Handling of Permanent Storage

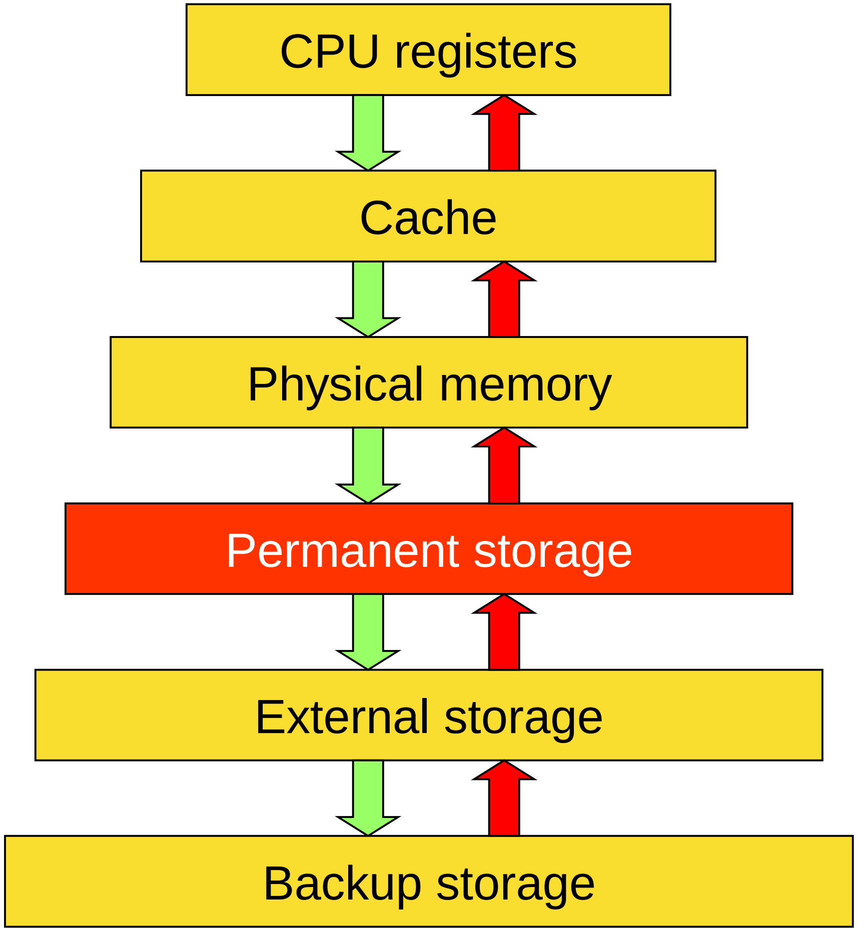
Tamás Kovácsházy, Phd

20th topic

Handling of Permanent Storage

Méréstechnika és
Információs Rendszerek
Tanszék

Permanent storage



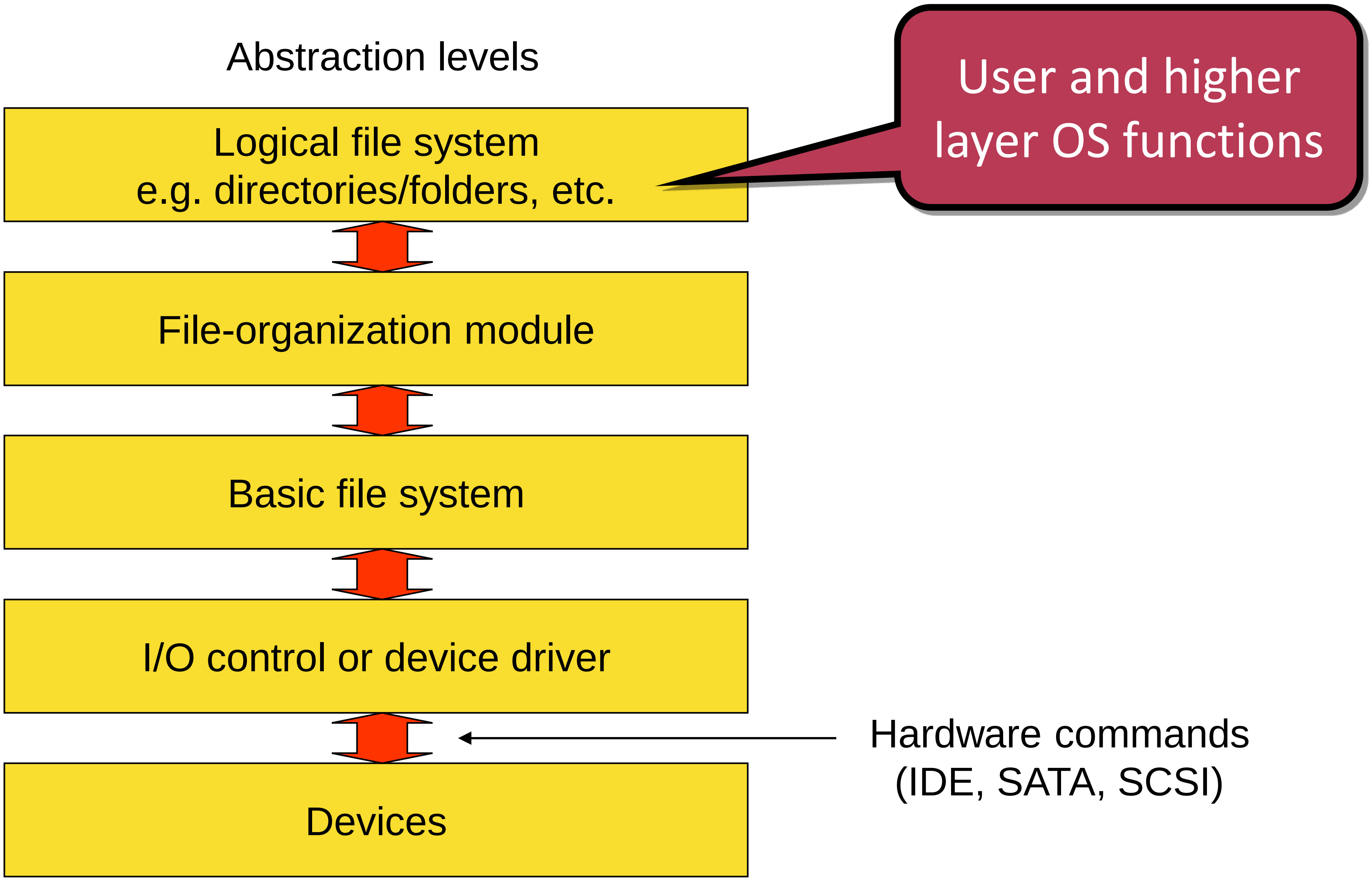
Handling of the Permanent storage

- Permanent storage or “storage”:
  - Typically compared to the physical memory
  - It offers orders of magnitudes bigger storage capacity
  - Also orders of magnitudes slower
    - Throughput
    - Latency
  - Nonvolatile storage
    - If properly used, otherwise it do losses data
    - Data security is a major issue!
  - Block based
    - OS handles this storage based on blocks
      - No byte access, a full block must be handled
    - A block can be read, written or erased
    - Programs cannot be executed directly from storage, it must be loaded into memory first!
    - One exception
      - NOR flash memory is organized as bytes (as regular memory)
      - It is possible to execute the OS or other programs directly from NOR flash

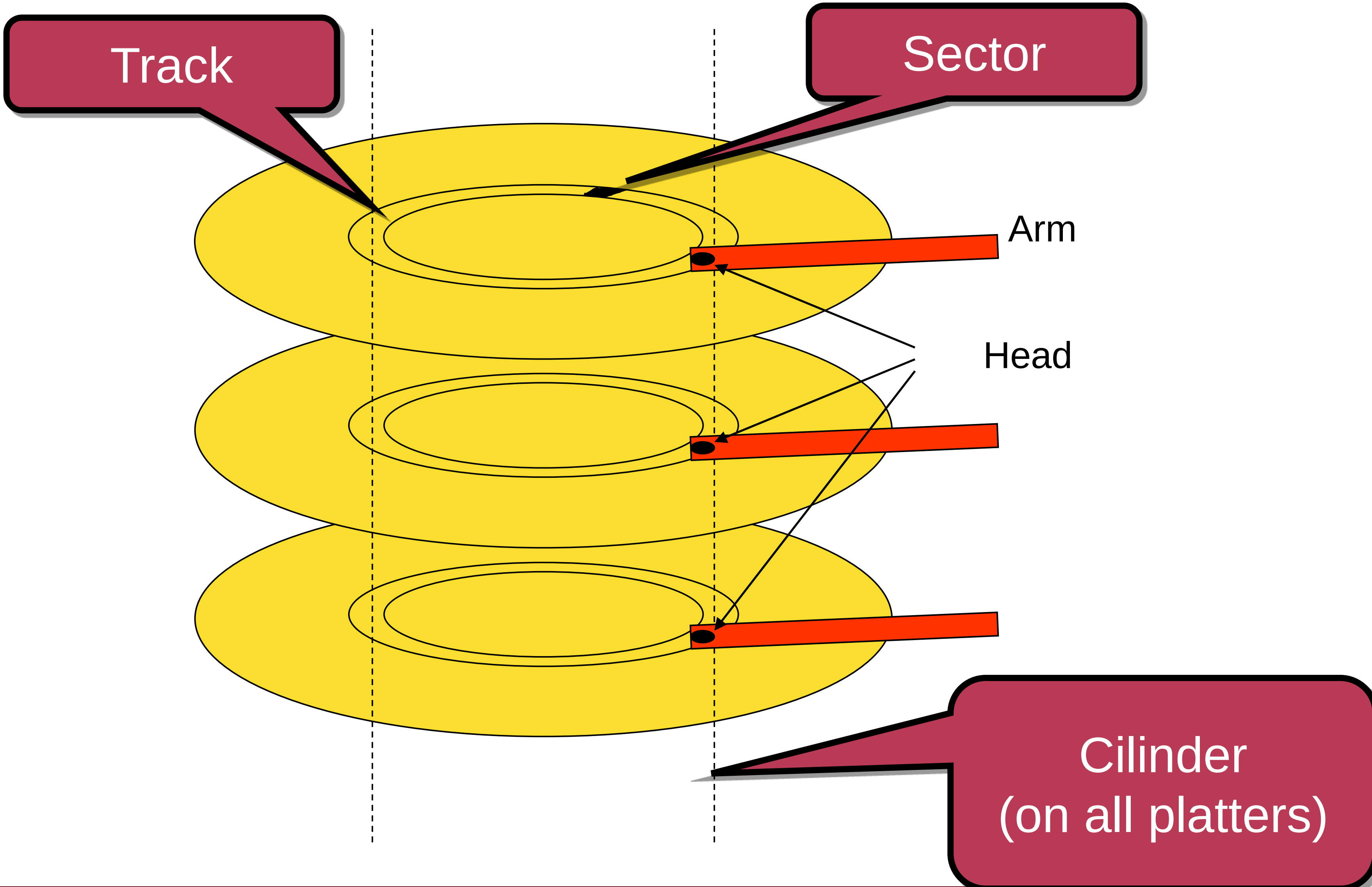
How we map files to storage?

- The file is the logical unit of data storage on the permanent strorage (file)
  - It has a name (named collection).
  - We reference it by its name
  - It’s size can vary, practically any size is allowed
    - There is a hard limit coming from the physical storage and the filesystem
- The main task of the operating system regarding permanent storage is the mapping of files (logical unit) to blocks (physical unit)
- This task is solved by the OS using a multilayer hierarchical system, solving the problem on multiple abstraction level
- On the lowest layer you will find some special HW (HDD, Flash drive, etc.)
  - RAM drive is an exception here

Abstraction levels (simplified)



Devices: Hard Disk Drive



Hard disk drive terminology

- Hard Disk Drive (HDD):
  - Rotating magnetized platters
  - Read/write heads are mounted on a movable arm
  - A set of data available in a head position is a track
  - Tracks on multiple platters are handled as a cylinder
  - A track is split into sectors
  - The cylinder, track, and sector identifies the readable/writable data block
  - In practice we use logical addressing (Logical Block Addressing)
    - The device transforms logical block addresses to physical ones
  - LBA: 48 or 64 bit addressing today
  - The sector/block size is 512 byte or 4 Kbyte (new advancement)
    - We use both today
    - Performance may vary do to block size

The real speed of an HDD?

- Strongly depends on actual head position and how the requested data positioned relative to it, and how fast platters rotate
  - Average and maximum seek time (latency)
  - It takes time to position the head over the cylinder storing the data than waiting for the sector to move under the head…
- Multiple level of optimization:
  - Disk scheduling
    - What is the optimal order of serving the incoming requests?
    - Optimizing head movement
  - HDD level optimization (SATA NCQ, SCSI).
  - Operating system level optimization
  - Prefetch...
- Multiple level of caches:
  - HDD level cache (16-64 Mbyte typically).
  - Hard disk controller cache (on expensive RAID controllers)
  - Operating system level cache:
    - Disk cache, dynamically changing size as memory requirements change

Devices: NAND flash storage

- Low level hardware interface is identical to the interface of HDDs
  - Solid State Disk (SSD) with SATA or IDE interface
  - PEN drive with USB interface
    - Flash card readers work just like PEN drives but with removable FLASH memory
- Reading the data is fast independently from the location of the data on the device:
  - There are no heads to move, and no sectors to move into position
  - More like RAM, but addressed with blocks (the unit of reading, writing, erasing)
- Writing (erasing actually) is problematic:
  - A flash memory cell can be erased a very limited number of times
    - Cheap PEN drive: some thousand of times
    - High end server SSD: some millions of times
  - Writing/erasing is slower also than reading
    - It is executed in parallel on multiple flash chips on bigger devices (SSD)
  - Wear leveling (device and/or OS level)
    - E.g. on OS level special file system can provide this: JFFS2, YAFFS, UDF (on optical medium), ZFS has built in support for it
  - TRIM (SSD + OS support required): The number of writes can be reduced to avoid the effects of „write amplification”
    - Write amplification: Accessing a block results multiple writes on other blocks
    - E.g. Last access time must be written if a block is read!

Device attachment

- How the storage device is attached to the computer?
- Host-Attached Storage:
  - Direct connection to the computer: SATA/eSATA, IDE, SCSI, SAS, etc.
  - Indirect connection:
    - USB, Firewire based tunnel
    - RAID (Redundant Array of Inexpensive Disks)
- Storage-Area Networks (SAN):
  - A network tunnel between the host and the storage device using **block based access**
    - File level access handled on the host!
  - Dedicated storage solution: Fibre channel
  - Ethernet and/or TCP/IP based: iSCSI, AoE
- Network-Attached Storage (NAS):
  - A network tunnel between the host and the storage device using a **file based access**
  - TCP/IP based: NFS, SMB/CIFS

USB

- USB mass storage device class.
- SCSI command set is tunneled through the USB bus transparently
  - The OS sees it as a device connected to the SCSI bus
  - USB only creates a tunnel between the device and the OS

RAID

- Facts:
  - HDDs and SSDs are relatively cheap
  - They are not reliable enough (HDD: moving components, SSD: wear)
  - HDDs are slow, and SSDs are not fast enough compared to physical memory
- Idea:
  - Use multiple one of them…
  - Redundant usage of multiple devices may increase reliability
  - Parallel usage of multiple devices may increase speed
  - Let’s create a virtual disk from multiple physical disks
    - The virtual disk will be handled by the OS...
    - This virtual disk is called as “RAID array”
- Implementations:
  - HW RAID controllers
  - SW RAID solutions
    - Motherboard RAID solutions are like this nearly exceptionally
    - Some server motherboards may have a HW RAID controller

RAID levels

- RAID 0-6 and nested levels
- RAID 0-1 levels are typically implemented using SW RAID and from small number of disks (typically using 2 disks)
- RAID 2-4 levels are rarely used today
- RAID 5 and 6 are the typical today for larger number of disks
  - 4 disks or more
- Nested RAID levels:
  - RAID 1+0 or RAID 0+1.
- There are manufacturer specific proprietary RAID levels also…

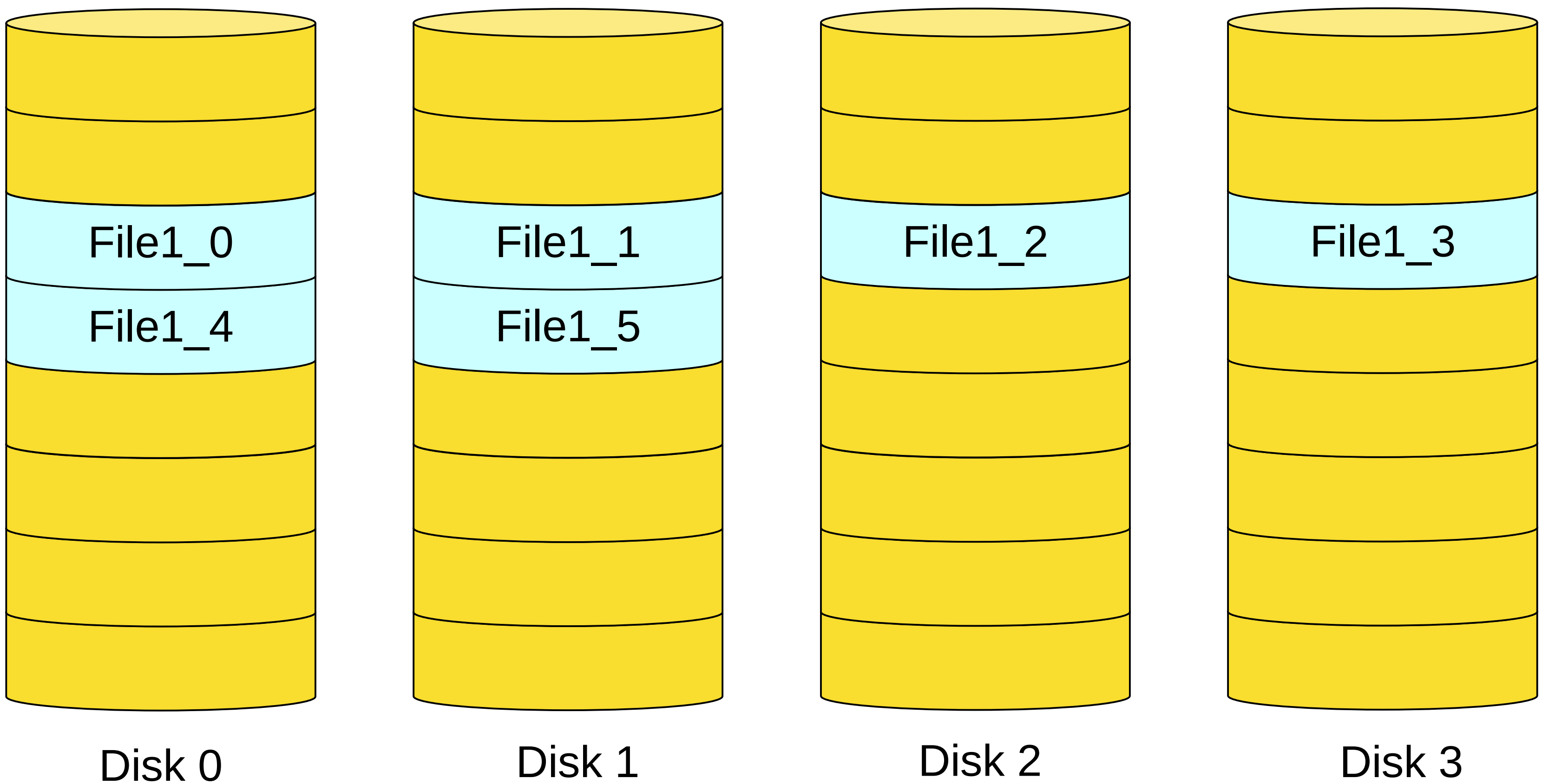
MŰEGYETEM 1782

Méréstechnika és Információs Rendszerek Tanszék

RAID level 0

- RAID 0 (striped disks).
- Multiple disks are used in parallel
- The file is split into small parts stored and those parts are stored on the disks:
  - This parts can be access in parallel if they are located on different disks
    - The storage capacity of the disks adds up
    - In case of N disks the read and write speed is multiplied by somewhat lower than N (due to overhead)
    - The access time (latency) is close to the access time of one disk due to overhead of the RAID controller
    - If any of the disks fail the file cannot be accessed

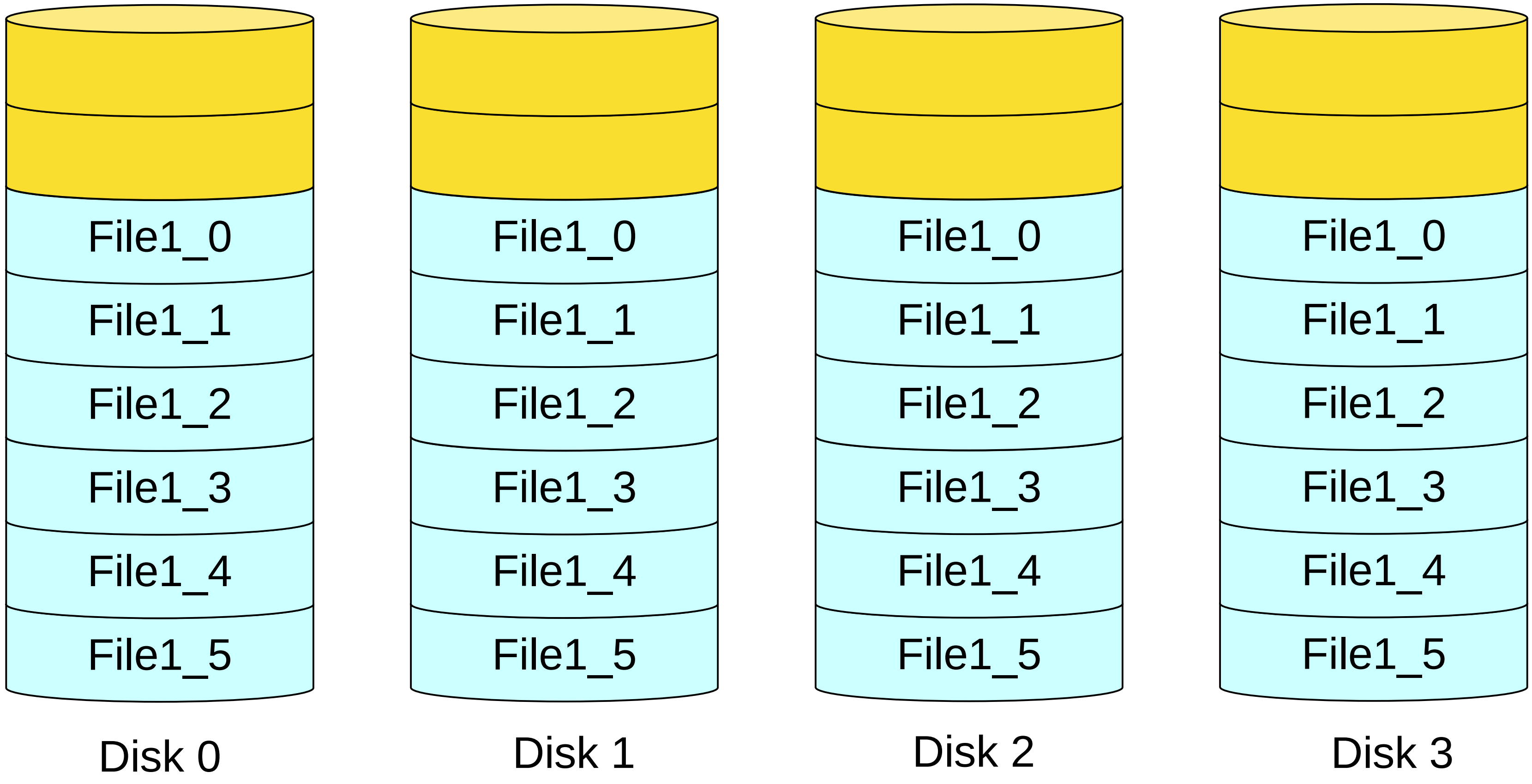
RAID level 0


File1_0
File1_4
File1_1
File1_5
File1_2
File1_3
Disk 0
Disk 1
Disk 2
Disk 3

RAID level 1

- RAID 1 (mirroring).
- Using multiple redundant disks
- All parts of a file is written to all (N) disks
  - Assuming that the disks are identical the available storage capacity is the capacity of one disk
  - Read and write speed is somewhat slower then for one disks (due to overhead)
  - Access time is increased due to overhead
  - In a special case read speed my be close to N times faster
    - If we assume that the failure of one disk can be identified without explicitly reading the content from all disks and doing a majority vote
    - In this case the parts of the file can be read just like at RAID 0
  - One operation disk is enough (N-1 is allowed to fail)

RAID level 1

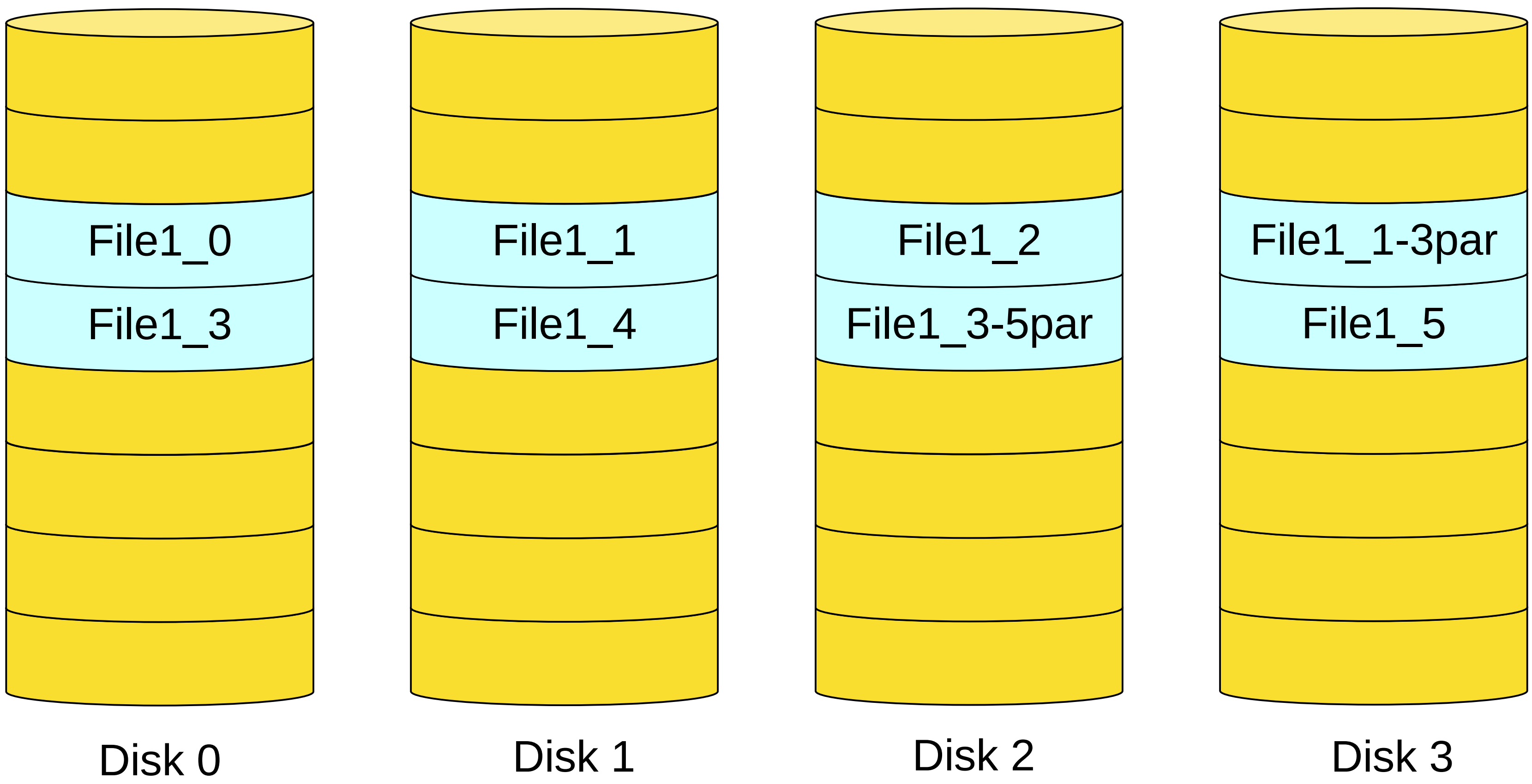


File1_0
File1_1
File1_2
File1_3
File1_4
File1_5
Disk 0
File1_0
File1_1
File1_2
File1_3
File1_4
File1_5
Disk 1
File1_0
File1_1
File1_2
File1_3
File1_4
File1_5
Disk 2
File1_0
File1_1
File1_2
File1_3
File1_4
File1_5
Disk 3

RAID level 5

- RAID 5 (block interleaved distributed parity).
- Multiple disks are used redundantly and in parallel
- Data and parity are stored on N+1 diszk
  - Read and write speed are close to the speed of a RAID 0 array (if HW controller is used)
  - Storage capacity is N times the storage capacity of one disk
  - If 1 disk fails the data can be accessed
  - If 2 or more disks fail the data is lost
  - In case of one identified disk failure the data not necessarily reconstructable!
    - Silent errors may be present on the "good" disks
    - These silent errors are identified when the array is reconstructed after the replacement of the failed disk

RAID level 5



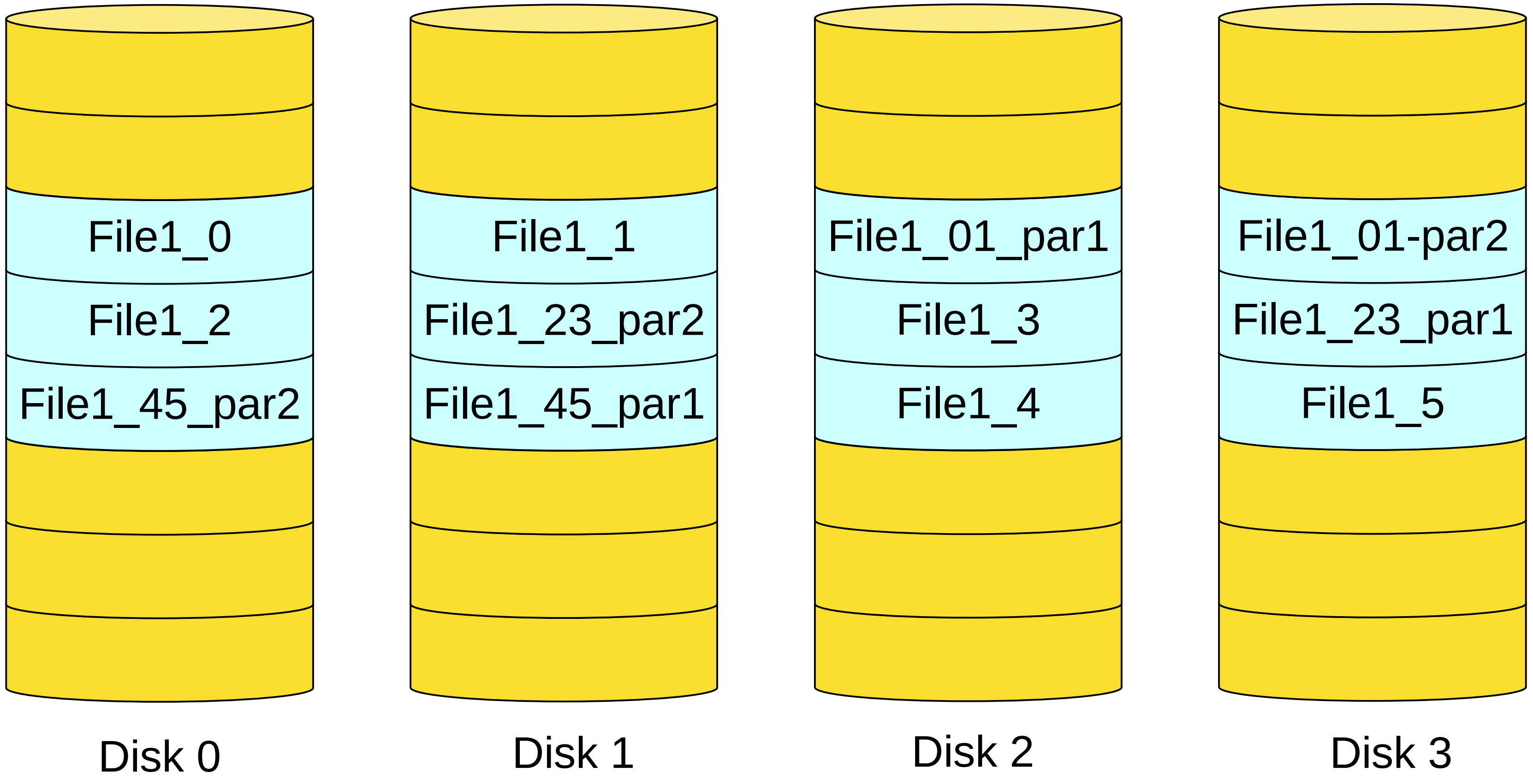
File1_0
File1_3
File1_1
File1_4
File1_2
File1_3-5par
File1_1-3par
File1_5
Disk 0
Disk 1
Disk 2
Disk 3

RAID level 6

- RAID 6 (block interleaved dual distributed parity)
- Multiple disk are used redundantly and in parallel
- The data and parity are stored on N+2 disks.
  - Read and write speed are close to the speed of a RAID 0 array (if HW controller is used)
  - Storage capacity is N times the storage capacity of one disk
  - If 2 disks fail the data can be accessed
    - If no silent errors are present!
    - If one disk fails it must be replaced immediately to maintain the error margin for the one silent error
  - If 3 or more disks fail the data is lost
  - If we always replace a failed disk (one error assumption) another silent/hidden error can be corrected, i.e., the survivability of the array is better

MŰEGYETEM 1782

RAID level 6


File1_0
File1_2
File1_45_par2
File1_1
File1_23_par2
File1_45_par1
File1_01_par1
File1_3
File1_4
File1_01-par2
File1_23_par1
File1_5
Disk 0
Disk 1
Disk 2
Disk 3

Disadvantages of RAID

- Falsified sense of data security:
  - It can correct only the isolated and random failure of a drive.
  - It cannot save data if the power supply fries all the HDDs with some overvoltage (can happen any time)
  - It cannot correct data loss due to SW problems, viruses, and malicious user access, etc.
  - It is not a replacement for data backup, only increases availability and/or speed of the storage subsystem
- HW RAID controllers are expensive>
  - 8 port SATA RAID with RAID 5/6 support in the 800 USD range
  - It is in the same price range as the disks attached to it
  - A second low end server can be bought from this
- SW RAID is slow, they are for RAID 0/1 primarily
  - Implementation of RAID 5/6 is slow in software due to complex coding required for parity computation

Advantages of RAID

- RAID 1/5/6 reduces the possibility of unscheduled downtime due to random disks failures
  - The HDD is the weakest element of computer systems:
    - E.g. Google HDD statistics
    - http://labs.google.com/papers/disk_failures.pdf
    - Temperature does not influence HDD lifetime as much as earlier reports suggested
    - SMART (Self-Monitoring, Analysis, and Reporting Technology) is efficient in predicting disk failures, but disks fail without any signs found in SMART previously (some errors cannot be predicted based on SMART)
    - Who uses SMART? (Everybody should try it!)
      - Smartmontools + GSmartControl or HDD Guardian
      - Some HDDs and SSDs have compatibility problems (see list)
  - Optical drives can also fail miserably (the story of the failing CD drive)
- RAID 0/5/6 speeds up disk read and write speed
  - HDD is the weakest link also from this point of view

Storage Area Network (SAN)

- A network tunnel is built between the storage and the host
  - Low level, block based solution
  - Send SCSI commands most cases
  - Practically, it is a storage virtualization solution:
    - Provides the same properties as local storage, but it is more scalable, can be freely partitioned, bootable, etc.
    - It works like storage attached locally
    - In one time it can be attached to only one host (some exceptions can be found in clusters).
- Solutions:
  - Special protocol: Fibre channel (expensive, requires dedicated HW)
  - Ethernet and/or TCP/IP based: iSCSI, AoE.
    - Cheap or free of charge (OS built in), partial or full SW solution, for booting special firmware required for the network cards (e.g., to boot a PC from iSCSI a special firmware is required for the network cards, for example Intel provides that for server network cards).
- Conventions:
  - Target: The server device on the network, which makes the storage available on the network (the storage is attached to it directly or it accesses it through nested SAN layers)
  - Initiator: The client that uses the storage by accessing it through the target device
  - Naming conventions to identify the designated storage
    - iSCSI Qualified Name (IQN)
  - Initiator level access control
    - The decision is made if a given initiator can access the or not (block based access)

Abstraction levels (simplified)

Abstraction levels

- Logical file system
  e.g. directories/folders, etc.
- File-organization module
- Basic file system
- I/O control or device driver
- Devices

User and higher layer OS functions

Hardware commands (IDE, SATA, SCSI)

Méréstechnika és Információs Rendszerek Tanszék

Basic file system

- The main task of this layer is writing and reading blocks on the physical disks
- It has to do caching also
  - Buffer cache:
    - A different cache is used for paging (page file cache) and for caching of physical blocks
    - Practically it is a double caching scheme (the two caches are developed for different reasons in parallel)
  - Unified buffer cache:
    - The cache operate only on block level, there is no page file cache
  - Unified virtual memory:
    - Paging and OS level filesystem cache is unified in a single cache
    - The file is mapped to virtual memory
    - E.g. Linux and Windows use it
    - Linux: As long as there is free memory it is used for file cache

File-organization module

- The task of this layer is to map logical blocks (parts of files) to physical blocks (allocation)
- Solutions
  - Contiguous allocation
  - Linked allocation
  - Indexed allocation
- Free-space management:
  - Bit vector
  - Linked list
  - Grouping of free spaces
  - Counting
  - Space maps
  - We do not address free-space management in this class...

Contiguous allocation 1st slide

- The file occupies a contiguous set of physical blocks
  - Access is simple and fast in case of a HDD (sectors and tracks can be accessed contiguously from HDD).
  - Growing files cause serious problems:
    - How big space should be allocated for growing files to allow growth?
  - Finding space for new files is problematic, external fragmentation cannot be avoided
    - After erasing a file the physical blocks storing the file are put on the free list
    - For this free space a smaller or equal sized file can be written later
      - The same algorithms can be used as with memory allocation (first fit, next fit, best fit, worst fit)

Contiguous allocation 2nd slide

- Growing files:
  - Best fit allocation strategy is very dangerous
  - After exhausting the free space the file needs to be copied to bigger free space which is extremely resource intensive
- Reducing external fragmentation:
  - Copying the whole drive to a new one and then back to the old one (off-line).
    - A scheduled downtime is required to do it
    - It takes long time and requires lot of resources
  - Reducing external fragmentation on-line (defragmentation)
    - Resource intensive also
    - The performance of the system degrades during the process

Linked allocation 1st

- Data structures on the disk store the identification of the first and the last block of the file
- All blocks may store the identification of the next and previous block in the file
- The blocks storing the file can be located anywhere on the disk
- No external fragmentation
- Problems:
  - Sequential file access is simple and fast, but indexing into files are resource intensive (direct access to the n^{th} block requires to read all blocks before or after the block).
  - Storing the identification of the next and previous block in the blocks reduces space available in the block
  - Fragile: damage to a block renders the file inaccessible
  - It increases head movement (seek) if the blocks are not contiguous on the disk
    - Lot of time is spent seeking for the parts of the file

Linked allocation 2nd

- For example the FAT file system uses this method
- Defragmentation means a different process here:
  - The aim is to reduce head movement during file access
    - It is not reasonable to do it on an SSD
      - There is no head movement, and writes caused by the process reduces the lifetime of the SSD
  - The aim is achieved by organizing the file into contiguous physical blocks
  - It is also called defragmentation...
  - This is why it is reasonable to run the defragmentation program under Windows on FAT file systems
    - It speeds up file access
    - Not only read speed, but write speed also increased!
      - Free space is also organized into contiguous regions...

Indexed allocation

- It uses special index blocks to store information about the files:
  - Some blocks are allocated to store indexes (metadata)
  - Other blocks store the files (and only the content of the files)
- It is efficient for both sequential and random access
  - It is enough to read only the index blocks to locate any part of the file
- Fragile:
  - No readable index block makes the file inaccessible
  - Index blocks can be replicated easily
- It causes lot of head movement if the blocks of a file is spread on the disk
  - A defragmentation algorithm similar to the algorithms applied to the linked allocation may be used to minimize head movement and speed up disk access

Logical file system

- Operating system specific
  - An operating system specific API is on the top of this layes
  - Typical API functions:
    - Create, Delete, Read, Write, Set/Get attributes, etc.
- Storing metadata (everything required to store the file except the data of the file itself)
  - Due to metadata and other inefficiencies we can store less file than the size of the storage medium would suggest
- File:
  - Abstract data type (object or file pointer)
  - It has a name, type, and other properties
  - File locking is also provided
- Directory/Folder
- Volume/Drive

File

- The file is the logical unit of information storage on the permanent storage
- Properties:
  - Name:
    - Naming conventions, OS specific (see the Windows/UNIX differences).
    - Is it a unique identifier in a folder/directory?
      - Most cases it is, but there are exceptions in some older OSs...
  - Type (specifies how the OS handles the file):
    - E.g. Windows uses the extension (.*), but in UNIX file type is partially mapped to the file system (regular file, symbolic link, device, etc.) partially decided based on extension and file content
  - Access times
    - E.g. Creation time, Last modification time (write), last access time (read or write)
  - Access rights (User and type of access are specified)
  - Other OS specific information

Directories/Folders

- Hierarchic storage of information...
- Implementations:
  - Single-level, used in early operating systems
  - Two-level, it was used even in the 1990s (e.g. IBM OS/400).
  - Tree-structured file system
  - Acyclic-graph file system
  - General graph file system

Acyclic-graph file system

- A file or subdirectory can be mapped to multiple directory (but not into itself or under itself)
- It exists only in one instance, it is only mapped into multiple directory!!!
- E.g. UNIX/Linux hard or symbolic links
  - The „hard or symbolic link” type linkage identifiable through file properties
    - It is possible to iterate through he file system due to it, which is a must...
  - What if a file or directory linked into multiple directories is deleted?
    - Only the reference is deleted, the file is preserved
    - Not deleted as long as all references are deleted
    - The file is deleted with all the links

General graph file system

- It is not used in operating systems
  - Searching for files is algorithmically complex
  - How we can stop the search?
  - The WEB can be considered as implementing a general graph structure
    - It is not a real file system, but search on the WEB is quite problematic (It is hidden from us by Google and other companies, though)

Drives or volumes

- It is the highest layer on the level of logical file system
- It is mapped to a physical or logical partition on the physical storage
- How they appear in the operating system?
  - They have a unique identifier in the OS (e.g. Windows drive letters, e.g. C:)
    - It is the first branch in the tree of the file system…
  - They can be mapped to any location in the file system (UNIX/Linux mount, newest Windows OSs)

Data structures on the device

- Low level data structures
  - Boot control block
    - Loaded by the firmware (BIOS or EFI in cases of the PC platform) to load the operating system
  - Volume control block
    - Stores partition/volume specific data
    - Partition size and location, unpartitioned space, etc. is store here
  - File system specific information
    - Description of the directory/folder structure
    - File descriptions (File Control Block)

Data loss...

- The files may be present in the memory and also on the permanent storage
  - These two versions may be different:
    - E.g. the copy in the memory may be never...
  - The metadata and allocation structures can be also under modification
  - A malfunction or loss of power may cause inconsistency
- Consistency check
  - Can be done on-line, but repair can be done off-line some cases
- Keeping the file system consistent continously
  - Transaction oriented file systems
    - Other names: Log-structured, log-based transaction oriented, journaling
    - E.g. NTFS, EXT3 és EXT4.
    - It does not provide data security, it keeps at least a working copy, but not necessarily the new one!
- Safe system shutdown even in case of power loss
  - Uninterruptible Power Supply (UPS) with properly configured software to shut down the system in case of long power problems
- Backup and system restore
  - Schedule of backups
  - Backups must be tested if restore is possible using them (can be done on a test system)
  - Without a successful restore from backup data security cannot be guaranteed!

Some common file systems 1st

- FAT (File Allocation Table)
  - 8+3 character file name, long file names are stored in a special file...
  - FAT16 (the first one was FAT12)
    - Max. 2GB partition size
    - 32767 directory entry
    - It is even used today on smaller pendrives and memory cards
  - FAT32
    - 2 TByte (TiB in the SI system) partition size
      - For other reasons there is a 64 GByte or 128 GByte-os partition size limit in earlier version of Windows
    - Maximum file size: 4 GByte-1 byte
      - This is why it is impossible to copy some large files to portable storage...
- NTFS (New Technology File System)
  - 2^{64} Byte (16 EB) - 1 KByte max. file size, 2^{32}-1 files, etc.
  - 2^{64} sectors on a partition.
  - 256 character long file names
  - Transaction based
  - Needs defragmentation

Some common file systems 1st

- FAT (File Allocation Table)
  - 8+3 character file name, long file
  - FAT16 (the first one was FAT12)
    - Max. 2GB partition size
    - 32767 directory entry
    - It is even used today on smaller
  - FAT32
    - 2 TByte (TiB in the SI system) par
      - For other reasons there is a 64 r version of Windows
    - Maximum file size: 4 GByte-1 byte
      - This is why it is impossible to copy arge files to portable storage...
- NTFS (New Technology File System)
  - 2^{64} Byte (16 EB) - 1 KByte max. file size, 2^{32}-1 files, etc.
  - 2^{64} sectors on a partition.
  - 256 character long file names
  - Transaction based
  - Needs defragmentation

The FAT and NTFS file systems are case insensitive regarding the file names for historical reasons…

Some common file systems 2nd

- EXT2
  - Default file system in earlier Linux versions
  - There exists a Windows driver for it
    - It does not support EXT2 on the system partition
  - Max. file size: 16 GByte - 2 TByte (depends on the block size)
  - Max. number of files: 10^{18}
  - Max. file name length: 255 byte (case sensitive).
  - Max. partition size: 2-32 TB (depends on Linux kernel).
  - Fragmentation is slow, defragmentation is rarely needed, and can be done only off-line
- EXT3
  - Enhanced EXT2, transaction based
  - Htree based indexing, makes possible to make more directories
  - It is easy to convert file systems between EXT2 and EXT3
- EXT4: Further enhancements to EXT3 (bigger storage, extents, etc.).
  - Linux distributions use it as the default file system
  - Last in the EXT filesystem family, BTRFS will follow with lot of new features
- CD-ROM/DVD file system (ISO 9660, Rock Ridge, Joliet, El Torito extensions)
  - The max. file size is 2/4 Gbyte
  - This is why files are split to smaller parts on DVDs

Future filesystems

- ZFS (It was introduced late 2005)
  - Developed by SUN Microsystems, now by Oracle
  - Open source, but not GPL license (OpenZFS is GNU, but with limited features)
    - Cannot be introduced into the Linux kernel…
  - Lot of new features…
    - Extreme focus on data integrity achieving much better data integrity than other available filesystem solutions (including BTRFS),
    - Copy on write transactional model support,
    - Support for RAID like RAID-Z features (use of HW RAID is not recommended under ZFS)
    - On-line resilvering and scrub for detecting and repairing file system integrity and detecting silent errors on physical disks,
    - Snapshot and clone support,
    - Storage pool support (LVM like features),
    - Multiple-level file system caching including RAM, fast disk (SSD or fast HDD) caching,
    - Filesystem compression and encryption
    - Deduplication,
    - Clustering and high availability,
- Btrfs (B-tree file system)
  - GPL licenced open source alternative
  - Less features and not as stable as ZFS

NAS (Network-Attached Storage)

- File system level file sharing
  - Most cases printers can be shared also using this technology...
- Examples:
  - Network File System (NFS).
    - Primarily UNIX type OSs, but there exists Windows implementation also
  - Server Message Block / Common Internet File System (SMB/CIFS)
    - Primarily Windows, but it is available on UNIX type OSs (SAMBA).
- File system level sharing
  - The network transports directory and file level commands (open, close, read, write files)
  - It is typically multiuser
  - Access rights are handled on the user and file level
  - Server and client file handling conventions may be different causing problems
    - E.g. UNIX and Windows file names and properties are very different
- The HTTP protocol is a totally different thing, it is a file access protocol
  - Primarily the complete file is read or written

Monitoring file systems

- Low level (block based) monitoring
  - Sysinternals: Disk Monitor (diskmon.exe)
  - Must be executed as system administrator
- High level (file based) monitoring
  - Sysinternals: Process Monitor (procmon.exe)
- It is necessary to observe that on low level much less activity is detected
  - Why?
  - Caching…