer to the list of free inodes
Pointer to the list of free data blocks

Logical View of a File System

- At least one inode for "/" (inode number: 2)
- Optimized location of inodes and data blocks depends on the disk structure
  - HDD: data blocks of a file are likely to be placed one after each other
  - HDD: files in the same directory are likely to be placed on the same track

TELECOM Paris

IP PARIS

Outline

TELECOM Paris

IP PARIS

File Systems: Syscalls

Files

- *open()*, *close()*
- *read()*, *write()*
- *lseek()*
- *fsync()*
- *rename()*
- *stat()*
- *link()*
- *unlink()*

Directories

- *mkdir()*
- *rmdir()*

Also, in section 3:

- *opendir()*
- *readdir()*
- *closedir()*
- ...

Example of Code: Content of a Directory

```c
int main(int argc, char **argv) {
  if (argc != 2) {
    printf ("usage: %s <directory name>\n", argv[0]);
    exit(0);
  }

  printf("Listing of directory %s:\n", argv[1]);
  listDir(argv[1]);
}
```

Example of Code: Content of a Directory (Cont.)

```
void listDir(char *dirName) {
   DIR* dir; struct dirent *dirEntry; struct stat inode; char name[1000];

   dir = opendir(dirName);

   if (dir == 0) {perror ("Failed opening directory"); exit(1);}

   while (( dirEntry = readdir(dir) ) != 0) {
     sprintf(name,"%s/%s", dirName, dirEntry->d_name);
     lstat (name, &inode);
     // test the type of the inode
     if (S_ISDIR(inode.st_mode))
          printf("dir ");
     else if (S_ISREG(inode.st_mode))
          printf ("file ");
     else if (S_ISLNK(inode.st_mode))
          printf ("lnk ");
     printf(" %s\n", dirEntry->d_name);
    }
```

Example of Code: Content of a Directory (Cont.)

```
$ gcc -Wall -o myls contentOfDirectory.c

$ ./myls .
Listing of directory .:
file  myls
dir  ..
dir  .
file  createFileWithEmptySpace.c
file  test1
file  contentOfDirectory.c
file  createFileWithEmptySpace
```