

Taller Ext2

Departamento de Computación, FCEyN,
Universidad de Buenos Aires, Buenos Aires, Argentina

Sistemas Operativos, primer cuatrimestre de 2019

(2) El disco

- ¿Qué es? Un montón de bits agrupados en bloques.
- A cada bloque lo accedo con su LBA (Logical Block Addressing).
- En esta primera parte, nos vamos a preocupar sólo por leerlo
`void Ext2FS::read_block(unsigned int block_address, unsigned char * buffer);`
- ¿Qué tamaño tiene cada bloque? (buscar en `_superblock`)

(3) Superblock



```
struct Ext2FSSuperblock {
__le32 s_inodes_count;          /* Inodes count */
__le32 s_blocks_count;         /* Blocks count */
__le32 s_r_blocks_count;       /* Reserved blocks count */
__le32 s_free_blocks_count;     /* Free blocks count */
__le32 s_free_inodes_count;     /* Free inodes count */
__le32 s_first_data_block;      /* First Data Block */
__le32 s_log_block_size;        /* Block size */
...
__le32 s_blocks_per_group;      /* # Blocks per group */
...
__le32 s_inodes_per_group;      /* # Inodes per group */
...
__le16 s_magic;                 /* Magic signature */
__le32 s_first_ino;             /* First non-reserved inode */
__le16 s_inode_size;            /* size of inode structure */
}
```

(4) Inodo

- La representación de un archivo
- Un archivo puede ser desde un archivo regular, hasta un directorio, un pipe, un socket, un device, etc.
- Hoy, para nosotros, una struct de FSInode

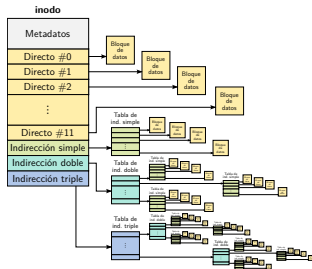
(5) FSInode

```
struct Ext2FSInode {  
    unsigned short mode;  
    unsigned short uid;  
    unsigned int size;  
    unsigned int atime;  
    unsigned int ctime;  
    unsigned int mtime;  
    unsigned int dtime;  
    unsigned short gid;  
    unsigned short links_count;  
    unsigned int blocks;  
    unsigned int flags;  
    unsigned int os_dependant_1;  
    unsigned int block[15];  
    unsigned int generation;  
    unsigned int file_acl;  
    unsigned int directory_acl;  
    unsigned int faddr;  
    unsigned int os_dependant_2[3];  
}
```

- ¿Dónde están los datos? 
- ¿Dónde está el nombre del archivo?  Porque la gente no anda preguntando por números de inodos.

(6) Inodo - Datos

- 15 Punteros a bloques con distintos sabores:
 - 12 Punteros a bloques de datos directos
 - 1 Puntero indirecto a bloque de datos
 - 1 Puntero con una doble indirección a bloque de datos
 - 1 Puntero con una triple indirección a bloque de datos.



- ¿Por qué hicieron este quilombo? ⚠
- ¿Son punteros a direcciones de memoria? ⚠
- ¿Los bloques a los que apuntan, están en memoria o en disco? ⚠

(7) Enunciado - Parte 1

- Completar la implementación del siguiente método de la clase Ext2FS (archivo ext2fs.cpp):

```
unsigned int Ext2FS::get_block_address(struct Ext2FSInode * inode, unsigned int n)
```

- Para poder recorrer el contenido de un archivo, necesito recorrer sus datos bloque a bloque empezando por el primer bloque del archivo (es decir, el bloque número cero). La función `get_block_address` deberá devolverme el bloque *n* de **datos** del archivo correspondiente al inodo *inode*.

(8) ¿Qué tengo solucionado?

- Clase Ext2FS parcialmente
- Estructuras de Ext2FS
 - Ext2FSSuperblock (Superblock)
 - Ext2FSInode (Inode)
- Funciones auxiliares de Ext2FS
 - read_block: Lee un bloque de disco
 - superblock: Devuelve el superbloque

(9) Documentacion

- <http://www.nongnu.org/ext2-doc/ext2.html>
- <http://e2fsprogs.sourceforge.net/ext2intro.html>
- <http://wiki.osdev.org/Ext2>
- <http://oreilly.com/catalog/linuxkernel2/chapter/ch17.pdf>