Operating systems

ELTE IK.

Dr. Illés Zoltán

zoltan.illes@elte.hu

Financed from the financial support ELTE won from the Higher Education Restructuring Fund of the Hungarian Government

We have talked about…

- Operating system generations
  - Typical hw constructions
- Op. Systems definitions
- Notions:
  - Files, directories, processes
- System calls
- System structures
  - Today: Client-server model, with layered features
- Data storage
  - Types, efficiency, hw, sw

What comes today…

- Files
  - File types
- Directories
  - Directory architecture
- File systems
- File system access scheduling
- …

Filesystem

- File: logical grouping a data set with a name (filename), with other parameters e.g. Size, date, etc.
- Directory: logical grouping of files and directories
- Filesystem: a method, how to organize this storage, data system on a physical disk.

Files

- A file: a base unit of information storage.
- Most common property: name.
- Stored on a storage device, HDD, SSD, SD card, etc.
  - The standard output, input are also a special file.
- Generally there are 3 types of files:
  - User files.
  - Temporary files
  - Administration, configuration files. Part of op.system.

File properties

- Filename: a string
  - There are restrictions for containing characters
    - Length, special characters are not allowed(*, \,&,etc.)
    - Restrictions are up to operating system.
- Other attributes (additional information)
  - Size of file, owner, last modification date, hidden or read only file, general access rigths, etc…
- Physical location on the drive(e.g. LBA address)
  - Real file, link (hard), link (soft)

Directories

- A special entry, shows the place in the file system where the containing files and directory entry are recorded.
- Directory architectures
  - Systems without any catalogs, e.g. Early tape systems
  - One level, two lewel catalogs
    - Not used today
  - Multi level, hierarchical catalog structure
    - Tree structure
    - Effective search
    - Today typically is used.
- Absolute, relative names
  - PATH environment variable

Access rigths

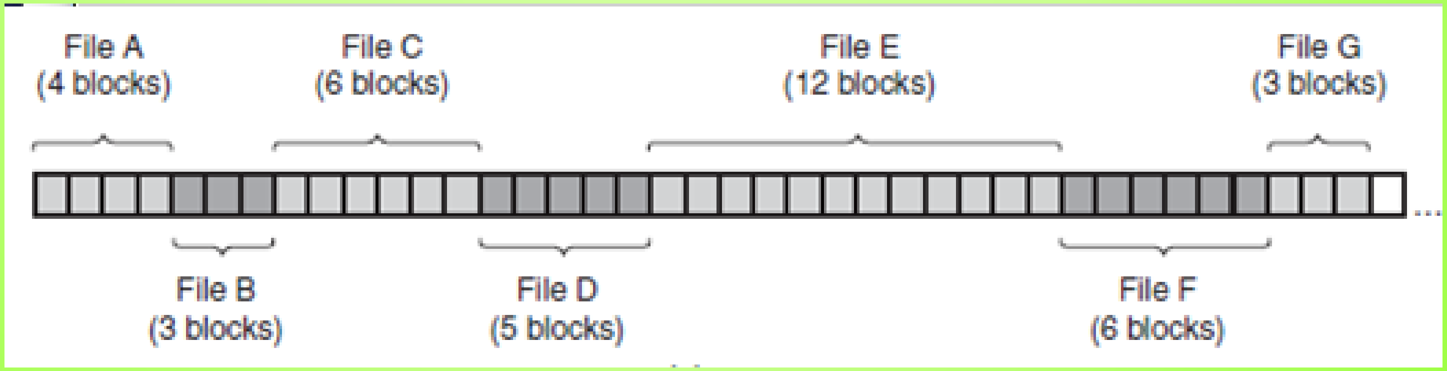
- There are no generally used security system
- Typical access rigths:
  - Read
  - Write, Create, delete
  - Execution
  - Modify
  - Full control
- Recording of access rigths
  - As Attributes
  - ACL (Access Control List), getfacl,setfacl,chacl

Partition description

- At the beginning of partition: a Superblock (e.g. FAT, the 0. block) describes the properties of file system.
- Next one is usually a used/free block registration (e.g. FAT, File Allocation Table)
- Next part is the directory structure (inode), file, directory entries, and file data
- Where can we store a new file?
- What kind of strategy can we choose?

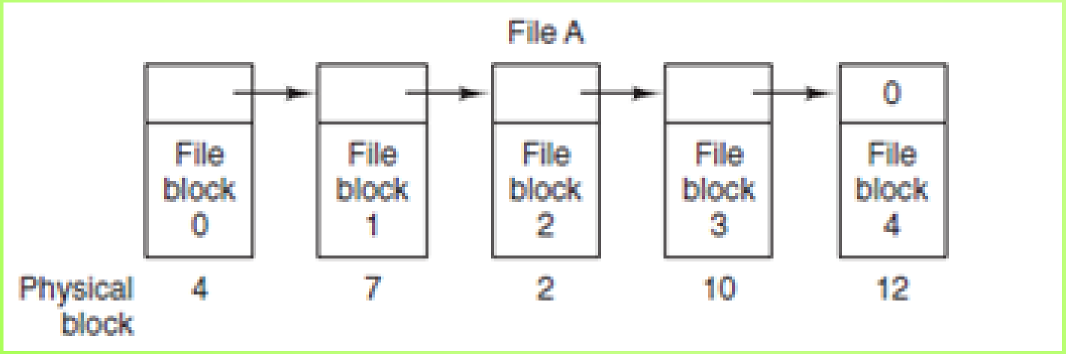
Storage strategy of files I.

- Contiguous storage place reservation
  - First Fit
  - Best Fit
  - Worst Fit (Put the file in such a free disk gap, so the remaining place (free place) should be the biggest)
  - In all of above are lost (waste) place.



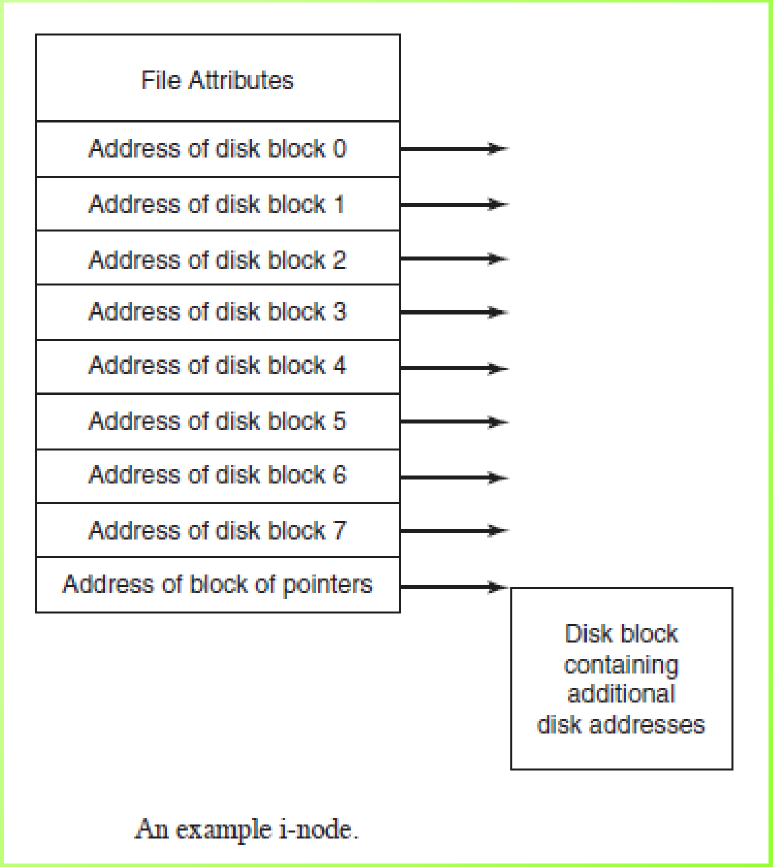
Storage strategy of files II.

- **Chained storage**
  - **No waste of place(only from the block size)**
  - **The file data is stored as a chained list of blocks**
    - The access of last blocks of files will be slow.
  - **Free- Reserved blocks : File Allocation Table,FAT**
    - It might be big, and the FAT is always in the memory!



Storage strategy of files III.

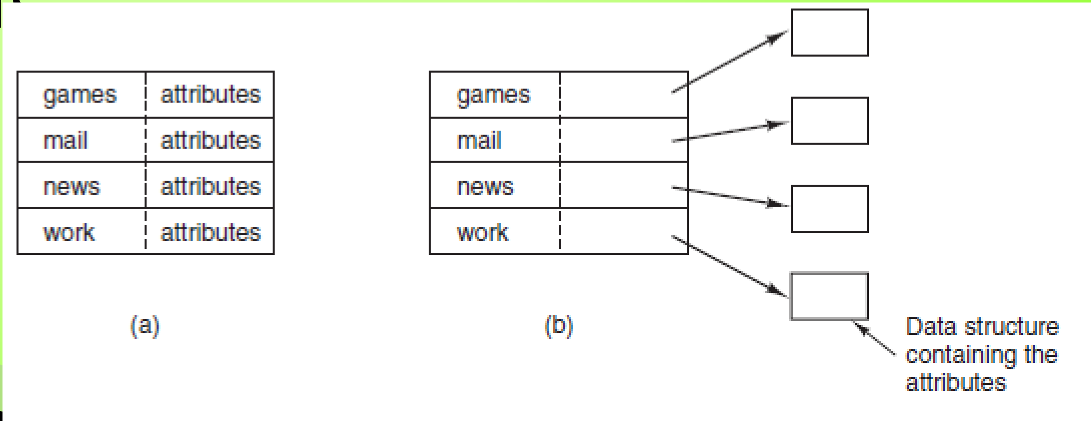
- Indextable storage
  - The directory catalog contains the address of file node
  - The inode address gives the file data.



An example i-node.

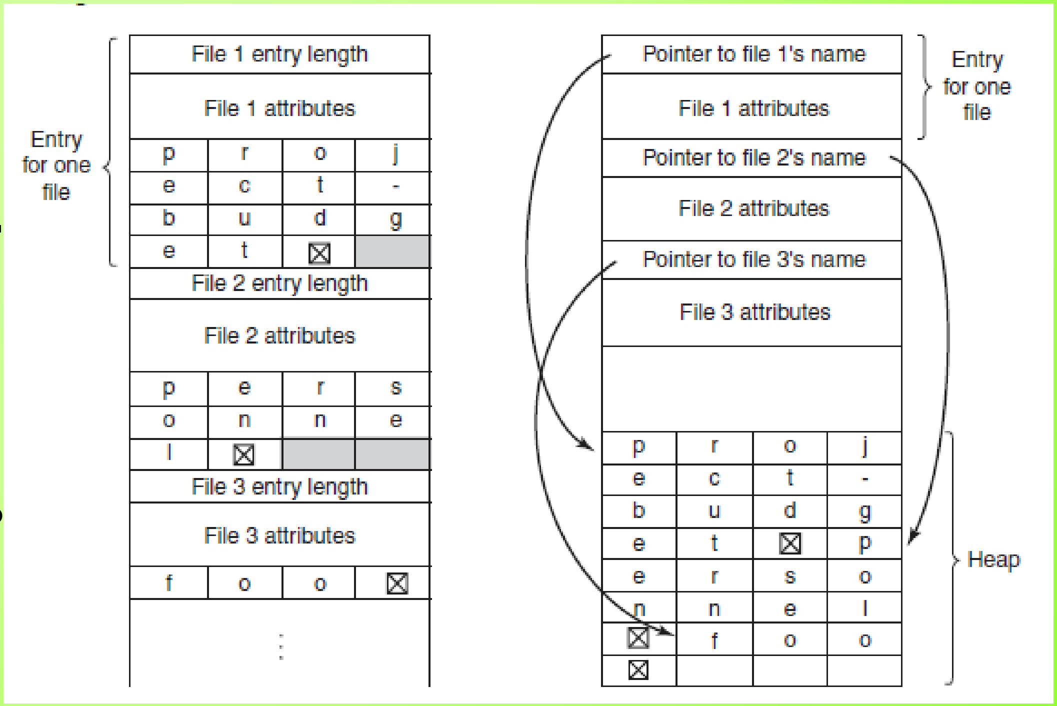
Implementation of a directory

- Main function to bind names and locations searching for filenames (linear, hash table, using cash)
- A directory entry may contain:
  - The disk address of the entire file (contiguous file allocation)
  - The number of the first block (linked list scheme)
  - The number of the i-node.
- Where does it store the attributes? There are two typical methods:
  - Fixed length
  - In i-nodes



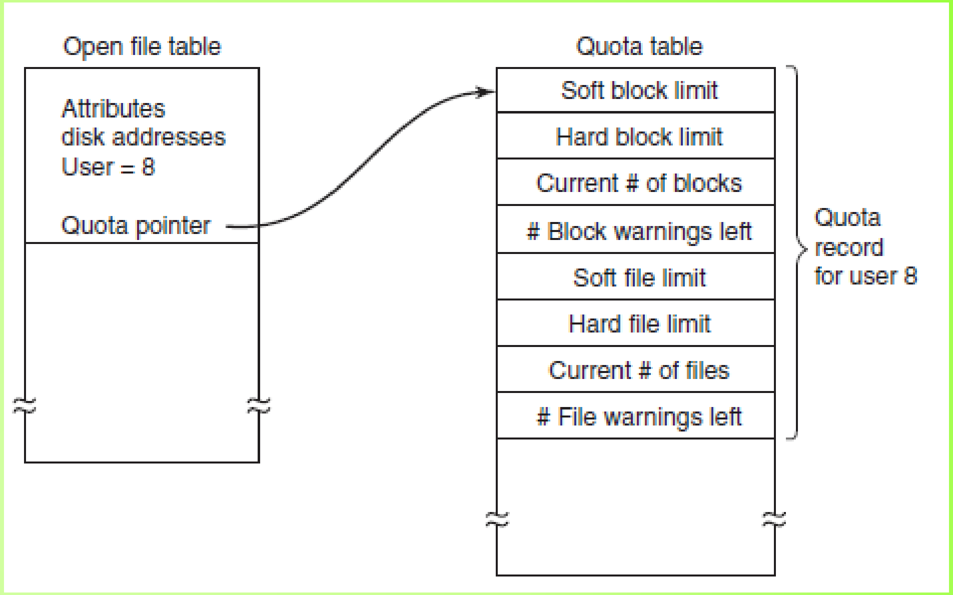
Filename implementation

- Recently – fix length (8+3)
- Nowadays – usually max 255 characters
  - How can we avoid place wasting?
    - With different length, at the first place storing the length of the entry.
    - With same length entries using pointers to various length of filenames



Disk quotas

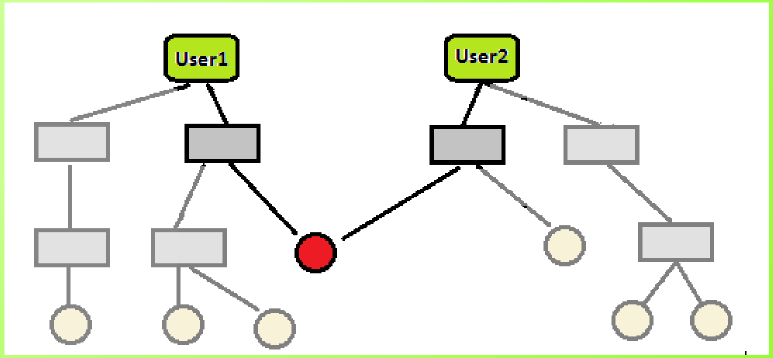
- If user opens a file
  - Opens file table (in memory)
  - Quota table (in memory)



- Adding new blocks -> change quota table
- If there is an attempt to log in
  - Over soft limit, counted warnings (reached max)- disabled
  - Over hard limit – log in is not permitted

Shared files

- How can the system manage to be able to see all of the changes the other user made?
- There are mainly two solutions
  - The file blocks are not listed in the directory entry, they are in a structure and directory entry point to it. (e.g. UNIX where the data structure is the i-node) So user1 and user2 points the same structure. (What happens if the owner want to delete it?)
  - With a new „link” file, a link to the original filename. (symbolic link) (Slow access.)

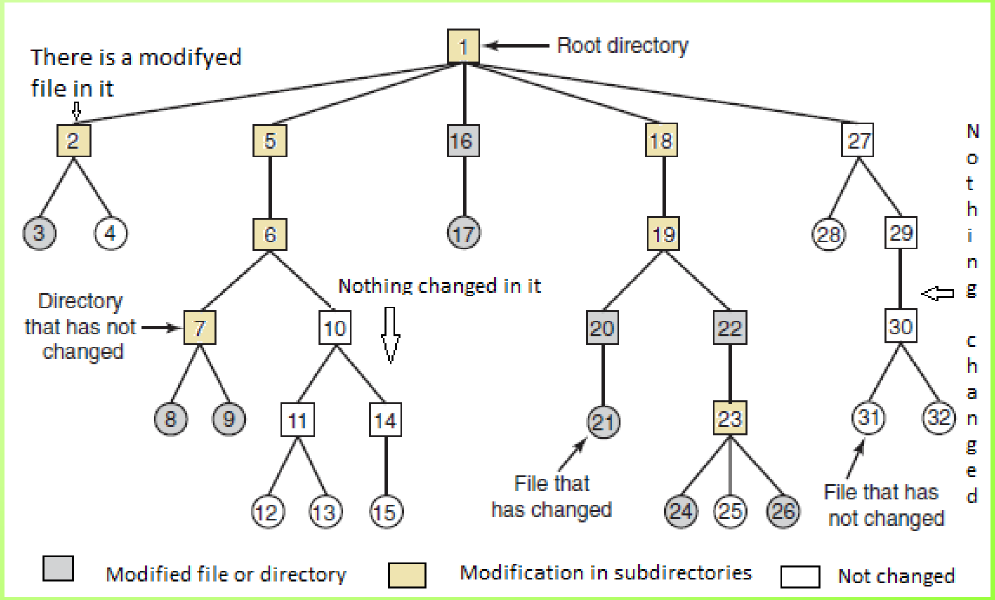


Backups

- Physical dump (copys everything from the 0th block)
  - Advantage (simple, quick)
  - Disadvantage (copy unused, failed blocks..)
- Logical dump (copys only the newer or modified files)
  - Advantage (may restore a required directory or file too, not only the entire disk)
  - Disadvantage (complicate algorithm)
- Mixed backups
  - Time to time a physical dump
  - Frequently a logical dump
- Restore (physical dump, logical dump1, … last

Logical dump – algorithm

- 4 step algorithm (used in most UNIX systems)
  - Mark each directory+modified files
  - Unmark directory in which there are no modified files or directories.
  - Dump the marked directories with attributes.
  - Dumo the marked files with attributes.



Root directory
There is a modifyed file in it
Directory that has not changed
Nothing changed in it
File that has changed
File that has not changed
Nothing changed
1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18
19
20
21
22
23
24
25
26
27
28
29
30
31
32
Modified file or directory
Modification in subdirectories
Not changed

Questions in logical dumping

- Free block list is not a file – so it is not dumped (But it can be restored, while it is the complement of used ones.)
- Links – several directories may contain a link to a modified file. (Restore all.)
- A file may contain holes (e.g. seek+write) – Allocate a huge area and fill it with 0?? (No)
- What to do with not real files (e.g. named pipe) – do not dump.

File, directory functions

- File
  - Open a file
  - Functions: read, write, append
  - Close a file
- File data
  - Binary- a sequence of bytes
  - Text file- a sequence of chars
- File access: sequentially, random
- Directory functions
  - Creating, list table of contents, delete

Filesystem types

- Filesystem types on HDD:
  - FAT, NTFS, EXT2FS, XFS, etc.
- On tape systems, typically used on backup system
  - LTFS- Linear Tape File System
- CD, DVD, Magneto-opto Disc filesystem
  - CDFS, UDF (Universal Disc Format)
- RAM drives (today rarelly used)
- FLASH memory (FAT32)
- Network drive
  - NFS
- Other pseudo filesystems
  - Zip, tar.gz, ISO

Journaled filesystems

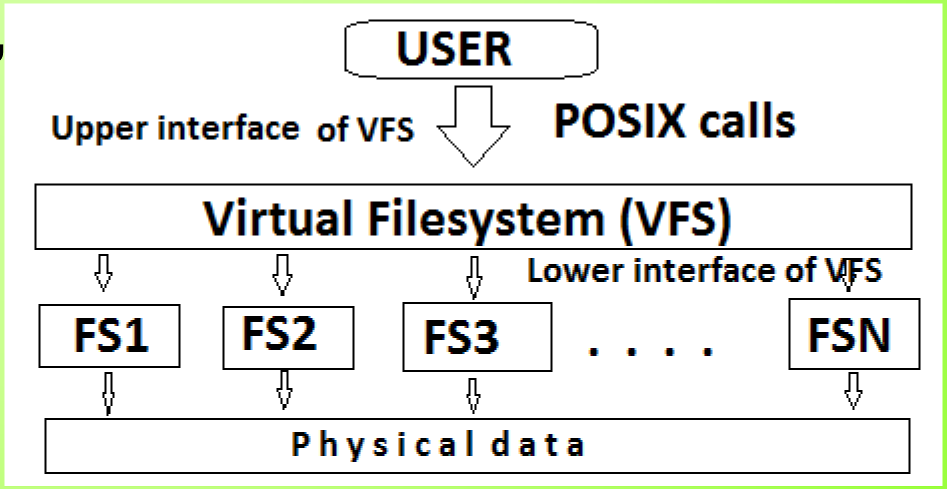
- On case of disk fails, power-cut, etc. The file system may be inconsistent, the storage content will be inaccessible.
- JFS often called as an LFS (Log-structured File System) too.
- As a database system, LFS based on a transaction system: task + log
  - On case of failure, the LFS repairs itself, file system check can run automatically too.
  - Log is stored too(sometimes on other artitions, disks)
- It needs more CPU load- but gives more safety!

Filesystem support

- Recent operating systems supports a lot of file systems.
  - E.g.: Linux 2.6 kernel supports more than 50.
- Mounting filesystems
  - Mount command, mounts all given in /etc/filesystems, during boot
  - Mount automatically (eg. USB drive)
  - Mount manually (Linux, mount parancs)
- Windows: Drive letters
  - A,B,C,…
- UNIX: uniform filesystem, one entry, /

Parallel usage of several filesystems on the same system

- Many different filesystems may be used on the same computer parallel
  - E.g. Windows: NTFS, FAT-32, UDF (DVD,CD) etc.
    - The systems uses them separately. Which one is the actual? The drive letter decides it: (c:, a:)
  - E.g.:UNIX: ext2,ext3,
    - The modern UNIX systems try to integrate them. They use a virtual filesystem (VFS) –



Application- Disk relation

- Layered architecture
  - User lewel
    - An application can access disk information via system libraries.
    - Text, binary operation
  - Operating system lewel
    - Filesystem implementations
      - Access, rights
    - Volume manager
    - Device driver
      - Based on BIOS
  - Hardware lewel
    - I/O device,IDE, SATA, SAS,etc.

FAT

- File Allocation Table
  - Rather old, but still alive filesystem!
- The FAT table acts as a disc allocation map, a bit describe a block allocation, 0-free, 1-reserved
  - E.g.: Fat12, FDD, Cluster size12 bits.
  - For safety 2 FAT's
- Chained storage
  - In the directory a file entry contains the file name, date,etc, and the first block index, where file data is stored.
  - The last bytes of block shows to the next block.
  - The last bytes FFF, if no more data is in the file.
- One directory entry size is fixed, it's equal 32 bytes. (max. 8.3 name, size, last modification date, etc)
- System,Hidden,Archive,Read only, directory attributes

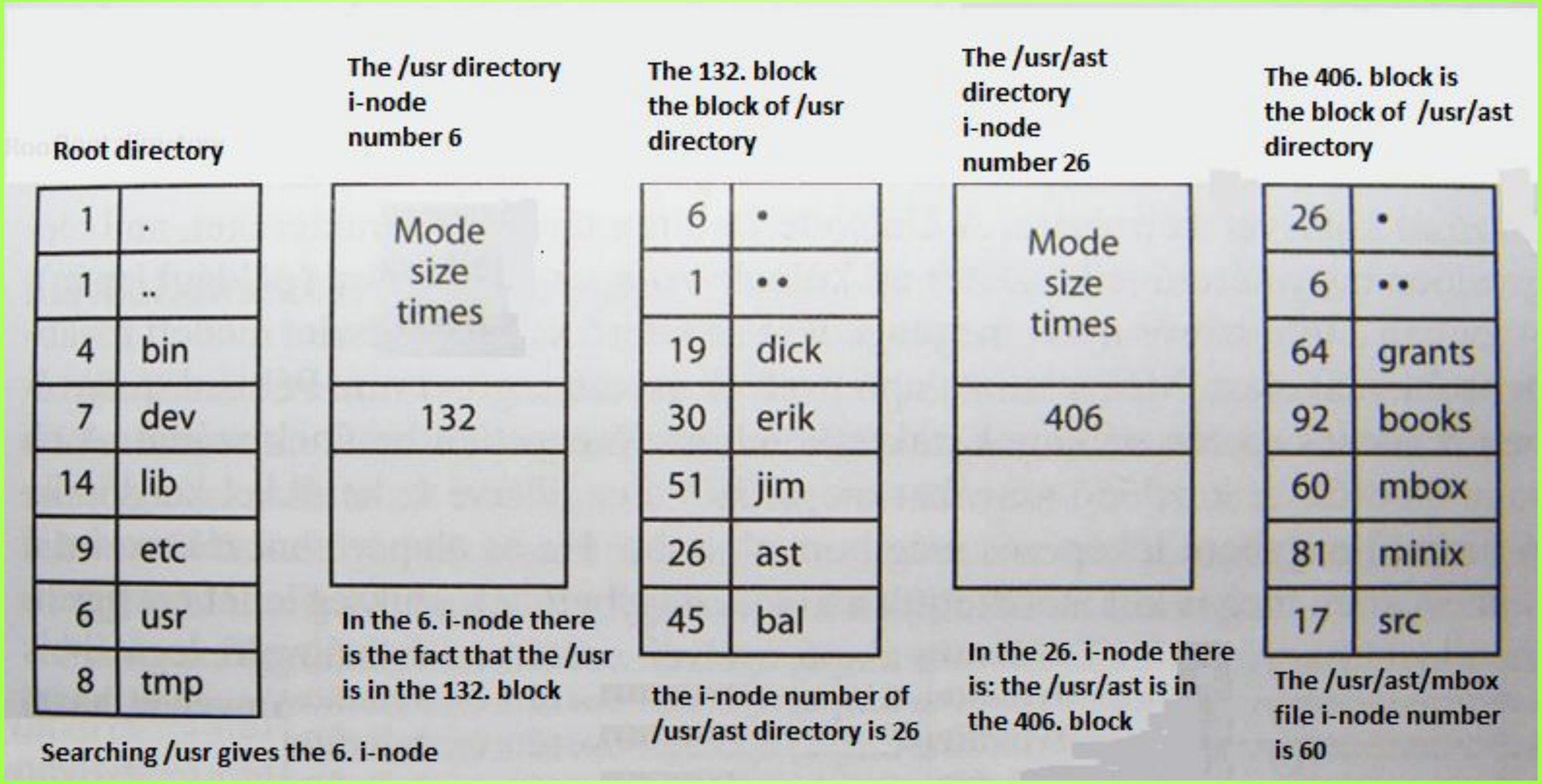
FAT properties

- FAT16, 16 bits cluster description, 4 byte (2x2) the first block address
  - Max. partition size: 4 GB (using 64kb blocks), using 32kb and less blocks, the max size is 2 GB.
  - Reserved directory blocks (on FDD this is the track 0)
    - On FDD 512 directory entry
    - On HDD 32736 directory entry(16 bits, signed)
- FAT32 (from 1996)
  - 28 bits cluster size
  - The maximum file size is 4 (2) GB.
  - 2 TB partition size (512 byte block size)
- 32MB-ig, 1 block = 1 sector(512bájt)
  - 64 MB, 1 block=1KB (2 sector), 128MB, 1 block=2KB
  - 1 block max. 64 KB lehet.
- Long file name support
- Defragmentation necessary.

UNIX directory structure

- Indextable directory structure
- After MBR, after partition boot sector follows a partition superblock
- Next is the free-reserved allocation part.
- i-node table, including root inode table
- Modular, lots of many table, quick access to file entry, this is the directory catalog structure.
- A file pointed by an i-node entry!
    - An i-node table entry contains 15 fields, the first 12 points to the file blocks.
    - If it is not enough, the 13th field points to a next i-node, that again +15 fields.
    - If it is not enough, the 14th field points to a next i-node, that again +15 fields, and repeat the first i-node architechture.

i-node chaining example


Root directory
The /usr directory i-node number 6
The 132. block the block of /usr directory
The /usr/ast directory i-node number 26
The 406. block is the block of /usr/ast directory
Mode size times
132
Mode size times
406
In the 6. i-node there is the fact that the /usr is in the 132. block
the i-node number of /usr/ast directory is 26
In the 26. i-node there is: the /usr/ast is in the 406. block
The /usr/ast/mbox file i-node number is 60
Searching /usr gives the 6. i-node

Root directory

1	.
1	..
4	bin
7	dev
14	lib
9	etc
6	usr
8	tmp

The 132. block the block of /usr directory

6	•
1	••
19	dick
30	erik
51	jim
26	ast
45	bal

The 406. block is the block of /usr/ast directory

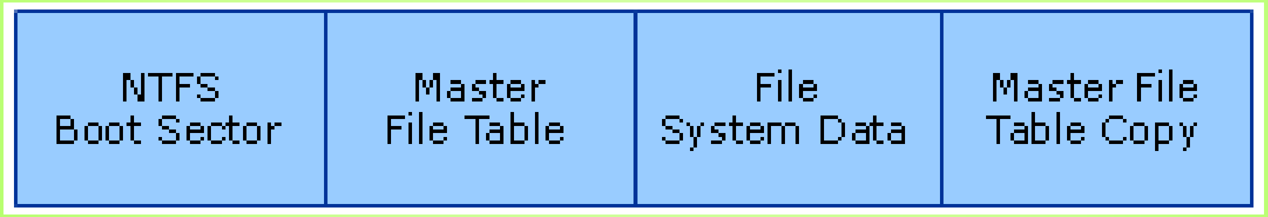
26	•
6	••
64	grants
92	books
60	mbox
81	minix
17	src

NTFS

- New Technology File System
  - FAT-NTFS efficiency border: appr. 400 MB.
- The max. filename length is 255 chars, 8+3 secondary name
- Fine tuning security settings
- Defragmentation also necessary.
- Encrypted filesystem, eventing subsystem
- POSIX support
  - Hard link (fsutil command), time stamps, case sensitive (sometimes, sometimes not☺)
- Compressed file, directory, user quote support
- NTFS is cluster based, default 4kb, so 1 cluster is 8 sectors (8*512byte)

NTFS partition architecture

- A Master File Table is a table.
- A Files System Data too.



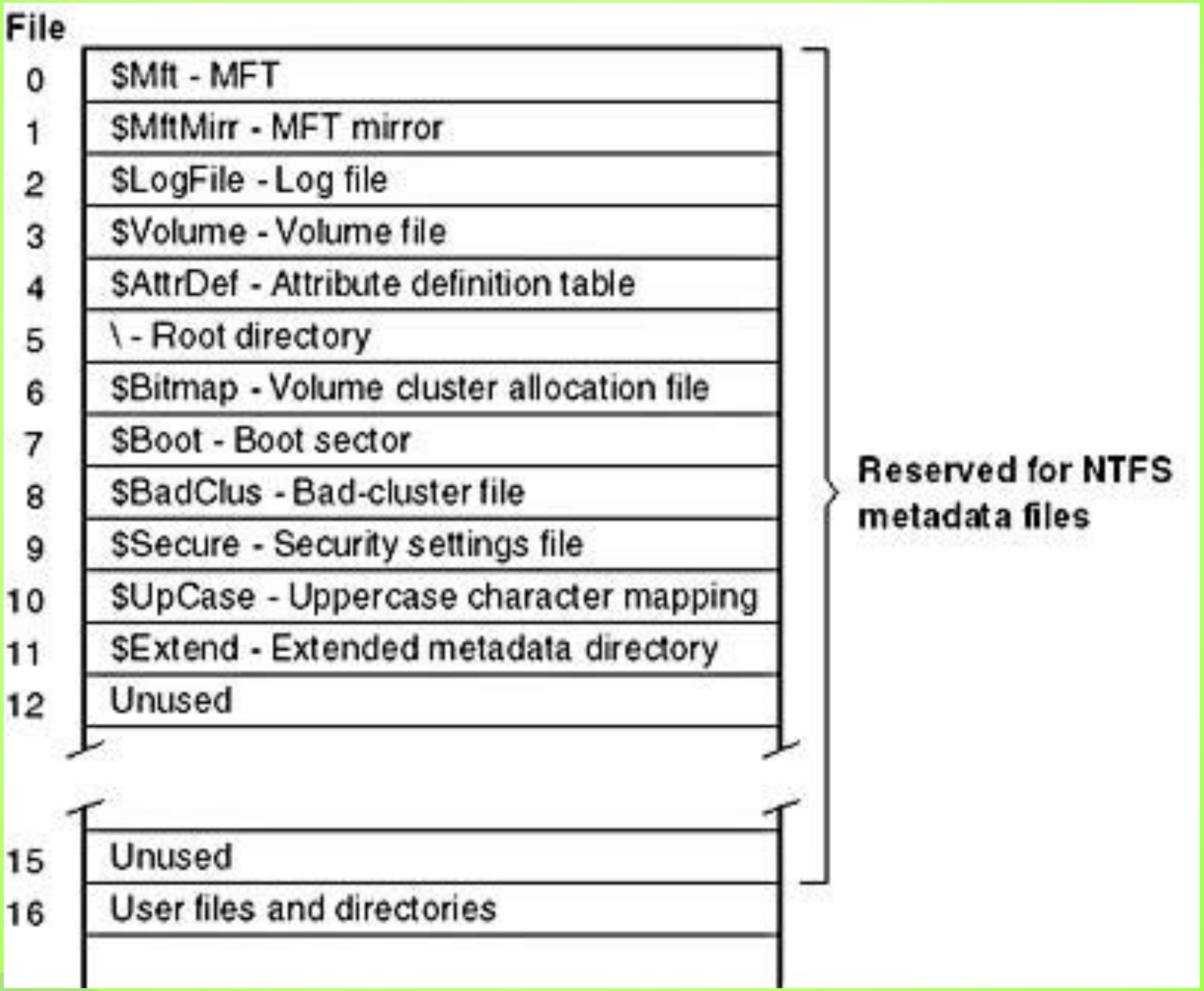
NTFS partition Boot sector

- Boot sector
  - JMP +0x52 (EB 52)
  - OEMID (8 byte, MSWINx.y)
  - BPB (Bios Parameter Block)
    - Bytes per sector (512)
    - Sectors per cluster (8)
  - Extended BPB
    - Total Sector number (8 byte-on tárolva)
    - LCN – Logical Cluster Number for MFT
    - Volume serial number
  - Loader code (loads ntldr.dll, and ntfs.sys,ntoskrnl.exe)
  - Sector end(0xAA55)

MFT

- NTFS partition begins with an MFT (Master File Table)
  - 16 attributes reserved for a file entry.
  - Each attributes is max. 1kb. If it is not enough, the last attr. points to the next entry.
  - Each data is an attribut, an entry can contain more data. (eg: preview picture)
  - Logical file size max is 2^64 byte
  - If file size is < 1kb, so the file content in the attribut, direct file.
  - No file size maximum.

NTFS table details



File	
0	$Mft - MFT
1	$MftMirr - MFT mirror
2	$LogFile - Log file
3	$Volume - Volume file
4	$AttrDef - Attribute definition table
5	\ - Root directory
6	$Bitmap - Volume cluster allocation file
7	$Boot - Boot sector
8	$BadClus - Bad-cluster file
9	$Secure - Security settings file
10	$UpCase - Uppercase character mapping
11	$Extend - Extended metadata directory
12	Unused
15	Unused
16	User files and directories

Reserved for NTFS metadata files

Thanks for your attention!

zoltan.illes@elte.hu