System Programming: File Management

Raymond Namyst

Dept. of Computer Science

University of Bordeaux, France

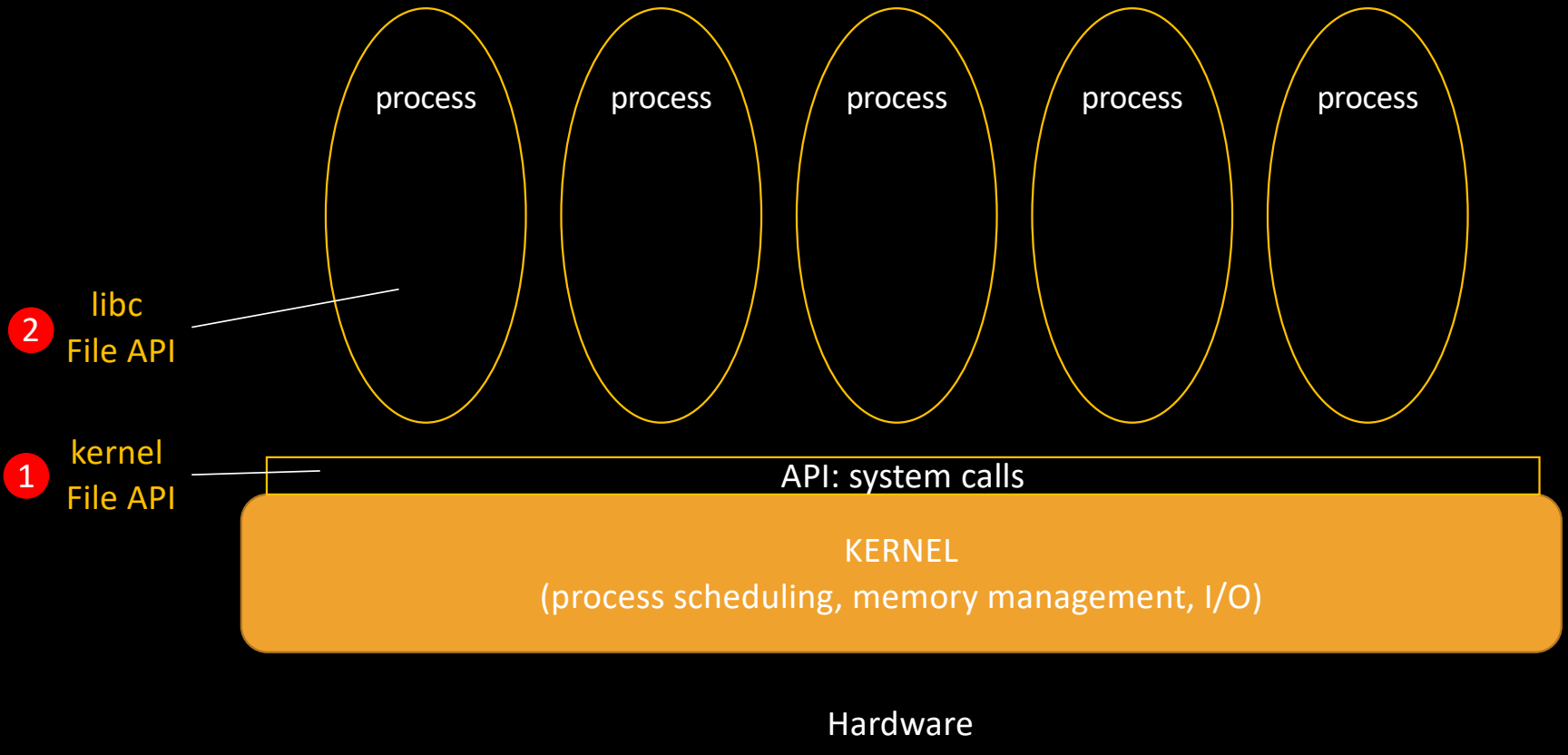
https://gforgeron.gitlab.io/progsys/

The concept of File

- Concept of major importance in Unix
  - "*All is File*" philosophy
    - Regular disk files, but also
      - Terminal
      - Devices (mouse, keyboard)
      - Network sockets
      - Etc.

- User view of a Disk File
  - Contiguous series of bytes
  - Known length, but may expand/shrink dynamically
  - Access rights (`rwx`)
  - Can be referenced by multiple links (paths)

Two File Management APIs



The concept of File

- Before we can read from/write into a file, we must *open* it
  - Why can’t we just read directly ?
    - read (“/net/cremi/dupont/myfile.txt”, buffer1, …)
    - read (“/net/cremi/dupont/myfile.txt”, buffer2, …)
    - …

The concept of File

- Before we can read from/write into a file, we must *open* it
  - Why can't we just read directly ?
    - read ("/net/cremi/dupont/myfile.txt", buffer1, …)
    - read ("/net/cremi/dupont/myfile.txt", buffer2, …)
    - …
  - Because of efficiency!
    - To access file "/net/cremi/dupont/myfile.txt", the OS must check
      - That there is a "net" entry in the "/" directory
      - That "/net" is a directory and that the user can traverse it (x)
      - That there is a "cremi" entry in the "/net" directory
      - That "/net/cremi" is a directory and that the user can traverse it (x)
      - …

Opening Files

- Before we can read from/write into a file, we must *open* it

```
int open(const char *path, int oflag, ...);
```

  - Open performs the appropriate checks, and returns a *file descriptor*
    - This file descriptor is a *key* which will accelerate upcoming read/write operations

Opening Files

- Before we can read from/write into a file, we must *open* it

```
int open(const char *path, int oflag, ...);
```

  - Open performs the appropriate checks, and returns a *file descriptor*
    - This file descriptor is a *key* which will accelerate upcoming read/write operations
  - `oflag`:
    - `O_RDONLY, O_WRONLY or O_RDWR`
    - Optional: `O_CREAT, O_TRUNC, O_SYNC`, etc.
  - When a file is created, the third parameter sets access rights (octal notation)
    - 0750 = 111 101 000 = `rwxr-x---`
    - 0666 = 110 110 110 = `rw-rw-rw-`