Operating Systems : FileSystems

Gabriel Laskar <gabriel@lse.epita.fr>

LSE Security System

- need for space bigger that virtual memory
- data persistence
- data sharing

- large storage
- static
- decoupled from processes

Vocabulary

- Block
- Partition
- Filesystem
- Directory
- File

How a disk works?

- accessed by blocks
- command queue (read/write/flush)
- interrupt when ready or finished

LSE Security System

Filesystems

- Structure
  - Files: logical unit for data storage
  - Directories: logical organisation for information
  - Partitions: high level organisation
- unified view of informations
- abstraction from physical layer
- format, types and semantics can be defined

File

- Name: unique identifier
- Format: hint about the internal structure of the file
- Type
- attributes: depends of the FS
  - dates
  - owner
  - ACLs
  - archive
  - hidden

Most of the time, metadatas are stored in directories

LSE Security System

Types of files

- MS-DOS: only some files can be executed (com, exe, bat)
- Mac OS: information about application creator are stored, in order to relaunch it on opening
- Unix: no types (except for chardev, blockdev), every files are the same

Access Schemes

- Sequential access
  - positioning offset
  - lseek(2)
- Random access
  - preadv(2), pwritev(2)

Operations on files

- open(2)/creat
- read(2)/write(2)
- lseek(2)
- fstat(2)
- other
  - append modes
  - truncate
  - renaming
  - reading/writing attributes

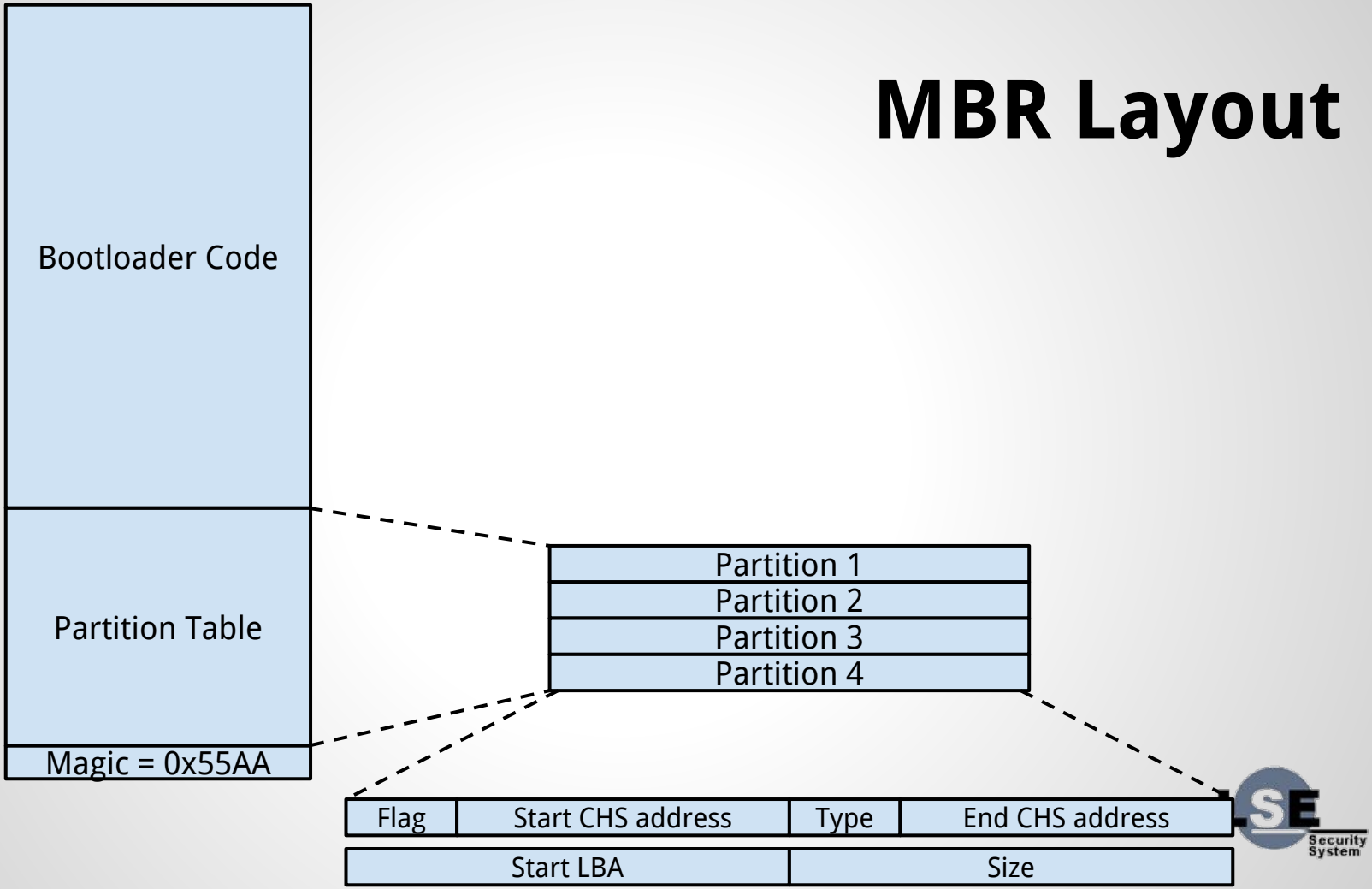
LSE Security System

Multiple level of organisation

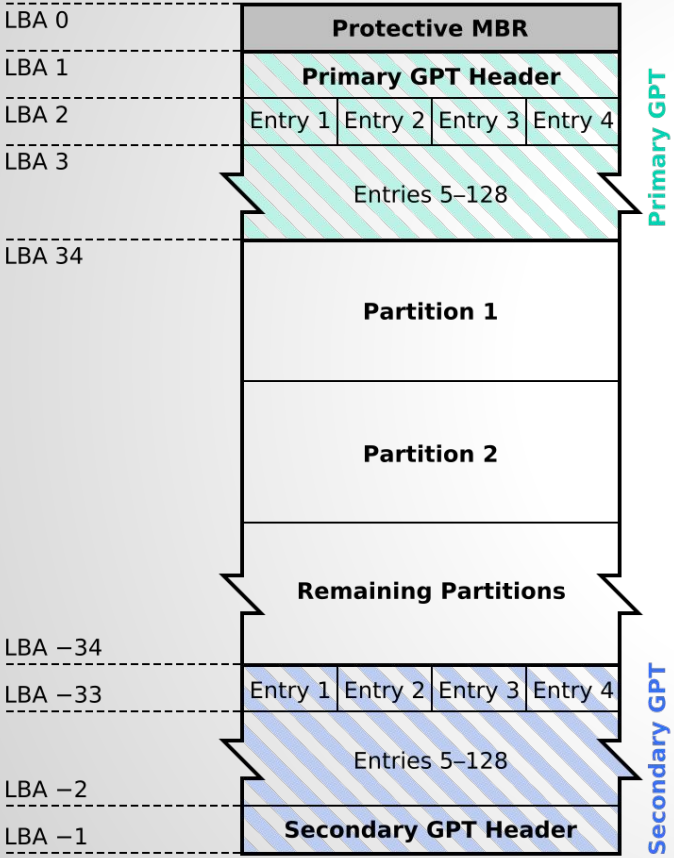
- partition tables
  - mbr
  - gpt
- partitions
  - primary
  - extended
- volumes
  - lvm
  - bsd volumes
- filesystems
  - ext2/3/4, FAT, ntfs, ffs, ufs, ...

LSE Security System

MBR Layout

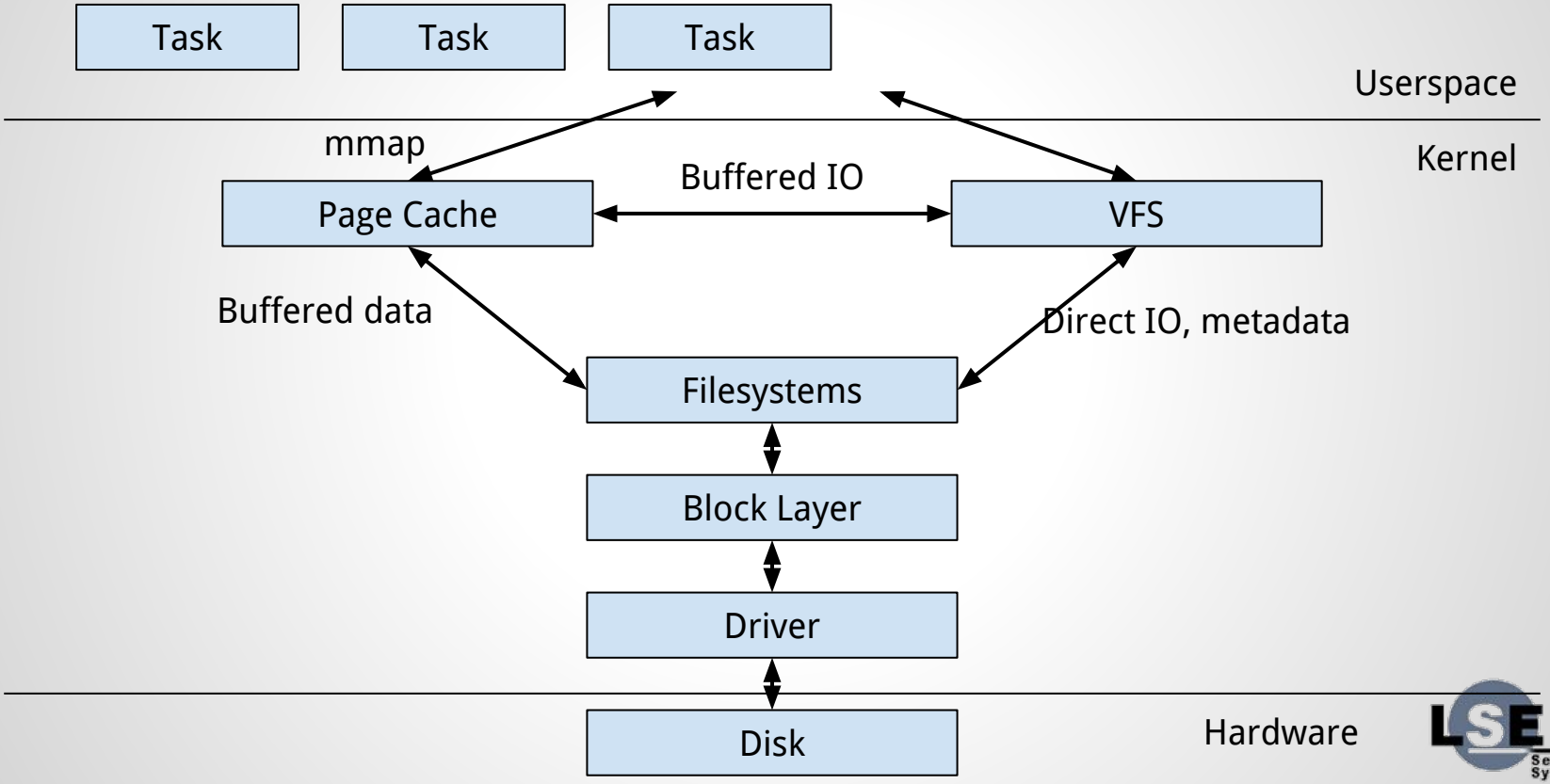

Bootloader Code
Partition Table
Magic = 0x55AA
Partition 1
Partition 2
Partition 3
Partition 4
Flag
Start CHS address
Type
End CHS address
Start LBA
Size

GUID Partition Table Scheme



type_guid		partition_guid	
first_lba	last_lba	attributes	...
partition name			

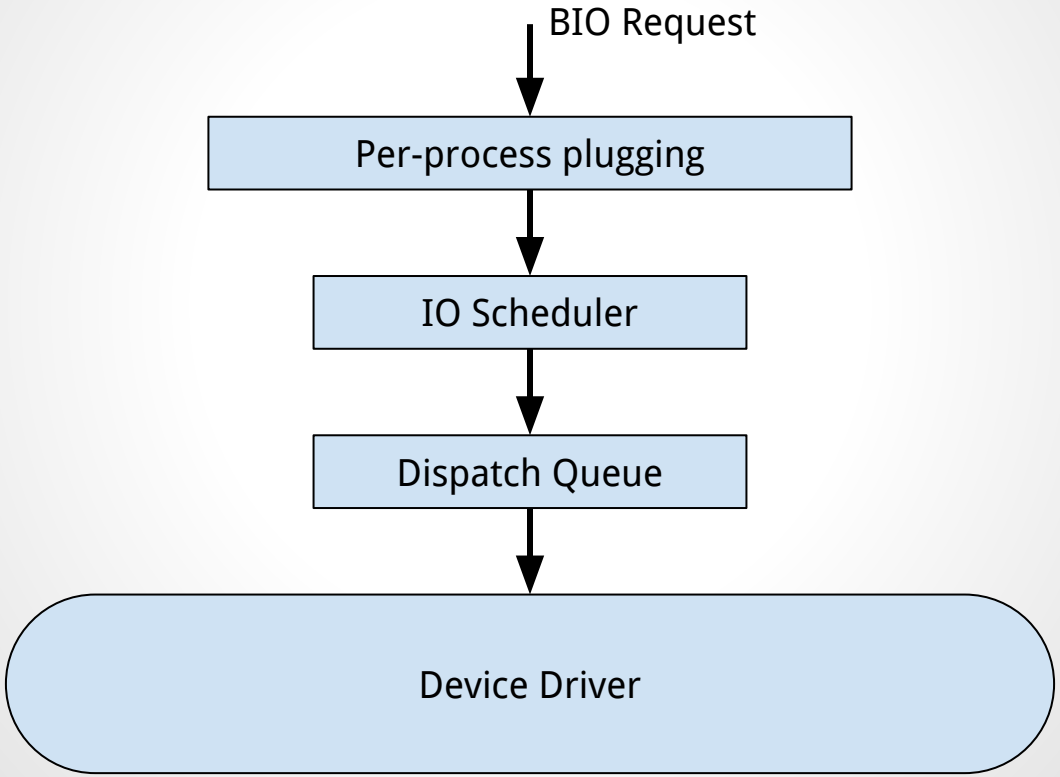
Linux Kernel IO Architecture



Block Layer Basics

- Works with IO requests
  - starting sector, length, read / write / special
  - Can have hints (SYNC) and other flags (FUA, FLUSH)
- Life of a request
  - Created in block layer when IO submitted by a filesystem
  - Can be delayed, merged (IO scheduler, multi queue handling)
  - Dispatched into a device driver
  - Completed when IO is finished

Submission Handling in Block Layer



IO Schedulers

Decide when and in which order IO requests are submitted

- NOOP - just pass requests into dispatch queue
- Deadline
  - Prefer reads over writes
  - Sorts waiting requests to reduce seeking
  - Aims to dispatch each request at least after its deadline has expired
- CFQ
  - Prefers sync requests over async
  - Tries to achieve fairness among tasks
  - Support for IO priorities, cgroups, sync requests idling, ...

LSE Security System

Virtual File System

- Abstraction for file access
- Concrete filesystems are specialisation of this fs
- Concrete filesystems are aggregated into a unique tree

LSE Security System

mount(2)

```c
#include <sys/mount.h>

int mount(const char *source, const char *target,
      const char *filesystemtype, unsigned long mountflags,
      const void *data);
```

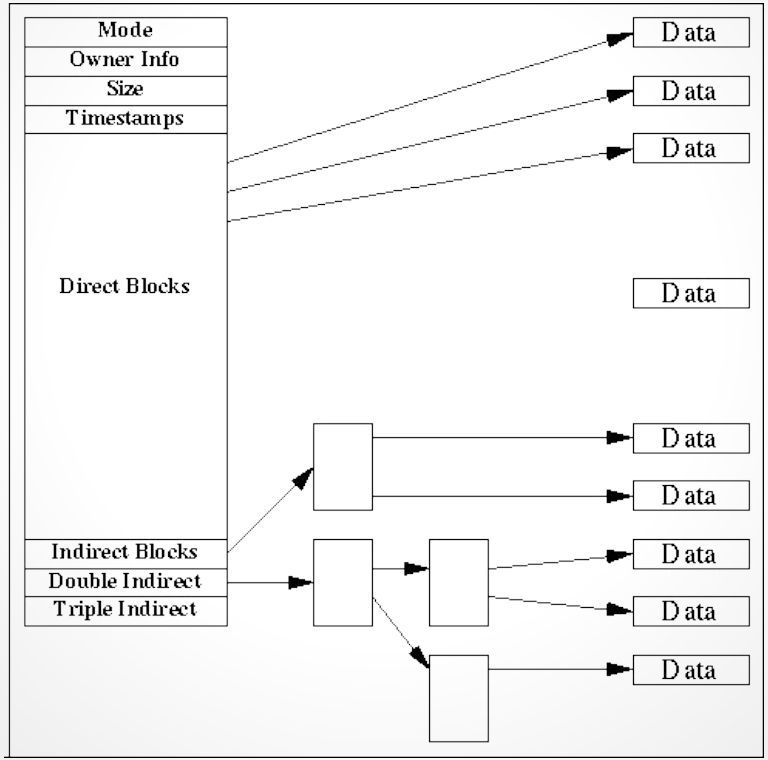
```c
mount("/dev/sdb12", "/mnt/example", "ext4", 0, NULL);
```

LSE Security System

Superblock

- base of the file system structure
- located inside the partition
- contains the informations about the configuration of the filesystem

ext2 inodes


Mode
Owner Info
Size
Timestamps
Direct Blocks
Indirect Blocks
Double Indirect
Triple Indirect
Data
Data
Data
Data
Data
Data
Data
Data
Data

Let’s discover

- inodes
- dentry
- fdtable
- ext2 / ramfs
- basic syscalls
  - read
  - write
  - lseek
  - open
  - close

LSE Security System

Modern Filesystems

- how to:
  - create a snapshot
  - maintain versions
  - rollback...
- btrfs and zfs

LSE Security System

Devices and IO

- what is a char device?
- what is a block device?
- mknod
- implementation (rtc-omap, evdev)
- subsystems in linux