# Storage Area Networks (18CS822)

SEMESTER – VIII

## Module I

**Storage System: Introduction to Information Storage:** Information Storage, Evolution of Storage Architecture, Data Center Infrastructure, Virtualization and Cloud Computing.
**Data Center Environment:** Application Database Management System (DBMS), Host (Compute), Connectivity, Storage, Disk Drive Components, Disk Drive Performance, Host Access to Data, Direct-Attached Storage, Storage Design Based on Application

# STORAGE SYSTEM

## Introduction to Information Storage

### Why Information management?

- Information is increasingly important in our daily lives. We have become information dependents.
- We live in on-command, on-demand world that means we need information when and where it is required.
- We access the Internet every day to perform searches, participate in social networking, send and receive e-mails, share pictures and videos, and scores of other applications. Equipped with a growing number of content-generating devices, more information is being created by individuals than by businesses.
- The importance, dependency, and volume of information for the business world also continue to grow at astounding rates.
- Businesses depend on fast and reliable access to information critical to their success. Some of the business applications that process information include airline reservations, telephone billing systems, e-commerce, ATMs, product designs, inventory management, e-mail archives, Web portals, patient records, credit cards, life sciences, and global capital markets.
- The increasing criticality of information to the businesses has amplified the challenges in protecting and managing the data.
- Organizations maintain one or more data centers to store and manage information. **A data center** is a facility that contains information storage and other physical information technology (IT) resources for computing, networking, and storing information.

### A. Information Storage

Business organizations use/process data to derive information that is critical to their day-to-day operations. Storage is a repository that enables users to store and retrieve this digital data.

### 1. Data

- Data is a collection of raw facts from which conclusions may be drawn.
- Eg: Handwritten letter, a printed book, a family photograph, a movie on videotape, e-mail message, an e-book, a bitmapped image, or a digital movie are all examples of data.

The data can be generated using a computer and stored in strings of 0s and1s (as shown in Fig 1.1), is called digital data and is accessible by the user only after it is processed by a computer.

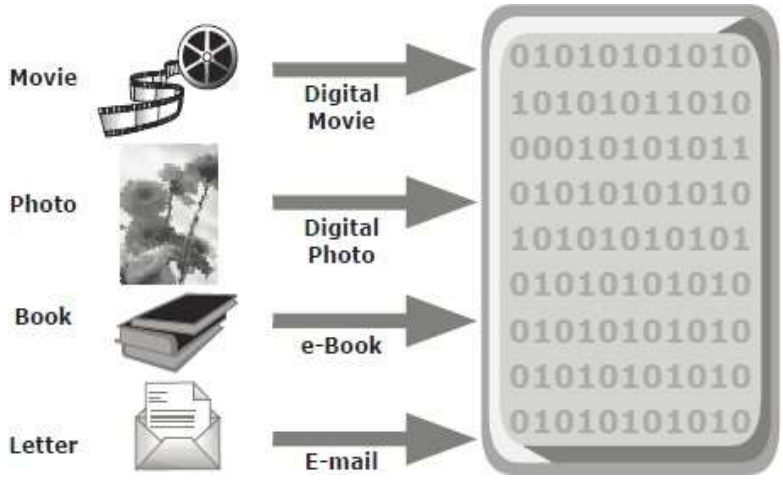



**Fig 1.1:** Digital data

The following is a list of some of the factors that have contributed to the growth of digital data:

1. **Increase in data processing capabilities**: Modern-day computers provide a significant increase in processing and storage capabilities. This enables the conversion of various types of content and media from conventional forms to digital formats.

2. **Lower cost of digital storage:** Technological advances and decrease in the cost of storage devices have provided low-cost solutions and encouraged the development of less expensive data storage devices. This cost benefit has increased the rate at which data is being generated and stored.

3. **Affordable and faster communication technology:** The rate of sharing digital data is now much faster than traditional approaches. A handwritten letter may take a week to reach its destination, whereas it only takes a few seconds for an e-mail message to reach its recipient.

4. **Proliferation of applications and smart devices:** Smart phones, tablets, and newer digital devices, along with smart applications, have significantly contributed to the generation of digital content.

## 2. Types of Data

Data can be classified as structured or unstructured (see Fig1.2) based on how it is stored and managed.

- **Structured data:**

- Structured data is organized in rows and columns in a rigidly defined format so that applications can retrieve and process it efficiently.
- Structured data is typically stored using a database management system (DBMS).

➢ **Unstructured data:**

- Data is unstructured if its elements cannot be stored in rows and columns, and is therefore difficult to query and retrieve by business applications.
- Example: e-mail messages, business cards, or even digital format files such as .doc,.txt, and .pdf.

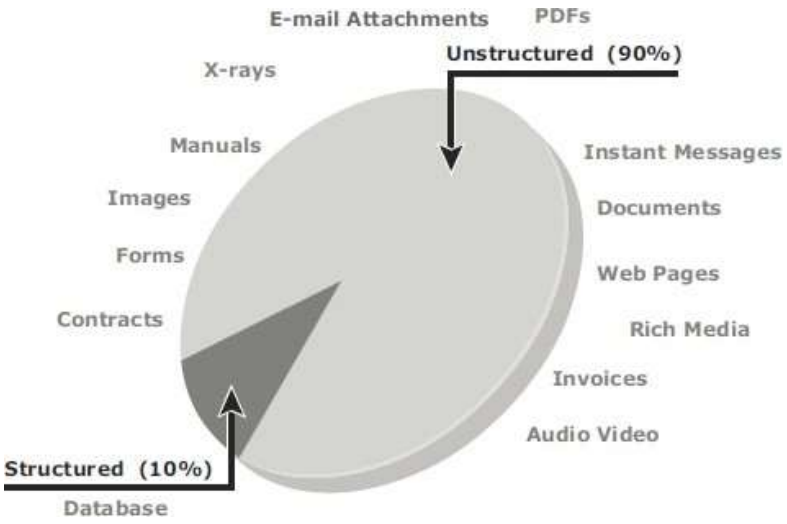



**Fig 1.2: Types of data**

**3. Big Data**

➢ Big data refers to data sets whose sizes are beyond the capability of commonly used software tools to capture, store, manage, and process within acceptable time limits.

➢ It includes both structured and unstructured data generated by a variety of sources, including business application transactions, web pages, videos, images, emails, social media, and so on.

➢ The big data ecosystem (see Fig 1.3) consists of the following:

1. Devices that collect data from multiple locations and also generate new data about this data (metadata).
2. Data collectors who gather data from devices and users.
3. Data aggregators that compile the collected data to extract meaningful information.
4. Data users and buyers who benefit from the information collected and aggregated by others in the data value chain.

➢ Big data Analysis in real time requires new techniques, architectures, and tools that provide:

1. High performance,
2. Massively parallel processing (MPP) data platforms,
3. Advanced analytics on the datasets.

➢ Big data Analytics provide an opportunity to translate large volumes of data into right decisions.

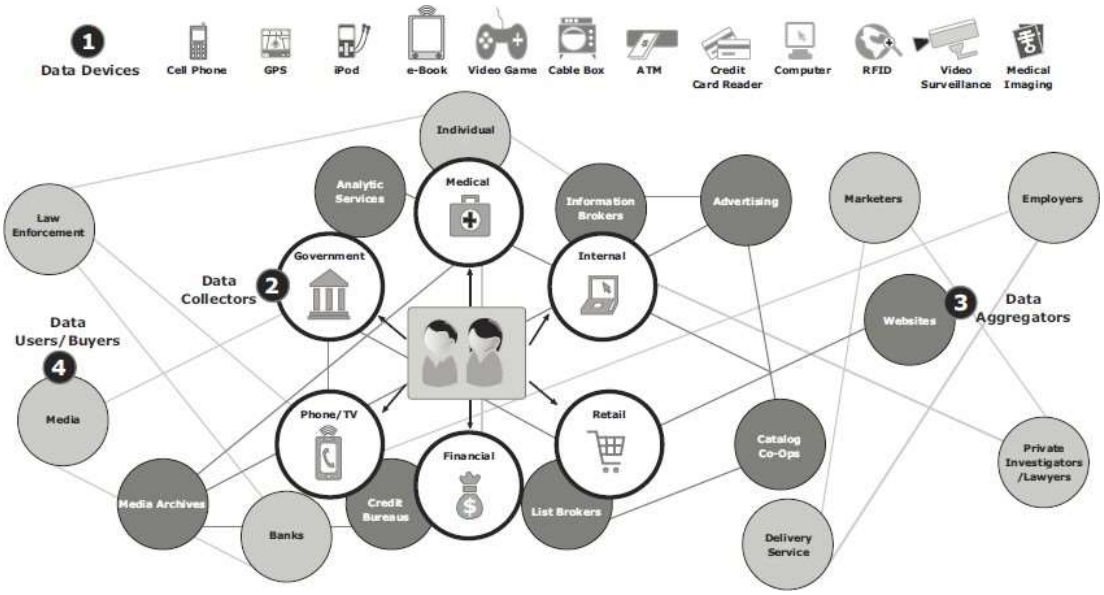



**Fig 1.3: Big data Ecosystem**

## 4. Information

- Data, whether structured or unstructured, does not fulfill any purpose for individuals or businesses unless it is presented in a meaningful form.
- Information is the intelligence and knowledge derived from data.
- Businesses analyze raw data in order to identify meaningful trends. On the basis of these trends, a company can plan or modify its strategy.
- For example, a retailer identifies customers' preferred products and brand names by analyzing their purchase patterns and maintaining an inventory of those products.
- Because information is critical to the success of a business, there is an ever present concern about its availability and protection.

## 5. Storage

- Data created by individuals or businesses must be stored so that it is easily accessible for further processing.
- In a computing environment, devices designed for storing data are termed storage devices or simply storage.
- The type of storage used varies based on the type of data and the rate at which it is created and used.
- Devices such as memory in a cell phone or digital camera, DVDs, CD-ROMs, and hard disks in personal computers are examples of storage devices.
- Businesses have several options available for storing data including internal hard disks, external disk arrays and tapes.

## B. Evolution of Storage Architecture

- Historically, organizations had centralized computers (mainframe) and information storage devices (tape reels and disk packs) in their data center.
- The evolution of open systems and the affordability and ease of deployment that they offer made it possible for business units/departments to have their own servers and storage.
- In earlier implementations of open systems, the storage was typically internal to the server. This approach is referred to as **server-centric storage architecture** (see Fig 1.4 [a]).
- In this server-centric storage architecture, each server has a limited number of storage devices, and any administrative tasks, such as maintenance of the server or increasing storage capacity, might result in unavailability of information.
- The rapid increase in the number of departmental servers in an enterprise resulted in unprotected, unmanaged, fragmented islands of information and increased capital and operating expenses.
- To overcome these challenges, storage evolved from **server-centric to information-centric architecture** (see Fig 1.4 [b]).

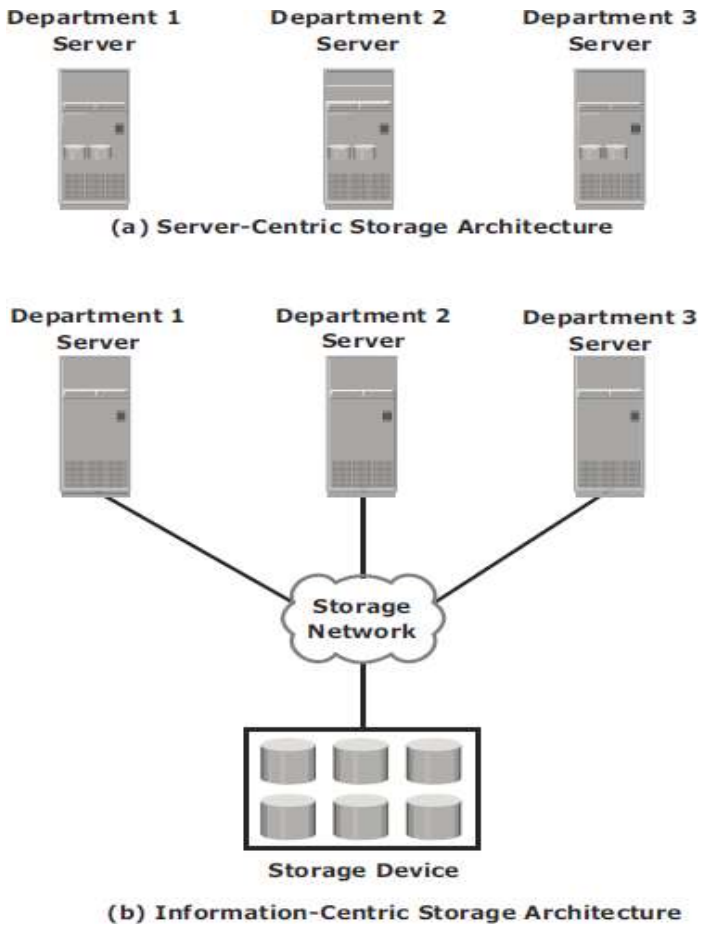



**Fig 1.4: Evolution of storage architecture**

- In information-centric architecture, storage devices are managed centrally and independent of servers.
- These centrally-managed storage devices are shared with multiple servers.
- When a new server is deployed in the environment, storage is assigned from the same shared storage devices to that server.
- The capacity of shared storage can be increased dynamically by adding more storage devices without impacting information availability.
- In this architecture, information management is easier and cost-effective.
- Storage technology and architecture continues to evolve, which enables organizations to consolidate, protect, optimize, and leverage their data to achieve the highest return on information assets.

## C. Data Center Infrastructure

- Organizations maintain data centers to provide centralized data processing capabilities across the enterprise.
- The data center infrastructure includes computers, storage systems, network devices, dedicated power backups, and environmental controls (such as air conditioning and fire suppression).

### I. Key Data Center Elements [Also known as Core elements of a data center]

Five core elements are essential for the basic functionality of a data center:

1)**Application:** An application is a computer program that provides the logic for computing operations. Eg: order processing system.

2)**Database:** More commonly, a database management system (DBMS) provides a structured way to store data in logically organized tables that are interrelated. A DBMS optimizes the storage and retrieval of data.

**3)Host or compute:** A computing platform (hardware, firmware, and software) that runs applications and databases**.**

4)**Network:** A data path that facilitates communication among various networked devices.

5)**Storage array:** A device that stores data persistently for subsequent use.

- These core elements are typically viewed and managed as separate entities, but all the elements must work together to address data processing requirements.
- Fig 1.5 shows an example of an order processing system that involves the five core elements of a data center and illustrates their functionality in a business process.

1) A customer places an order through a client machine connected over a LAN/ WAN to a host running an order-processing application.

2) The client accesses the DBMS on the host through the application to provide order- related information, such as the customer name, address, payment method, products ordered, and quantity ordered.

3) The DBMS uses the host operating system to write this data to the database located on physical disks in the storage array.

4) The Storage Network provides the communication link between the host and the storage array and transports the request to read or write commands between them.

5) The storage array, after receiving the read or write request from the host, performs the necessary operations to store the data on physical disks.

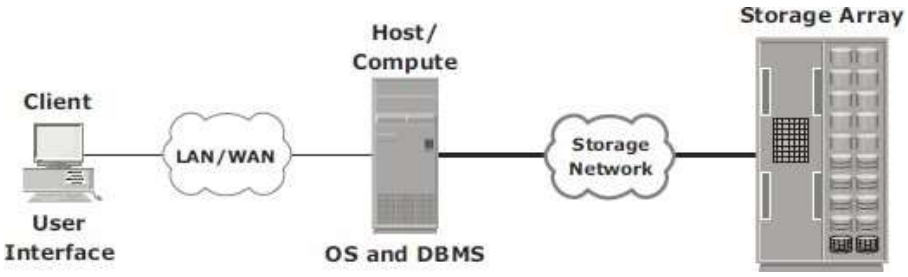



**Fig 1.5: Example of an online order transaction system**

## II. Key characteristics for Data Center Elements

Key characteristics of data center elements are:

1)**Availability:** All data center elements should be designed to ensure accessibility. The inability of users to access data can have a significant negative impact on a business.

2)**Security:** Polices, procedures, and proper integration of the data center core elements that will prevent unauthorized access to information must be established. Specific mechanisms must enable servers to access only their allocated resources on storage arrays.

3)**Scalability:** Data center operations should be able to allocate additional processing capabilities (eg: servers, new applications, and additional databases) or storage on demand, without interrupting business operations. The storage solution should be able to grow with the business.

4)**Performance**: All the core elements of the data center should be able to provide optimal performance and service all processing requests at high speed. The infrastructure should be able to support performance requirements.

5)**Data integrity:** Data integrity refers to mechanisms such as error correction codes or parity bits which ensure that data is written to disk exactly as it was received. Any variation in data during its retrieval implies corruption, which may affect the operations of the organization.

6)**Capacity:** Data center operations require adequate resources to store and process large amounts of data efficiently. When capacity requirements increase, the data center must be able to provide additional capacity without interrupting availability, or, at the very least, with minimal disruption. Capacity may be managed by reallocation of existing resources, rather than by adding new resources.

7) **Manageability:** A data center should perform all operations and activities in the most efficient manner. Manageability can be achieved through automation and the reduction of human (manual) intervention in common tasks.

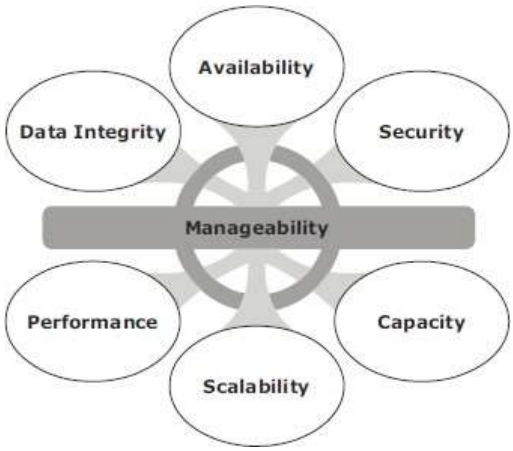



**Fig1.6: Key characteristics of data center elements**

**III. Managing a data center:** It involves many tasks. Namely –

1. Monitoring: it is a continuous process of gathering information on various elements and services running in a data center. The aspects of a data that are monitored include security, performance, availability, and capacity.
2. Reporting: it is done periodically on resource performance, capacity, and utilization. Reporting tasks help to establish business justifications and chargeback of costs associated with data center operations.
3. Provisioning: it is a process of providing the hardware, software, and other resources required to run a data center. Provisioning activities primarily include resources management to meet capacity, availability, performance, and security requirements.

**IV. Virtualization**

- Virtualization is a technique of abstracting physical resources, such as compute, storage, and network, and making them appear as logical resources.
- Virtualization has existed in the IT industry for several years and in different forms.
- Common examples of virtualization are virtual memory used on computer systems and partitioning of raw disks.
- Virtualization enables pooling of physical resources and providing an aggregated view of the physical resource capabilities. For example, storage virtualization enables multiple pooled storage devices to appear as a single large storage entity.
- Similarly, by using compute virtualization, the CPU capacity of the pooled physical servers can be viewed as the aggregation of the power of all CPUs (in megahertz).
- Virtualization also enables centralized management of pooled resources.

- Virtual resources can be created and provisioned from the pooled physical resources. For example, a virtual disk of a given capacity can be created from a storage pool or a virtual server with specific CPU power and memory can be configured from a compute pool.
- These virtual resources share pooled physical resources, which improves the utilization of physical IT resources.
- Based on business requirements, capacity can be added to or removed from the virtual resources without any disruption to applications or users.
- With improved utilization of IT assets, organizations save the costs associated management of new physical resources. Moreover, fewer physical resources means less space and energy, which leads to better economics and green computing.

### V. Cloud Computing

- Cloud computing enables individuals or businesses to use IT resources as a service over the network. It provides highly scalable and flexible computing that enables provisioning of resources on demand. Users can scale up or scale down the demand of computing resources, including storage capacity, with minimal management effort or service provider interaction.
- Cloud computing empowers self-service requesting through a fully automated request- fulfillment process.
- Cloud computing enables consumption-based metering; therefore, consumers pay only for the resources they use, such as CPU hours used, amount of data transferred, and gigabytes of data stored. Cloud infrastructure is usually built upon virtualized data centers, which provide resource pooling and rapid provisioning of resources.

## Data Center Environment

### Key Data center Elements

1. **Application:** Today data centers are essential and integrated parts of many businesses [small/medium/large].

- An application is a computer program that provides the logic for computing operations.
- The application sends requests to the underlying operating system to perform read/write (R/W) operations on the storage devices. Applications can be layered on the data base, which in turn uses OS services to perform R/W operations on the storage devices.
- Applications deployed in a data center environment are commonly categorized as business applications, infrastructure management applications, data protection applications, and security applications.
- Some examples of these applications are e-mail, enterprise resource planning (ERP), decision support system (DSS), resource management, backup, authentication and antivirus applications, and so on.

## 2. DBMS

A database is a structured way to store data in logically organized tables that are interrelated. A database helps to optimize the storage and retrieval of data. A DBMS controls the creation, maintenance, and use of a database. The DBMS processes an application's requests for data and instruct the OS to transfer the appropriate data from the storage.

## 3. Host (or) Compute

The users store and retrieve data through applications. The computers on which applications run are referred to as hosts. Hosts can range from simple desktops, laptops, mobile devices to servers, complex clusters of servers.

Hosts can be physical or virtual machines.

Compute virtualization software enables creating virtual machines on top of a physical compute infrastructure.

A host consists of -

✓ CPU: The CPU consists of four components-Arithmetic Logic Unit (ALU), control unit, registers, and L1 cache

✓ Memory: There are two types of memory on a host, Random Access Memory (RAM) and Read-Only Memory (ROM)

✓ I/O devices: keyboard, mouse, monitor

✓ a collection of software to perform computing operations- This software includes the operating system, file system, logical volume manager, device drivers, and soon.

The following section details various software components that are essential parts of a host system.

### a) Operating System

In a traditional computing environment, an operating system controls all aspects of computing.

It works between the application and the physical components of a compute system.

In a virtualized compute environment, the virtualization layer works between the operating system and the hardware resources.

**Functions of OS**

data access

monitors and responds to user actions and the environment

organizes and controls hardware components

manages the allocation of hardware resources

provides basic security for the access and usage of all managed resources performs basic storage management tasks

manages the file system, volume manager, and device drivers.

**Memory Virtualization**

- Memory has been, and continues to be, an expensive component of a host.
- It determines both the size and number of applications that can run on a host.
- Memory virtualization is an operating system feature that virtualizes the physical memory (RAM) of a host.
- It creates virtual memory with an address space larger than the physical memory space present in the compute system.
- The operating system utility that manages the virtual memory is known as the virtual memory manager (VMM).
- The space used by the VMM on the disk is known as a swap space.
- A swap space (also known as page file or swap file) is a portion of the disk drive that appears to be physical memory to the operating system.
- In a virtual memory implementation, the memory of a system is divided into contiguous blocks of fixed-size pages.
- A process known as paging moves inactive physical memory pages on to the swap file and brings them back to the physical memory when required.

**b) Device Drivers**

- A device driver is special software that permits the operating system to interact with a specific device, such as a printer, a mouse, or a disk drive.

**c) Volume Manager**

- In the early days, disk drives appeared to the operating system as a number of continuous disk blocks. The entire disk drive would be allocated to the file system or other data entity used by the operating system or application.

**Disadvantages:**

- ✓ Lack of flexibility.
- ✓ When a disk drive ran out of space, there was no easy way to extend the file system's size.
- ✓ As the storage capacity of the disk drive increased, allocating the entire disk drive for the file system often resulted in underutilization of storage capacity

**Solution:** evolution of Logical Volume Managers (LVMs)

- LVM enabled dynamic extension of file system capacity and efficient storage management.
- The LVM is software that runs on the compute system and manages logical and physical
- storage. LVM is an intermediate layer between the file system and the physical disk.
- LVM can partition a larger-capacity disk into virtual, smaller-capacity volumes (called Partitioning) or aggregate several smaller disks to form a larger virtual volume. The process is called concatenation.

- Disk partitioning was introduced to improve the flexibility and utilization of disk drives.
- In partitioning, a disk drive is divided into logical containers called logical volumes (LVs) (see Fig 1.7)

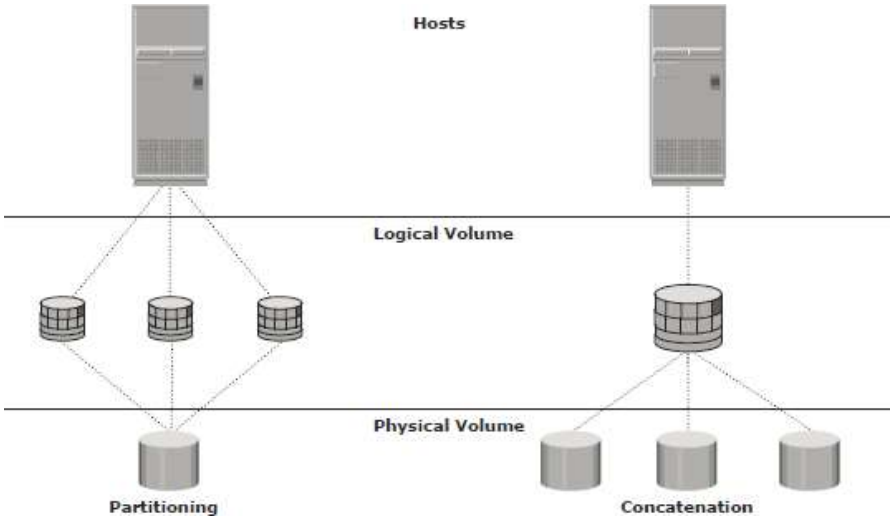



**Fig 1.7: Disk Partitioning and concatenation**

- Concatenation is the process of grouping several physical drives and presenting them to the host as one big logical volume.
- The basic LVM components are **physical volumes, volume groups, and logical**
- **volumes**. Each physical disk connected to the host system is a **physical volume (PV).**
- A **volume group** is created by grouping together one or more physical volumes. A unique physical volume identifier (PVID) is assigned to each physical volume when it is initialized for use by the LVM. Each physical volume is partitioned in to equal-sized data blocks called **physical extents** when the volume group is created.
- **Logical volumes** are created within a given volume group. A logical volume can be thought of as a disk partition, whereas the volume group itself can be thought of as a disk.

**d) File System**

- A file is a **collection of related records** or data stored as a unit with a name. A file system is a
- hierarchical structure of files.
- A file system enables easy access to data files residing within a diskdrive, a disk partition, or a logical volume.
- It provides users with the functionality to create, modify, delete, and access files.
- Access to files on the disks is controlled by the permissions assigned to the file by the owner, which are also maintained by the file system.
- A file system organizes data in a structured hierarchical manner via the use of directories, which are containers for storing pointers to multiple files.
- All file systems maintain a pointer map to the directories, sub directories, and files that are

part of the file system.

- Examples of common file systems are:
  - ✓ FAT32 (File Allocation Table) for Microsoft Windows
  - ✓ NT File System(NTFS) for Microsoft Windows - NT
  - ✓ UNIX File System(UFS) for UNIX
  - ✓ Extended File System(EXT2/3) for Linux
- The file system also includes a number of other related records, which are collectively called the **metadata**.
- For example, the metadata in a UNIX environment consists of the **superblock, the inodes, and the list of data blocks free and in use.**
- A superblock contains important information about the file system, such as the file system type, creation and modification dates, size, and layout.
- An inode is associated with every file and directory and contains information such as the file length, ownership, access privileges, time of last access/modification, number of links, and the address of the data.
- A file system block is the smallest "unit" allocated for storing data.
- The following list shows the process of mapping user files to the disk storage subsystem with an LVM (see Fig 1.8)

1. Files are created and managed by users and applications.
2. These files reside in the file systems.
3. The file systems are mapped to file system blocks.
4. The file system blocks are mapped to logical extents of a logical volume.
5. These logical extents in turn are mapped to the disk physical extents either by the operating system or by the LVM.
6. These physical extents are mapped to the disk sectors in a storage subsystem. If there is no LVM, then there are no logical extents. Without LVM, file system blocks are directly mapped to disk sectors.

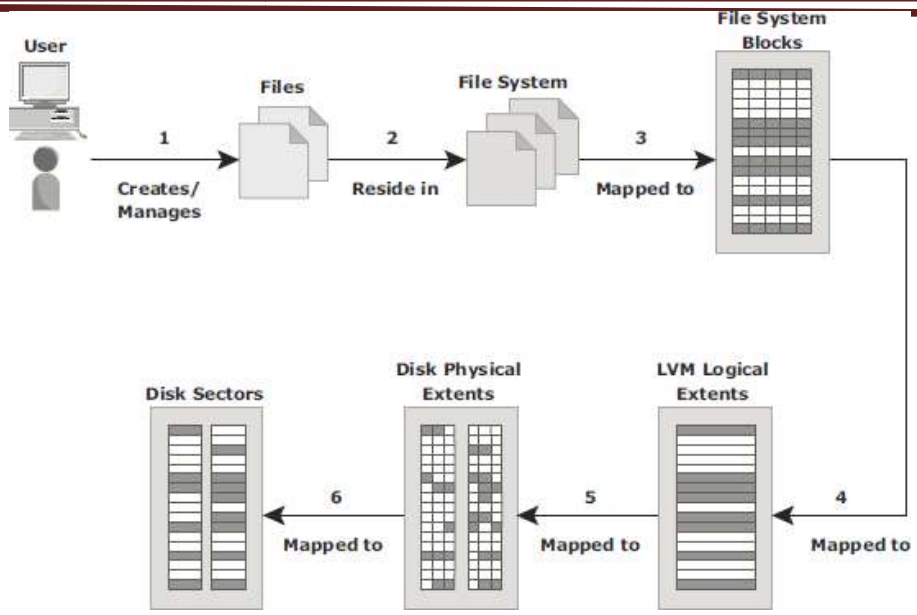



**Fig1.8: Process of mapping user files to disk storage**

- The file system tree starts with the root directory. The root directory has a number of
- subdirectories. A file system can be either:
  - ✓ A journaling file system
  - ✓ A nonjournaling file system

**Nonjournaling file system:** Nonjournaling file systems cause a potential loss of files because they use separate writes to update their data and metadata. If the system crashes during the write process, the metadata or data might be lost or corrupted. When the system reboots, the file system attempts to update the metadata structures by examining and repairing them. This operation takes a long time on large file systems. If there is insufficient information to re-create the wanted or original structure, the files might be misplaced or lost, resulting in corrupted file systems.

**Journaling file system: Journaling File System** uses a separate area called a *log* or *journal*. This journal might contain all the data to be written (physical journal) or just the metadata to be updated (logical journal). Before changes are made to the file system, they are written to this separate area. After the journal has been updated, the operation on the file system can be performed. If the system crashes during the operation, there is enough information in the log to "*replay*" the log record and complete the operation. Nearly all file system implementations today use journaling

Advantages:

- ➢ Journaling results in a quick file system check because it looks only at the active, most recently accessed parts of a large file system.
- ➢ Since information about the pending operation is saved, the risk of files being lost is reduced.

Disadvantage:

- They are slower than other file systems. This slow down is the result of the extra operations that have to be performed on the journal each time the file system is changed.
- But the advantages of lesser time for file system checks and maintaining file system integrity far outweighs its disadvantage.

**e) Compute Virtualization**

- Compute virtualization is a technique for masking or abstracting the physical hardware from the operating system. It enables multiple operating systems to run concurrently on single or clustered physical machines.
- This technique enables creating portable virtual compute systems called *virtual machines* (VMs) running its own operating system and application instance in an isolated manner.
- Compute virtualization is achieved by a virtualization layer that resides between the hardware and virtual machines called the *hypervisor*. The hypervisor provides hardware resources, such as CPU, memory, and network to all the virtual machines.
- A virtual machine is a logical entity but appears like a physical host to the operating system, with its own CPU, memory, network controller, and disks. However, all VMs share the same underlying physical hardware in an isolated manner.
- Before Compute virtualization:
  - ✓ A physical server often faces resource-conflict issues when two or more applications running on the same server have conflicting requirements. As a result, only one application can be run on a server at a time, as shown in Fig 1.9 (a).
  - ✓ Due to this, organizations will need to purchase new physical machines for every application they deploy, resulting in expensive and inflexible infrastructure.

✓ Many applications do not fully utilize complete hardware capabilities available to them. Resources such as processors, memory and storage remain underutilized.

✓ Compute virtualization enables users to overcome these challenges (see Fig1.9(b)).

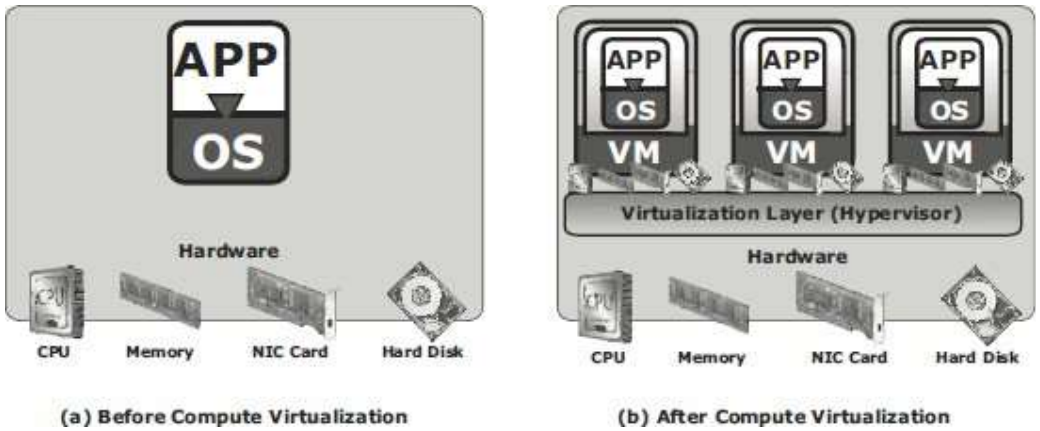



**Fig 1.9: Server Virtualization**

- After Compute virtualization:
  - ✓ This technique significantly improves server utilization and provides server consolidation.
  - ✓ *Server consolidation* enables organizations to run their data center with fewer physical servers.
  - ✓ This, in turn,
    - reduces cost of new server acquisition,
    - reduces operational cost,
    - saves data center floor and rack space.
  - ✓ Individual VMs can be restarted, upgraded, or even crashed, without affecting the other VMs.
  - ✓ VMs can be copied or moved from one physical machine to another (non-disruptive migration) without causing application downtime. This is required for maintenance activities

## 4. Connectivity

- Connectivity refers to the interconnection between hosts or between a host and peripheral devices, such as printers or storage devices.
- Connectivity and communication between host and storage are enabled using:
  - ✓ Physical components
  - ✓ Interface protocols.

### a) Physical Components of Connectivity

- The physical components of connectivity are the hardware elements that connect the host to storage. our focus is on connectivity between host and storage device.
- Three physical components of connectivity between the host and storage are (refer Fig1.10):
  - ✓ The host interface device
  - ✓ port
  - ✓ cable [Copper/fibre optic]

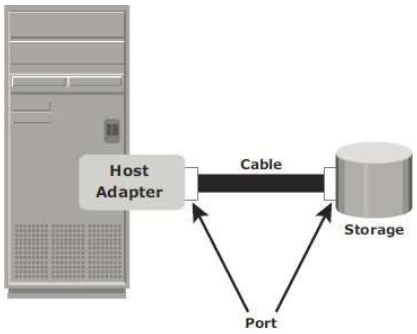



**Fig 1.10: Physical components of connectivity**

- A *host interface device* or *host adapter* connects a host to other hosts and storage devices.
  - ✓ Eg: host bus adapter (HBA) and network interface card(NIC).
  - ✓ HBA is an application-specific integrated circuit (ASIC) board that performs I/O interface

functions between the host and storage, relieving the CPU from additional I/O processing workload.

- ✓ A host typically contains multiple HBAs.

- A *port* is a specialized outlet that enables connectivity between the host and external devices. An HBA may contain one or more ports to connect the host.
- *Cables* connect hosts to internal or external devices using copper or fiber optic media.

**b) Interface Protocols**

- A protocol enables communication between the host and storage.
- Protocols are implemented using interface devices (or controllers) at both source and
- destination. The popular interface protocols used for host to storage communications are:

  i. Integrated Device Electronics/Advanced Technology Attachment (IDE/ATA)
  ii. Small Computer System Interface (SCSI),
  iii. Fibre Channel (FC)
  iv. Internet Protocol (IP)

**i. IDE/ATA and Serial ATA:**

- **IDE/ATA** is a popular interface protocol standard used for connecting storage devices, such as disk drives and CD-ROM drives.
- This protocol supports parallel transmission and therefore is also known as *Parallel ATA (PATA*) or simply ATA.
- IDE/ATA has a variety of standards and names.
- The Ultra DMA/133 version of ATA supports a throughput of **133MB per second**.
- In a master-slave configuration, an ATA interface supports two storage devices per connector.
- If performance of the drive is important, sharing a port between two devices is not
- recommended. The serial version of this protocol is known as Serial ATA (SATA) and supports single bit serial transmission.
- *High performance* and *low cost* SATA has replaced PATA in newer systems.

SATA revision 3.0 provides a data transfer rate up to **6 Gb/s.**

**ii. SCSI and Serial SCSI:**

- **SCSI** has emerged as a preferred connectivity protocol in high-end computers.
- This protocol supports parallel transmission and offers improved **performance**, **scalability,** And **compatibility** compared to ATA.
- The high cost associated with SCSI limits its popularity among home or personal desktop
- users. SCSI supports up to16 devices on a single bus and provides data transfer rates up to
- **640 MB/s. Serial attached SCSI (SAS)** is a point-to-point serial protocol that provides an

alternative to parallel SCSI.

- A newer version of serial SCSI (SAS2.0) supports a data transfer rate up to **6Gb/s.**

**iii. Fibre Channel (FC):**

- **Fibre Channel** is a widely used protocol for high-speed communication to the storage device. Fibre Channel interface provides gigabit network speed.
- It provides a serial data transmission that operates over copper wire and optical fiber. The latest version of the FC interface (16FC) allows transmission of data up to **16Gb/s.**

**iv. Internet Protocol (IP):**

- IP is a network protocol that has been traditionally used for **host-to-host traffic**.
- With the emergence of new technologies, an IP network has become a viable option for host-to- storage communication.
- IP offers several advantages:
  - ✓ cost
  - ✓ maturity
  - ✓ enables organizations to leverage their existing IP-based network.
- **iSCSI** and **FCIP** protocols are common examples that leverage IP for host-to-storage communication.

**5. Storage**

- Storage is a core component in a data center.
- A storage device uses magnetic, optic, or solid state
- media. Disks, tapes, and diskettes use magnetic media,
- CD/DVD uses optical media.

Removable Flash memory or Flash drive uses solid state media.

**a) Tapes**

- In the past, **tapes** were the most popular storage option for backups because of their low
- cost. Tapes have various limitations in terms of performance and management, as listed below:

i. Data is stored on the tape linearly along the length of the tape. Search and retrieval of data are done sequentially, and it invariably takes several seconds to access the data. As a result, **random data access is slow and time-consuming**.

ii. In a shared computing environment, data stored on tape **cannot be accessed by multiple applications simultaneously**, restricting its use to one application at a time.

iii. On at a pedrive, the read/write head touches the tape surface, so the tape degrades or wears out after repeated use.

iv. The storage and retrieval requirements of data from the tape and the overhead

associated with managing the tape media are significant.

- Due to these limitations and availability of low-cost disk drives, tapes are no longer a preferred choice as a backup destination for enterprise-class data centers.

**b) Optical Disc Storage:**

- It is popular in small, single-user computing environments.
- It is frequently used by individuals to store photos or as a backup medium on personal or laptop computers.
- It is also used as a distribution medium for small applications, such as games, or as a means to transfer small amounts of data from one computer system to another.
- The capability to **write once and read many (WORM)** is one advantage of optical disc storage. Eg: CD-ROM
- Collections of optical discs in an array, called a **jukebox**, are still used as a fixed-content storage solution.
- Other forms of optical discs include CD-RW, Blu-ray disc, and other variations of DVD.

**c) Disk Drives:**

- **Disk drives** are the most popular storage medium used in modern computers for storing and accessing data for performance-intensive, online applications.
- Disks support rapid access to random data locations. Disks have large capacity. Disk storage
- arrays are configured with multiple disks to provide increased capacity and enhanced performance.
- Disk drives are accessed through predefined protocols, such as ATA, SATA, SAS, and FC.
- These protocols are implemented on the disk interface controllers.
- Disk interface controllers were earlier implemented as separate cards, which were connected to the motherboard.
- Modern disk interface controllers are integrated with the disk drives; therefore, disk drives are known by the protocol interface they support, for example SATA disk, FC disk, etc.

**6. Disk Drive Components**

The key components of a hard disk drive are platter, spindle, read-write head, actuator arm assembly, and controller board.

I/O operations in a HDD are performed by rapidly moving the arm across the rotating flat platters coated with magnetic particles. Data is transferred between the disk controller and magnetic platters through the read-write (R/W) head which is attached to the arm. Data can be recorded and erased on magnetic platters any number of times. Following sections detail the different components of the disk drive, the mechanism for organizing and storing data on disks, and the factors that affect disk performance.

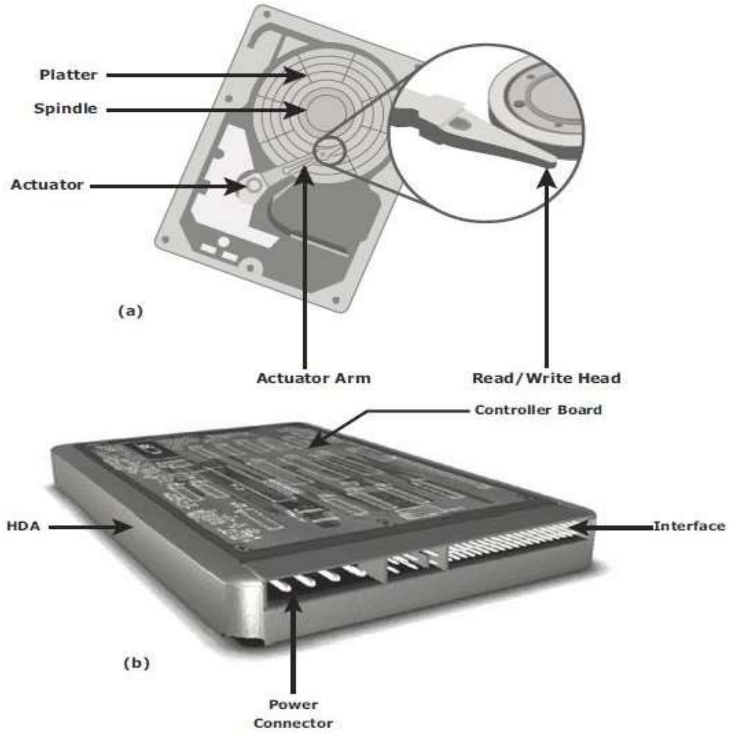



**Fig 1.11: Disk drive components**

**Platter**

A typical HDD consists of one or more flat circular disks called platters. The data is recorded on these platters in binary codes (0s and 1s). The set of rotating platters is sealed in a case, called the Head Disk Assembly (HDA). A platter is a rigid, round disk coated with magnetic material on both surfaces (top and bottom). The data is encoded by polarizing the magnetic area, or domains, of the disk surface. Data can be written to or read from both surfaces of the platter. The number of platters and the storage capacity of each platter determine the total capacity of the drive.

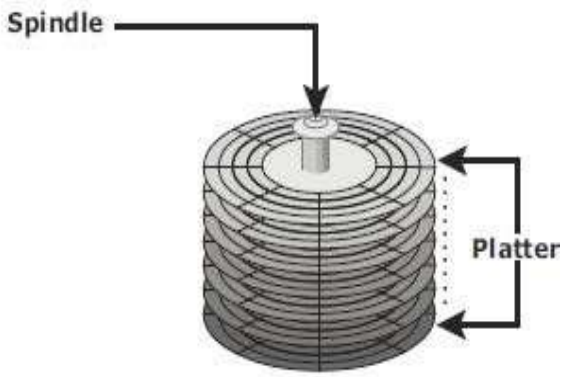



**Fig 1.12: Spindle and platter**

**Spindle**

A spindle connects all the platters and is connected to a motor. The motor of the spindle rotates with a constant speed. The disk platter spins at a speed of several thousands of revolutions per minute (rpm). Common spindle speeds are 5,400 rpm, 7,200 rpm, 10,000 rpm, and15,000 rpm. The speed of the platter is increasing with improvements in technology, although the extent to which it can be improved is limited.

**Read/Write Head**

Read/Write(R/W) heads, as shown in Figure below, read and write data from or to platters. Drives have two R/W heads per platter, one for each surface of the platter. The R/W head changes the magnetic polarization on the surface of the platter when writing data. While reading data, the head detects the magnetic polarization on the surface of the platter. During reads and writes, the R/W head senses the magnetic polarization and never touches the surface of the platter. When the spindle is rotating, there is a microscopic air gap maintained between the R/W heads and the platters, known as the head flying height. This air gap is removed when the spindle stops rotating and the R/W head rests on a special area on the platter near the spindle. This area is called the landing zone. The landing zone is coated with a lubricant to reduce friction between the head and the platter.

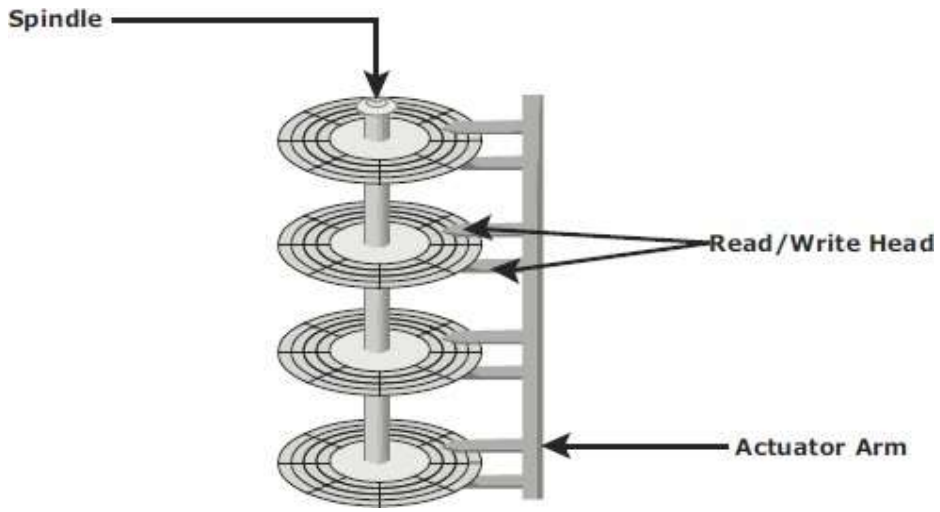



**Fig 1.13: Actuator arm assembly**

The logic on the disk drive ensures that heads are moved to the landing zone before they touch the surface. If the drive malfunctions and the R/W head accidentally touches the surface of the platter outside the landing zone, a head crash occurs. In a head crash, the magnetic coating on the platter is scratched and may cause damage to the R/W head. A head crash generally results in data loss.

**Actuator Arm Assembly**

R/W heads are mounted on the actuator arm assembly, which positions the R/W head at the location on the platter where the data needs to be written or read. The R/W heads for all platters on a drive are attached to one actuator arm assembly and move across the platters simultaneously.

**Drive Controller Board**

The controller is a printed circuit board, mounted at the bottom of a disk drive. It consists of a microprocessor, internal memory, circuitry, and firmware. The firmware controls the power to the spindle motor and the speed of the motor. It also manages the communication between the drive and the host. In addition, it controls the R/W operations by moving the actuator arm and switching between different R/W heads, and performs the optimization of data access.

**Physical Disk Structure**

Data on the disk is recorded on tracks, which are concentric rings on the platter around the spindle, as shown in below Figure. The tracks are numbered, starting from zero, from the outer edge of the platter. The number of tracks per inch (TPI) on the platter (or the track density) measure show tightly the tracks are packed on a platter.

Each track is divided into smaller units called sectors. A sector is the smallest, individually addressable unit of storage. The track and sector structure is written on the platter by the drive manufacturer using a low-level formatting operation. The number of sectors per track varies according to the drive type.

The first personal computer disks had 17 sectors per track. Recent disks have a much larger number of sectors on a single track. There can be thousands of tracks on a platter, depending on the physical dimensions and recording density of the platter.

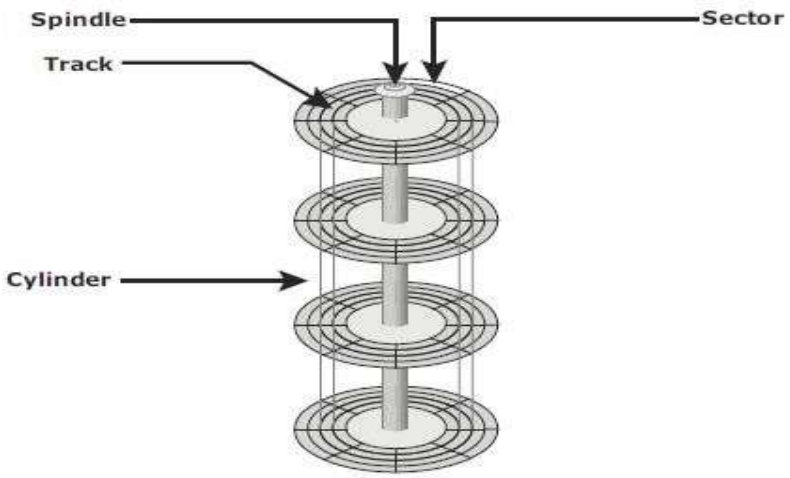



**Fig 1.14: Disk structure: sectors, tracks, and cylinders**

Typically, a sector holds 512 bytes of user data, although some disks can be formatted with larger sector sizes. In addition to user data, a sector also stores other information, such as the sector number, head number or platter number, and track number. This information helps the controller to

locate the data on the drive. A cylinder is a set of identical tracks on both surfaces of each drive platter. The location of R/W heads is referred to by the cylinder number, not by the track number.

**Zoned Bit Recording**

Platters are made of concentric tracks; the outer tracks can hold more data than the inner tracks because the outer tracks are physically longer than the inner tracks. On older disk drives, the outer tracks had the same number of sectors as the inner tracks, so data density was low on the outer tracks. This was an inefficient use of the available space, as shown in Figure(a). Zoned bit recording uses the disk efficiently. As shown in Figure (b), this mechanism groups tracks into zones based on their distance from the center of the disk. The zones are numbered, with the outer most zone being zone 0. An appropriate number of sectors per track are assigned to each zone, so a zone near the center of the platter has fewer sectors per track than a zone on the outer edge. However, tracks within a particular zone have the same number of sectors.

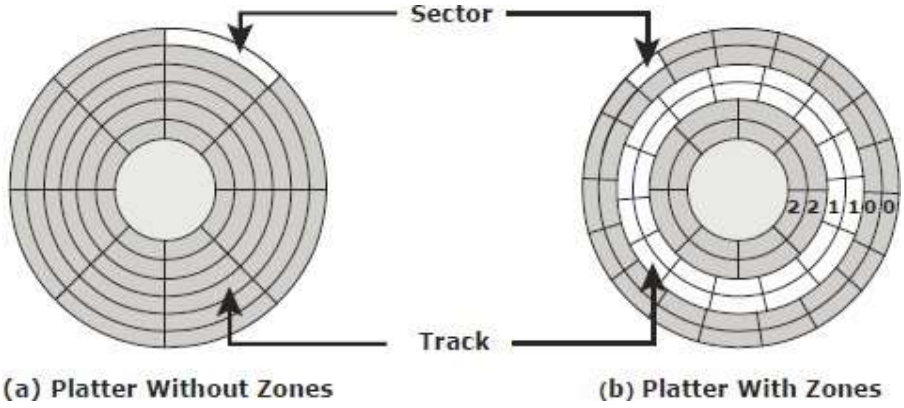



**Fig1.15: Zoned bit recording**

**Logical Block Addressing**

Earlier drives used physical addresses consisting of the cylinder, head, and sector (CHS) number to refer to specific locations on the disk, as shown in Figure (a), and the host operating system had to be aware of the geometry of each disk used. Logical block addressing (LBA), as shown in

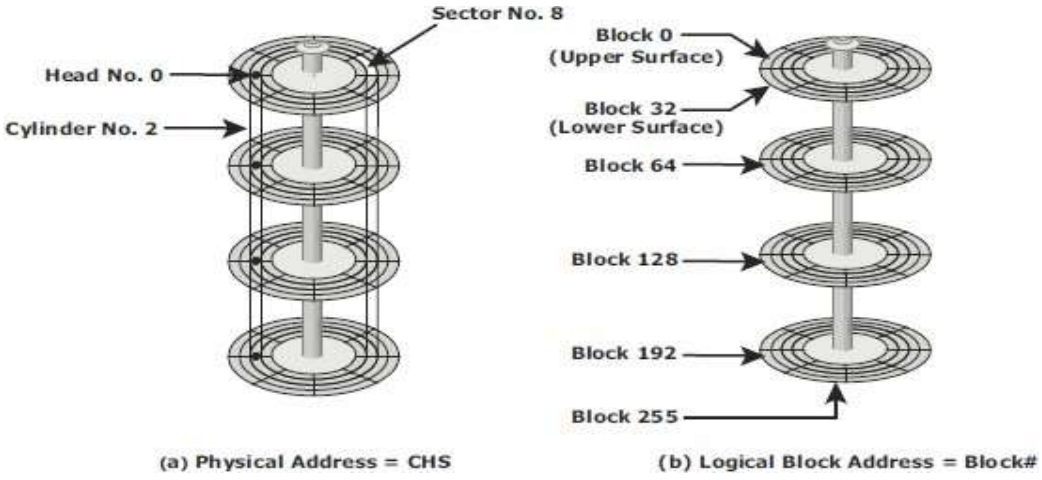



**Fig 1.16: Physical address and logical block address**

Figure (b), simplifies addressing by using a linear address to access physical blocks of data. The disk controller translates LBA to a CHS address, and the host needs to know only the size of the disk drive in terms of the number of blocks. The logical blocks are mapped to physical sectors on a 1:1 basis.

In Figure 2-10 (b), the drive shows eight sectors per track, eight heads, and four cylinders. This means a total of 8×8×4=256 blocks, so the block number ranges from 0 to 255. Each block has its own unique address. Assuming that the sector holds 512 bytes, a 500 GB drive with a formatted capacity of 465.7 GB has in excess of 976,000,000 blocks.

## 7. Disk Drive Performance

A disk drive is an electromechanical device that governs the overall performance of the storage system environment. The various factors that affect the performance of disk drives are discussed in this section.

**Disk Service Time**

Disk service time is the time taken by a disk to complete an I/O request. Components that contribute to the service time on a disk drive are seek time, rotational latency, and data transfer rate.

**Seek Time**

The seek time (also called access time) describes the time taken to position the R/W heads across the platter with a radial movement (moving along the radius of the platter). In other words, it is the time taken to position and settle the arm and the head over the correct track. Therefore, the lower the seek time, the faster the I/O operation. Disk vendors publish the following seek time specifications:

**Full Stroke:** The time taken by the R/W head to move across the entire width of the disk, from the innermost track to the outermost track.

**Average:** The average time taken by the R/W head to move from one random track to another, normally listed as the time for one-third of a full stroke.

**Track-to-Track:** The time taken by the R/W head to move between adjacent tracks.

Each of these specifications is measured in milli seconds. The seek time of a disk is typically specified by the drive manufacturer. The average seek time on a modern disk is typically in the range of 3 to 15 milli seconds. Seek time has more impact on the read operation of random tracks rather than adjacent tracks.

To minimize the seek time, data can be written to only a subset of the available cylinders. This results in lower usable capacity than the actual capacity of the drive. For example, a 500 GB disk drive is setup to use only the first 40 percent of the cylinders and is effectively treated as a 200 GB drive. This is known as short-stroking the drive.

**Rotational Latency**

To access data, the actuator arm moves the R/W head over the platter to a particular track while the platter spins to position the requested sector under the R/W head. The time taken by the platter to rotate and position the data under the R/W head is called rotational latency. This latency depends on the rotation speed of the spindle and is measured in milliseconds. The average rotational latency is one-half of the time taken for a full rotation. Similar to the seek time, rotational latency has more impact on the reading/writing of random sectors on the disk than on the same operations on adjacent sectors.

Average rotational latency is approximately 5.5 ms for a 5,400-rpm drive, and around 2.0 ms for a 15,000- rpm (or 250-rps revolution per second) drive as shown here:

Average rotational latency for a 15,000 rpm (or250rps) drive = 0.5/250 = 2 milliseconds.

**Data Transfer Rate**

The data transfer rate (also called transfer rate) refers to the average amount of data per unit time that the drive can deliver to the HBA. It is important to first understand the process of read/write operations to calculate data transfer rates. In a read operation, the data first moves from disk platters to R/W heads; then it moves to the drive's internal buffer. Finally, data moves from the buffer through the interface to the host HBA. In a write operation, the data moves from the HBA to the internal buffer of the disk drive through the drive's interface. The data then moves from the buffer to the R/W heads. Finally, it moves from the R/W heads to the platters.

The data transfer rates during the R/W operations are measured in terms of internal and external transfer rates, as shown in below figure.

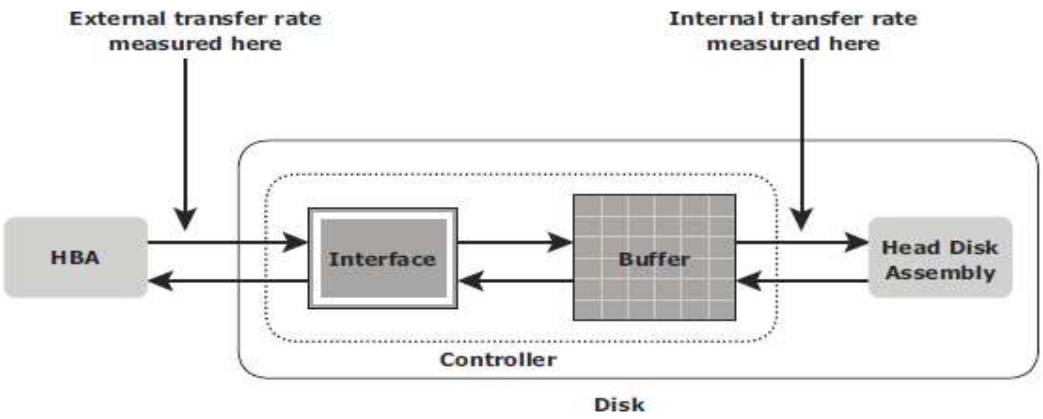



**Fig 1.17: Data transfer rate**

Internal transfer rate is the speed at which data moves from a platter's surface to the internal buffer (cache) of the disk. The internal transfer rate takes into account factors such as the seek time and rotational latency. External transfer rate is the rate at which data can move through the interface to the HBA. The external transfer rate is generally the advertised speed of the interface, such as 133 MB/s for ATA. The sustained external transfer rate is lower than the interface speed.

**Disk I/O Controller Utilization**

Utilization of a disk I/O controller has a significant impact on the I/O response time. To understand this impact, consider that a disk can be viewed as a black box consisting of two elements:

**Queue:** The location where an I/O request waits before it is processed by the I/O controller

**Disk I/O Controller:** Processes I/Os waiting in the queue one by one

The I/O requests arrive at the controller at the rate generated by the application. This rate is also called the arrival rate. These requests are held in the I/O queue, and the I/O controller processes them one by one, as shown in Figure 1-18. The I/O arrival rate, the queue length, and the time taken by the I/O controller to process each request determines the I/O response time. If the controller is busy or heavily utilized , the queue size will be large and the response time will be high.

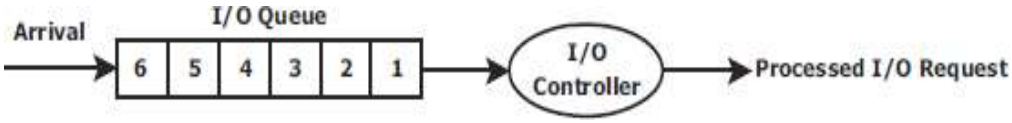



**Fig1.18:I/O processing**

Based on the fundamental laws of disk drive performance, the relationship between controller utilization and average response time is given as

**Average response time (TR)=Service time (TS)/(1 –Utilization)**

Where TS is the time taken by the controller to serve an I/O.

As the utilization reaches 100 percent — that is, as the I/O controller saturates — the response time is closer to infinity. In essence, the saturated component, or the bottleneck, forces the serialization of I/O requests, meaning that each I/O request must wait for the completion of the I/O requests that preceded it. Figure 1-19 shows a graph plotted between utilization and response time.

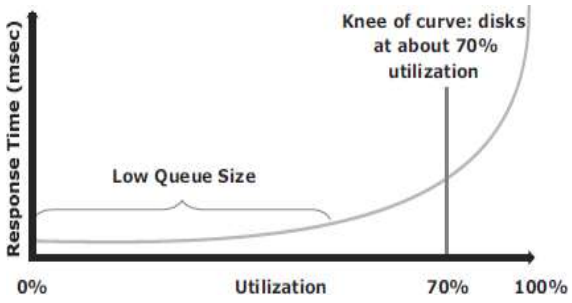



**Fig 1.19: Utilization versus response time**

The graph indicates that the response time changes are nonlinear as the utilization increases. When the average queue sizes are low, the response time remains low. The response time increases slowly with added load on the queue and increases exponentially when the utilization exceeds 70 percent.Therefore, forperformance-sensitive applications, it is common to utilize disks below their 70 percent of I/O serving capability.

## 8. Host Access to Data

Data is accessed and stored by applications using the underlying infrastructure. The key components of this infrastructure are the operating system (or file system), connectivity, and storage. The storage device can be internal and (or) external to the host. In either case, the host controller card accesses the storage devices using predefined protocols, such as IDE/ATA,SCSI, or Fibre Channel (FC). IDE/ATA and SCSI are popularly used in small and personal computing environments for accessing internal storage. FC and iSCSI protocols are used for accessing data from an external storage device (or subsystems). External storage devices can be connected to the host directly or through the storage network. When the storage is connected directly to the host, it is referred as direct-attached storage (DAS).

Understanding access to data over a network is important because it lays the foundation for storage networking technologies. Data can be accessed over a network in one of the following ways: block level, file level, or object level. In general, the application requests data from the file system (or operating system) by specifying the filename and location. The file system maps the file attributes to the logical block address of the data and sends the request to the storage device. The storage device converts the logical block address (LBA) to a cylinder-head-sector (CHS) address and fetches the data.

In a block-level access, the file system is created on a host, and data is accessed on a network at the block level, as shown in Figure 1-20(a). In this case, raw disks or logical volumes are assigned to the host for creating the file system. In a file-level access, the file system is created on a separate file server or at the storage side, and the file-level request is sent over a network, as shown in Figure1-20 (b).Because data is accessed at the file level, this method has higher overhead, as compared to the data accessed at the block level. Object-level access is an intelligent evolution, whereby data is accessed over a network in terms of self-contained objects with a unique object identifier.

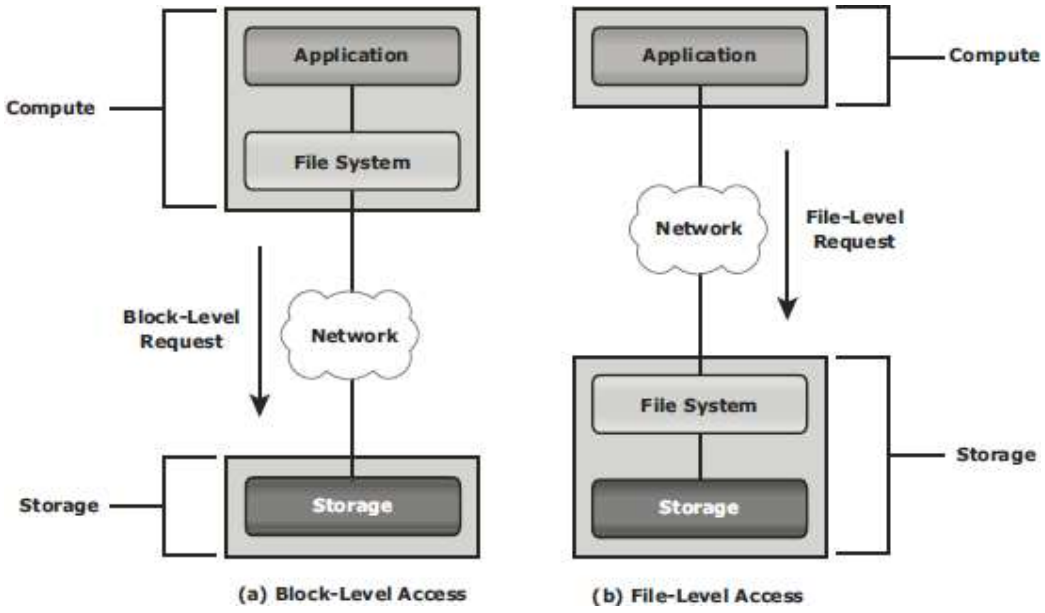



**Fig 1.20: Host access to storage**

## 9. Direct-Attached Storage

DAS is an architecture in which storage is connected directly to the hosts. The internal disk drive of a host and the directly-connected external storage array are some examples of DAS. Although the implementation of storage networking technologies is gaining popularity, DAS has remained suitable for localized data access in a small environment, such as personal computing and workgroups.

DAS is classified as internal or external, based on the location of the storage device with respect to the host. In internal DAS architectures, the storage device is internally connected to the host by a serial or parallel bus (see Figure 1-21 [a]). The physical bus has distance limitations and can be sustained only over a shorter distance for high speed connectivity. In addition, most internal buses can support only a limited number of devices, and they occupy a large amount of space inside the host, making maintenance of other components difficult.

On the other hand, in external DAS architectures, the host connects directly to the external storage device, and data is accessed at the block level (see Figure 1-21[b]). In most cases, communication between the host and the storage device takes place over a SCSI or FC protocol. Compared to internal DAS, an external DAS overcomes the distance and device count limitations and provides centralized management of storage devices.

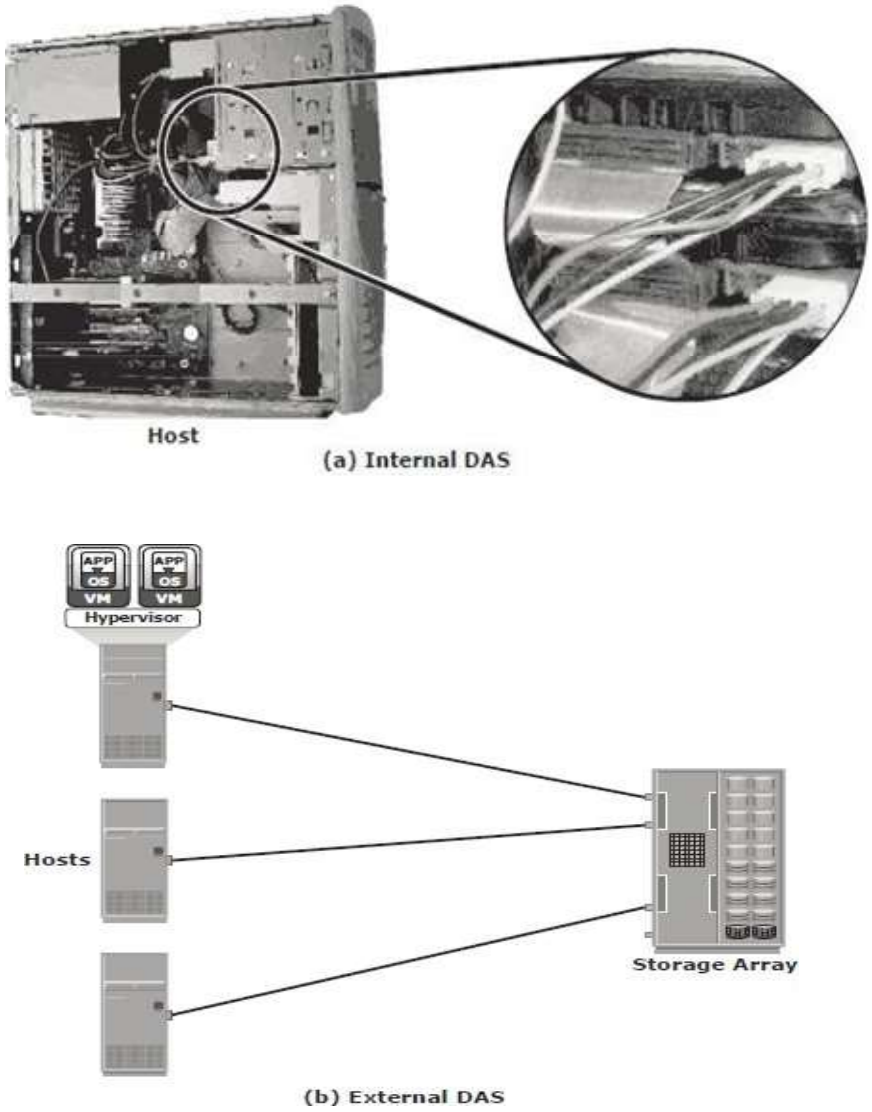



**Fig 1.21: Internal and external DAS architecture DAS**

**Benefits and Limitations**

DAS requires a relatively lower initial investment than storage networking architectures. The DAS configuration is simple and can be deployed easily and rapidly. The setup is managed using host-based tools, such as the host OS, which makes storage management tasks easy for small environments. Because DAS has a simple architecture, it requires fewer management tasks and less hardware and software elements to set up and operate.

However, DAS does not scale well. A storage array has a limited number of ports, which restricts the number of hosts that can directly connect to the storage.

When capacities are reached, the service availability may be compromised. DAS does not make optimal use of resources due to its limited capability to share front-end ports. In DAS environments, unused resources cannot be easily reallocated, resulting in islands of over-utilized and under-utilized storage pools.

## 10. Storage Design Based on Application

### Requirements and Disk Performance

Determining storage requirements for an application begins with determining the required storage capacity. This is easily estimated by the size and number of file systems and database components used by applications. The I/O size, I/O characteristics, and the number of I/Os generated by the application at peak workload are other factors that affect disk performance, I/O response time, and design of storage systems. The I/O block size depends on the file system and the data base on which the application is built. Block size in a database environment is controlled by the underlying database engine and the environment variables. The disk service time (TS) for an I/O is a key measure of disk performance; TS, along with disk utilization rate (U), determines the I/O response time for an application. The total disk service time (TS) is the sum of the seek time (T), rotational latency (L), and internal transfer time (X):

$$TS = T + L + X$$

Consider an example with the following specifications provided for a disk:

- The average seek time is 5 ms in a random I/O environment; therefore, T=5 ms.
- Disk rotation speed of 15,000 revolutions per minute or 250 revolutions per second—from which rotational latency (L) can be determined, which is one-half of the time taken for a full rotation or L = (0.5/250 rps expressed in ms).
- 40MB/s internal data transfer rate, from which the internal transfer time (X) is derived based on the block size of the I/O — for example, an I/O with a block size of 32 KB; therefore X = 32 KB/40 MB.

Consequently, the time taken by the I/O controller to serve an I/O of block size 32 KB is

(TS)=5ms + (0.5/250) + 32 KB/40 MB = 7.8 ms.

Therefore, the maximum number of I/Os serviced per second or IOPS is

(1/TS) =1/(7.8 ×10-3) = 128IOPS.

Table 1-1 lists the maximum IOPS that can be serviced for different block sizes using the previous disk specifications.

**Table 1-1: IOPS Performed by Disk Drive**

| BLOCK SIZE | $T_S = T + L + X$ | IOPS = $1/T_S$ |
|---|---|---|
| 4 KB | 5 ms + (0.5/250 rps) + 4 K/40 MB = 5 + 2 + 0.1 = 7.1 | 140 |
| 8 KB | 5 ms + (0.5/250 rps) + 8 K/40 MB = 5 + 2 + 0.2 = 7.2 | 139 |
| 16 KB | 5 ms + (0.5/250 rps) + 16 K/40 MB = 5 + 2 + 0.4 = 7.4 | 135 |
| 32 KB | 5 ms + (0.5/250 rps) + 32 K/40 MB = 5 + 2 + 0.8 = 7.8 | 128 |
| 64 KB | 5 ms + (0.5/250 rps) + 64 K/40 MB = 5 + 2 + 1.6 = 8.6 | 116 |

The IOPS ranging from 116 to 140 for different block sizes represents the IOPS that can be achieved at potentially high levels of utilization (close to 100 percent).

The application response time, R, increases with an increase in disk controller utilization. For the same preceding example, the response time (R) for an I/O with a block size of 32 KB at 96 percent disk controller utilization is

R =TS/(1 – U)=7.8/(1– 0.96)=195 ms

If the application demands a faster response time, then the utilization for the disks should be maintained below 70 percent. For the same 32-KB block size, at 70-percent disk utilization, the response time reduces drastically to 26ms. However, at lower disk utilization, the number of IOPS a disk can perform is also reduced. In the case of a 32-KB block size, a disk can perform 128I OPS at almost 100 percent utilization, where as the number of IOPS it can perform at 70-percent utilization is 89 (128x0.7). This indicates that the number of I/Os a disk can perform is an important factor that needs to be considered while designing the storage requirement for an application. Therefore, the storage requirement for an application is determined in terms of both the capacity and IOPS. If an application needs 200 GB of disk space, then this capacity can be provided simply with a single disk. However, if the application IOPS requirement is high, then it results in performance degradation because just a single disk might not provide the required response time for I/O operations. The total number of disks required (DR) for an application is computed as follows:

DR = Max (DC, DI)

Where DC is the number of disks required to meet the capacity, and DI is the number of disks required to meet the application IOPS requirement. Let's see an example.

Consider an example in which the capacity requirement for an application is 1.46 TB. The number of IOPS generated by the application at peak workload is estimated at 9,000 IOPS. The vendor specifies that a 146-GB, 15,000-rpm drive is capable of doing a maximum 180 IOPS.

In this example, the number of disks required to meet the capacity requirements will be 1.46TB/146GB = 10 disks.

To meet the application IOPS requirements, the number of disks required is 9,000/180=50.However, if the application is response-time sensitive, the number of IOPS a disk drive can perform should be calculated based on 70-percent disk utilization. Considering this, the number of IOPS a disk can perform at 70 percent utilization is 180 ¥ 0.7 = 126 IOPS. Therefore, the number of disks required to meet the application IOPS requirement will be 9,000/126 = 72.

As a result, the number of disks required to meet the application requirements will be Max (10,72)=72 disks.

The preceding example indicates that from a capacity-perspective, 10 disks are sufficient; however, the number of disks required to meet application performance is 72. To optimize disk

requirements from a performance perspective, various solutions are deployed in a real-time environment. Examples of these solutions are disk native command queuing, use of flash drives, RAID, and the use of cache memory.

## Question Bank

1. Explain the evolution of storage technology.
2. Explain the architecture and evolution of storage technology.
3. Define data center. Explain the core elements of data center.
4. Discuss the key characteristics of a data center, with a neat diagram.
5. What is a data center? Explain key characteristics of data center elements.
6. Discuss the tasks of key management activities.
7. Write short notes on i) Virtualization ii) Cloud Computing.
8. Discuss various software components that are essential parts of a host system.
9. Discuss file system and the process of mapping user files to disk storage.
10. Define Connectivity. Discuss the physical components of connectivity.
11. Discuss the interface protocols used for host to storage communication.
12. Write a short note on storage.
13. Discuss volume manager and compute virtualization in detail.
14. Briefly Explain Disk Drive Components.
15. Write a note on disk drive performance.
16. What is DAS? Explain its benefits and Limitations.

==========

# Storage Area Networks (18CS822)

SEMESTER – VIII

## Module II

**Data Protection** - RAID: RAID Implementation Methods, RAID Array Components, RAID Techniques, RAID Levels, RAID Impact on Disk Performance, RAID Comparison. **Intelligent Storage Systems:** Components of an Intelligent Storage System, Types of Intelligent Storage Systems. **Fibre Channel Storage Area Networks - Fibre Channel:** Overview, The SAN and Its Evolution, Components of FC SAN.

## Data Protection – RAID

Earlier days data was stored on a single large, expensive disk drive called Single Large Expensive Drive [SLED]. Use of single disks could not meet the required performance levels because they were capable of serving only a limited number of I/Os.

RAID [Redundant Arrays of Inexpensive Disks] is an enabling technology that leverages multiple drives as part of a set that provides data protection against drive failures.

In 1987, Patterson, Gibson, and Katz at the University of California, Berkeley, published a paper titled "A Case for Redundant Arrays of Inexpensive Disks [RAID]". This paper described the use of small-capacity, inexpensive disk drives as an alternative to large-capacity drives common on mainframe computers.

### 1) RAID Implementation Methods

The two methods of RAID implementation are hardware and software. Both have their advantagesand disadvantages.

**i) Software RAID**

Software RAID uses host-based software to provide RAID functions. It is implemented at theoperating-system level and does not use a dedicated hardware controller to manage the RAID array.

Software RAID implementations offer cost and simplicity benefits when compared with hardwareRAID. However, they have the following limitations:

- Performance: Software RAID affects overall system performance. This is due to additional CPUcycles required to perform RAID calculations.
- Supported features: Software RAID does not support all RAID levels.
- Operating system compatibility: Software RAID is tied to the host operating system; hence, upgrades to software RAID or to the operating system should be validated for compatibility. Thisleads to inflexibility in the data-processing environment.

### ii) Hardware RAID

In hardware RAID implementations, a specialized hardware controller is implemented either onthe host or on the array.

Controller card RAID is a host-based hardware RAID implementation in which a specialized RAID controller is installed in the host, and disk drives are connected to it. Manufacturers also integrate RAID controllers on motherboards. A host-based RAID controller is not an efficient solution in a data center environment with a large number of hosts.

The external RAID controller is an array-based hardware RAID. It acts as an interface between the host and disks. It presents storage volumes to the host, and the host manages these volumes as physical drives. The key functions of the RAID controllers are as follows:

- Management and control of disk aggregations
- Translation of I/O requests between logical disks and physical disks
- Data regeneration in the event of disk failures

## 2. RAID Array Components

A RAID array is an enclosure that contains a number of disk drives and sup- porting hardware to implement RAID. A subset of disks within a RAID array can be grouped to form logical associations calledlogical arrays, also known as a RAID set or a RAID group.

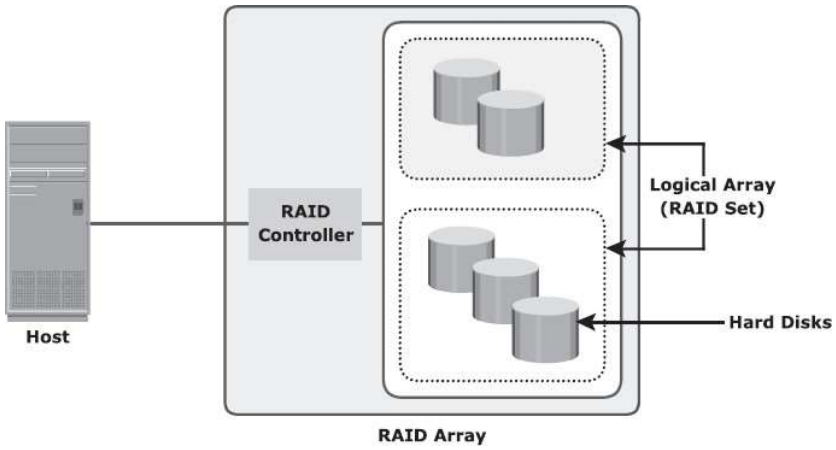



**Figure: Components of a RAID array**

## 3. RAID Techniques

RAID techniques — striping, mirroring, and parity — form the basis for defining various RAIDlevels. These techniques determine the data availability and performance characteristics of a RAID set.

**i) Striping**

Striping is a technique to spread data across multiple drives (more than one) to use the drives in parallel. All the read-write heads work simultaneously, allowing more data to be processed in a shorter timeand increasing performance, compared to reading and writing from a single disk.

Within each disk in a RAID set, a predefined number of contiguously address- able disk blocks aredefined as a strip. The set of aligned strips that spans across all the disks within the RAID set is called a stripe. Figure shows physical and logical representations of a striped RAID set. Strip size (also called stripe depth) describes the number of blocks in a strip and is the maximum amount of data that can be written to or read from a single disk in the set, assuming that the accessed data starts at the beginning of the strip. All strips in a stripe have the same number of blocks. Having a smaller strip size means that data is broken into smaller pieces while spread across the disks. Stripe size is a multiple of strip size by the number of data disks in the RAID set. For example, in a five disk striped RAID set witha strip size of 64 KB, the stripe size is 320 KB(64KB x 5). Stripe width refers to the number of data strips in a stripe. Striped RAID does not provide any data protection unless parity or mirroring is used, as discussed in the following sections.

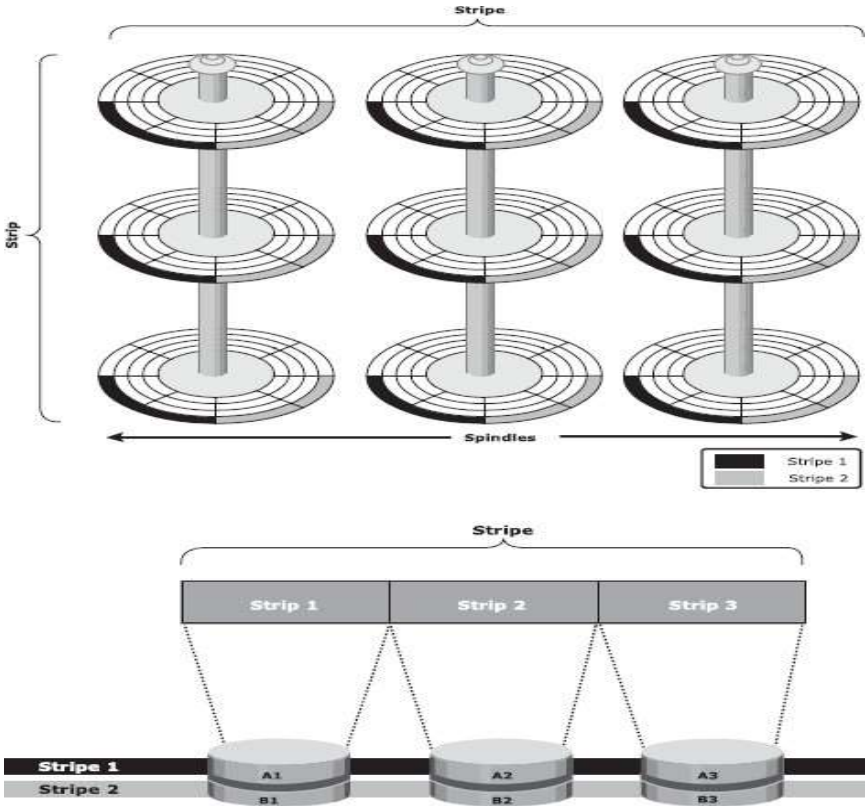



**Figure: Striped RAID set**

**ii) Mirroring**

Mirroring is a technique whereby the same data is stored on two different disk drives, yielding two copies of the data. If one disk drive failure occurs, the data is intact on the surviving disk drive and the controller continues to service the host's data requests from the surviving disk of a mirrored pair.

When the failed disk is replaced with a new disk, the controller copies the data from the surviving disk of the mirrored pair. This activity is transparent to the host. In addition to providing complete data redundancy, mirroring enables fast recovery from disk failure. However, disk mirroring provides only data protection and is not a substitute for data backup. Mirroring constantly captures changes in the data, whereas a backup captures point-in-time images of the data.

Mirroring involves duplication of data —the amount of storage capacity needed is twice the amount of data being stored. Therefore, mirroring is considered expensive and is preferred for mission-critical applications that cannot afford the risk of any data loss. Mirroring improves read performance because read requests can be serviced by both disks. However, write performance is slightly lower than that in a single disk because each write request manifests as two writes on the disk drives. Mirroring does not deliver the same levels of write performance as a striped RAID.

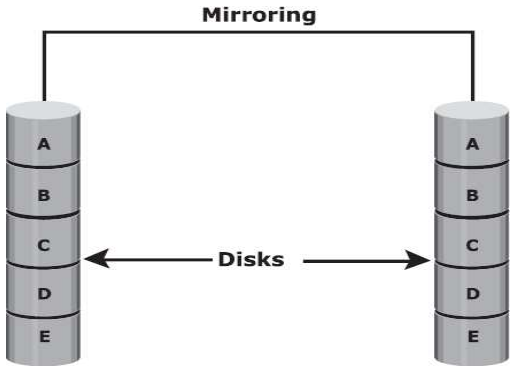



**Figure: Mirrored disks in an array**

**iii) Parity**

Parity is a method to protect striped data from disk drive failure without the cost of mirroring. An additional disk drive is added to hold parity, a mathematical construct that allows re-creation of the missing data. Parity is a redundancy technique that ensures protection of data without maintaining a full set of duplicate data. Calculation of parity is a function of the RAID controller.

Parity information can be stored on separate, dedicated disk drives or distributed across all the drives in a RAID set. Figure shows a parity RAID set. The first four disks, labeled "Data Disks," contain the data. The fifth disk, labeled "Parity Disk," stores the parity information, which, in this case, is the sum of the elements in each row. Now, if one of the data disks fails, the missing value can be calculated by subtracting the sum of the rest of the elements from the parity value. Here, for simplicity, the computation of parity is represented as an arithmetic sum of the data. However, parity calculation is a bitwise XOR operation.

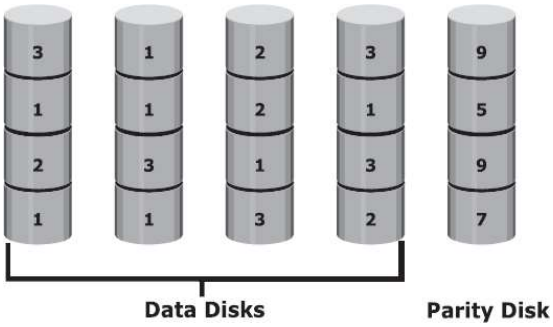



Figure 3-4: Parity

**XOR OPERATION**

A bit-by-bit Exclusive -OR (XOR) operation takes two bit patterns of equal length and performs the logical XOR operation on each pair of corresponding bits. The result in each position is 1 if the two bits are different, and 0 if they are the same. The truth table of the XOR operation is shown next. (A and B denote the inputs and C, the output after performing the XOR operation.) If any of the data from A, B, or C is lost, it can be reproduced by performing an XOR operation on the remaining available data. For example, if a disk containing all the data from A fails, the data can be regenerated by performing an XOR between B and C.

| A | B | C |
|---|---|---|
| 0 | 0 | 0 |
| 0 | 1 | 1 |
| 1 | 0 | 1 |
| 1 | 1 | 0 |

Compared to mirroring, parity implementation considerably reduces the cost associated with data protection. Consider an example of a parity RAID configuration with five disks where four disks hold data, and the fifth holds the parity information. In this example, parity requires only 25 percent extra disk space compared to mirroring, which requires 100 percent extra disk space. However, there are some disadvantagesof using parity. Parity information is generated from data on the data disk. Therefore, parity is recalculated every time there is a change in data. This recalculation is time-consuming and affects the performance of the RAID array.

For parity RAID, the stripe size calculation does not include the parity strip. For example in a five(4 + 1) disk parity RAID set with a strip size of 64 KB, the stripe size will be 256 KB (64 KB x 4).

## 4. RAID Levels

Application performance, data availability requirements, and cost determine the RAID level selection. These RAID levels are defined on the basis of striping, mirroring, and parity techniques. Some RAID levels use a single technique, whereas others use a combination of techniques. The following table shows the commonly used RAID levels.

| LEVELS | BRIEF DESCRIPTION |
|---|---|
| RAID 0 | Striped set with no fault tolerance |
| RAID 1 | Disk mirroring |
| Nested | Combinations of RAID levels. Example: RAID 1 + RAID 0 |
| RAID 3 | Striped set with parallel access and a dedicated parity disk |
| RAID 4 | Striped set with independent disk access and a dedicated parity disk |
| RAID 5 | Striped set with independent disk access and distributed parity |
| RAID 6 | Striped set with independent disk access and dual distributed parity |

### RAID 0

RAID 0 configuration uses data striping techniques, where data is striped across all the disks withina RAID set. Therefore it utilizes the full storage capacity of a RAID set. To read data, all the strips are put back together by the controller. Figure shows RAID 0 in an array in which data is striped across five disks. When the number of drives in the RAID set increases, performance improves because more data can be read or written simultaneously. RAID 0 is a good option for applications that need high I/O throughput.

However, if these applications require high availability during drive failures, RAID 0 does not provide dataprotection and availability.

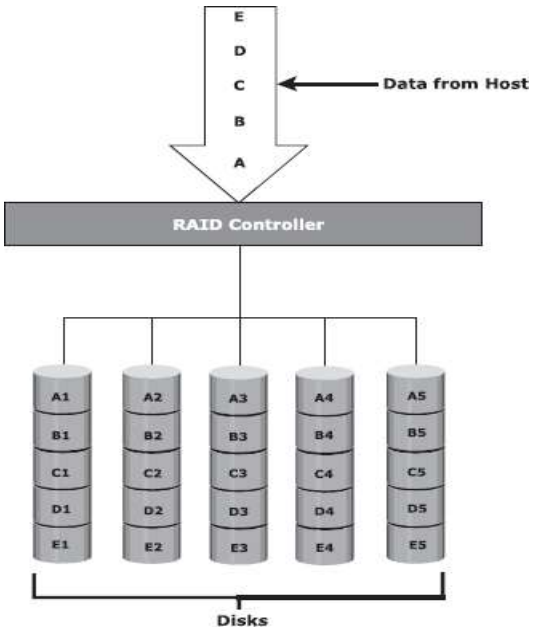



**Figure: RAID 0**

**RAID 1**

RAID 1 is based on the mirroring technique. In this RAID configuration, data is mirrored to providefault tolerance. A RAID 1 set consists of two disk drives and every write is written to both disks. The mirroring is transparent to the host. During disk failure, the impact on data recovery in RAID 1 is the leastamong all RAID implementations. This is because the RAID controller uses the mirror drive for data recovery. RAID 1 is suitable for applications that require high availability and cost is no constraint.

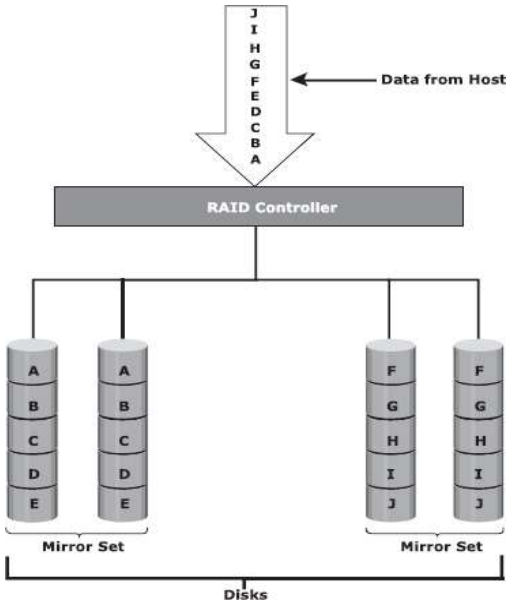


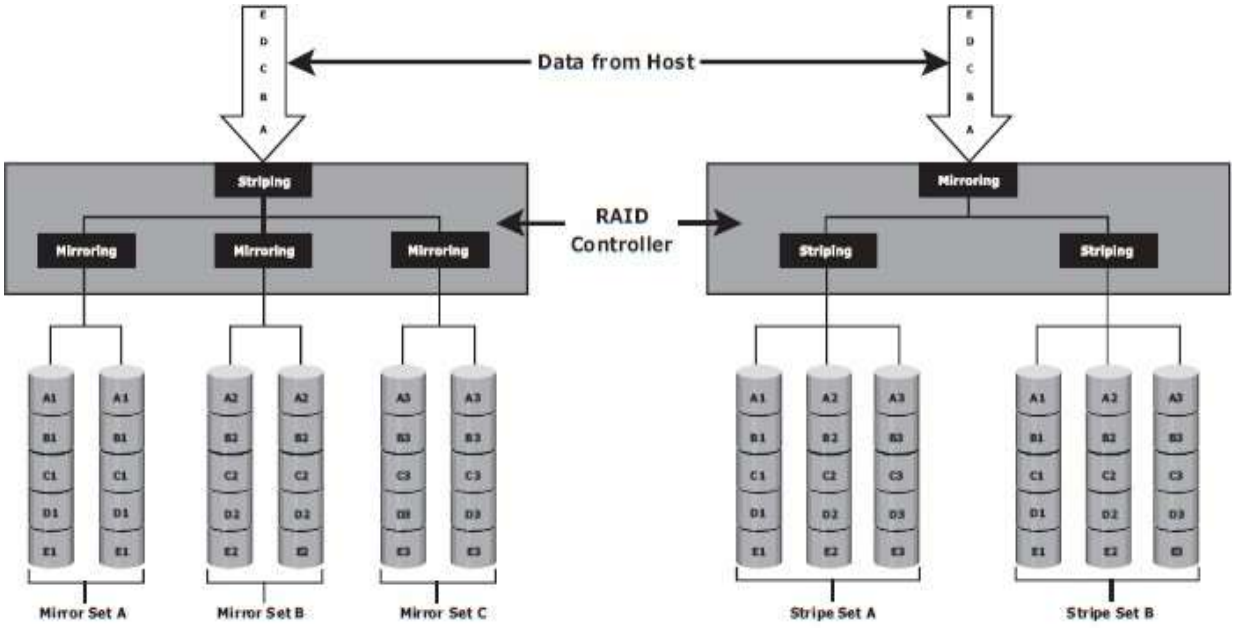

**Figure: RAID 1**

**Nested RAID**

Most data centers require data redundancy and performance from their RAID arrays. RAID 1+0 and RAID 0+1combine the performance benefits of RAID 0 with the redundancy benefits of RAID 1. They use striping and mirroring techniques and combine their benefits. These types of RAID require an even numberof disks, the minimum being four.

RAID 1+0 is also known as RAID 10 (Ten) or RAID 1/0. Similarly, RAID 0+1 is also known as RAID 01 or RAID 0/1. RAID 1+0 performs well for workloads with small, random, write-intensive I/Os. Some applications that benefit from RAID 1+0 include the following:

- High transaction rate Online Transaction Processing (OLTP)
- Large messaging installations
- Database applications with write intensive random access workloads
- A common misconception is that RAID 1+0 and RAID 0+1 are the same. Under normal conditions, RAID levels 1+0 and 0+1 offer identical benefits. However, rebuild operations in the case of disk failure differ between the two.




(a) RAID 1+0 (b) RAID 0+1

RAID 1+0 is also called striped mirror. The basic element of RAID 1+0 is a mirrored pair, which means that data is first mirrored and then both copies of the data are striped across multiple disk drive pairs in a RAID set. When replacing a failed drive, only the mirror is rebuilt. In other words, the disk array controller uses the surviving drive in the mirrored pair for data recovery and continuous operation. Data from the surviving disk is copied to the replacement disk.

To understand the working of RAID 1+0, consider an example of six disks forming a RAID 1+0

(RAID 1 first and then RAID 0) set. These six disks are paired into three sets of two disks, where each set acts as a RAID 1 set (mirrored pair of disks). Data is then striped across all the three mirrored sets to formRAID 0. Following are the steps performed in RAID 1+0:

Drives 1+2 = RAID 1 (Mirror Set A) Drives 3+4 = RAID 1 (Mirror Set B) Drives 5+6 = RAID 1 (Mirror Set C)

Now, RAID 0 striping is performed across sets A through C. In this configuration, if drive 5 fails, then the mirror set C alone is affected. It still has drive 6 and continues to function and the entire RAID 1+0 array also keeps functioning. Now, suppose drive 3 fails while drive 5 was being replaced. In this casethe array still continues to function because drive 3 is in a different mirror set. So, in this configuration, upto three drives can fail without affecting the array, as long as they are all in different mirror sets.

RAID 0+1 is also called a mirrored stripe. The basic element of RAID 0+1 is a stripe. This means that the process of striping data across disk drives is performed initially, and then the entire

stripe is mirrored. In this configuration if one drive fails, then the entire stripe is faulted. Consider the same example of six disks to understand the working of RAID 0+1 (that is, RAID 0 first and then RAID 1). Here, six disks are paired into two sets of three disks each. Each of these sets, in turn, act as a RAID 0 set that contains three disks and then these two sets are mirrored to form RAID 1. Following are the steps performed in RAID 0+1:

Drives 1 + 2 + 3 = RAID 0 (Stripe Set A)

Drives 4 + 5 + 6 = RAID 0 (Stripe Set B)

Now, these two stripe sets are mirrored. If one of the drives, say drive 3, fails, the entire stripe set A fails. A rebuild operation copies the entire stripe, copying the data from each disk in the healthy stripe to an equivalent disk in the failed stripe. This causes increased and unnecessary I/O load on the surviving disks and makes the RAID set more vulnerable to a second disk failure.

**RAID 3**

RAID 3 stripes data for performance and uses parity for fault tolerance. Parity information is stored on a dedicated drive so that the data can be reconstructed if a drive fails in a RAID set. For example, in a set of five disks, four are used for data and one for parity. Therefore, the total disk space required is 1.25 times the size of the data disks. RAID 3 always reads and writes complete stripes of data across all disks because the drives operate in parallel. There are no partial writes that update one out of many strips in a stripe. Figure 3-8 illustrates the RAID 3 implementation.

RAID 3 provides good performance for applications that involve large sequen- tial data access, such as data backup or video streaming.

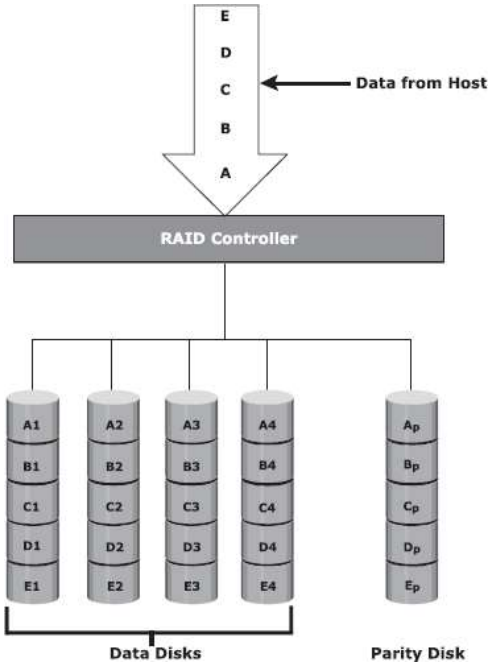



**Figure: RAID 3**

**RAID 4**

Similar to RAID 3, RAID 4 stripes data for high performance and uses parity for improved fault tolerance. Data is striped across all disks except the parity disk in the array. Parity information is stored ona dedicated disk so that the data can be rebuilt if a drive fails.

Unlike RAID 3, data disks in RAID 4 can be accessed independently so that specific data elementscan be read or written on a single disk without reading or writing an entire stripe. RAID 4 provides good read throughput and reason- able write throughput.

**RAID 5**

RAID 5 is a versatile RAID implementation. It is similar to RAID 4 because it uses striping. The drives (strips) are also independently accessible. The difference between RAID 4 and RAID 5 is the paritylocation. In RAID 4, parity is written to a dedicated drive, creating a write bottleneck for the parity disk. InRAID 5, parity is distributed across all disks to overcome the write bottleneck of a dedicated parity disk.

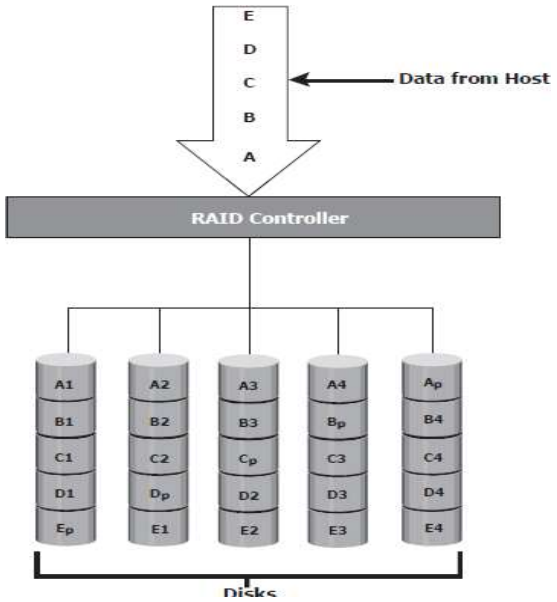



**Figure: RAID 5**

RAID 5 is good for random, read-intensive I/O applications and preferred for messaging, data mining, medium-performance media serving, and relational database management system (RDBMS) implementations, in which database administrators (DBAs) optimize data access.

**RAID 6**

RAID 6 works the same way as RAID 5, except that RAID 6 includes a second parity element to enable survival if two disk failures occur in a RAID set. Therefore, a RAID 6 implementation requires at least four disks. RAID 6 distributes the parity across all the disks. The write penalty (explained later in this chapter) in RAID 6 is more than that in RAID 5; therefore, RAID 5 writes perform better than RAID 6. Therebuild operation in RAID 6 may take longer than that in RAID 5 due to the presence of two parity sets.

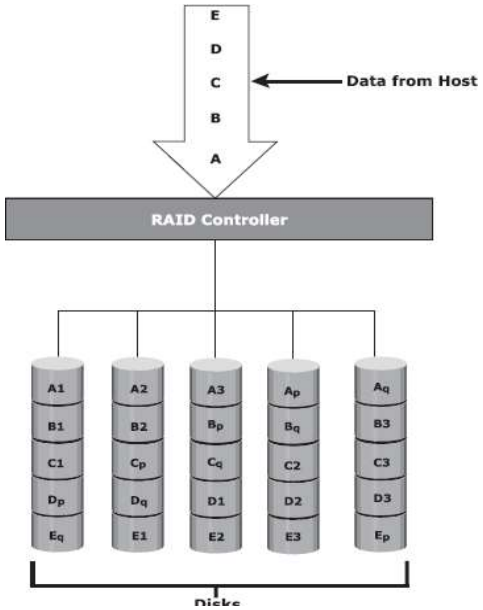



Figure: RAID 6

## 5. RAID Impact on Disk Performance

When choosing a RAID type, it is imperative to consider its impact on disk performance andapplication IOPS.

In both mirrored and parity RAID configurations, every write operation translates into more I/O overhead for the disks, which is referred to as a write penalty. In a RAID 1 implementation, every write operation must be performed on two disks configured as a mirrored pair, whereas in a RAID 5 implementation, a write operation may manifest as four I/O operations. When performing I/Os to a disk configured with RAID 5, the controller has to read, recalculate, and write a parity segment for every data write operation.

Figure illustrates a single write operation on RAID 5 that contains a group of five disks.

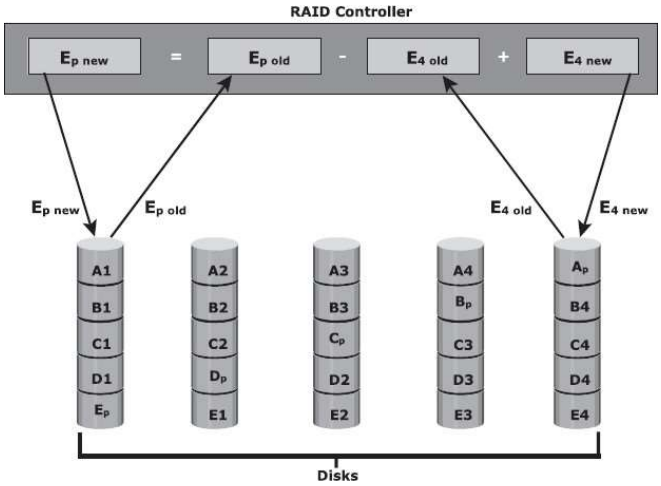



**Figure: Write penalty in RAID 5**

The parity (P) at the controller is calculated as follows:

Ep = E1 + E2 + E3+ E4 (XOR operations)

Whenever the controller performs a write I/O, parity must be computed by reading the old parity (Ep old) and the old data (E4 ,d) from the disk, which means two read I/Os. Then, the new parity (Ep new)is computed as follows:

Ep new = Ep old - $E_4$ old + E4 new (XOR operations)

After computing the new parity, the controller completes the write I/O by writing the new data and the new parity onto the disks, amounting to two write I/Os. Therefore, the controller performs two disk reads and two disk writes for every write operation, and the write penalty is 4.

In RAID 6, which maintains dual parity, a disk write requires three read operations: two parity and one data. After calculating both new parities, the controller performs three write operations: two parity andan I/O. Therefore, in a RAID 6 implementation, the controller performs six I/O operations for each write I/O, and the write penalty is 6.

## 6. Application IOPS and RAID Configurations

When deciding the number of disks required for an application, it is important to consider the impact of RAID based on IOPS generated by the application. The total disk load should be computed by considering the type of RAID configuration and the ratio of read compared to write from the host.

The following example illustrates the method to compute the disk load in different types of RAID. Consider an application that generates 5,200 IOPS, with 60 percent of them being reads.

The disk load in RAID 5 is calculated as follows:

RAID 5 disk load (reads + writes) = 0.6 x 5,200 + 4 x (0.4 x 5,200) [because the write penalty for RAID 5 is 4]

= 3,120 + 4 x 2,080

= 3,120 + 8,320

= 11,440 IOPS

The disk load in RAID 1 is calculated as follows:

RAID 1 disk load = 0.6 x 5,200 + 2 x (0.4 x 5,200) [because every write manifests as two writes to the disks]

= 3,120 + 2 x 2,080

= 3,120 + 4,160

= 7,280 IOPS

The computed disk load determines the number of disks required for the application. If in this example a disk drive with a specification of a maximum 180 IOPS needs to be used, the number of disks required tomeet the workload for the RAID configuration would be as follows:

RAID 5: 11,440/180 = 64 disks

RAID 1: 7,280/180 = 42 disks (approximated to the nearest even number)

## 7. RAID Comparison

Table compares the common types of RAID levels.

| RAID | MIN. DISKS | STORAGE EFFICIENCY % | COST | READ PERFORMANCE | WRITE PERFORMANCE | WRITE PENALTY | PROTECTION |
|---|---|---|---|---|---|---|---|
| 0 | 2 | 100 | Low | Good for both random and sequential reads | Good | No | No protection |
| 1 | 2 | 50 | High | Better than single disk | Slower than single disk because every write must be committed to all disks | Moderate | Mirror protection |
| 3 | 3 | [(n-1)/n] × 100 where n= number of disks | Moderate | Fair for random reads and good for sequential reads | Poor to fair for small random writes and fair for large, sequential writes | High | Parity protection for single disk failure |
| 4 | 3 | [(n-1)/n] × 100 where n= number of disks | Moderate | Good for random and sequential reads | Fair for random and sequential writes | High | Parity protection for single disk failure |
| 5 | 3 | [(n-1)/n] × 100 where n= number of disks | Moderate | Good for random and sequential reads | Fair for random and sequential writes | High | Parity protection for single disk failure |
| 6 | 4 | [(n-2)/n] × 100 where n= number of disks | Moderate but more than RAID 5. | Good for random and sequential reads | Poor to fair for random writes and fair for sequential writes | Very High | Parity protection for two disk failures |
| 1+0 and 0+1 | 4 | 50 | High | Good | Good | Moderate | Mirror protection |

# Intelligent Storage Systems

Business-critical applications require high levels of performance, availability, security, and scalability. A disk drive is a core element of storage that governs the performance of any storage system. Some of the older disk-array technologies could not overcome performance constraints due to the limitations of disk drives and their mechanical components. RAID technology made an important contribution to enhancing storage performance and reliability, but disk drives, even with a RAID implementation, could not meet the performance requirements of today's applications.

With advancements in technology, a new breed of storage solutions, known as *intelligent storage systems*, has evolved. These intelligent storage systems are feature-rich RAID arrays that provide highly optimized I/O processing capabilities. These storage systems are configured with a large amount of memory (called *cache*) and multiple I/O paths and use sophisticated algorithms to meet the requirements of performance-sensitive applications. These arrays have an operating environment that intelligently and optimally handles the management, allocation, and utilization of storage resources. Support for flash drives and other modern-day technologies, such as virtual storage provisioning and automated storage tiering, has added a new dimension to storage system performance, scalability, and availability.

## 1. Components of an Intelligent Storage System

An intelligent storage system consists of four key components: *front end*, *cache*, *back end*, and *physical disks*. Figure 4-1 illustrates these components and their interconnections. An I/O request received from the host at the front-end port is processed through cache and back end, to enable storage and retrieval of data from the physical disk. A read request can be serviced directly from cache if the requested data is found in the cache. In modern intelligent storage systems, front end, cache, and back end are typically integrated on a single board (referred to as a *storage processor* or *storage controller*).

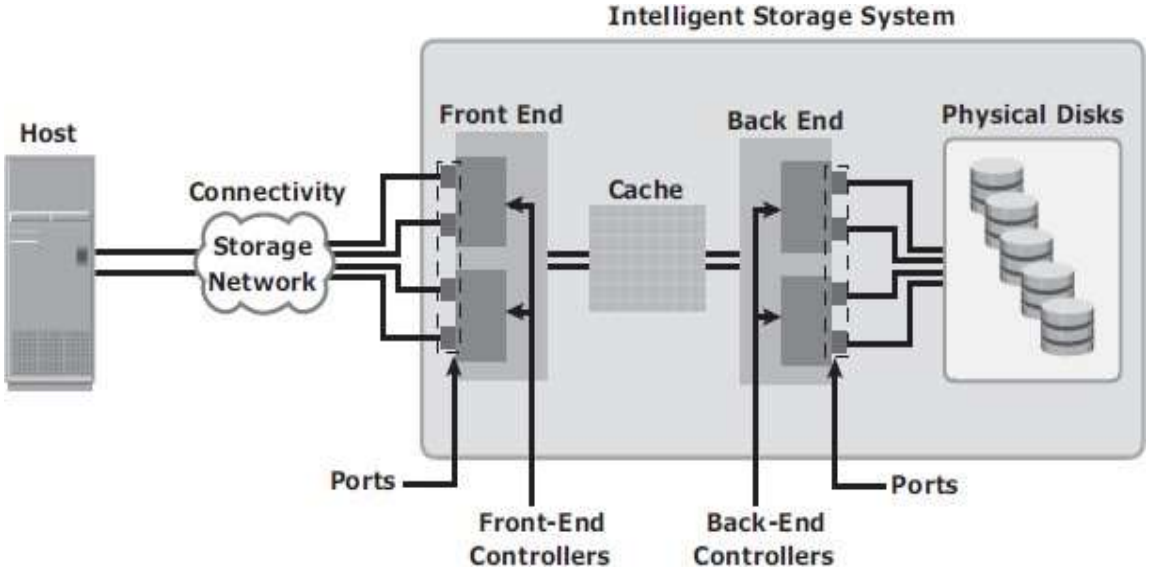



**Figure 4-1:** Components of an intelligent storage system

**a) Front End**

The front end provides the interface between the storage system and the host. It consists of two components: front-end ports and front-end controllers. Typically, a front end has redundant controllers for high availability, and each controller contains multiple ports that enable large numbers of hosts to connect to the intelligent storage system. Each front-end controller has processing logic that executes the appropriate transport protocol, such as Fibre Channel, iSCSI, FICON, or FCoE for storage connections.

*Front-end controllers* route data to and from cache via the internal data bus. When the cache receives the write data, the controller sends an acknowledgment message back to the host.

**b) Cache**

*Cache* is semiconductor memory where data is placed temporarily to reduce the time required to service I/O requests from the host. Cache improves storage system performance by isolating hosts from the mechanical delays associated with rotating disks or hard disk drives (HDD). Rotating disks are the slowest component of an intelligent storage system. Data access on rotating disks usually takes several millisecond because of seek time and rotational latency. Accessing data from cache is fast and typically takes less than a millisecond. On intelligent arrays, write data is first placed in cache and then written to disk.

***Structure of Cache***

Cache is organized into pages, which is the smallest unit of cache allocation. The size of a cache page is configured according to the application I/O size. Cache consists of the *data store* and *tag RAM*. The data store holds the data whereas the tag RAM tracks the location of the data in the data store and in the disk.

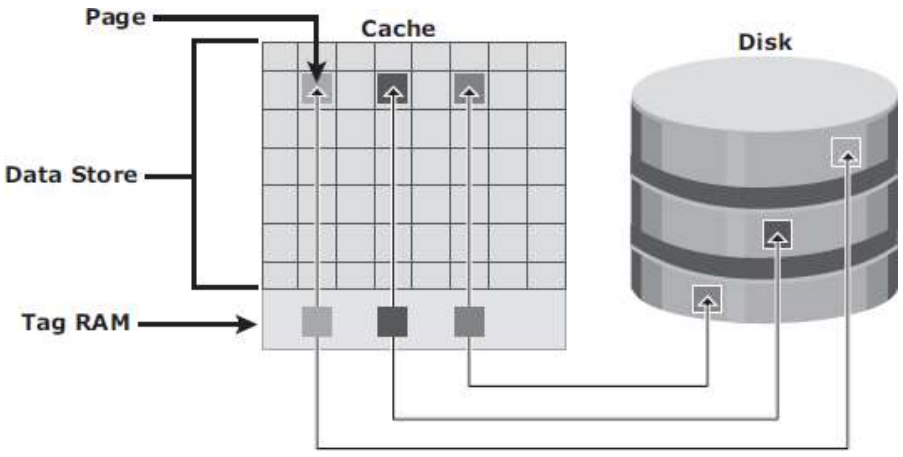



**Figure 4-2:** Structure of cache

Entries in tag RAM indicate where data is found in cache and where the data belongs on the disk. Tag RAM includes a *dirty bit* flag, which indicates whether the data in cache has been committed to the disk. It also contains time-based information, such as the time of last access, which is used to identify cached information that has not been accessed for a long period and may be freed up.

***Read Operation with Cache***

When a host issues a read request, the storage controller reads the tag RAM to determine whether the required data is available in cache. If the requested data is found in the cache, it is called a *read cache hit* or *read hit* and data is sent directly to the host, without any disk operation. This provides a fast response time to the host (about a millisecond). If the requested data is not found in cache, it is called a *cache miss* and the data must be read from the disk. The back end accesses the appropriate disk and retrieves the requested data. Data is then placed in cache and finally sent to the host through the front end. Cache misses increase the I/O response time.

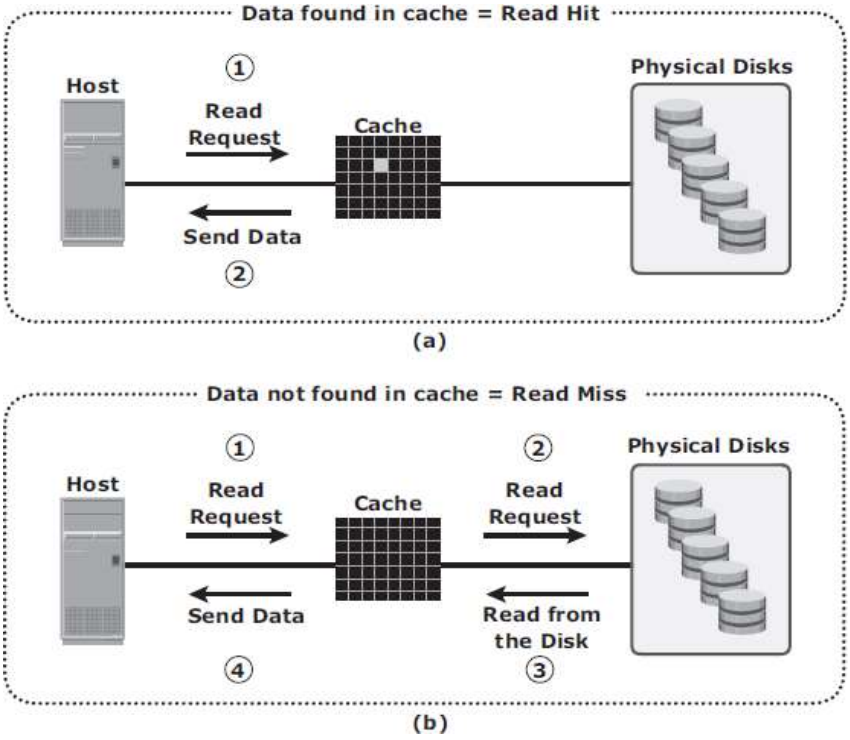



**Figure 4-3:** Read hit and read miss

A *prefetch* or *read-ahead* algorithm is used when read requests are sequential. In a sequential read request, a contiguous set of associated blocks is retrieved. Several other blocks that have not yet been requested by the host can be read from the disk and placed into cache in advance. When the host subsequently requests these blocks, the read operations will be read hits. This process significantly improves the response time experienced by the host. The intelligent storage system offers fixed and variable prefetch sizes. In *fixed prefetch*, the intelligent storage system prefetches a fixed amount of data. It is most suitable when host I/O sizes are uniform. In *variable prefetch*, the storage system prefetches an amount of data in multiples of the size of the host request. *Maximum prefetch* limits the number of data blocks that can be prefetched to prevent the disks from being rendered busy with prefetch at the expense of other I/Os.

Read performance is measured in terms of the *read hit ratio*, or the *hit rate*, usually expressed as a percentage. This ratio is the number of read hits with respect to the total number of read requests. A higher read hit ratio improves the read performance.

***Write Operation with Cache***

Write operations with cache provide performance advantages over writing directly to disks. When an I/O is written to cache and acknowledged, it is completed in far less time (from the host's perspective) than it would take to write directly to disk. Sequential writes also offer opportunities for optimization because many smaller writes can be coalesced for larger transfers to disk drives with the use of cache.

A write operation with cache is implemented in the following ways:

- **Write-back cache:** Data is placed in cache and an acknowledgment is sent to the host immediately. Later, data from several writes are committed (de-staged) to the disk. Write response times are much faster because the write operations are isolated from the mechanical delays of the disk. However, uncommitted data is at risk of loss if cache failures occur.
- **Write-through cache:** Data is placed in the cache and immediately written to the disk, and an acknowledgment is sent to the host. Because data is committed to disk as it arrives, the risks of data loss are low, but the write-response time is longer because of the disk operations.

Cache can be bypassed under certain conditions, such as large size write I/O. In this implementation, if the size of an I/O request exceeds the predefined size, called *write aside size*, writes are sent to the disk directly to reduce the impact of large writes consuming a large cache space. This is particularly useful in an environment where cache resources are constrained and cache is required for small random I/Os.

***Cache Implementation***

Cache can be implemented as either dedicated cache or global cache. With dedicated cache, separate sets of memory locations are reserved for reads and writes. In global cache, both reads and writes can use any of the available memory addresses. Cache management is more efficient in a global cache implementation because only one global set of addresses has to be managed.

Global cache allows users to specify the percentages of cache available for reads and writes for cache management. Typically, the read cache is small, but it should be increased if the application being used is read-intensive. In other global cache implementations, the ratio of cache available for reads versus writes is dynamically adjusted based on the workloads.

***Cache Management***

Cache is a finite and expensive resource that needs proper management. Even though modern intelligent storage systems come with a large amount of cache, when all cache pages are filled, some pages have to be freed up to accommodate new data and avoid performance degradation. Various cache management algorithms are implemented in intelligent storage systems to proactively maintain a set of free pages and a list of pages that can be potentially freed up whenever required. The most commonly used algorithms are discussed in the following list:

- **Least Recently Used (LRU):** An algorithm that continuously monitors data access in cache and identifies the cache pages that have not been accessed for a long time. LRU either frees up these pages or marks them for reuse. This algorithm is based on the assumption that data that has not been accessed for a while will not be requested by the host. However, if a page contains write data that

has not yet been committed to disk, the data is first written to disk before the page is reused.

- **Most Recently Used (MRU):** This algorithm is the opposite of LRU, where the pages that have been accessed most recently are freed up or marked for reuse. This algorithm is based on the assumption that recently accessed data may not be required for a while.

As cache fills, the storage system must take action to flush dirty pages (data written into the cache but not yet written to the disk) to manage space availability. *Flushing* is the process that commits data from cache to the disk. On the basis of the I/O access rate and pattern, high and low levels called *watermarks* are set in cache to manage the flushing process. *High watermark* (*HWM*) is the cache utilization level at which the storage system starts high-speed flushing of cache data. *Low watermark* (*LWM*) is the point at which the storage system stops flushing data to the disks. The cache utilization level, as shown in Figure 4-4, drives the mode of flushing to be used:

- **Idle flushing:** Occurs continuously, at a modest rate, when the cache utilization level is between the high and low watermark.
- **High watermark flushing:** Activated when cache utilization hits the high watermark. The storage system dedicates some additional resources for flushing. This type of flushing has some impact on I/O processing.
- **Forced flushing:** Occurs in the event of a large I/O burst when cache reaches 100 percent of its capacity, which significantly affects the I/O response time. In forced flushing, system flushes the cache on priority by allocating more resources.

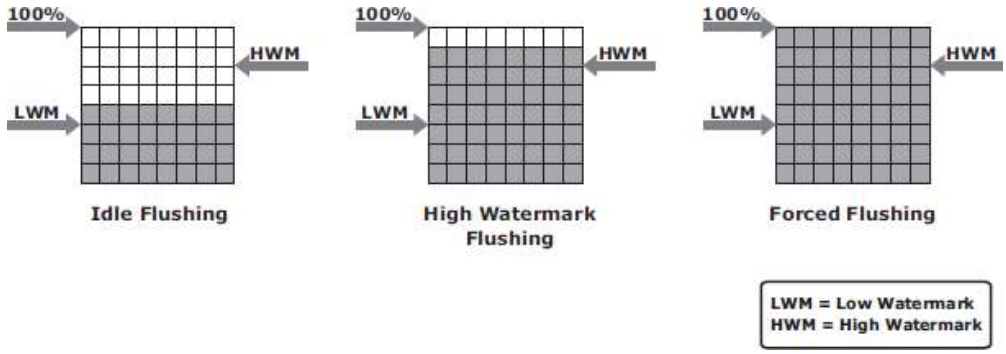



**Figure 4-4:** Types of flushing

***Cache Data Protection***

Cache is volatile memory, so a power failure or any kind of cache failure will cause loss of the data that is not yet committed to the disk. This risk of losing uncommitted data held in cache can be mitigated using *cache mirroring* and *cache vaulting*:

- **Cache mirroring:** Each write to cache is held in two different memory locations on two independent memory cards. If a cache failure occurs, the write data will still be safe in the mirrored location and can be committed to the disk. Reads are staged from the disk to the cache; therefore, if a cache failure occurs, the data can still be accessed from the disk. Because only writes are mirrored, this method results in better utilization of the available cache. In cache mirroring approaches, the problem of maintaining *cache coherency* is introduced. Cache coherency means that data in two

different cache locations must be identical at all times. It is the responsibility of the array operating environment to ensure coherency.

- **Cache vaulting:** The risk of data loss due to power failure can be addressed in various ways: powering the memory with a battery until the AC power is restored or using battery power to write the cache content to the disk. If an extended power failure occurs, using batteries is not a viable option. This is because in intelligent storage systems, large amounts of data might need to be committed to numerous disks, and batteries might not provide power for sufficient time to write each piece of data to its intended disk. Therefore, storage vendors use a set of physical disks to dump the contents of cache during power failure. This is called *cache vaulting* and the disks are called *vault drives*. When power is restored, data from these disks is written back to write cache andthen written to the intended disks.

Server flash-caching technology uses intelligent caching software and a PCI Express (PCIe) flash card on the host. This dramatically improves application performance by reducing latency, and accelerates throughput. Server flash caching technology works in both physical and virtual environments and provides performance acceleration for read-intensive workloads. This technology uses minimal CPU and memory resources from the server by offloading flash management onto the PCIe card. It intelligently determines which data would benefit by sitting in the server on PCIe flash and closer to the application. This avoids the latencies associated with I/O access over the network to the storage array. With this, the processing power required for an application's most frequently referenced data is offloaded from the back-end storage to the PCIe card. Therefore, the storage array can allocate greater processing power to other applications.

**c) Back End**

The *back end* provides an interface between cache and the physical disks. It consists of two components: back-end ports and back-end controllers. The back-end controls data transfers between cache and the physical disks. From cache, data is sent to the back end and then routed to the destination disk.

Physical disks are connected to ports on the back end. The back-end controller communicates with the disks when performing reads and writes and also provides additional, but limited, temporary data storage. The algorithms implemented on back-end controllers provide error detection and correction, along with RAID functionality. For high data protection and high availability, storage systems are configured with dual controllers with multiple ports. Such configurations provide an alternative path to physical disks if a controller or port failure occurs. This reliability is further enhanced if the disks are also dual-ported. In that case, each disk port can connect to a separate controller. Multiple controllers also facilitate load balancing.

**d) Physical Disk**

Physical disks are connected to the back-end storage controller and provide persistent data storage. Modern intelligent storage systems provide support to a variety of disk drives with different speeds and types, such as FC, SATA, SAS, and flash drives. They also support the use of a mix of flash, FC, or SATA within the same array.

## 2. Storage Provisioning

Storage provisioning is the process of assigning storage resources to hosts based on capacity, availability, and performance requirements of applications running on the hosts. Storage provisioning can be performed in two ways: traditional and virtual.

Virtual provisioning leverages virtualization technology for provisioning storage for applications.

### Traditional Storage Provisioning

In traditional storage provisioning, physical disks are logically grouped together and a required RAID level is applied to form a set, called a RAID set. The number of drives in the RAID set and the RAID level determine the availability, capacity, and performance of the RAID set. It is highly recommend that the RAID set be created from drives of the same type, speed, and capacity to ensure maximum usable capacity, reliability, and consistency in performance. For example, if drives of different capacities are mixed in a RAID set, the capacity of the smallest drive is used from each disk in the set to make up the RAID set's overall capacity. The remaining capacity of the larger drives remains unused. Likewise, mixing higher revolutions per minute (RPM) drives with lower RPM drives lowers the overall performance of the RAID set.

RAID sets usually have a large capacity because they combine the total capacity of individual drives in the set. Logical units are created from the RAID sets by partitioning (seen as slices of the RAID set) the available capacity into smaller units.

These units are then assigned to the host based on their storage requirements. Logical units are spread across all the physical disks that belong to that set. Each logical unit created from the RAID set is assigned a unique ID, called a logical unit number (LUN). LUNs hide the organization and composition of the RAID set from the hosts. LUNs created by traditional storage provisioning methods are also referred to as thick LUNs to distinguish them from the LUNs created by virtual provisioning methods.

Figure 4-5 shows a RAID set consisting of five disks that have been sliced, or partitioned, into two LUNs: LUN 0 and LUN 1. These LUNs are then assigned to Host1 and Host 2 for their storage requirements.

When a LUN is configured and assigned to a non-virtualized host, a bus scan is required to identify the LUN. This LUN appears as a raw disk to the operating system. To make this disk usable, it is formatted with a fi le system and then the file system is mounted.

In a virtualized host environment, the LUN is assigned to the hypervisor, which recognizes it as a raw disk. This disk is configured with the hypervisor file system, and then virtual disks are created on it. Virtual disks are fi les on the hypervisor file system. The virtual disks are then assigned to virtual machines and appear as raw disks to them. To make the virtual disk usable to the virtual machine, similar steps are followed as in a non-virtualized environment. Here, the LUN space may be shared and accessed simultaneously by multiple virtual machines.

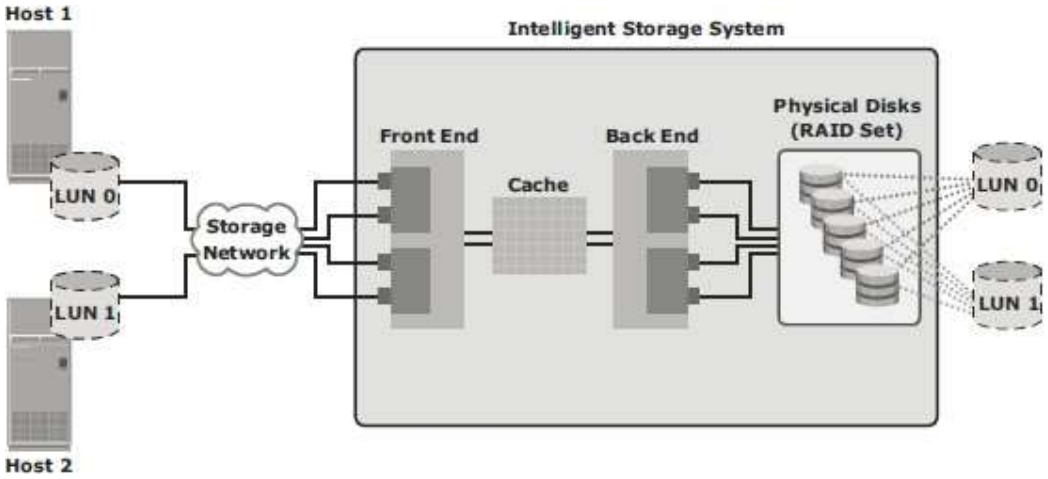



**Figure 4-5: RAID set and LUNs**

Virtual machines can also access a LUN directly on the storage system. In this method the entire LUN is allocated to a single virtual machine. Storing data in this way is recommended when the applications running on the virtual machine are response-time sensitive, and sharing storage with other virtual machines may impact their response time. The direct access method is also used when a virtual machine is clustered with a physical machine. In this case, the virtual machine is required to access the LUN that is being accessed by the physical machine.

LUN Expansion: MetaLUN

MetaLUN is a method to expand LUNs that require additional capacity or performance. A metaLUN can be created by combining two or more LUNs. A metaLUN consists of a base LUN and one or more component LUNs. MetaLUNs can be either concatenated or striped.

Concatenated expansion simply adds additional capacity to the base LUN. In this expansion, the component LUNs are not required to be of the same capacity as the base LUN. All LUNs in a concatenated metaLUN must be either protected (parity or mirrored) or unprotected (RAID 0). RAID types within a metaLUN can be mixed. For example, a RAID 1/0 LUN can be concatenated with a RAID 5 LUN.

However, a RAID 0 LUN can be concatenated only with another RAID 0 LUN. Concatenated expansion is quick but does not provide any performance benefit. (See Figure 4-6.)

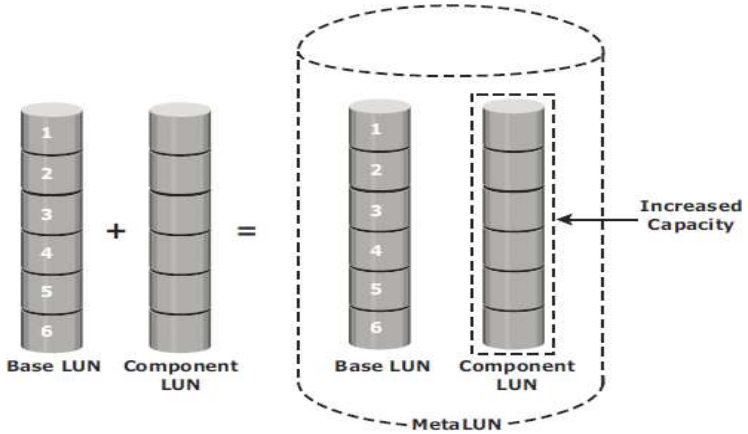



**Figure 4-6: Concatenated metaLUN**

Striped expansion restripes the base LUN's data across the base LUN and component LUNs. In striped expansion, all LUNs must be of the same capacity and RAID level. Striped expansion provides improved performance due to the increased number of drives being striped (see Figure 4-7).

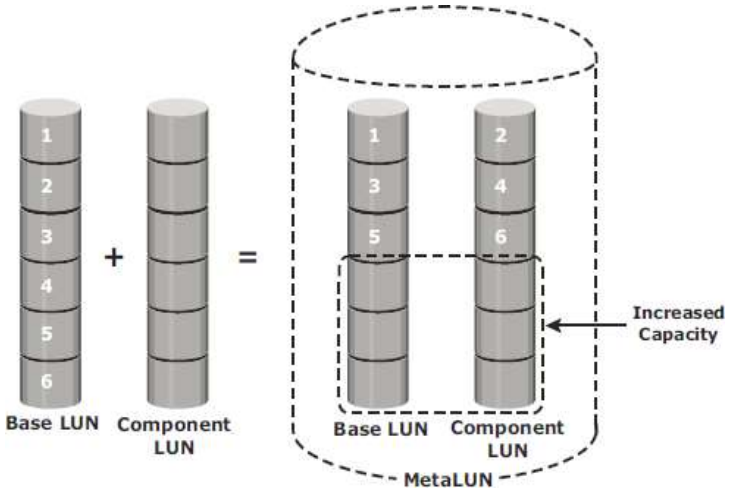



**Figure 4-7: Striped metaLUN**

All LUNs in both concatenated and striped expansion must reside on the same disk-drive type: either all Fibre Channel or all ATA.

**Virtual Storage Provisioning**

Virtual provisioning enables creating and presenting a LUN with more capacity than is physically allocated to it on the storage array. The LUN created using virtual provisioning is called a thin LUN to distinguish it from the traditional LUN.

Thin LUNs do not require physical storage to be completely allocated to them at the time they are created and presented to a host. Physical storage is allocated to the host "on-demand" from a shared pool of physical capacity. A shared pool consists of physical disks. A shared pool in virtual provisioning is analogous to a RAID group, which is a collection of drives on which LUNs are created. Similar to a RAID group, a shared pool supports a single RAID protection level. However, unlike a RAID group, a shared pool might contain large numbers of drives. Shared pools can be homogeneous (containing a single drive type) or heterogeneous (containing mixed drive types, such as flash, FC, SAS, and SATA drives).

Virtual provisioning enables more efficient allocation of storage to hosts. Virtual provisioning also enables oversubscription, where more capacity is presented to the hosts than is actually available on the storage array. Both shared pool and thin LUN can be expanded nondisruptively as the storage requirements of the hosts grow. Multiple shared pools can be created within a storage array, and a shared pool may be shared by multiple thin LUNs. Figure 4-8 illustrates the provisioning of thin LUNs.

**Comparison between Virtual and Traditional Storage Provisioning**

Administrators typically allocate storage capacity based on anticipated storage requirements. This generally results in the over provisioning of storage capacity, which then leads to higher costs and lower capacity utilization. Administrators often over-provision storage to an application for various reasons, such as, to avoid frequent provisioning of storage if the LUN capacity is exhausted, and to reduce disruption to application availability. Over provisioning of storage often leads to additional storage acquisition and operational costs.

Virtual provisioning addresses these challenges. Virtual provisioning improves storage capacity utilization and simplifies storage management. Figure 4-9 shows an example, comparing virtual provisioning with traditional storage provisioning.

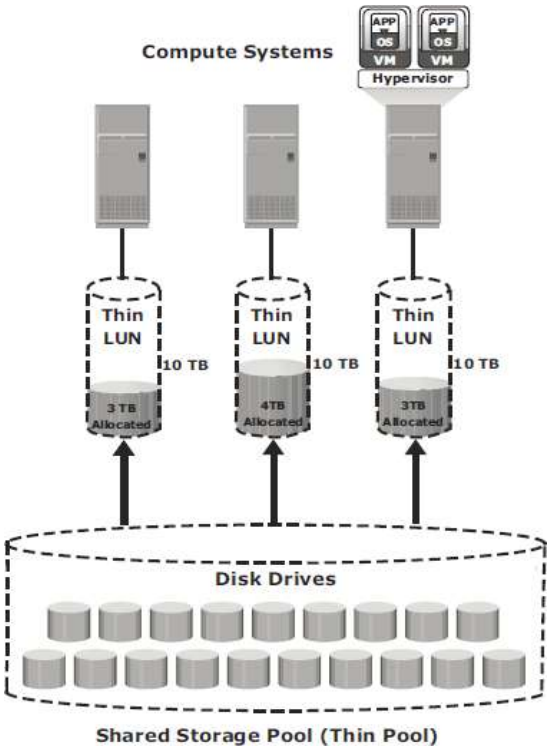



**Figure 4-8: Virtual provisioning**

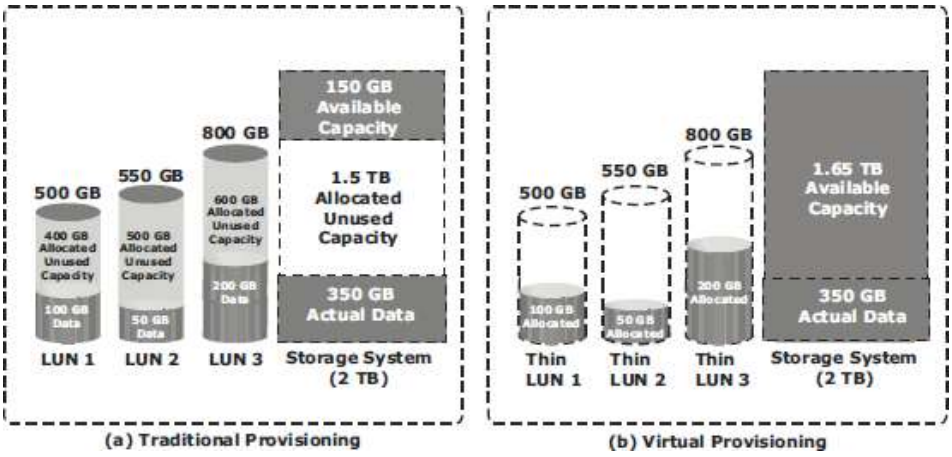



**Figure 4-9: Traditional versus virtual provisioning**

With traditional provisioning, three LUNs are created and presented to one or more hosts (see Figure 4-9 [a]). The total storage capacity of the storage system is 2 TB. The allocated capacity of LUN 1 is 500 GB, of which only 100 GB is consumed, and the remaining 400 GB is unused. The size of LUN 2 is 550 GB, of which 50 GB is consumed, and 500 GB is unused. The size of LUN 3 is 800 GB, of which 200 GB is consumed, and 600 GB is unused. In total, the storage system has 350 GB of data, 1.5 TB of allocated but unused capacity, and only 150 GB of remaining capacity available for other applications.

Now consider the same 2 TB storage system with virtual provisioning (see Figure 4-9 [b]). Here, three thin LUNs of the same sizes are created. However, there is no allocated unused capacity. In total, the storage system with virtual provisioning has the same 350 GB of data, but 1.65 TB of capacity is available for other applications, whereas only 150 GB is available in traditional storage provisioning. Use Cases for Thin and Traditional LUNs Virtual provisioning and thin LUN offer many benefits, although in some cases traditional LUN is better suited for an application. Thin LUNs are appropriate for applications that can tolerate performance variations. In some cases, performance improvement is perceived when using a thin LUN, due to striping across a large number of drives in the pool. However, when multiple thin LUNs contend for shared storage resources in a given pool, and when utilization reaches higher levels, the performance can degrade. Thin LUNs provide the best storage space efficiency and are suitable for applications where space consumption is difficult to forecast. Using thin LUNs benefits organizations in reducing power and acquisition costs and in simplifying their storage management.

Traditional LUNs are suited for applications that require predictable performance. Traditional LUNs provide full control for precise data placement and allow an administrator to create LUNs on different RAID groups if there is any workload contention. Organizations that are not highly concerned about storage space efficiency may still use traditional LUNs.

Both traditional and thin LUNs can coexist in the same storage array. Based on the requirement, an administrator may migrate data between thin and traditional LUNs.

**LUN Masking**

LUN masking is a process that provides data access control by defining which LUNs a host can access. The LUN masking function is implemented on the storage array. This ensures that volume access by hosts is controlled appropriately, preventing unauthorized or accidental use in a shared environment.

For example, consider a storage array with two LUNs that store data of the sales and finance departments. Without LUN masking, both departments can easily see and modify each other's data, posing a high risk to data integrity and security. With LUN masking, LUNs are accessible only to the designated hosts.

## 3. Types of Intelligent Storage Systems

Intelligent storage systems generally fall into one of the following two categories:

- High-end storage systems
- Midrange storage systems

Traditionally, high-end storage systems have been implemented with *active active configuration*, whereas

midrange storage systems have been implemented with *active-passive configuration*. The distinctions between these two implementations are becoming increasingly insignificant.

**High-End Storage Systems**

High-end storage systems, referred to as *active-active arrays*, are generally aimed at large enterprise applications. These systems are designed with a large number of controllers and cache memory. An active-active array implies that the host can perform I/Os to its LUNs through any of the available controllers.

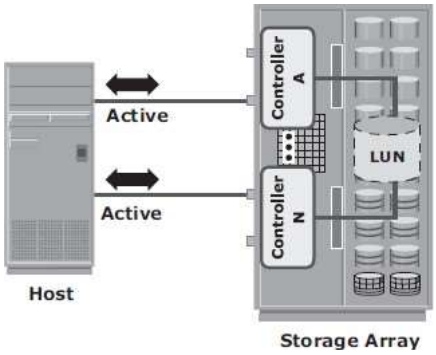



**Figure 4-10:** Active-active configuration

To address enterprise storage needs, these arrays provide the following capabilities:

- Large storage capacity
- Large amounts of cache to service host I/Os optimally
- Fault tolerance architecture to improve data availability
- Connectivity to mainframe computers and open systems hosts
- Availability of multiple front-end ports and interface protocols to serve a large number of hosts
- Availability of multiple back-end controllers to manage disk processing
- Scalability to support increased connectivity, performance, and storage capacity requirements
- Ability to handle large amounts of concurrent I/Os from a number of hosts and applications
- Support for array-based local and remote data replication

In addition to these features, high-end systems possess some unique features that are required for mission-critical applications.

**Midrange Storage Systems**

Midrange storage systems are also referred to as *active-passive arrays* and are best suited for small- and medium-sized enterprise applications. They also provide optimal storage solutions at a lower cost. In an active-passive array, a host can perform I/Os to a LUN only through the controller that owns the LUN. As shown in Figure 4-11, the host can perform reads or writes to the LUN only through the path to controller A because controller A is the owner of that LUN. The path to controller B remains passive and no I/O activity is performed through this path.

Midrange storage systems are typically designed with two controllers, each of which contains host interfaces, cache, RAID controllers, and interface to disk drives.

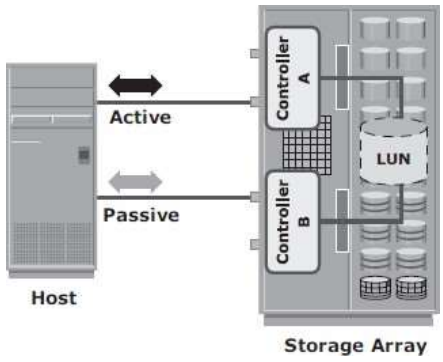



**Figure 4-11:** Active-passive configuration

Midrange arrays are designed to meet the requirements of small and medium enterprise applications; therefore, they host less storage capacity and cache than high-end storage arrays. There are also fewer front-end ports for connection to hosts. However, they ensure high redundancy and high performance for applications with predictable workloads. They also support array-based local and remote replication.

## Fibre Channel Storage Area Networks

Organizations are experiencing an explosive growth in information. This information needs to be stored, protected, optimized, and managed efficiently. Data center managers are burdened with the challenging task of providing low-cost, high-performance information management solutions. An effective information management solution must provide the following:

- **Just-in-time information to business users:** Information must be available to business users when they need it. 24 x 7 data availability is becoming one of the key requirements of today's storage infrastructure. The explosive growth in storage, proliferation of new servers and applications, and the spread of mission-critical data throughout enterprises are some of the challenges that need to be addressed to provide information availability in real time.
- **Integration of information infrastructure with business processes:** The storage infrastructure should be integrated with various business processes without compromising its security and integrity.
- **Flexible and resilient storage infrastructure:** The storage infrastructure must provide flexibility and resilience that aligns with changing business requirements. Storage should scale without compromising the performance requirements of applications and, at the same time, the total cost of managing information must be low.

*Direct-attached storage* (DAS) is often referred to as a stovepiped storage environment. Hosts "own" the storage, and it is difficult to manage and share resources on these isolated storage devices. Efforts to organize this dispersed data led to the emergence of the *storage area network* (SAN). SAN is a high-speed, dedicated network of servers and shared storage devices. A SAN provides storage consolidation and facilitates centralized data management. It meets the storage demands efficiently with better economies of scale and also provides effective maintenance and protection of data. Virtualized SAN and block storage

virtualization provide enhanced utilization and collaboration among dispersed storage resources. The implementation of virtualization in SAN provides improved productivity, resource utilization, and manageability. Common SAN deployments are Fibre Channel (FC) SAN and IP SAN. Fibre Channel SAN uses Fibre Channel protocol for the transport of data, commands, and status information between servers (or hosts) and storage devices. IP SAN uses IP-based protocols for communication.

## 1. Fibre Channel: Overview

The FC architecture forms the fundamental construct of the FC SAN infrastructure. *Fibre Channel* is a high-speed network technology that runs on high-speed optical fiber cables and serial copper cables. The FC technology was developed to meet the demand for increased speeds of data transfer between servers and mass storage systems. Although FC networking was introduced in 1988, the FC standardization process began when the American National Standards Institute (ANSI) chartered the Fibre Channel Working Group (FCWG). By 1994, the new high-speed computer interconnection standard was developed and the Fibre Channel Association (FCA) was founded with 70 charter member companies. Technical Committee T11, which is the committee within International Committee for Information Technology Standards (INCITS), is responsible for Fibre Channel interface standards.

High data transmission speed is an important feature of the FC networking technology. The initial implementation offered a throughput of 200 MB/s (equivalent to a raw bit rate of 1Gb/s), which was greater than the speeds of Ultra SCSI (20 MB/s), commonly used in DAS environments. In comparison with Ultra SCSI, FC is a significant leap in storage networking technology. The latest FC implementations of 16 GFC (Fibre Channel) offer a throughput of 3200 MB/s (raw bit rates of 16 Gb/s), whereas Ultra640 SCSI is available with a throughput of 640 MB/s. The FC architecture is highly scalable, and theoretically, a single FC network can accommodate approximately 15 million devices.

## 2. The SAN and Its Evolution

A SAN carries data between servers (or *hosts*) and storage devices through Fibre Channel network. A SAN enables storage consolidation and enables storage to be shared across multiple servers. This improves the utilization of storage resources compared to direct-attached storage architecture and reduces the total amount of storage an organization needs to purchase and manage. With consolidation, storage management becomes centralized and less complex, which further reduces the cost of managing information.

SAN also enables organizations to connect geographically dispersed servers and storage.

In its earliest implementation, the FC SAN was a simple grouping of hosts and storage devices connected to a network using an FC hub as a connectivity device. This configuration of an FC SAN is known as a *Fibre Channel Arbitrated Loop* (FC-AL). Use of hubs resulted in isolated FC-AL SAN islands because hubs provide limited connectivity and bandwidth. The inherent limitations associated with hubs gave way to high-performance FC *switches*. Use of switches in SAN improved connectivity and performance and enabled FC SANs to be highly scalable. This enhanced data accessibility to applications across the enterprise. Now, FC-AL has been almost abandoned for FC SANs due to its limitations but still survives

as a back-end connectivity option to disk drives. Figure 5-2 illustrates the FC SAN evolution from FC-AL to enterprise SANs.

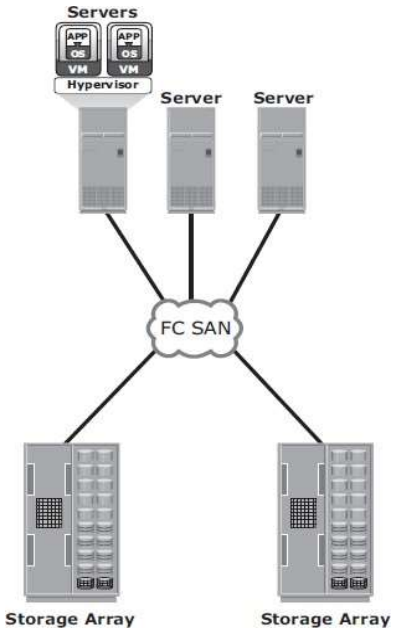
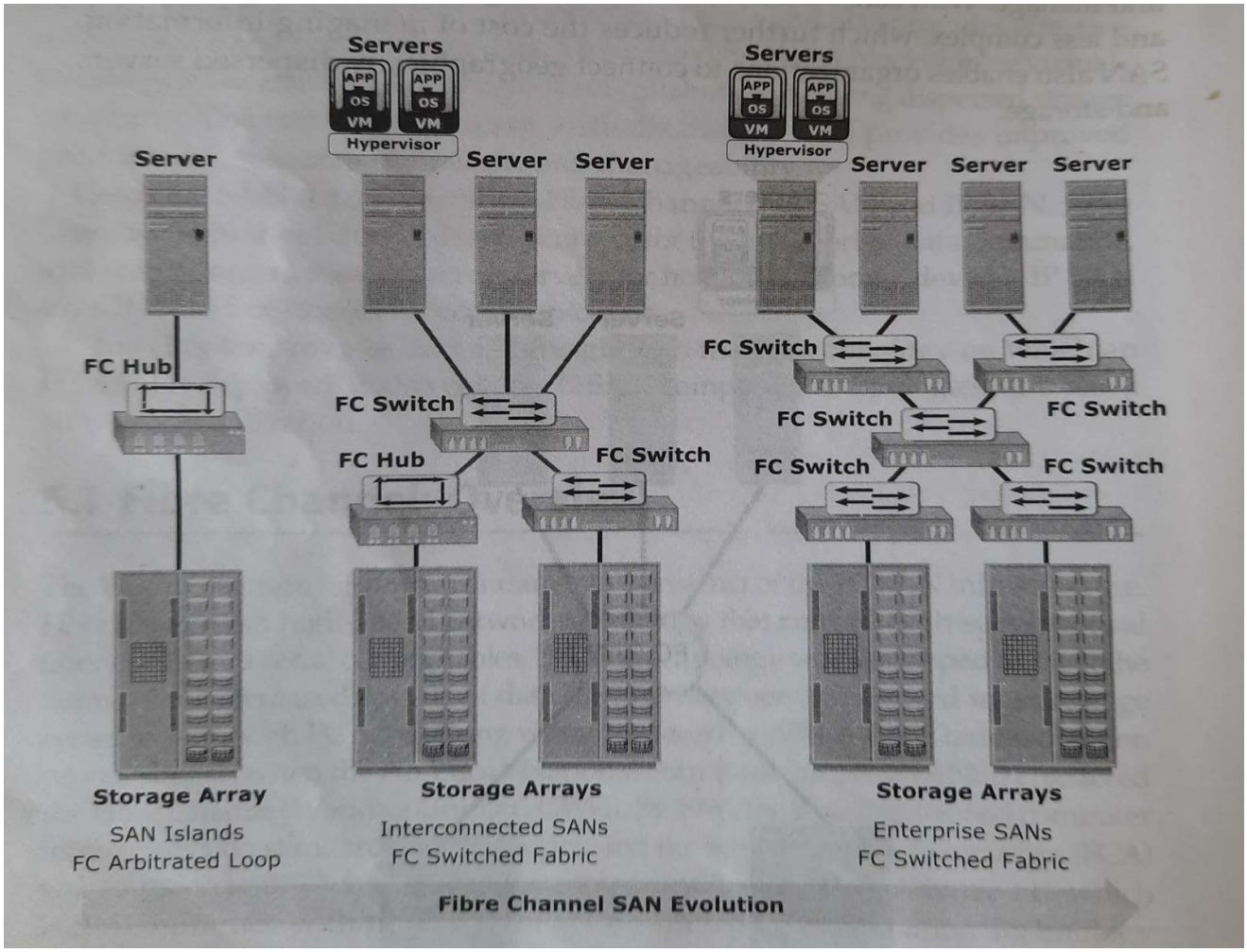



**Figure 5-1:** FC SAN implementation



**Node Ports**

In a Fibre Channel network, the end devices, such as hosts, storage arrays, and tape libraries, are all referred to as *nodes*. Each node is a source or destination of information. Each node requires one or more ports to provide a physical interface for communicating with other nodes. These ports are integral components of host adapters, such as HBA, and storage front-end controllers or adapters. In an FC environment a port operates in full-duplex data transmission mode with a *transmit* (Tx) link and a *receive* (Rx).

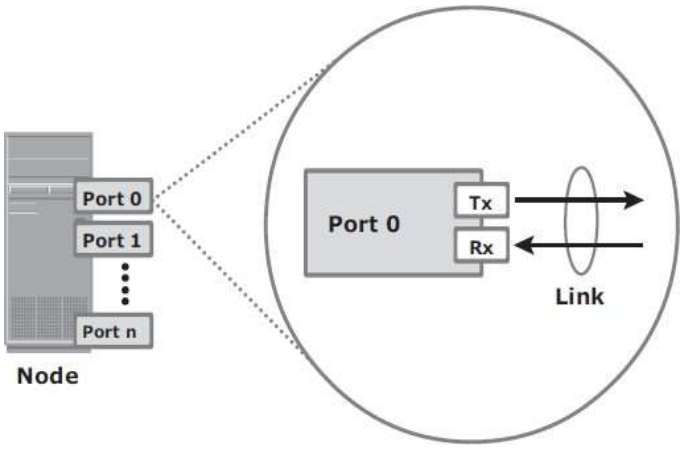



**Figure 5-3:** Nodes, ports, and links

**Cables and Connectors**

SAN implementations use optical fiber cabling. Copper can be used for shorter distances for back-end connectivity because it provides an acceptable signal-to noise ratio for distances up to 30 meters. Optical fiber cables carry data in the form of light. There are two types of optical cables: multimode and single-mode.

*Multimode fiber* (MMF) cable carries multiple beams of light projected at different angles simultaneously onto the core of the cable. Based on the bandwidth, multimode fibers are classified as OM1 (62.5µm core), OM2 (50µm core), and laser-optimized OM3 (50µm core). In an MMF transmission, multiple light beams traveling inside the cable tend to disperse and collide. This collision weakens the signal strength after it travels a certain distance — a process known as *modal dispersion*. An MMF cable is typically used for short distances because of signal degradation (attenuation) due to modal dispersion.

*Single-mode fiber* (SMF) carries a single ray of light projected at the center of the core. These cables are available in core diameters of 7 to 11 microns; the most common size is 9 microns. In an SMF transmission, a single light beam travels in a straight line through the core of the fiber. The small core and the single light wave help to limit modal dispersion. Among all types of fiber cables, singlemode provides minimum signal attenuation over maximum distance (up to 10 km). A single-mode cable is used for long-distance cable runs, and distance usually depends on the power of the laser at the transmitter and sensitivity of the receiver.

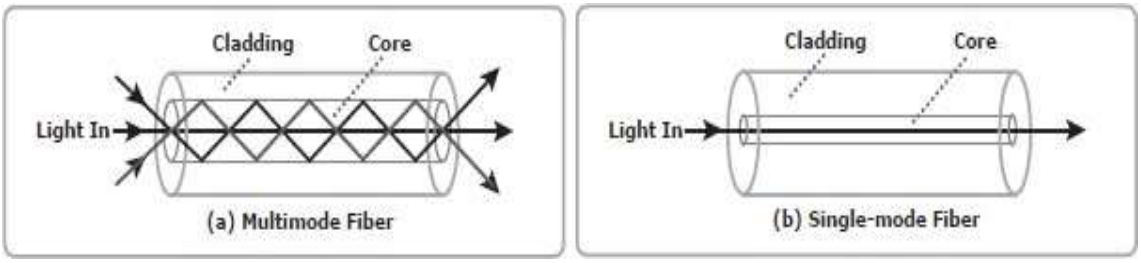



**Figure 5-4:** Multimode fiber and single-mode fiber

MMFs are generally used within data centers for shorter distance runs, whereas SMFs are used for longer distances. A connector is attached at the end of a cable to enable swift connection and disconnection of the cable to and from a port. A *Standard connector* (SC) and a *Lucent connector* (LC) are two commonly used connectors for fiber optic cables. *Straight Tip* (ST) is another fiber-optic connector, which is often used with fiber patch panels.

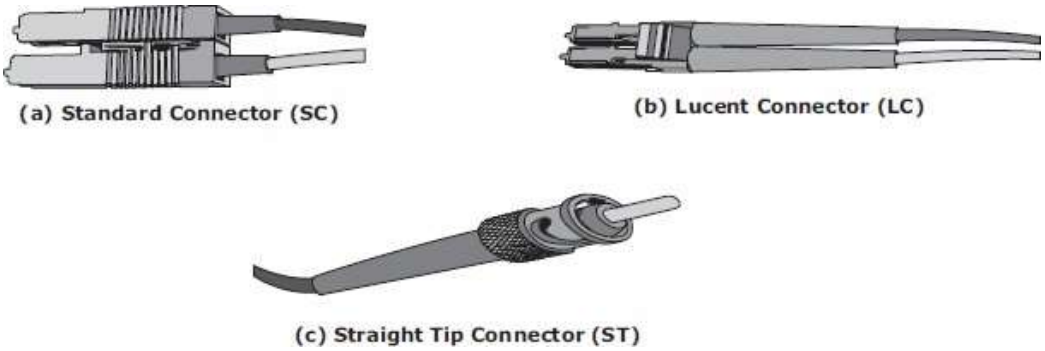



**Figure 5-5:** SC, LC, and ST connectors

**Interconnect Devices**

FC hubs, switches, and directors are the interconnect devices commonly used in FC SAN.

*Hubs* are used as communication devices in FC-AL implementations. Hubs physically connect nodes in a logical loop or a physical star topology. All the nodes must share the loop because data travels through all the connection points. Because of the availability of low-cost and high-performance switches, hubs are no longer used in FC SANs.

*Switches* are more intelligent than hubs and directly route data from one physical port to another. Therefore, nodes do not share the bandwidth. Instead, each node has a dedicated communication path.

*Directors* are high-end switches with a higher port count and better fault tolerance capabilities.

Switches are available with a fixed port count or with modular design. In a modular switch, the port count is increased by installing additional port cards to open slots. The architecture of a director is always modular, and its port count is increased by inserting additional line cards or blades to the director's chassis. High-end switches and directors contain redundant components to provide high availability. Both switches and directors have management ports (Ethernet or serial) for connectivity to SAN management servers.

A port card or blade has multiple ports for connecting nodes and other FC switches. Typically, a Fibre

Channel transceiver is installed at each port slot that houses the transmit (Tx) and receive (Rx) link. In a transceiver, the Tx and Rx links share common circuitry. Transceivers inside a port card are connected to an application specifi c integrated circuit, also called port ASIC. Blades in a director usually have more than one ASIC for higher throughput.

**SAN Management Software**

SAN management software manages the interfaces between hosts, interconnect devices, and storage arrays. The software provides a view of the SAN environment and enables management of various resources from one central console.

It provides key management functions, including mapping of storage devices, switches, and servers, monitoring and generating alerts for discovered devices, and *zoning.*

Scalability and performance are the primary differences between switches and hubs. Addressing in a switched fabric supports more than 15 million nodes within the fabric, whereas the FC-AL implemented in hubs supports only a maximum of 126 nodes.

Fabric switches provide full bandwidth between multiple pairs of ports in a fabric, resulting in a scalable architecture that supports multiple simultaneous communications.

Hubs support only one communication at a time. They provide a low-cost connectivity expansion solution. Switches, conversely, can be used to build dynamic, high-performance fabrics through which multiple communications can take place simultaneously. Switches are more expensive than hubs.

## Question Bank

1. What is RAID? Explain the RAID levels RAID 0, 1 and Nested.
2. Define RAID. Explain RAID levels 3, 4, 5, 6.
3. With neat diagram explain the structure of read and write operations with cache.
4. Explain Fibre Channel connectivity with FC SAN implementation.
5. Describe with neat diagram the components of Intelligent Storage System.
6. Differentiate between Software and Hardware RAID.
7. Discuss how parity method is used in increasing fault tolerance in RAID levels.
8. Compare virtual and traditional storage provisioning.
9. What are RAID implementation methods?