Infrastructures et Stockage et de Traitement de Données - IST

Leonard Cseres | June 20, 2025

Storage Systems - Levels of Abstraction

- **Block Storage:** Raw blocks of data. Accessed via **Block-level API**. Examples: Hard disk, SSD.
- **File Storage:** Data organized as files and directories. Accessed via **File system-level API**. Uses a file system on top of block storage.
- **Database Storage:** Data managed within a database. Accessed via **Database-level API**. Often uses file storage and block storage underneath.

Block Storage Layer

- Provides low-level persistent storage.
- Examples: Hard disk (SSD or magnetic), floppy disk, tape, CD, DVD, Blu-ray.
- Most are **random access**, some for backup are **serial access** (magnetic tape).
- Abstraction: **block device** with numbered blocks of fixed size (typically **4 KB**).

Magnetic Disk / Hard Drive

- Spindle rotates platters at **5'400 - 12'000 rounds per minute**.
- Data stored as magnetic orientation on surfaces.
- Read-write head floats on air cushion (~**3 nm**).
- **Track:** Circular path on a surface.
- **Cylinder:** Tracks across all surfaces at the same distance from the spindle.
- **Sector:** Division of a track, typically stores **4 kiB**.
- **Logical Block Addressing (LBA):** Specifies starting block number and count for read/write.
- Controller optimizations:
  - **Command queueing and out-of-order execution:** Up to **32 commands**.
  - **Write caching and coalescing:** Buffers writes in RAM.
  - **Intelligent seek optimization:** Minimizes head movement.
  - **Read-ahead caching:** Anticipates sequential reads.
  - Remaps failing sectors.
- Communication with host OS using **AHCI protocol**.

Solid-State Disks (SSD)

- Memory cells with floating gates store electrical charge (**2-10 years** retention).
- Information encoding: **2 levels (SLC - 1 bit), 4 levels (MLC - 2 bits), 8 levels (TLC - 3 bits), 16 levels (QLC - 4 bits)**.
- Trade-off: more levels = higher capacity, lower reliability (**reduced Program/Erase cycles**).
- **Page:** Cells of a word line (**4-8 KB**), smallest unit for read/write.
- **Block:** Group of **128-256 pages**, smallest unit for erase.
- **Plane:** Group of **1024 blocks**, independent I/O.
- **Die:** Piece of wafer with **1-4 planes**.
- **TSOP:** Circuit board with **1-8 dies**.
- Issues compared to magnetic disks:
  - **Coarse-grained erase:** Entire block must be erased before writing.
  - **Cell wear out:** Limited erase cycles (**1k-100k cycles**).
- Controller functions:
  - **Garbage collection:** Moves valid pages, erases full dirty blocks.
  - **Wear levelling:** Rotates data to wear cells uniformly.
  - **Over-provisioning:** Extra **10%** memory cells.
- **TRIM command:** OS informs SSD when pages are no longer used.
- States of a page: **free, used, dirty**.

Magnetic Disks vs Solid-State Disks

- **Magnetic Disk:**
  - Speed: Slower (head movements)
  - Lifespan: Age (**3-6 years**)
  - Cost: Cheaper (e.g., 4 TB CHF 80)
  - Fragility: Fragile (moving parts)
  - Secure Erase: By OS
- **Solid-State Disk (SSD):**
  - Speed: Faster (sequential **56x**, 4K **225x** in one test)
  - Lifespan: Overwrites (e.g., **600 TB written** for a 1 TB SSD)
  - Cost: More expensive (e.g., 4 TB CHF 200)
  - Fragility: Shock-resistant (no moving parts)
  - Secure Erase: **ATA Secure Erase command** by SSD

File Systems

- Layer on top of block storage, provides abstraction of files, directories, metadata, links.
- **Virtual File System (VFS):** Manages files and directories inside the kernel.
- Common file systems: **ext2, ext3 (deprecated), ext4 (standard Linux), Btrfs, ISO 9660, ReiserFS, XFS, ZFS, NTFS, FAT/V-FAT, HFS+, APFS**.
- **Data structures on disk (Ext2/3/4):**
  - **Block groups:** Reduce fragmentation.
  - **Superblock:** Basic file system info (magic number, block size - **1024-4096 bytes**).
  - **Group descriptor:** Block numbers of bitmaps.
  - **Directory:** Special file with file names and **Inode numbers**.
  - **Inode:** Metadata (permissions, size, times) and pointers to data blocks (direct, single, double, triple indirection).
- **Commands:**
  - **Disk Partitioning:**
    - `fdisk`: Basic interactive disk partitioning tool.
    - `sfdisk`: Scriptable disk partitioning tool.
    - `parted`: Powerful disk partitioning tool.
    - `gparted`: Graphical version of `parted`.
  - **File System Creation (Formatting):**
    - `mkfs`: Create a file system of a given type (e.g., `mkfs -t ext4 /dev/sda1`).
  - **Mounting and Unmounting:**
    - `mount`: Mount a file system (e.g., `mount /dev/sda1 /mnt`).
    - `umount`: Unmount a file system (e.g., `umount /mnt`).
  - **File System Information:**
    - `df`: Shows mounted file systems, their size, used and available space (e.g., `df -h`).
    - `findmnt`: Shows mounted file systems in a hierarchy, their type, and options (e.g., `findmnt /mnt`).
  - **File System Verification/Repair:**
    - `fsck`: Verifies and repairs a file system (must not be mounted) (e.g., `fsck /dev/sda1`).
  - **Link Creation:**
    - `ln <target> <link_name>`: Creates a hard link by default (e.g., `ln file_a file_b`).
    - `ln -s <target> <link_name>` or `ln --symbolic <target> <link_name>`: Creates a symbolic link (e.g., `ln -s /path/to/file file_link`).

File System Links

- **Hard Links:** New directory entry pointing to the same Inode. Limited to the same file system/partition.
- **Symbolic Links (symlinks):** Special file with its own Inode, containing the path to the original file. Can span file systems.
- `ln`: Create links (`ln file hard_link`, `ln -s file symbolic_link`).
- Properties:

File System Links Properties

Property	Hard Links	Symbolic Links
Different file system		T
Point to directory		T
Distinguish original and link		T
Point to another link		T
Original moved, link breaks		T
Original deleted/recreated, link breaks	T	

Virtual Storage

- Blocks replaced by pointers in an **index (table)** to physical blocks.
- Physical disks can be local (LVM) or remote (SAN).
- Motivation: Big capacity, dynamic resizing, flexibility, sharing.
- **Snapshots:** Read-only, near-instant copy of a virtual disk (only index copied initially).
- **Copy-on-write:** Before modifying a block in the active disk after a snapshot, a copy is made.

Logical Volume Management (LVM)

- Layer of indirection between block devices and file system.
- Linux: **LVM**, Windows: **LDM**.
- Features: Flexible capacity, resizeable pools, online data relocation, convenient naming, disk striping, mirroring, snapshots.
- Components: **Physical Volumes (PVs)**, **Volume Groups (VGs)**, **Logical Volumes (LVs)**.
- **Physical Extents (PEs)** and **Logical Extents (LEs):** Chunks of storage (default **4 MB**).
- Commands:
  - `pvscan`: Scan existing PVs (`pvscan`)
  - `vgscan`: Scan existing VGs (`vgscan`)
  - `lvscan`: Scan existing LVs (`lvscan`)
  - `pvcreate`: Create PV (`pvcreate /dev/sda1`)
  - `vgcreate`: Create VG (`vgcreate myvg /dev/sda1 /dev/sdb1`)
  - `lvcreate`: Create LV (`lvcreate -L 10G -n mylv myvg`)
  - `vgextend`: Extend VG (`vgextend myvg /dev/sdc1`)
  - `lvextend`: Extend LV (`lvextend -L +5G /dev/myvg/mylv`)
  - `lvreduce`: Reduce LV (`lvreduce -L -2G /dev/myvg/mylv`)
  - `vgremove`: Remove VG (`vgremove myvg`)
  - `lvremove`: Remove LV (`lvremove /dev/myvg/mylv`)
  - `vgchange -ay`: Activate VG (`vgchange -ay myvg`)
  - `vgchange -an`: Deactivate VG (`vgchange -an myvg`)
  - `lvdisplay`: Display LV info (`lvdisplay /dev/myvg/mylv`)
  - `vgdisplay`: Display VG info (`vgdisplay myvg`)
  - `pvdisplay`: Display PV info (`pvdisplay /dev/sda1`)

Networked Storage

- **Direct-Attached Storage (DAS):** Directly connected to a computer.
- **Network-Attached Storage (NAS):** File server providing file-level access over a network (e.g., NFS, SMB).
- **Storage Area Network (SAN):** High-performance network providing block-level access to servers (e.g., iSCSI, Fibre Channel).

Cloud Block Storage - AWS Elastic Block Store (EBS)

- Virtual disk (**EBS volume**) in AWS.
- Must be in the same **Availability Zone (AZ)** as the EC2 instance.
- One volume per instance.
- Replicated within an AZ for durability.
- **99.9% availability**, **99.999% annual durability**.
- **Snapshots:** Incremental backups to S3, replicated across AZs.
- EBS volume types:
  - **General Purpose SSD (gp2/gp3):** Most workloads.

  - **Provisioned IOPS SSD (io1/io2):** High-performance, consistent IOPS (up to **64'000 IOPS**).
  - **Throughput Optimized HDD (st1):** Big data, data warehouses (up to **500 MiB/s**).
  - **Cold HDD (sc1):** Infrequently accessed, lowest cost (up to **250 MiB/s**).
- **Maximum volume size: 16 TiB**.
- Snapshot pricing: ~**$50/TB-month**.

Object Storage - AWS Simple Storage Service (S3)

- Data as **objects** in flat containers called **buckets**.
- Unlimited capacity, supports very big objects (up to **5 TB** per object).
- Accessed via **RESTful API** (HTTP).
- No real directories, but key names can have prefixes (like paths).
- Operations: **CREATE, RETRIEVE, UPDATE (delete and create new), DELETE (CRUD)**.
- Buckets are globally unique, located in one **Region**, cannot be renamed.
- Objects have a key (unique within the bucket), can have user-defined metadata.
- Storage classes: **Standard, Intelligent-Tiering, Standard-Infrequent Access, One Zone-Infrequent Access, Glacier Instant Retrieval, Glacier Flexible Retrieval, Glacier Deep Archive**.
- Cost components: **Storage, Requests (PUT, GET, LIST, etc.), Data Transfer OUT**.
- S3 bucket URLs: **Path-style** (deprecated) and **virtual hosted-style**.

Cloud File Storage - Amazon Elastic File System (EFS)

- File storage in the AWS Cloud, shared, elastic.
- Petabyte-scale, low-latency.
- Supports **NFSv4**.
- Compatible with Linux EC2 instances.
- Mount targets in VPC subnets, one per AZ.

Serverless Computing

- Build and run applications without server management.
- Provider handles provisioning, maintenance, scaling.
- No compute cost when idle.
- **Function-as-a-Service (FaaS):** Code runs in stateless functions triggered by events.
- **AWS Lambda:** First public FaaS (**2014**).
- Lambda handler receives **event** and **context** objects.
- Deployment parameters: **Memory (128 MB - 3 GB), CPU, Timeout (default 3s, max 15 min), Concurrency, Environment variables**.
- Pricing: **Resources consumed (time x memory) + number of invocations**.
- Execution environment lifecycle: **Init, Invoke, Shutdown**. Environment can be reused for caching (**512 MB in /tmp**).
- **Cold start:** Delay when invoking a function for the first time or after inactivity. Mitigation: reduce dependencies, provisioned concurrency.
- **S3 Object Lambda:** Transform S3 data on the fly with a Lambda function before returning to application.
- **S3 Access Points:** Alias for bucket names with specific access policies.

Identity and Access Management (IAM)

- Securely control access to AWS resources.
- **Identity (Authentication):** Verify who the user is (userid/password, Access Key ID/Secret Key, token, MFA).
- **Access Management (Authorization):** Determine who can do what to which resource under which circumstances (ACLs, Permission Policies).
- **IAM User:** An individual or application.
- **IAM Group:** Collection of users with the same permissions.
- **IAM Role:** Identity that can be assumed by users, applications, or services, provides temporary credentials.
- **Permission Policy:** JSON document defining Allow or Deny rules for actions on resources.
- **Principal:** IAM entity making a request.
- **Action:** Operation to be performed (e.g., s3:GetObject, ec2:StartInstances).
- **Resource:** AWS resource to act upon (identified by **ARN - Amazon Resource Name**).
- **Effect:** Allow or Deny. **Explicit Deny** overrides any Allows.
- Policy evaluation: Implicit Deny by default if no matching Allow policy. Deny overrides Allow.
- **Identity-based policies:** Attached to IAM users, groups, or roles.
- **Resource-based policies:** Attached to a resource (e.g., S3 bucket policy).

Key Points

- **MLC SSDs:** Lower durability, lower cost, slower performance than SLC SSDs.
- **SSD Recommendation:** Extreme temperatures/humidity, quick recovery (high read throughput).
- **Magnetic Disk Recommendation:** Forensic data recovery (straightforward block allocation).
- **Ext2/3/4 Data Storage:** File name in directory, metadata in inode, content in blocks.
- **Symbolic Link Behavior:** Broken if original removed, reflects content of recreated original.
- **LVM for Databases:** Flexible resizing, snapshots for backups with minimal downtime.
- **EBS Snapshots:** Incremental, stored on S3, deletion can affect other snapshots.
- **Data Sharing:** S3 easiest for multiple users (REST), EFS for concurrent access control (network file system), EBS requires OS-level configuration.
- **MySQL on Cloud:** EBS better fit (block storage for file system), S3 inefficient for frequent small writes.
- **S3 Cost Components (Photo Storage):** Storage, Access (typically negligible), Transfer OUT (substantial).
- **IAM Policy Elements (IF-THEN):** IF: Resource, Principal, Action; THEN: Effect.
- **IAM Policy Properties:** Define permissions regardless of method, applied to users/groups/roles, no priorities, only matching policies evaluated.
- **Data Ingestion (S3 to Data Lake):** Lambda more cost-effective for few stores, EC2 instance might be better for many stores with longer download times.
- **Data Lakehouse Recommendation:** Suitable for structured (POS, inventory) and unstructured (customer feedback) data needing different types of analysis (SQL, transformations).
- **Lack of Data Catalog:** Impacts application by requiring hardcoding or discovery of data location, format, schema, partitioning.