2015-16 Code: 50538

MALLA REDDY ENGINEERING COLLEGE

(*Autonomous)*

IV Year B.Tech IT-II Sem

L	T	P	C
3	-	-	3

STORAGE AREA NETWORKS
(Professional Elective)

Prerequisite: Basic Network Communication.
Objective: To understand Storage Area Network characteristics and components.

Module I: Introduction
A: Basics of SAN **[09 Periods]**
Introduction to Storage Technology Review data creation and the amount of data being created and understand the value of data to a business, challenges in data storage and data management. Solutions available for data storage, Core elements of a data center infrastructure, role of each element in supporting business activities.

Module II: Architecture
A: Architecture and RAID Levels **[09 Periods]**
Storage Systems Architecture, Hardware and software components of the host environment, Key protocols and concepts used by each component, Physical and logical components of a connectivity environment, Major physical components of a disk drive and their function, logical constructs of a physical disk, access characteristics and performance implications, Concept of RAID and its components, Different RAID levels and their suitability for different application environments: RAID0, RAID1, RAID3, RAID5, RAID0+1, RAID 1+0, RAID 6, Compare and contrast integrated and modular storage systems. High-level architecture and working of an intelligent storage system.

Module III: Topologies
A: Components and Topologies **[09 Periods]**
Introduction to Networked Storage Evolution of networked storage, Architecture, components, and topologies of FC-SAN, NAS and IP-SAN, Benefits of the different networked storage options, Understand the need for long-term archiving solutions and describe how CAS fulfils the need. Understand the appropriateness of the different networked storage options for different application environments.

Module IV: Transactions of Data
A: San Transactions of data **[09 Periods]**
Information Availability & Monitoring & Managing Data center List reasons for planned/unplanned outages and the impact of downtime, Impact of downtime, Differentiate between business continuity (BC) and disaster recovery (DR), RTO and RPO, Identify single points of failure in a storage infrastructure and list solutions to mitigate these failures. Architecture of backup/recovery and the different backup/recovery topologies, replication technologies and their role in ensuring information availability and business continuity. Remote replication technologies and their role in providing disaster recovery and business continuity capabilities, Identify key areas to monitor in a data center, Industry standards for data center monitoring and management, Key metrics to monitor for different components in a storage infrastructure. Key management tasks in a data center.

Module V: Security
A: Realistic solution and Storage **[09 Periods]**

Securing Storage and Storage Virtualization, Information security, Critical security attributes for information systems, Storage security domains, List and analyzes the common threats in each domain.
Virtualization technologies, block-level and file-level virtualization technologies and processes.

Case Studies

The technologies described in the course are reinforced with EMC examples of actual solutions.
Realistic case studies enable the participant to design the most appropriate solution for given sets of criteria.

Text Book:

1. EMC Corporation, Information Storage and Management, Wiley.

References:

1. Robert Spalding, “Storage Networks: The Complete Reference”, Tata McGraw Hill, Osborne, 2003.
2. Marc Farley, “Building Storage Networks”, Tata McGraw Hill Osborne, 2001.
3. Meeta Gupta, Storage Area Network Fundamentals, Pearson Education Limited, 2002.

Outcomes:

On successful completion of this course the student will be able to

1. Describe the aspects of a Storage Area Network.
2. Explain Technologies such as Fiber Channel (FC), iSCSI, Network Attatched Storage (NAS) and Fiber Channel over Ethernet (FCoE)
3. Install, Configure and Manage a Fibre Channel Storage Area Network (FC SAN)
4. Install, Configure and Manage iSCSI Storage Area Network (iSCSI)

Storage Area Networks

Module I: Introduction

A: Basics of SAN

Chapter Objective

This chapter describes the evolution of information storage architecture from simple direct-attached models to complex networked topologies. It introduces the information lifecycle management (ILM) strategy, which aligns the information technology (IT) infrastructure with business priorities.

Introduction to Information Storage and Management, Storage System Environment

Introduction

Information is increasingly important in our daily lives. We have become information dependents of the twenty-first century, living in an 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.

Information created by individuals gains value when shared with others. Figure 1-1 depicts this virtuous cycle of information.

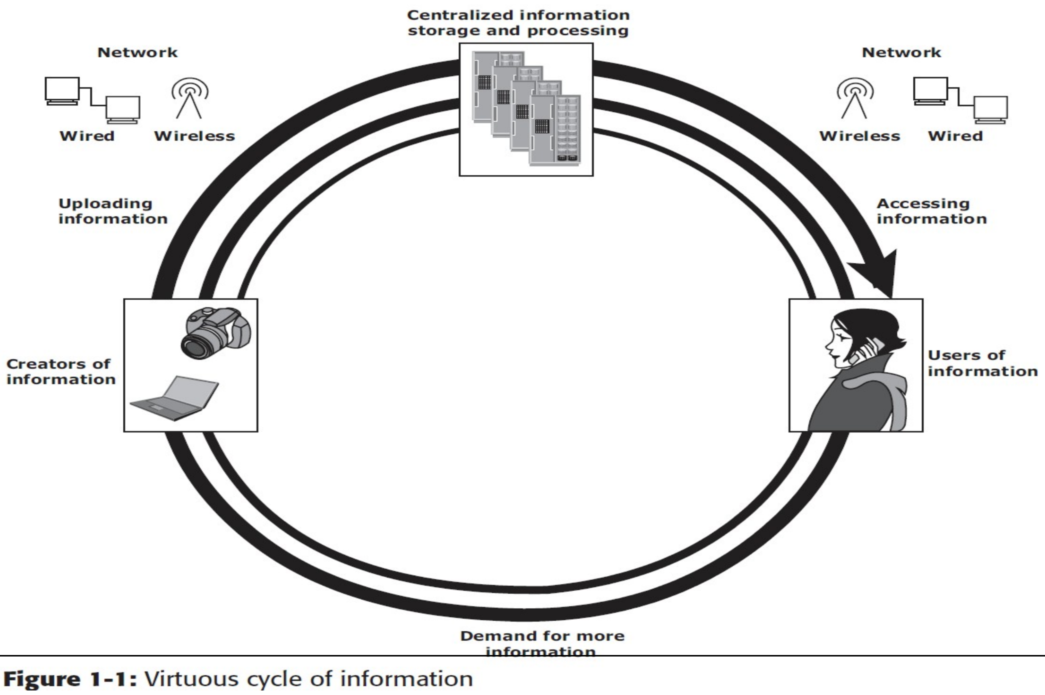


Figure 1-1: Virtuous cycle of information

Information Storage

Businesses use 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.

Data

Data is a collection of raw facts from which conclusions may be drawn. Handwritten letters, a printed book, a family photograph, a movie on video tape, printed and duly signed copies of mortgage papers, a bank's ledgers, and an account holder's passbooks are all examples of data.

The data can be generated using a computer and stored in strings of 0s and 1s, as shown in Figure 1-2. Data in this form is called *digital data* and is accessible by the user only after it is processed by a computer.

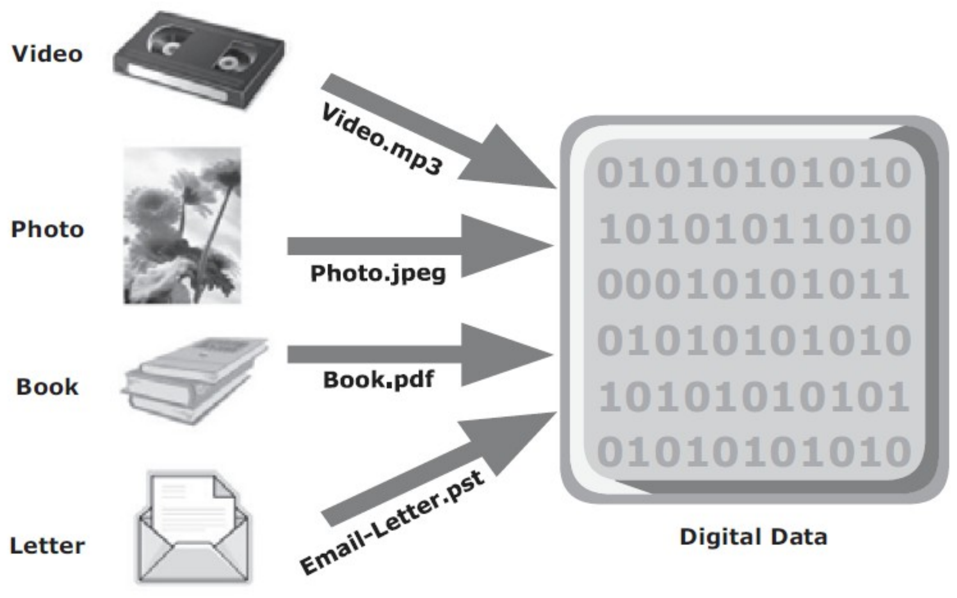


Figure 1-2: Digital data

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.

The importance and the criticality of data vary with time. Most of the data created holds significance in the short-term but becomes less valuable over time. This governs the type of data storage solutions used. Individuals store data on a variety of storage devices, such as hard disks, CDs, DVDs, or Universal Serial Bus (USB) flash drives.

Types of Data

Data can be classified as structured or unstructured (see Figure 1-3) based on how it is stored and managed. 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).
Data is unstructured if its elements cannot be stored in rows and columns, and is therefore difficult to query and retrieve by business applications. For example, customer contacts may be stored in various forms such as sticky notes, e-mail messages, business cards, or even digital format files such as .doc, .txt, and .pdf.

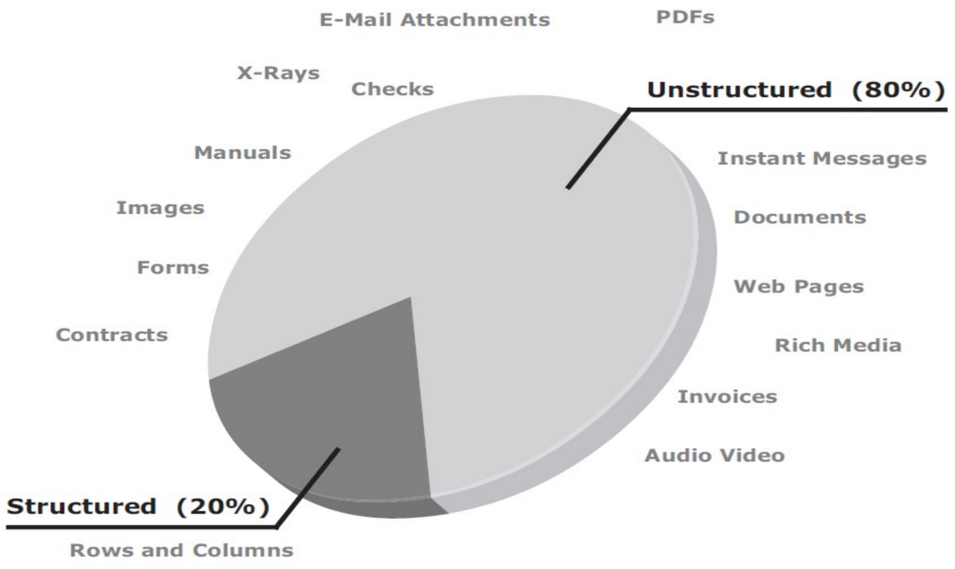


Figure 1-3: Types of data

Information

Information is the intelligence and knowledge derived from data. Data, whether structured or unstructured, does not fulfill any purpose for individuals or businesses unless it is presented in a meaningful form. Businesses need to analyze data for it to be of value.
Effective data analysis not only extends its benefits to existing businesses, but also creates the potential for new business opportunities by using the information in creative ways.
Example: Job portal. In order to reach a wider set of prospective employers, job seekers post their résumés on various websites offering job search facilities. These websites collect the résumés and post them on centrally

accessible locations for prospective employers. In addition, companies post available positions on job search sites. Job-matching software matches keywords from résumés to keywords in job postings. In this manner, the job search engine uses data and turns it into information for employers and job seekers.

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*. Devices such as memory in a cell phone or digital camera, DVDs, CD-ROMs, and hard disks in personal computers are examples.

Evolution of Storage Technology and 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.

Originally, there were very limited policies and processes for managing the servers and the data created. To overcome these challenges, storage technology evolved from non-intelligent internal storage to intelligent networked storage (see Figure 1-4).

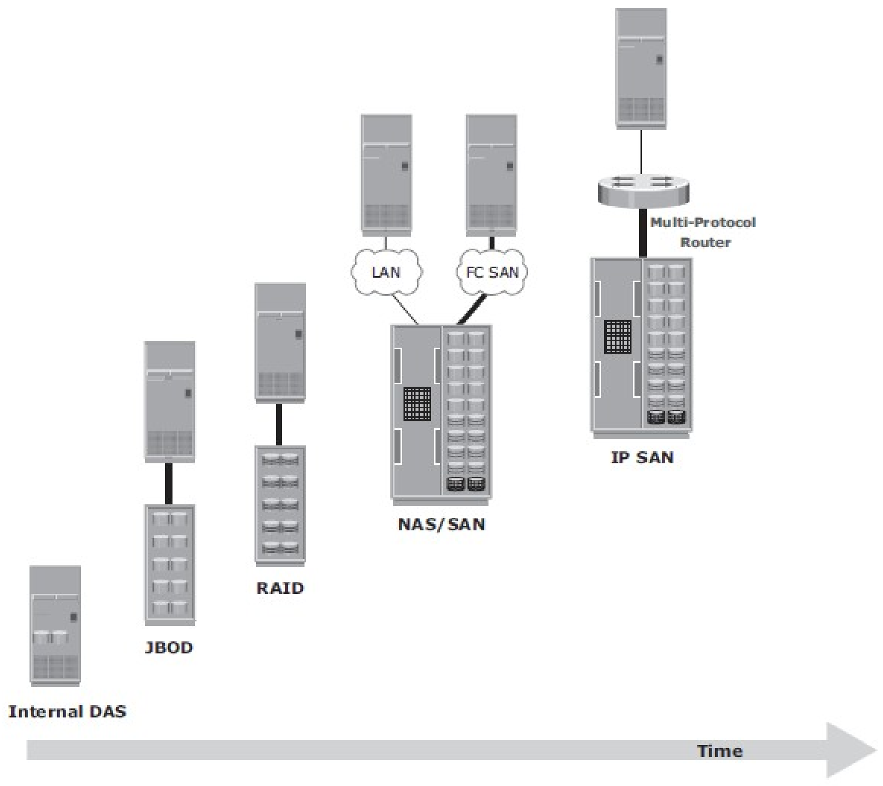


Figure 1-4: Evolution of storage architectures

The technology evolution includes:

1. Redundant Array of Independent Disks (RAID): This technology was developed to address the cost, performance, and availability requirements of data. It continues to evolve today and is used in all storage architectures such as DAS, SAN, and so on.

2. Direct-attached storage (DAS): This type of storage connects directly to a server (host) or a group of servers in a cluster. Storage can be either internal or external to the server. External DAS alleviated the challenges of limited internal storage capacity.

3. Storage area network (SAN): This is a dedicated, high-performance *Fibre Channel (FC)* network to facilitate *block-level* communication between servers and storage. Storage is partitioned and assigned to a server for accessing its data. SAN offers scalability, availability, performance, and cost benefits compared to DAS.

4. Network-attached storage (NAS): This is dedicated storage for *file serving* applications. Unlike a SAN, it connects to an existing communication network (LAN) and provides file access to heterogeneous clients. Because it is purposely built for providing storage to file server applications, it offers higher scalability, availability, performance, and cost benefits compared to general purpose file servers.

5. Internet Protocol SAN (IP-SAN): One of the latest evolutions in storage architecture, IP-SAN is a convergence of technologies used in SAN and NAS. IP-SAN provides block-level communication across a local or wide area network (LAN or WAN), resulting in greater consolidation and availability of data.

Data Center Infrastructure

Organizations maintain data centers to provide centralized data processing capabilities across the enterprise. Data centers store and manage large amounts of mission-critical data.

The data center infrastructure includes 1) computers, 2) storage systems, 3) network devices, 4) dedicated power backups, 5) and environmental controls (such as air conditioning and fire suppression).

Core Elements

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. Applications, such as an order processing system, can be layered on a database, which in turn uses operating system services to perform read/write operations to storage devices.

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. Server and operating system: A computing platform that runs applications and databases.

4. Network: A data path that facilitates communication between clients and servers or between servers and storage.

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

Example: Figure 1-5 shows an order processing system that involves the five core elements of a data center and illustrates their functionality in a business process.

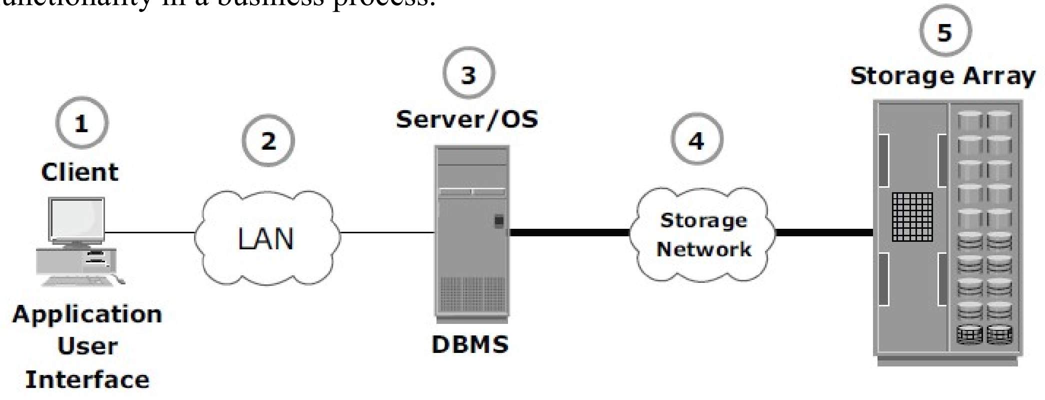


Figure 1-5: Example of an order processing system

Step 1: A customer places an order through the AUI of the order processing application software located on the client computer.

Step 2: The client connects to the server over the LAN and accesses the DBMS located on the server to update the relevant information such as the customer name, address, payment method, products ordered, and quantity ordered.

Step 3: The DBMS uses the server operating system to read and write this data to the database located on physical disks in the storage array.

Step 4: The Storage Network provides the communication link between the server and the storage array and transports the read or write commands between them.

Step 5: The storage array, after receiving the read or write commands from the server, performs the necessary operations to store the data on physical disks.

Key Requirements for Data Center Elements

Uninterrupted operation of data centers is critical to the survival and success of a business. It is necessary to have a reliable infrastructure that ensures data is accessible at all times. While the requirements, shown in Figure 1-6, are applicable to all elements of the data center infrastructure, our focus here is on storage systems.

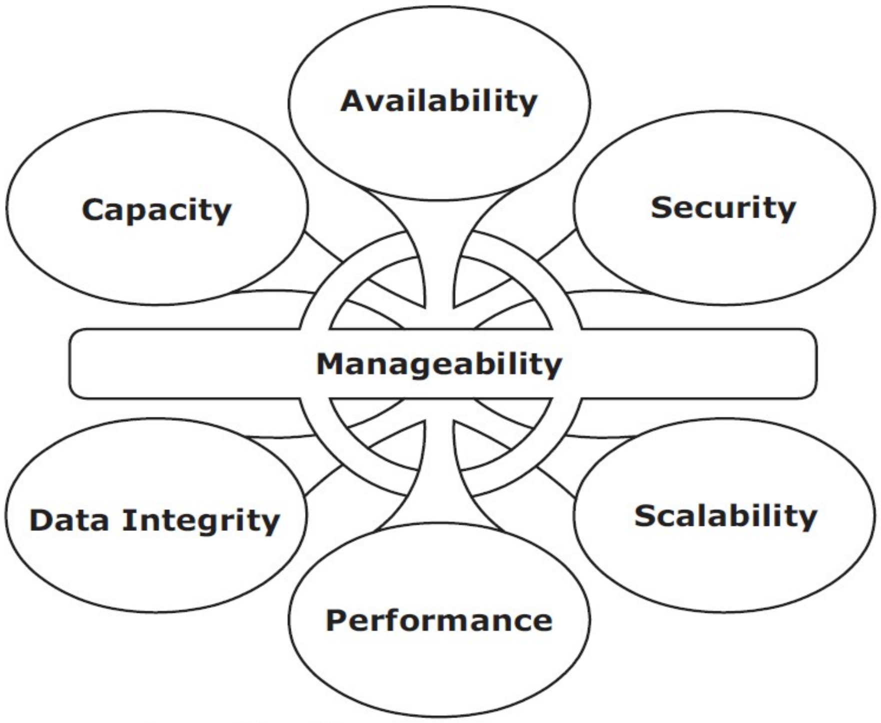


Figure 1-6: Key characteristics of data center elements

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. In addition to the security measures for client access, 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 or storage on demand, without interrupting business operations. Business growth often requires deploying more servers, new applications, and additional databases. 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.

Managing Storage Infrastructure

Managing a modern, complex data center involves many tasks. Key management activities include:

1 ***Monitoring*** is the continuous collection of information and the review of the entire data center infrastructure. The aspects of a data center that are monitored include security, performance, accessibility, and capacity.

2 ***Reporting*** 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*** is the process of providing the hardware, software, and other resources needed to run a data center. Provisioning activities include capacity and resource planning. Capacity planning ensures that the user's and the application's future needs will be addressed in the most cost-effective and controlled manner. Resource planning is the process of evaluating and identifying required resources, such as personnel, the facility (site), and the technology.

Key Challenges in Managing Information

In order to frame an effective information management policy, businesses need to consider the following key challenges of information management:

1 Exploding digital universe: The rate of information growth is increasing exponentially. Duplication of data to ensure high availability and repurposing has also contributed to the multifold increase of information growth.

2 Increasing dependency on information: The strategic use of information plays an important role in determining the success of a business and provides competitive advantages in the marketplace.

3 Changing value of information: Information that is valuable today may become less important tomorrow. The value of information often changes over time.

Information Lifecycle

The ***information lifecycle*** is the ―change in the value of information‖ over time. When data is first created, it often has the highest value and is used frequently. As data ages, it is accessed less frequently and is of less value to the organization.

For example, in a sales order application, the value of the information changes from the time the order is placed until the time that the warranty becomes void (see Figure 1-7). The value of the information is highest when a company receives a new sales order and processes it to deliver the product.

After order fulfillment, the customer or order data need not be available for real-time access. The company can transfer this data to less expensive secondary storage with lower accessibility and availability requirements unless or until a warranty claim or another event triggers its need.

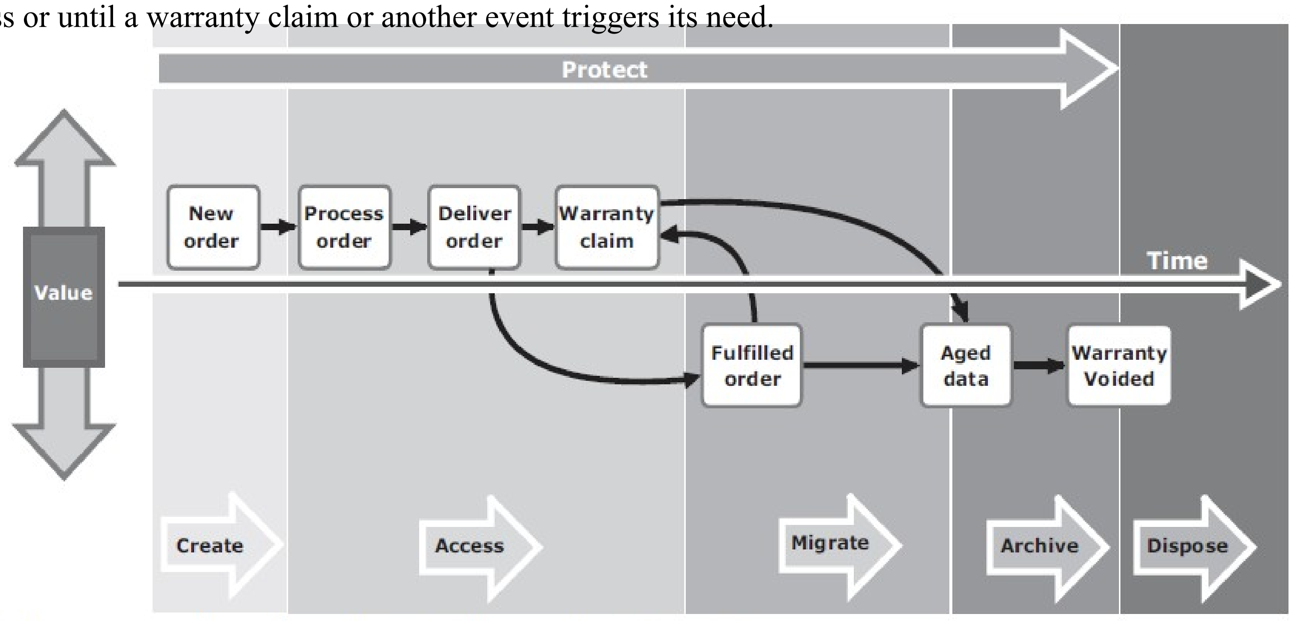


Figure 1-7: Changing value of sales order information

Information Lifecycle Management

Today's business requires data to be protected and available 24 × 7. Data centers can accomplish this with the optimal and appropriate use of storage infrastructure.

Information lifecycle management (ILM) is a proactive strategy that enables an IT organization to effectively manage the data throughout its lifecycle, based on predefined business policies. This allows an IT organization to optimize the storage infrastructure for maximum return on investment.

An ILM strategy should include the following characteristics:

1 Business-centric: It should be integrated with key processes, applications, and initiatives of the business to meet both current and future growth in information.

2 Centrally managed: All the information assets of a business should be under the purview of the ILM strategy.

3 Policy-based: The implementation of ILM should not be restricted to a few departments. ILM should be implemented as a policy and encompass all business applications, processes, and resources.

4 Heterogeneous: An ILM strategy should take into account all types of storage platforms and operating systems.

5 Optimized: Because the value of information varies, an ILM strategy should consider the different storage requirements and allocate storage resources based on the information's value to the business.

ILM Implementation

The process of developing an ILM strategy includes four activities—classifying, implementing, managing, and organizing:

1 Classifying data and applications on the basis of business rules and policies to enable differentiated treatment of information

2 Implementing policies by using information management tools, starting from the creation of data and ending with its disposal

3 Managing the environment by using integrated tools to reduce operational complexity

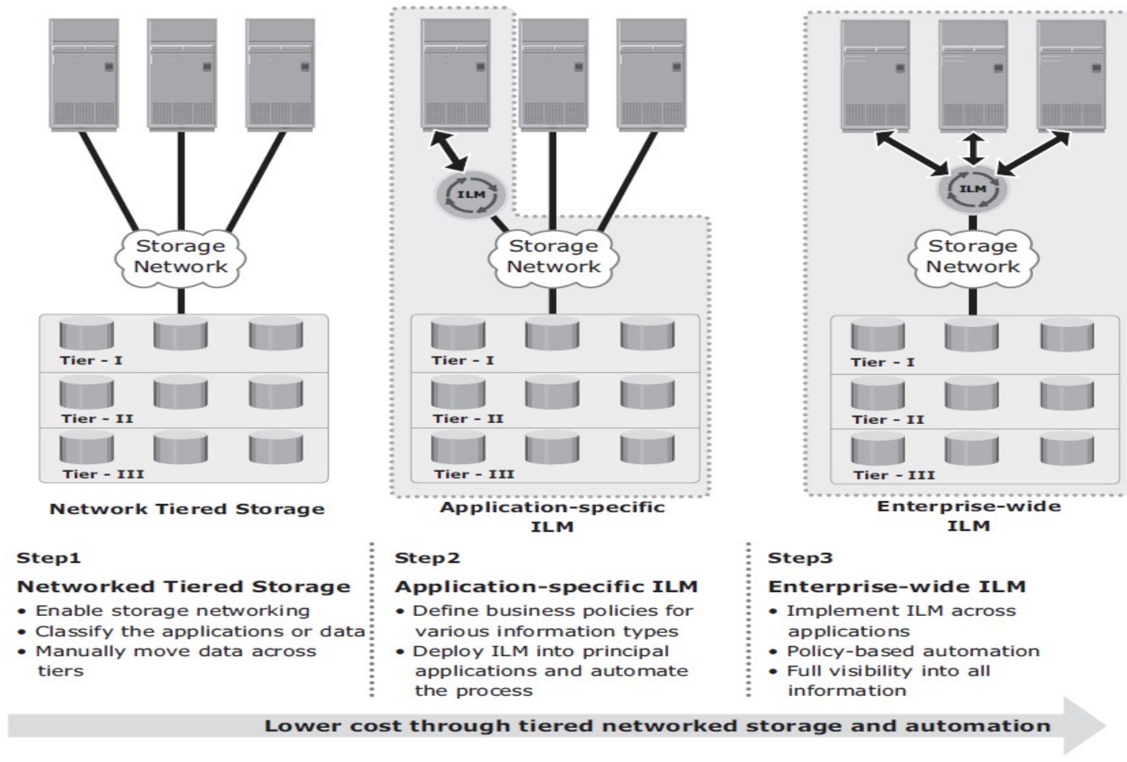
4 Organizing storage resources in tiers to align the resources with data classes, and storing information in the right type of infrastructure based on the information's current value.

Implementing ILM across an enterprise is an ongoing process. Figure 1-8 illustrates a three-step road map to enterprise-wide ILM.

Step 1 the goal is to implement a storage networking environment. Storage architectures offer varying levels of protection and performance and this acts as a foundation for future policy-based information management in Steps 2 and 3. The value of tiered storage platforms can be exploited by allocating appropriate storage resources to the applications based on the value of the information processed.

Step 2 takes ILM to the next level, with detailed application or data classification and linkage of the storage infrastructure to business policies. These classifications and the resultant policies can be automatically executed using tools for one or more applications, resulting in better management and optimal allocation of storage resources.

Step 3 of the implementation is to automate more of the applications or data classification and policy management activities in order to scale to a wider set of enterprise applications.



Storage Network
Tier - I
Tier - II
Tier - III
Network Tiered Storage
ILM
Application-specific ILM
Enterprise-wide ILM
Step1
Networked Tiered Storage
• Enable storage networking
• Classify the applications or data
• Manually move data across tiers
Step2
Application-specific ILM
• Define business policies for various information types
• Deploy ILM into principal applications and automate the process
Step3
Enterprise-wide ILM
• Implement ILM across applications
• Policy-based automation
• Full visibility into all information
Lower cost through tiered networked storage and automation

Figure 1-8: Implementation of ILM

ILM Benefits

Implementing an ILM strategy has the following key benefits that directly address the challenges of information management:

1 Improved utilization by using tiered storage platforms and increased visibility of all enterprise information.

2 Simplified management by integrating process steps and interfaces with individual tools and by increasing automation.

3 A wider range of options for backup, and recovery to balance the need for business continuity.

4 Maintaining compliance by knowing what data needs to be protected for what length of time.

5 Lower Total Cost of Ownership **(TCO)** by aligning the infrastructure and management costs with information value. As a result, resources are not wasted, and complexity is not introduced by managing low-value data at the expense of high-value data.

Module II: Architecture
A: Architecture and RAID Levels

Storage, as one of the core elements of a data center, is recognized as a distinct resource and it needs focus and specialization for its implementation and management. The data flows from an application to storage through various components collectively referred as a *storage system environment.* The three main components in this environment are the host, connectivity, and storage. These entities, along with their physical and logical components, facilitate data access.

Chapter Objective

This chapter details the storage system environment and focuses primarily on storage. It provides details on various hardware components of a disk drive, disk geometry, and the fundamental laws that govern disk performance. The connectivity between the host and storage facilitated by bus technology and interface protocols is also explained.

Components of a Storage System Environment

Host

Users store and retrieve data through applications. The computers on which these applications run are referred to as *hosts*. Hosts can range from simple laptops to complex clusters of servers.

Physical Components

A host has three key physical components:

■ Central processing unit (CPU)

■ Storage, such as internal memory and disk devices

■ Input/Output (I/O) devices

CPU

The CPU consists of four main components:

■**Arithmetic Logic Unit (ALU):** This is the fundamental building block of the CPU. It performs arithmetical and logical operations such as addition, subtraction, and Boolean functions (AND, OR, and NOT).

■ **Control Unit:** A digital circuit that controls CPU operations and coordinates the functionality of the CPU.

■ **Register:** A collection of high-speed storage locations. The registers store intermediate data that is required by the CPU to execute an instruction and provide fast access because of their proximity to the ALU. CPUs typically have a small number of registers.

■ **Level 1 (L1) cache:** Found on modern day CPUs, it holds data and program instructions that are likely to be needed by the CPU in the near future. The L1 cache is slower than registers, but provides more storage space.

Storage

Memory and storage media are used to store data, either persistently or temporarily. Memory modules are implemented using semiconductor chips, whereas storage devices use either magnetic or optical media.

There are two types of memory on a host:

■ **Random Access Memory (RAM):** This allows direct access to any memory location and can have data written into it or read from it. RAM is volatile; this type of memory requires a constant supply of power to maintain memory cell content. Data is erased when the system's power is turned off or interrupted.

■ **Read-Only Memory (ROM):** Non-volatile and only allows data to be read from it. ROM holds data for execution of internal routines, such as system startup.

I/O Devices

I/O devices enable sending and receiving data to and from a host. This communication may be one of the following types:

■**User to host communications:** Handled by basic I/O devices, such as the keyboard, mouse, and monitor. These devices enable users to enter data and view the results of operations.

■**Host to host communications:** Enabled using devices such as a Network Interface Card (NIC) or modem.

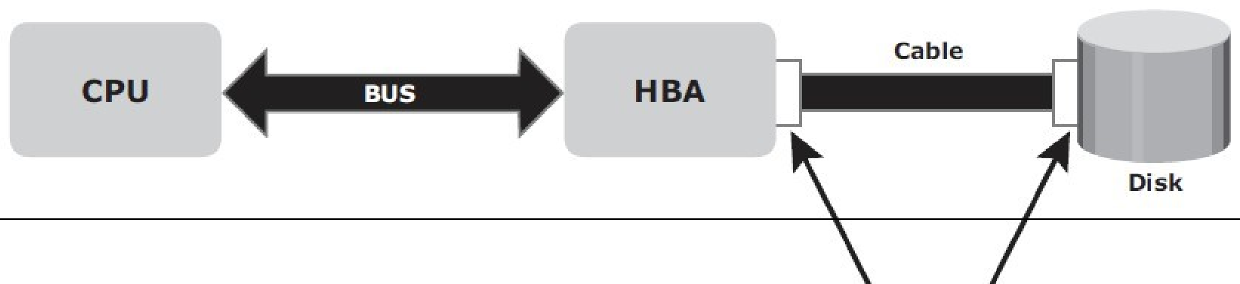
■**Host to storage device communications:** Handled by a *Host Bus Adaptor (HBA)*. HBA is an application-specific integrated circuit (ASIC) board that performs I/O interface functions between the host and the storage, relieving the CPU from additional I/O processing workload.

Connectivity

Connectivity refers to the interconnection between hosts or between a host and any other peripheral devices, such as printers or storage devices. The components of connectivity in a storage system environment can be classified as physical and logical. The *physical components* are the hardware elements that connect the host to storage and the *logical components* of connectivity are the protocols used for communication between the host and storage.

Physical Components of Connectivity

The three physical components of connectivity between the host and storage are Bus, Port, and Cable.



1. The ***bus*** is the collection of paths that facilitates data transmission from one part of a computer to another, such as from the CPU to the memory.
2. The ***port*** is a specialized outlet that enables connectivity between the host and external devices.
3. ***Cables*** connect hosts to internal or external devices using copper or fiber optic media.

Physical components communicate across a bus by sending bits (control, data, and address) of data between devices.

These bits are transmitted through the bus in either of the following ways:

■ **Serially:** Bits are transmitted sequentially along a single path. This transmission can be unidirectional or bidirectional.

■ **In parallel:** Bits are transmitted along multiple paths simultaneously. Parallel can also be bidirectional.

Buses, as conduits of data transfer on the computer system, can be classified as follows:

■ **System bus:** The bus that carries data from the processor to memory.

■ **Local or I/O bus:** A high-speed pathway that connects directly to the processor and carries data between the peripheral devices, such as storage devices and the processor.

Logical Components of Connectivity

PCI

PCI is a specification that standardizes how PCI expansion cards, such as network cards or modems, exchange information with the CPU. PCI provides the interconnection between the CPU and attached devices. The plug-and-play functionality of PCI enables the host to easily recognize and configure new cards and devices. The width of a PCI bus can be 32 bits or 64 bits. A 32-bit PCI bus can provide a throughput of 133 MB/s. *PCI Express* is an enhanced version of PCI bus with considerably higher throughput and clock speed.

IDE/ATA

IDE/ATA is the most popular interface protocol used on modern disks. This protocol offers excellent performance at relatively low cost. Details of IDE/ATA are provided in Chapter 5.

SCSI

SCSI has emerged as a preferred protocol in high-end computers. This interface is far less commonly used than IDE/ATA on personal computers due to its higher cost. SCSI was initially used as a parallel interface, enabling the connection of devices to a host. SCSI has been enhanced and now includes a wide variety of related technologies and standards. Chapter 5 provides details of SCSI.

Storage

A storage device uses magnetic or solid state media.

1. **Disks, tapes, and diskettes** use magnetic media.
2. **CD-ROM** is an example of a storage device that uses optical media
3. **Removable flash memory card** is an example of solid state media.
4. ***Tapes*** are a popular storage media used for backup because of their relatively low cost. Tape has the following limitations:

■ Data is stored on the tape linearly along the length of the tape. Search and retrieval of data is done sequentially, invariably taking several seconds to access the data. As a result, random data access is slow and time consuming. This limits tapes as a viable option for applications that require real-time, rapid access to data.

■ In a shared computing environment, data stored on tapecannot be accessed by multiple applications simultaneously, restricting its use to one application at a time.

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

■ The storage and retrieval requirements of data from tape and the overhead associated with managing tape media are significant.

5. ***Optical disk storage*** is popular in small, single-user computing environments. It is frequently used by individuals to store photos or as a backup medium on personal/laptop computers. It is also used as a distribution medium for single applications, such as games, or as a means of transferring small amounts of data from one self-contained system to another. Optical disks have limited capacity and speed, which limits the use of optical media as a business data storage solution.

Disk Drive Components

A disk drive uses a rapidly moving arm to read and write data across a flat platter coated with magnetic particles. Data is transferred from the magnetic platter through the R/W head to the computer. Several platters are assembled together with the R/W head and controller, most commonly referred to as a *hard disk drive (HDD)*. Data can be recorded and erased on a magnetic disk any number of times.

Key components of a disk drive are *platter, spindle, read/write head, actuator arm assembly, and controller* (Figure 2-2)

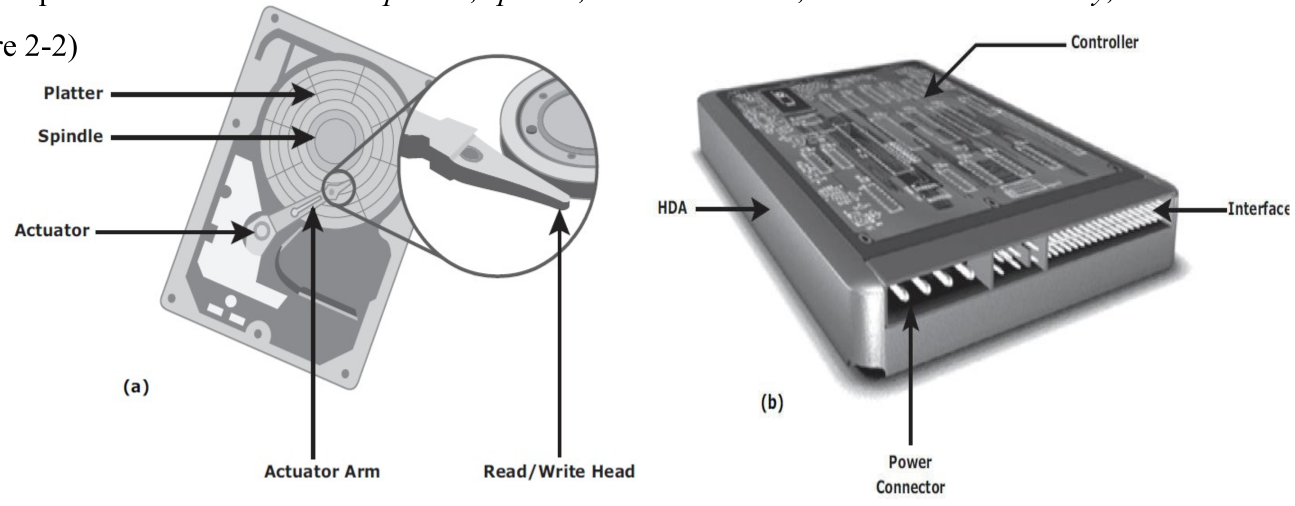


Figure 2-2: Disk Drive Components

Platter

A typical HDD consists of one or more flat circular disks called *platters* (Figure 2-3). The data is recorded on these platters in binary codes (0s and 1s). The set of rotating platters is sealed in a case, called a *Head Disk Assembly (HDA)*. 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.

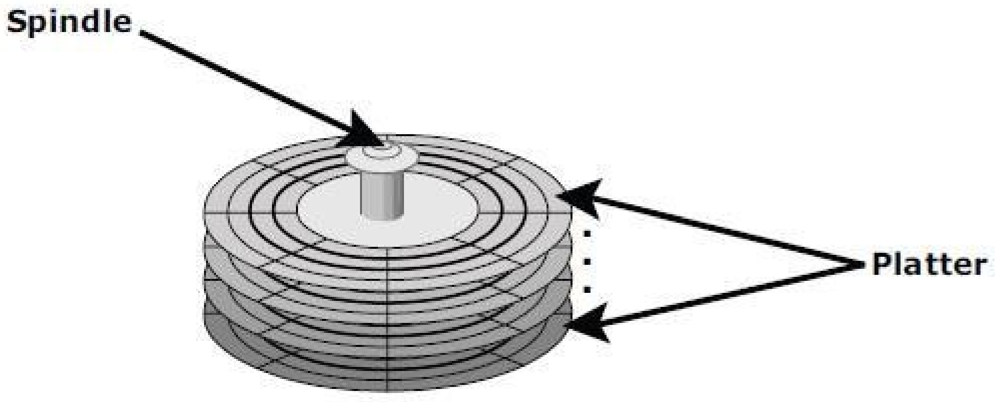


Figure 2-3: Spindle and platter

Spindle

A spindle connects all the platters, as shown in Figure 2-3, 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). Disk drives have spindle speeds of 7,200 rpm, 10,000 rpm, or 15,000 rpm. Disks used on current storage systems have a platter diameter of 3.5‖ (90 mm).

Read/Write Head

Read/Write (R/W) heads, shown in Figure 2-4, read and write data from or to a platter. 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, this head detects 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.

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

The R/W heads are mounted on the *actuator arm assembly* (refer to Figure 2-2 [a]), 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.

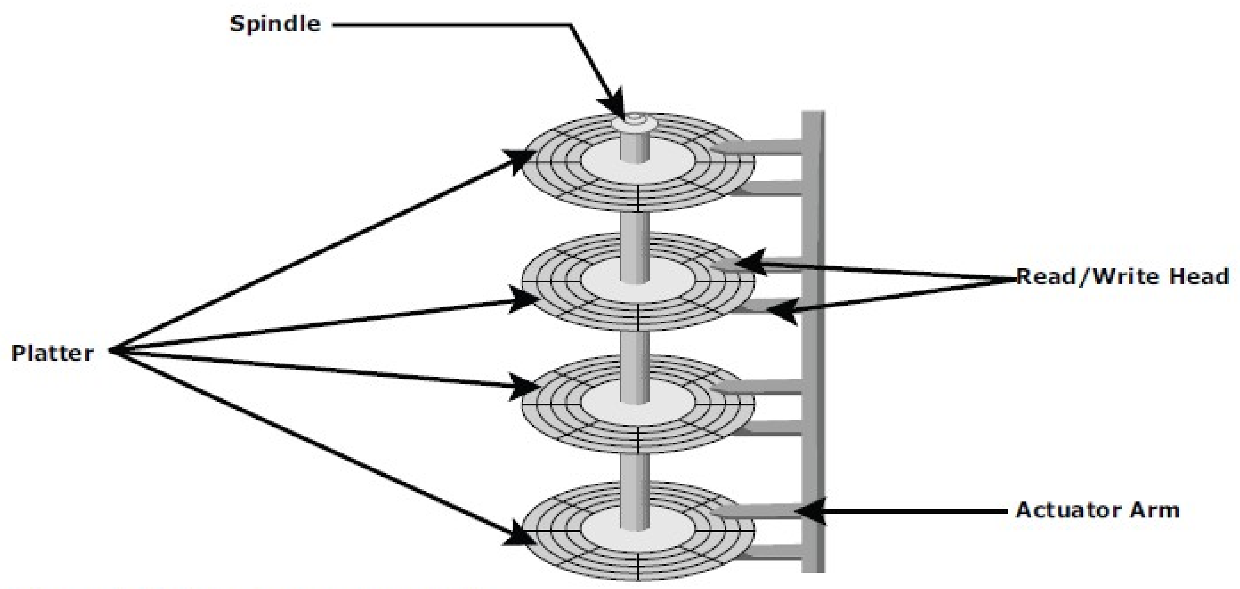


Figure 2-4: Actuator arm assembly

Controller

The *controller* (see Figure 2-2 [b]) 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 power to the spindle motor and the speed of the motor. It also manages communication between the drive and the host.

Physical Disk Structure

Data on the disk is recorded on *tracks*, which are concentric rings on the platter around the spindle, as shown in Figure 2-5. 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*) measures how 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.

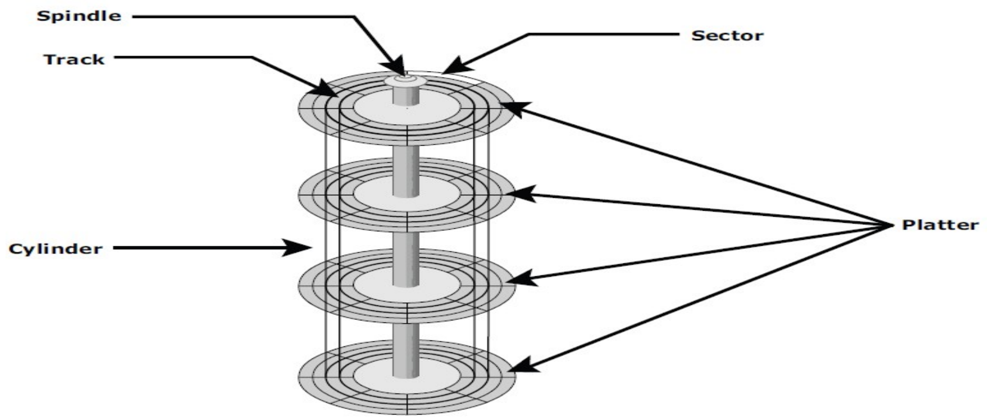


Figure 2-5: Disk structure: sectors, tracks, and cylinders

Zoned Bit Recording

Zone bit recording utilizes the disk efficiently. As shown in Figure 2-6 (b), this mechanism groups tracks into zones based on their distance from the center of the disk. The zones are numbered, with the outermost 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.

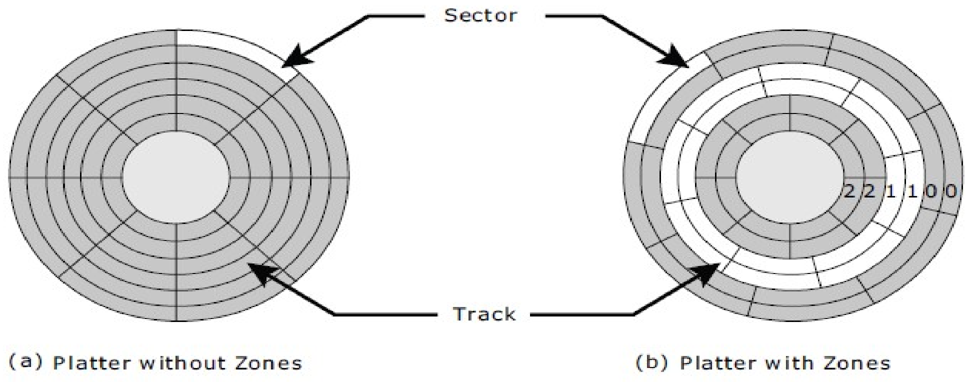


Figure 2-6: Zoned bit recording

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 2-7 (a), and the host operating system had to be aware of the geometry of each disk being used. *Logical block addressing (LBA)*, shown in Figure 2-7 (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 only needs to know the size of the disk drive in terms of the number of blocks.

In Figure 2-7 (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 will have in excess of 976,000,000 blocks.

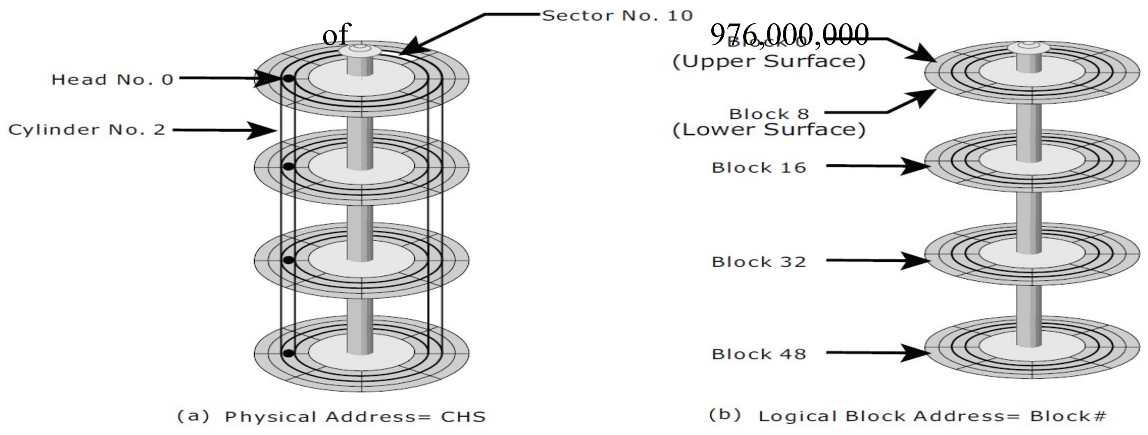


Figure 2-7: Physical address and logical block address

2.3 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.

2.3.1 Disk Service Time

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

1. **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 reposition and settle the arm and the head over the correct track. 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.

2. **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.

3. **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. In a *read operation*, the data first moves from disk platters to R/W heads, and 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.

Internal transfer rate is the speed at which data moves from a single track of a platter's surface to internal
buffer (cache) of the disk. Internal transfer rate takes into account factors such as the seek time.

External transfer rate is the rate at which data can be moved through the interface to the HBA. External transfer rate is generally the advertised speed of the interface, such as 133 MB/s for ATA.

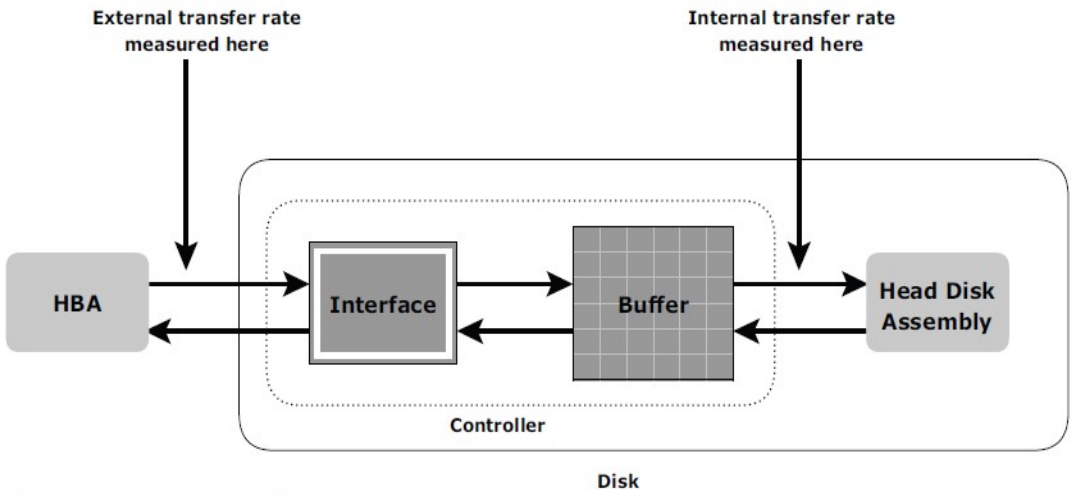

External transfer rate measured here
Internal transfer rate measured here
HBA
Interface
Buffer
Head Disk Assembly
Controller
Disk

Figure 2-8: Data transfer rate

Fundamental Laws Governing Disk Performance

To understand the laws of disk performance, a disk can be viewed as a black box consisting of two elements:

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

∎**Disk I/O Controller:** Processes I/Os that are 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 2-9. The I/O arrival rate, the queue length, and the time taken by the I/O controller to process each request determines the performance of the disk system, which is measured in terms of response time.

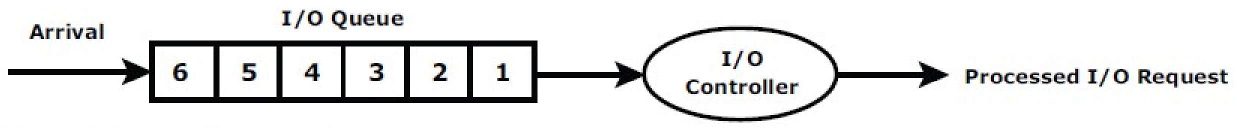


Figure 2-9: I/O processing

Little's Law is a fundamental law describing the relationship between the number of requests in a queue and the response time. The law states the following relation (numbers in parentheses indicate the equation number for crossreferencing):

N = a × R ------------------ (1)

where

–N‖ is the total number of requests in the queuing system (requests in the queue + requests in the I/O controller)

—a‖ is the arrival rate, or the number of I/O requests that arrive to the system per unit of time

–R‖ is the average response time or the turnaround time for an I/O request — the total time from arrival to departure from the system

The ***utilization law*** is another important law that defines the I/O controller utilization. This law states the relation:

U = a × Rs -----------------(2)

where

–U‖ is the I/O controller utilization

–Rs– is the *service time,* or the average time spent by a request on the controller**.** 1/RS is the *service rate.*

From the arrival rate ―a‖, the average inter-arrival time, Ra, can be computed as:

Ra = 1/a -------------------(3)

Consequently, *utilization* can be defined as the ratio of the service time to the average inter-arrival time, and is expressed as:

U = Rs /Ra---------------- (4)

The value of this ratio varies between 0 and 1.

In the following equation, the term average response rate (S) can be defined as the reciprocal of the average response time (R), and is derived as follows:

S = service rate – arrival rate

Consequently,

As a result,R = 1 / (service rate – arrival rate) R = 1 / (1/Rs - 1/Ra)

= 1 / (1/Rs-a) (from eq. 3)

= Rs / (1-a × Rs)

R = Rs / (1-U) -------------- (5) (from eq. 2)

Average response time (R) = service time / (1 – utilization) (from equation 2)

Utilization (U) can also be used to represent the average number of I/O requests on the controller, as shown in the following:

Number of requests in the queue (NQ) = Number of requests in the system (N)

– Number of requests on the controller or utilization (U). Number of requests in a queue is also referred to as *average queue size.*

Nq = N – U

= a × R – U (from eq. 1)

$= a \times (Rs / (1 – U)) – U$ (from eq. 5)

$= (Rs /Ra) / (1 – U) – U$ (from eq. 3)

$= U / (1 – U) – U$ (from eq. 4)

$= U (1 / (1 – U) – 1)$

$= U^2 / (1–U)$ -------------- (6)

The time spent by a request in the queue is equal to the time spent by a request in the system, or the average response time minus the time spent by a request on the controller for processing:

$= Rs / (1 – U) – Rs$ (from eq. 5)

$= U \times RS / (1 – U)$

$= U \times$ avg. response time

= Utilization × R----------------- (7)

Ex: Consider a disk I/O system in which an I/O request arrives at a rate of 100 I/Os per second. The service time, Rs, is 8 ms. The following measures of disk performance can be computed using the relationships developed above — utilization of I/O controller (U), total response time (R), average queue size [U2 / (1 – U)] and total time spent by a request in a queue (U × R), as follows:

Arrival rate (a) = 100 I/O/s; consequently, the arrival time
Ra = 1/a = 10 ms
Rs = 8 ms (given)
1. Utilization (U) = Rs / Ra = 8 / 10 = 0.8 or 80%
2. Response time (R) = Rs /(1 – U) = 8 / (1 – 0.8) = 40 ms
3. Average queue size = U2 / (1 – U) = (0.8)2 / (1 – 0.8) = 3.2
4. Time spent by a request in a queue = U × R, or the total response timeservice time = 32 ms

Now, if controller power is doubled, the service time is halved; consequently, Rs = 4 ms in this scenario.
1. Utilization (U) = 4 / 10 = 0.4 or 40%
2. Response time (R) = 4 / (1 – 0.4) = 6.67 ms
3. Average queue size = (0.4)2 / (1 – 0.4) = 0.26
4. Time spent by a request in a queue = 0.4 × 6.67 = 2.67 ms

As a result, it can be concluded that by reducing the service time (the sum of seek time, latency, and internal transfer rate) or utilization by half, the response time can be reduced drastically (almost six times in the preceding example). The relationship between utilization and response time is shown in Figure 2-10.

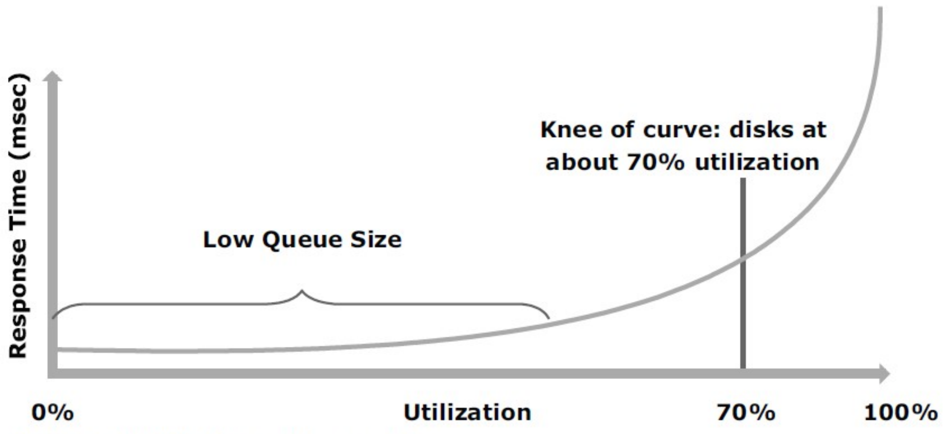


Figure 2-10: Utilization vs. Response time

Logical Components of the Host

The logical components of a host consist of the software applications and protocols that enable data communication with the user as well as the physical components. Following are the logical components of a host:

- Operating system
- Device drivers
- Volume manager
- File system
- Application

Operating System

An *operating system* controls all aspects of the computing environment. It works between the application and physical components of the computer system. One of the services it provides to the application is data access. The operating system also monitors and responds to user actions and the environment. It organizes and controls hardware components and manages the allocation of hardware resources.

Device Driver

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 hard drive. A device driver enables the operating system to recognize the device and to use a standard interface (provided as an *application programming interface*, or *API*) to access and control devices. Device drivers are hardware dependent and operating system specific.

Volume Manager

Disk partitioning was introduced to improve the flexibility and utilization of HDDs. In partitioning, an HDD is divided into logical containers called *logical volumes (LVs)* (see Figure 2-11).

Concatenation is the process of grouping several smaller physical drives and presenting them to the host as one logical drive (see Figure 2-11).

The evolution of *Logical Volume Managers (LVMs)* enabled the dynamic extension of file system capacity and efficient storage management. LVM is software that runs on the host computer and manages the logical and physical storage. LVM is an optional, intermediate layer between the file system and the physical disk. It can aggregate several smaller disks to form a larger virtual disk or to partition a larger-capacity disk into virtual, smaller-capacity disks, which are then presented to applications. The LVM provides optimized storage access and simplifies storage resource management. It hides details about the physical disk and the location of data on the disk; and it enables administrators to change the storage allocation without changing the hardware, even when the application is running.

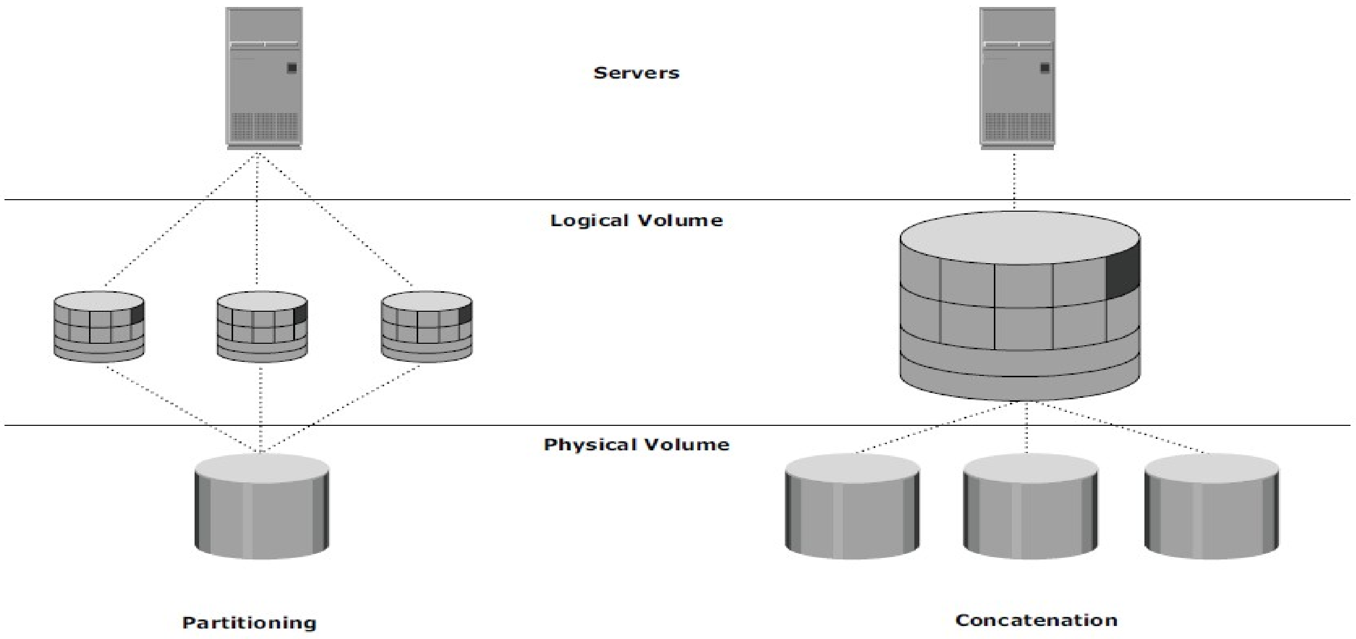


Figure 2-11: Disk partitioning and concatenation

The basic LVM components are the *physical volumes*, *volume groups*, and *logical volumes*. In LVM terminology, each physical disk connected to the host system is a *physical volume (PV)*. LVM converts the physical storage provided by the physical volumes to a logical view of storage, which is then used by the operating system and applications. 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.

Logical volumes are created within a given volume group. A logical volume can be thought of as a virtual disk partition, while the volume group itself can be thought of as a disk. A volume group can have a number of logical volumes. The size of a logical volume is based on a multiple of the physical extents.

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. File systems enable easy access to data files residing within a disk drive, a disk partition, or a logical volume. A file system needs host-based logical structures and software routines that control access to files. It provides users with the functionality to create, modify, delete, and access files. 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, subdirectories, and files that are part of the file system. Some of the common file systems are as follows:

- FAT 32 (File Allocation Table) for Microsoft Windows
- NT File System (NTFS) for Microsoft Windows
- UNIX File System (UFS) for UNIX
- Extended File System (EXT2/3) for Linux

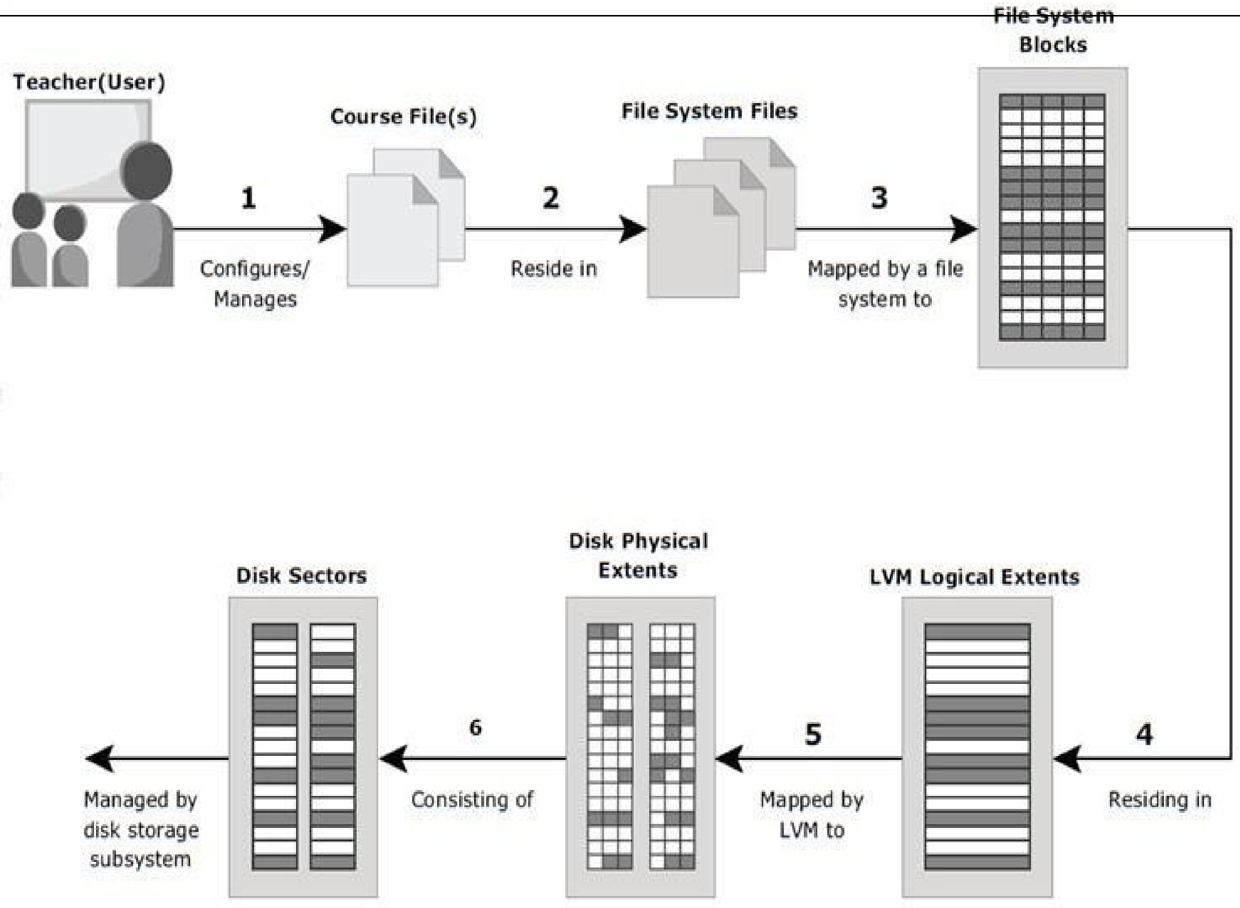


Figure 2-12: Process of mapping user files to disk storage

Figure 2-12 shows the following process of mapping user files to the disk storage subsystem with an LVM:

1. Files are created and managed by users and applications.
2. These files reside in the file systems.
3. The file systems are then mapped to units of data, or file system blocks.
4. The file system blocks are mapped to logical extents.
5. These in turn are mapped to disk physical extents either by the operating system or by the LVM.
6. These physical extents are mapped to the disk storage subsystem.

Application

An *application* is a computer program that provides the logic for computing operations. It provides an interface between the user and the host and among multiple hosts. Conventional business applications using databases have a three-tiered architecture — the application user interface forms the front-end tier; the computing logic forms, or the application itself is, the middle tier; and the underlying databases that organize the data form the back-end tier. 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 database, which in turn uses the OS services to perform R/W operations to storage devices. These R/W operations (I/O operations) enable transactions between the front-end and back-end tiers.

Data access can be classified as block-level or file-level depending on whether the application uses a logical block address or the file name and a file record identifier to read from and write to a disk.

Block-Level Access

Block-level access is the basic mechanism for disk access. In this type of access, data is stored and retrieved from disks by specifying the logical block address. The block address is derived based on the geometric configuration of the disks. Block size defines the basic unit of data storage and retrieval by an application. Databases, such as Oracle and SQL Server, define the block size for data access and the location of the data on the disk in terms of the logical block address when an I/O operation is performed.

File-Level Access

File-level access is an abstraction of block-level access. File-level access to data is provided by specifying the name and path of the file. It uses the underlying block-level access to storage and hides the complexities of logical block addressing (LBA) from the application and the DBMS.

Application Requirements and Disk Performance

The analysis of application storage requirements conventionally begins with determining storage capacity. This can be easily estimated by the size and number of file systems and database components that will be used by applications. The application I/O size and the number of I/Os the application generates are two important measures affecting disk performance and response time.

Consequently, the storage design and layout for an application commences with the following:

1. Analyzing the number of I/Os generated at peak workload
2. Documenting the application I/O size or block size.

Consider an example of a SCSI controller (SCSI interface) with a throughput of 160 MB/s and disk service time Rs = 0.3 ms. The computation of the rate (1 / [Rs + Transfer time]) at which I/Os are serviced in a typical database I/O with block sizes 4 KB, 8 KB, 16 KB, 32 KB, and 64 KB are shown in Table 2-1. The rate at which the application I/Os are serviced is termed I/Os per second (IOPS).

Table 2-1: Maximum IOPS Performed by SCSI Controller

BLOCK SIZE	TRANSFER TIME (MS)	IOPS= 1/(R_S + TRANSFER TIME)
4 KB	4 KB / 160 MB = 0.025	1 / (0.3 + 0.025) = 3,076
8 KB	8 KB / 160 MB = 0.05	1 / (0.3 + 0.05) = 2,857
16 KB	16 KB / 160 MB = 0.1	1 / (0.3 + 0.1) = 2,500
32 KB	32 KB / 160 MB = 0.2	1 / (0.3 + 0.2) = 2,000
64 KB	64 KB / 160 MB = 0.4	1 / (0.3 + 0.4) = 1,428

As a result, the number of IOPS per controller depends on the I/O block size and ranges from 1,400 (for 64 KB) to 3,100 (for 4 KB).

The disk service time (RS) is a key measure of disk performance; Rs along with disk utilization rate (U) determines the I/O response time for applications. As shown earlier in this chapter, the total disk service time (RS) is the sum of seek time (E), rotational latency (L), and the internal transfer time (X):

$$Rs = E + L + X$$

E is determined based on the randomness of the I/O request. L and X are measures provided by disk vendors as technical specifications of the disk.

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

- Average seek time of 5 ms in a random I/O environment, or E = 5 ms.
- Disk rotation speed of 15,000 rpm — from which rotational latency (L) can be determined, which is one half of the time taken for a full rotation or L = (0.5 / 15,000 rpm expressed in ms).
- 40 MB/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 or X = 32 KB / 40 MB.

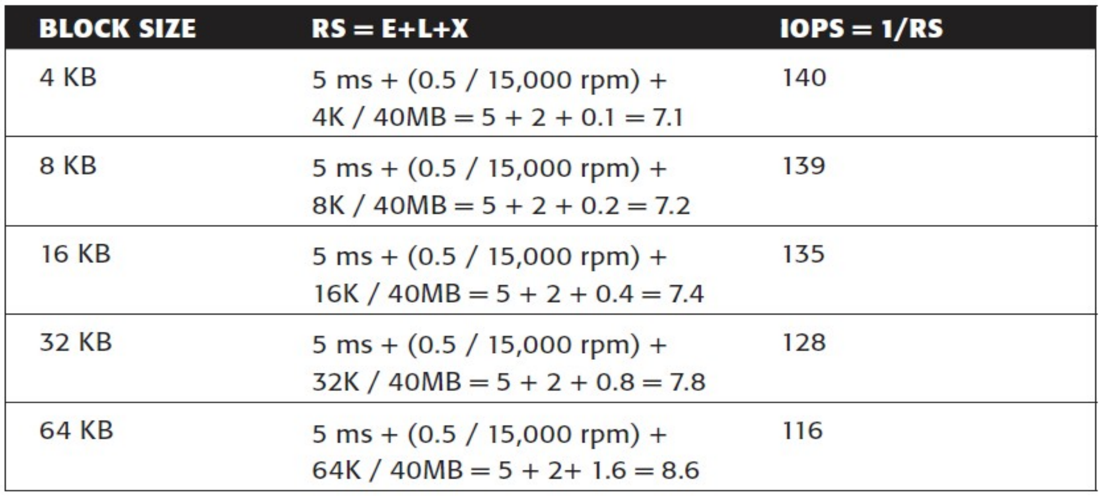
Consequently, Rs = 5 ms + (0.5 / 15,000) + 32 KB / 40 MB = 7.8 ms.

The maximum number of I/Os serviced per second or IOPS = 1/ RS.

In other words, for an I/O with a block size of 32 KB and RS = 7.8 ms, the maximum IOPS will be 1 / (7.8 × 10-3) = 128 IOPS.

Table 2-2 lists the maximum IOPS that can be serviced for different block sizes.

Table 2-2: Maximum IOPS Performed by Disk Drive



BLOCK SIZE	RS = E+L+X	IOPS = 1/RS
4 KB	5 ms + (0.5 / 15,000 rpm) + 4K / 40MB = 5 + 2 + 0.1 = 7.1	140
8 KB	5 ms + (0.5 / 15,000 rpm) + 8K / 40MB = 5 + 2 + 0.2 = 7.2	139
16 KB	5 ms + (0.5 / 15,000 rpm) + 16K / 40MB = 5 + 2 + 0.4 = 7.4	135
32 KB	5 ms + (0.5 / 15,000 rpm) + 32K / 40MB = 5 + 2 + 0.8 = 7.8	128
64 KB	5 ms + (0.5 / 15,000 rpm) + 64K / 40MB = 5 + 2+ 1.6 = 8.6	116

For the example in Table 2-2, the I/O response time (R) for an I/O with a block size of 64 KB will be approximately 215 ms when the controller works close to 96 percent utilization, as shown here:

$$\begin{aligned} R &= Rs / (1- U) \\ &= 8.6 / (1-0.96) \\ &= 215 \text{ ms} \end{aligned}$$

If an application demands a faster response time, then the utilization for the disks should be maintained below 70 percent, or the knee of the curve, after which the response time increases exponentially.

The total number of disks required (N) for an application is computed as follows:
If *C* is the number of disks required to meet the capacity and *I* is the number of disks required for meeting IOPS, then $N = \text{Max}(C, I)$

Disk vendors publish the disk potential in terms of IOPS based on the benchmark they carry out for different block sizes and application environments.

Consider an example in which the capacity requirements for an application are 1.46 TB. The peak workload generated by the application is estimated at 9,000 IOPS. The vendor specifies that a 146 GB, 15,000-rpm drive is capable of a maximum of 180 IOPS (U = 70%).

In this example, the number of disks required to meet the capacity requirements will be only 1.46 TB / 146 GB = 10 disks. To meet 9,000 IOPS, 50 disks will be required (9,000 / 180). As a result, the number of disks required to meet the application demand will be Max (10, 50) = 50 disks.

Data Protection, Intelligent Storage system

Data Protection: RAID Introduction

In the late 1980s, rapid adoption of computers for business processes stimulated the growth of new applications and databases, significantly increasing the demand for storage capacity. At that time, 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, due to their limitations.

HDDs are susceptible to failures due to mechanical wear and tear and other environmental factors. An HDD failure may result in data loss. The solutions available during the 1980s were not able to meet the availability and performance demands of applications.

An HDD has a projected life expectancy before it fails. ***Mean Time Between Failure (MTBF)*** measures (in hours) the average life expectancy of an HDD. Today, data centers deploy thousands of HDDs in their storage infrastructures. The greater the number of HDDs in a storage array, the greater the probability of a disk failure.

RAID is an enabling technology that leverages multiple disks as part of a set, which provides data protection against HDD failures. In general, RAID implementations also improve the I/O performance of storage systems by storing data across multiple HDDs.

Chapter Objective

This chapter details RAID technology, RAID levels, and different types of RAID implementations and their benefits.

Implementation of RAID

There are two types of RAID implementation, hardware and software. Both have their merits and demerits.

Software RAID

Software RAID uses host-based software to provide RAID functions. It is implemented at the operating-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 hardware RAID. Limitations of software RAID:

1. Performance: Software RAID affects overall system performance. This is due to the additional CPU cycles required to perform RAID calculations.

2. Supported features: Software RAID does not support all RAID levels.

3. 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.

Hardware RAID

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

Controller card RAID is host-based hardware RAID implementation in which a specialized **RAID controller** is installed in the host and HDDs are connected to it. **RAID Controller** interacts with the hard disks using PCI bus.

Manufacturers integrate RAID controllers on motherboards. This reduces the overall cost of the system, but does not provide the flexibility required for high-end storage systems.

The external RAID controller is an array-based hardware RAID. It acts as an interface between host and disks. It presents storage volumes to host, which manage the drives using the supported protocol.

Key functions of RAID controllers are:

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

RAID Array Components

A RAID array is an enclosure that contains a number of HDDs and the supporting hardware and software to implement RAID.

HDDs inside a RAID array are contained in smaller sub-enclosures. These sub-enclosures, or **physical arrays**, hold a fixed number of HDDs, and also include other supporting hardware, such as power supplies.

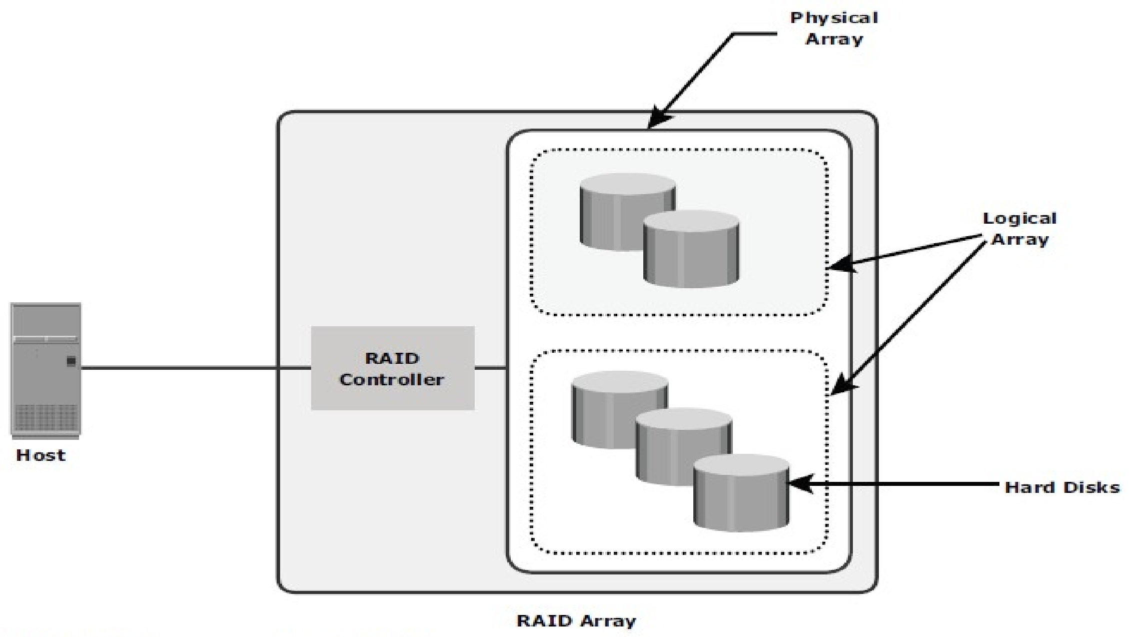


Figure 3-1: Components of RAID array

A subset of disks within a RAID array can be grouped to form logical associations called **logical arrays**, also known as a **RAID set** or a **RAID group** (see Figure 3-1). Logical arrays are comprised of **logical volumes (LV)**. The operating system recognizes the LVs as if they are physical HDDs managed by the RAID controller.

RAID Levels

RAID levels are defined on the basis of **1) striping**, **2) mirroring**, and **3) parity** techniques. These techniques determine the data availability and performance characteristics of an array. Some RAID arrays use one technique, whereas others use a combination of techniques.

Table 3-1: Raid Levels

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

Striping

A RAID set is a group of disks. Within each disk, a predefined number of contiguously addressable disk blocks are defined as ***strips***. The set of aligned strips that spans across all the disks within the RAID set is called a ***stripe*** (Figure 3-2).

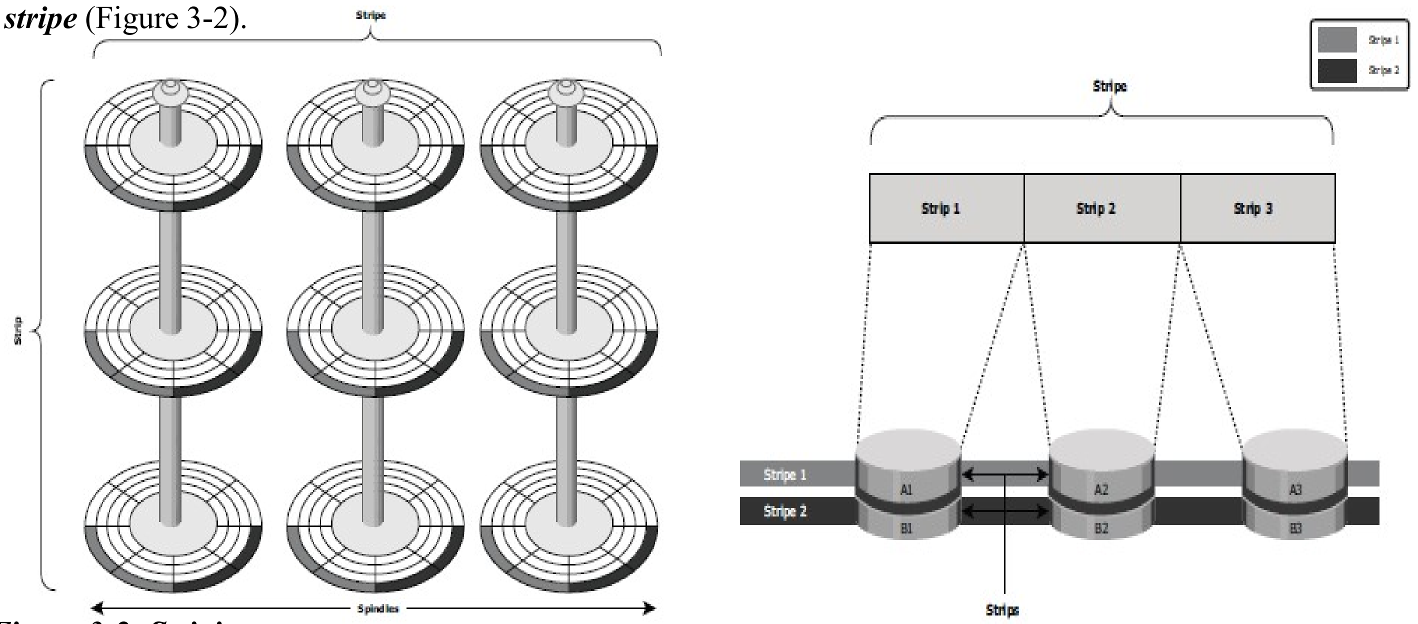


Figure 3-2: Striping

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 HDD in the set. All strips in a stripe have the same number of blocks.

Stripe size is a multiple of strip size by the number of HDDs in the RAID set. ***Stripe width*** refers to the number of data strips in a stripe.

Striped RAID does not protect data unless parity or mirroring is used. Striping significantly improves I/O performance.

Mirroring

Mirroring is a technique whereby data is stored on two different HDDs, yielding two copies of data. In the event of one HDD failure, the data is intact on the surviving HDD (see Figure 3-3) And the controller continues to service the host's data requests from the surviving disk.

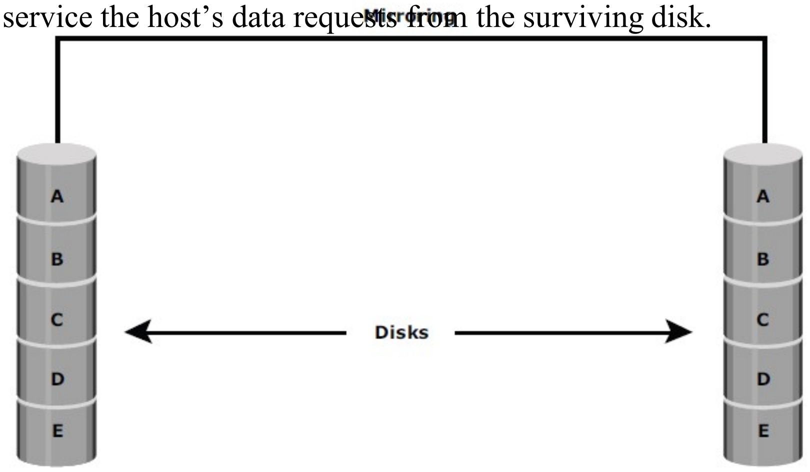


Figure 3-3: Mirrored disks in an array

When the failed disk is replaced with a new disk, the controller copies the data from the surviving disk of the mirrored pair.

Mirroring provide complete data redundancy, and also enables faster recovery from disk failure.

Mirroring is not a substitute for data backup. Mirroring constantly captures changes in the data, whereas a backup captures point-in-time images of data.

Mirroring involves duplication of data - 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 data loss.

Mirroring improves read performance because read requests can be serviced by both disks. However, write performance decreases, as each write request must perform two writes on the HDDs.

Parity

Parity is a method of protecting striped data from HDD failure without the cost of mirroring. An additional HDD is added to the stripe width to hold parity. Parity is a redundancy check that ensures full protection of data without maintaining a full set of duplicate data.

Parity information can be stored on separate, dedicated HDDs or distributed across all the drives in a RAID set. Figure 3-4 shows a parity RAID. The first four disks, labeled *D,* contain the data. The fifth disk, labeled *P*, stores the parity information, which is the sum of the elements in each row. Now, if one of the *D*s fails, the missing value can be calculated by subtracting the sum of the rest of the elements from the parity value.

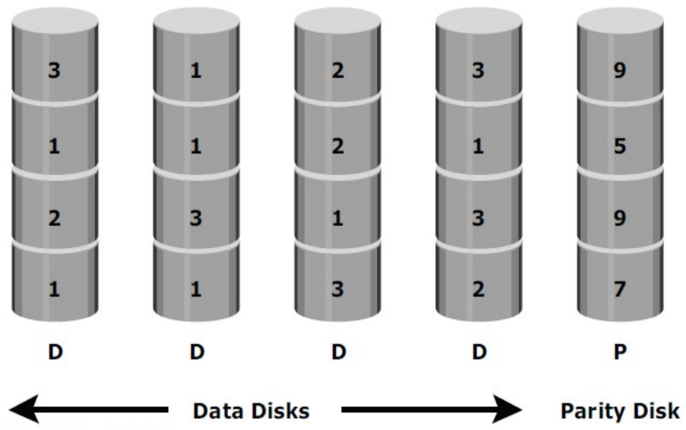


Figure 3-4: Parity RAID

The computation of parity is represented as a simple arithmetic operation on the data. Parity calculation is a *bitwise XOR* operation. Calculation of parity is a function of the RAID controller.

Advantage: Compared to mirroring, parity implementation considerably reduces the cost associated with data protection.

Consider a RAID configuration with five disks. Four of these disks hold data, and the fifth holds parity information. Parity requires 25 percent extra disk space compared to mirroring, which requires 100 percent extra disk space.

Disadvantage: 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 controller.

RAID 0

In a RAID 0 configuration, data is striped across the HDDs in a RAID set. It utilizes the full storage capacity by distributing strips of data over multiple HDDs.

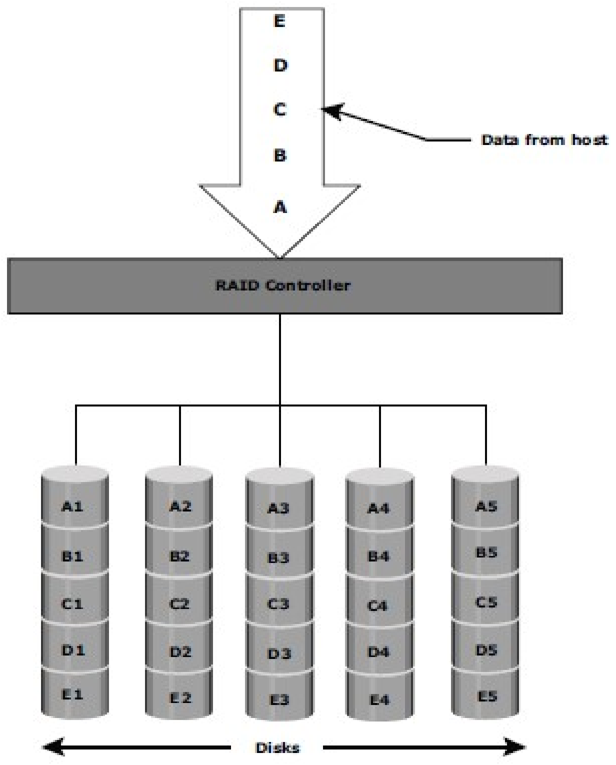
To read data, all the strips are put back together by the controller.

The stripe size is specified at a host level for software RAID and is vendor specific for hardware RAID.

Figure 3-5 shows RAID 0 on a storage array in which data is striped across 5 disks. When the number of drives in the array increases, performance improves because more data can be read or written simultaneously.

Adv: RAID 0 is used in applications that need high I/O throughput.

Disadv: RAID 0 does not provide data protection and availability in the event of drive failures.



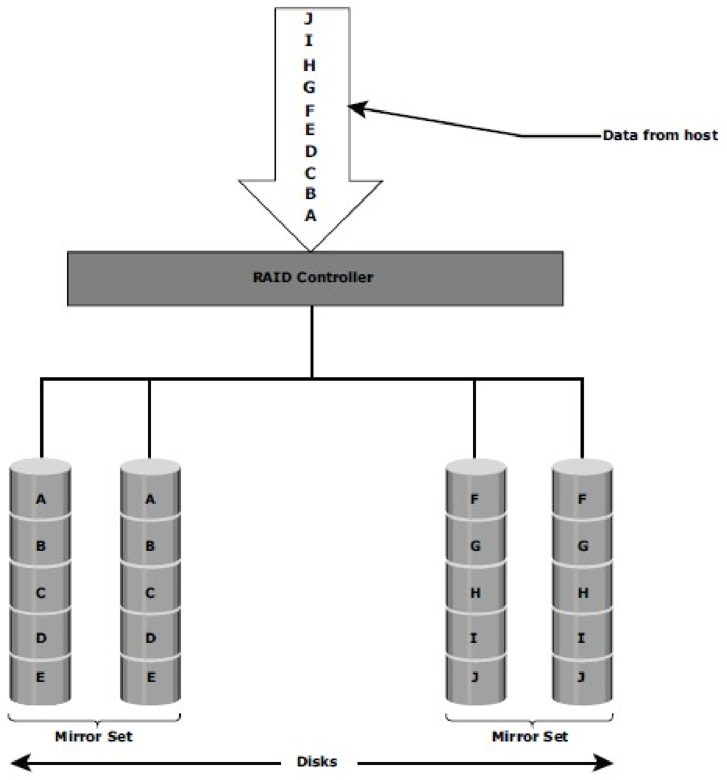


Figure 3-5: RAID 0

Figure 3-6: RAID 1

RAID 1

In a RAID 1 configuration, data is mirrored to improve fault tolerance (Figure 3-6). A RAID 1 group consists of at least two HDDs. As explained in mirroring, every write is written to both disks. In the event of disk failure, the impact on data recovery is the least among all RAID implementations. This is because the RAID controller uses the mirror drive for data recovery and continuous operation.

RAID 1 is suitable for applications that require high availability.

Nested RAID

Most data centers require data redundancy and performance from their RAID arrays.

RAID 0+1 and **RAID 1+0** combine 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 number of disks, the minimum being four (see Figure 3-7).

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 that use small, random, write-intensive I/O.

Some applications that benefit from RAID 1+0 include the following:

1. High transaction rate Online Transaction Processing (OLTP)
2. Large messaging installations
3. Database applications that require high I/O rate, random access, and high availability.

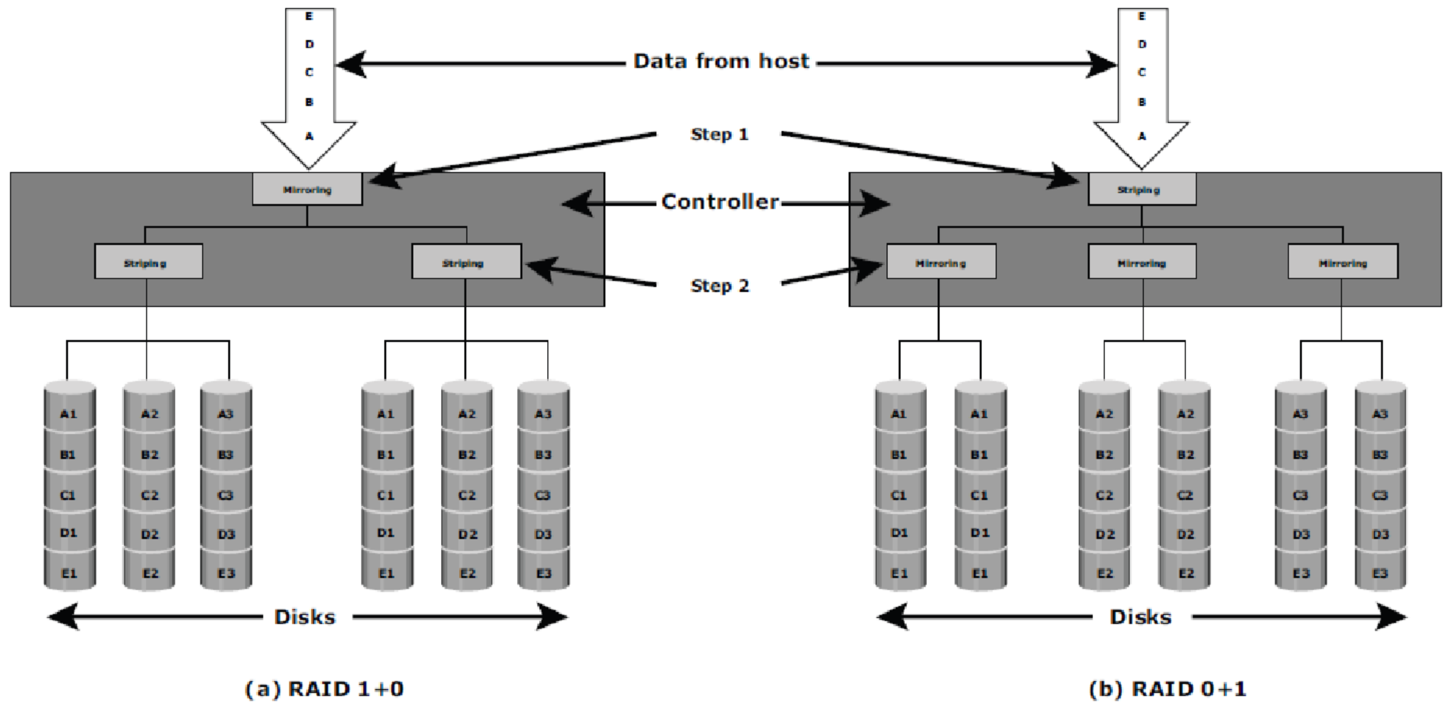


Figure 3-7: Nested RAID

A common misconception is that RAID 1+0 and RAID 0+1 are the same.

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 data are striped across multiple HDDs in a RAID set. When replacing a failed drive, only the mirror is rebuilt, i.e. 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.

RAID 0+1 is also called *mirrored stripe*. The basic element of RAID 0+1 is a stripe. This means that the process of striping data across HDDs is performed initially and then the entire stripe is mirrored. If one drive fails, then the entire stripe is faulted. A rebuild operation copies the entire stripe, copying data from each disk in the healthy stripe to an equivalent disk in the failed stripe.

Disadv: 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 high performance and uses parity for improved fault tolerance.

Parity information is stored on a dedicated drive so that data can be reconstructed if a drive fails.

For example, of five disks, four are used for data and one is used 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, as the drives operate in parallel. There are no partial writes that update one out of many strips. Figure 3-8 illustrates the RAID 3 implementation.\ RAID 3 provides good bandwidth for the transfer of large volumes of data. RAID 3 is used in applications that involve large sequential data access, such as video streaming.

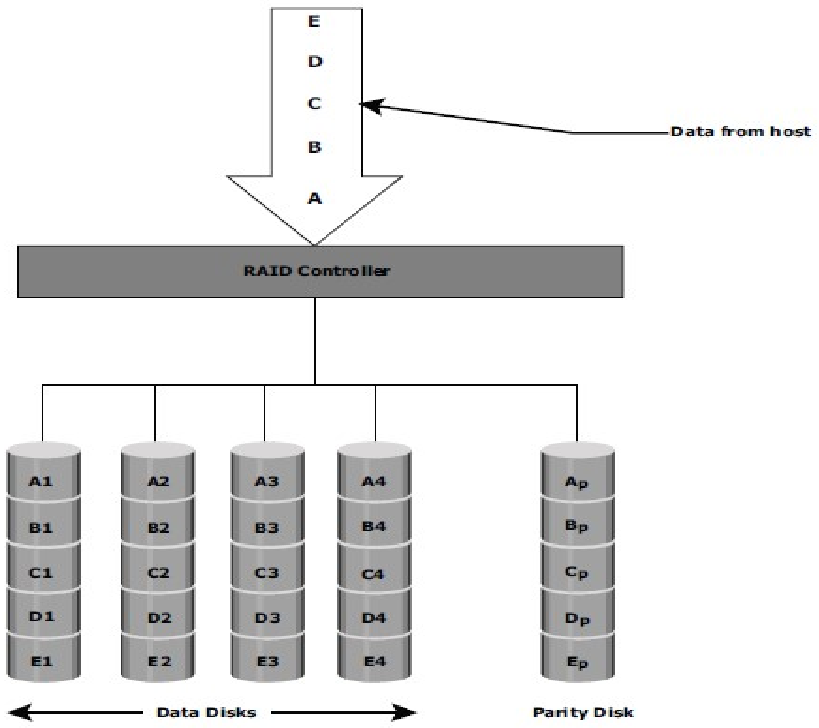


Figure 3-8: RAID 3

RAID 4

Similar to RAID 3, RAID 4 stripes data for high performance and uses parity for improved fault tolerance (refer to Figure 3-8). Data is striped across all disks except the parity disk. Parity information is stored on a dedicated disk so that the data can be rebuilt if a drive fails. Striping is done at the block level.

Unlike RAID 3, data disks in RAID 4 can be accessed independently so that specific data elements can be read or written on single disk without read or write of an entire stripe. RAID 4 provides good read throughput and reasonable write throughput.

RAID 5

RAID 5 is a very versatile RAID implementation. It is similar to RAID 4 because it uses striping and the drives (strips) are independently accessible.

The difference between RAID 4 and RAID 5 is the **parity location**. In RAID 4, parity is written to a dedicated drive, creating a write bottleneck for the parity disk. In RAID 5, **parity is distributed** across all disks. The distribution of parity in RAID 5 overcomes the write bottleneck. Figure 3-9 illustrates the RAID 5 implementation.

RAID 5 is preferred for messaging, data mining, medium-performance media serving, and relational database management system (RDBMS) implementations in which database administrators (DBAs) optimize data access.

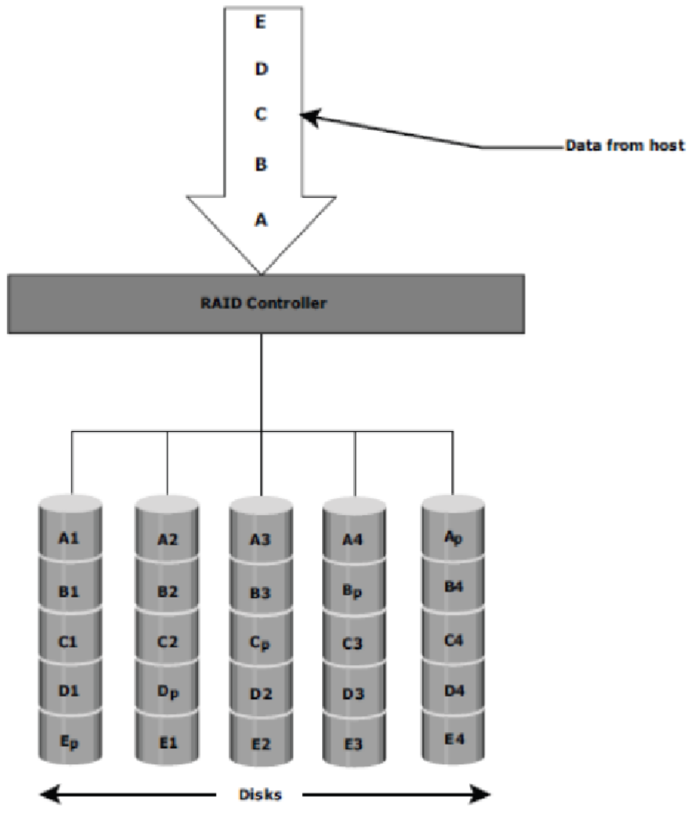


Figure 3-9: RAID 5

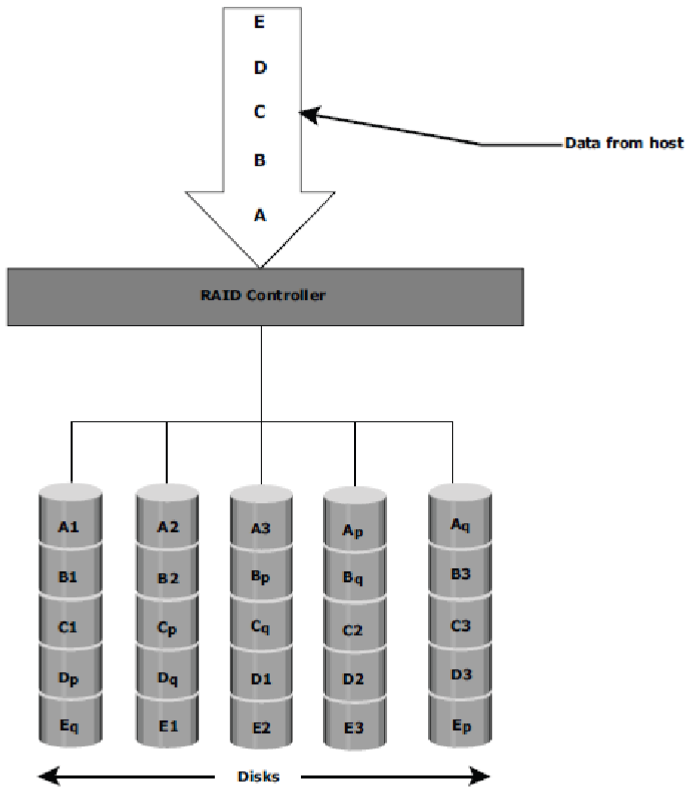


Figure 3-10: RAID 6

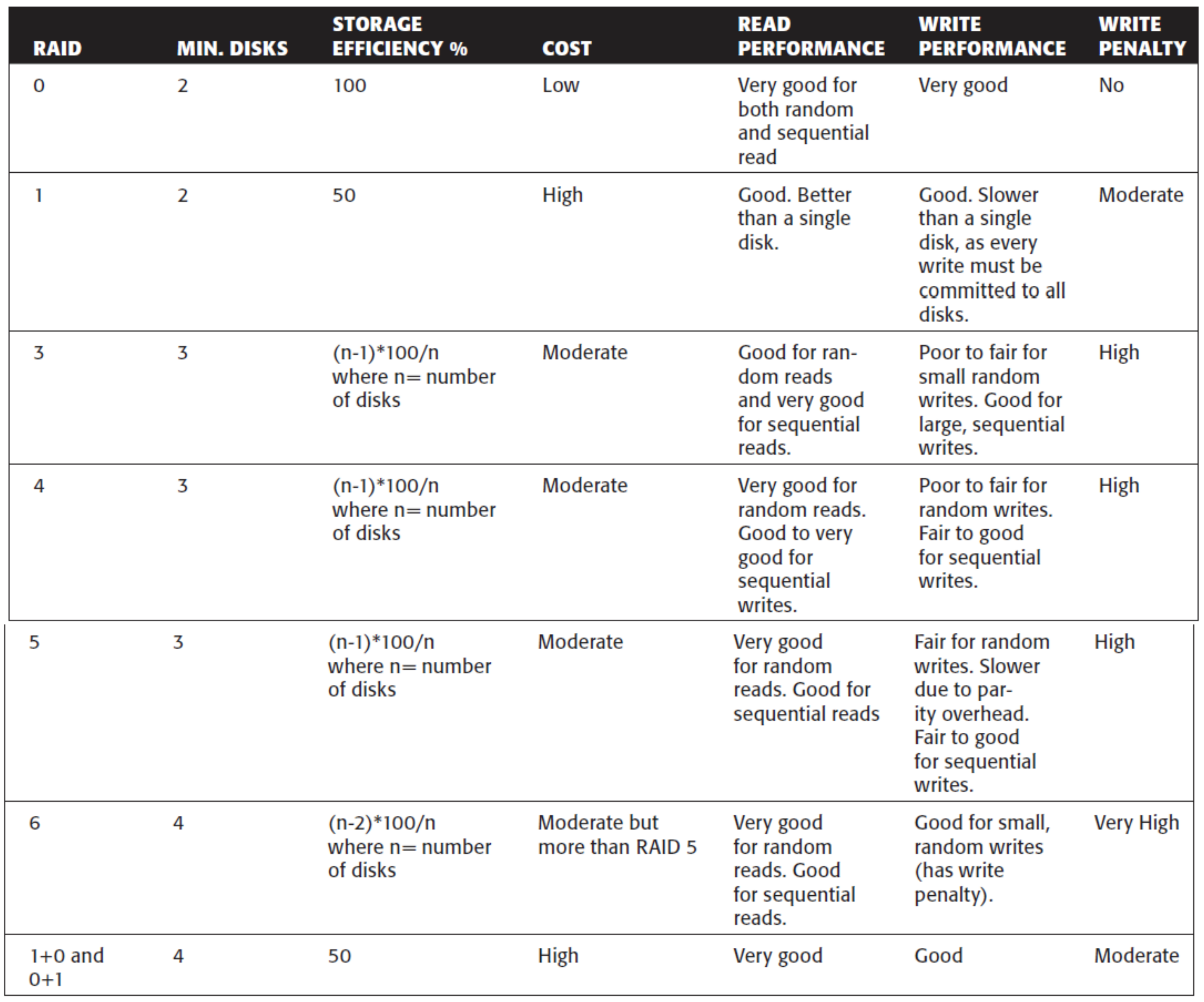
RAID 6

RAID 6 includes a second parity element to enable survival in the event of the failure of two disks in a RAID group (Figure 3-10). Therefore, a RAID 6 implementation requires at least four disks. RAID 6 distributes the parity across all the disks.

The write penalty in RAID 6 is more than that in RAID 5; therefore, RAID 5 writes perform better than RAID 6. The rebuild operation in RAID 6 may take longer than that in RAID 5 due to the presence of two parity sets.

RAID Comparison

Table 3-2: Comparison of Different RAID Types



RAID	MIN. DISKS	STORAGE EFFICIENCY %	COST	READ PERFORMANCE	WRITE PERFORMANCE	WRITE PENALTY
0	2	100	Low	Very good for both random and sequential read	Very good	No
1	2	50	High	Good. Better than a single disk.	Good. Slower than a single disk, as every write must be committed to all disks.	Moderate
3	3	(n-1)*100/n where n= number of disks	Moderate	Good for ran-dom reads and very good for sequential reads.	Poor to fair for small random writes. Good for large, sequential writes.	High
4	3	(n-1)*100/n where n= number of disks	Moderate	Very good for random reads. Good to very good for sequential writes.	Poor to fair for random writes. Fair to good for sequential writes.	High
5	3	(n-1)*100/n where n= number of disks	Moderate	Very good for random reads. Good for sequential reads	Fair for random writes. Slower due to par-ity overhead. Fair to good for sequential writes.	High
6	4	(n-2)*100/n where n= number of disks	Moderate but more than RAID 5	Very good for random reads. Good for sequential reads.	Good for small, random writes (has write penalty).	Very High
1+0 and 0+1	4	50	High	Very good	Good	Moderate

RAID Impact on Disk Performance

When choosing a RAID type, it is imperative to consider the impact to disk performance and application 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 ***write penalty***.

Figure 3-11 illustrates a single write operation on RAID 5 that contains a group of five disks. Four of these disks are used for data and one is used for parity.

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

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

Here, D1 to D4 is striped data across the RAID group of five disks.

Whenever controller performs a write I/O, parity must be computed by reading the old parity (Ep old) and the old data (E4 old) from the disk, i.e. two read I/Os.

The new parity (Ep new) is computed as follows:

Ep new = Ep old – E4 old + E4 new (XOR operations)

After computing the new parity, controller completes write I/O by writing the new data and new parity onto the disks, amounting to two write I/Os.

Therefore, controller performs two disk reads and two disk writes for every write operation, and the write penalty in RAID 5 implementations is 4.

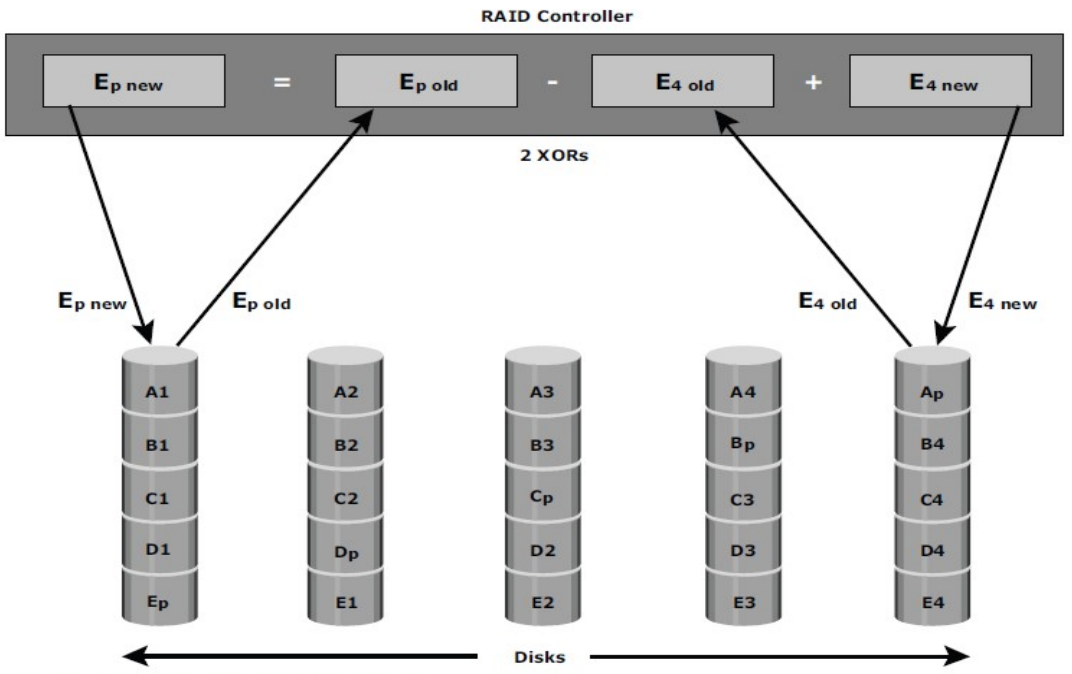


Figure 3-11: Write penalty in RAID 5

3.5.1 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 of computing 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 = 0.6 × 5,200 + **4** × (0.4 × 5,200) [because the write penalty for RAID 5 is 4]

= 3,120 + 4 × 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 × 5,200 + **2** × (0.4 × 5,200) [because every write manifests as two writes to the disks]

= 3,120 + 2 × 2,080

= 3,120 + 4,160

= 7,280 IOPS

Computed disk load determines the number of disks required for the application.

If in this example an HDD with a specification of a maximum 180 IOPS for the application needs to be used, the number of disks required to meet the workload for the RAID configuration as follows:

RAID 5: 11,440 / 180 = 64 disks

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

3.6 Hot Spares

A ***hot spare*** refers to a spare HDD in a RAID array that temporarily replaces a failed HDD of a RAID set.

A hot spare takes the identity of the failed HDD in the array.

One of the following methods of data recovery is performed depending on the RAID implementation:

1. If parity RAID is used, then the data is rebuilt onto the hot spare from the parity and the data on the surviving HDDs in the RAID set.

2. If mirroring is used, then the data from the surviving mirror is used to copy the data.

When the failed HDD is replaced with a new HDD, one of the following takes place:

1. The hot spare replaces the new HDD permanently. This means that it is no longer a hot spare, and a new hot spare must be configured on the array.

2. When a new HDD is added to the system, data from the hot spare is copied to it. The hot spare returns to its idle state, ready to replace the next failed drive.

A hot spare should be large enough to accommodate data from a failed drive. System can implement multiple hot spares to improve data availability.

A hot spare can be configured as ***automatic*** or ***user initiated**,* which specifies how it will be used in the event of disk failure.

In an ***automatic configuration***, when the recoverable error rates for a disk exceed a predetermined threshold, the disk subsystem tries to copy data from the failing disk to the hot spare automatically. If this task is completed before the damaged disk fails, then the subsystem switches to the hot spare and marks the failing disk as unusable.

Intelligent Storage System Introduction

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

With advancements in technology, a new breed of storage solutions known as an *intelligent storage system* has evolved. The intelligent storage systems detailed in this chapter are the feature-rich RAID arrays that provide highly optimized I/O processing capabilities. These arrays have an operating environment that controls the management, allocation, and utilization of storage resources.

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 the 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 cache.

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.

1. Front-end ports: enable hosts to connect to the intelligent storage system. Each front-end port has processing logic that executes the appropriate transport protocol, such as SCSI, Fibre Channel, or iSCSI, for storage connections. Redundant ports are provided on the front end for high availability.

2. Front-end controllers: route data to and from cache via the internal data bus.

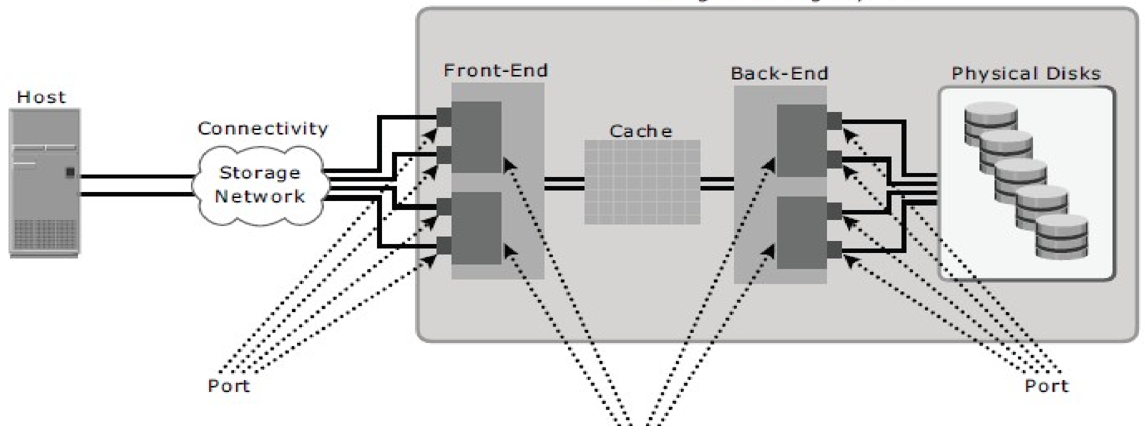


Figure 4-1: Components of an intelligent storage system

Front-End Command Queuing: *Command queuing* is a technique implemented on front-end controllers. It determines the execution order of received commands and can reduce unnecessary drive head movements and improve disk performance.

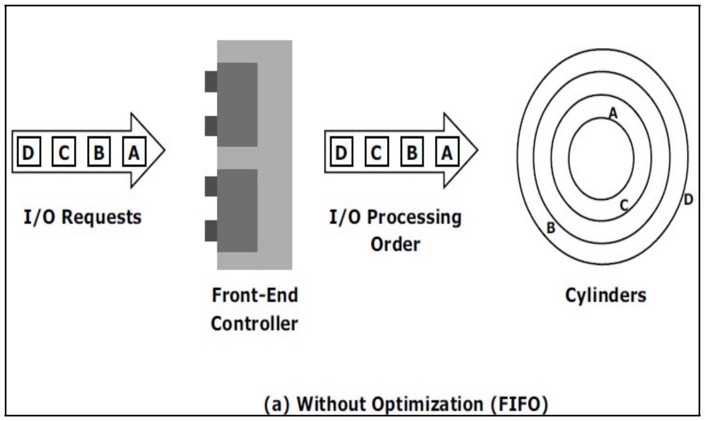
When a command is received for execution, the command queuing algorithms assigns a tag that defines a sequence in which commands should be executed. With command queuing, multiple commands can be executed concurrently based on the organization of data on the disk, regardless of the order in which the commands were received.

The most commonly used command queuing algorithms are as follows:

1. First In First Out (FIFO)**:** This is the default algorithm where commands are executed in the order in which they are received (Figure 4-2 [a]). There is no reordering of requests for optimization; therefore, it is inefficient in terms of performance.

2. Seek Time Optimization: Commands are executed based on optimizing read/write head movements, which may result in reordering of commands. Without seek time optimization, the commands are executed in the order they are received. For example, as shown in Figure 4-2(a), the commands are executed in the order A, B, C and D. The radial movement required by the head to execute C immediately after A. With seek time optimization, the command execution sequence would be A, C, B and D, as shown in Figure 4-2(b).

3. Access Time Optimization: Commands are executed based on the combination of seek time optimization and an analysis of rotational latency for optimal performance.



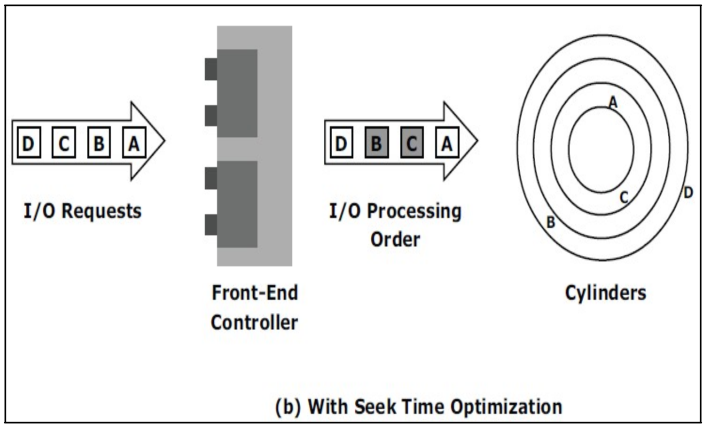


Figure 4-2: Front-end command queuing

Cache

Cache is an important component that enhances the I/O performance in an intelligent storage system. 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 physical disks, which are the slowest components of an intelligent storage system. Accessing data from cache takes less than a millisecond. Write data is placed in cache and then written to disk. After the data is securely placed in cache, the host is acknowledged immediately.

Structure of Cache: Cache is organized into pages or slots, 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 while tag RAM tracks the location of the data in the data store (see Figure 4-3) and in disk.

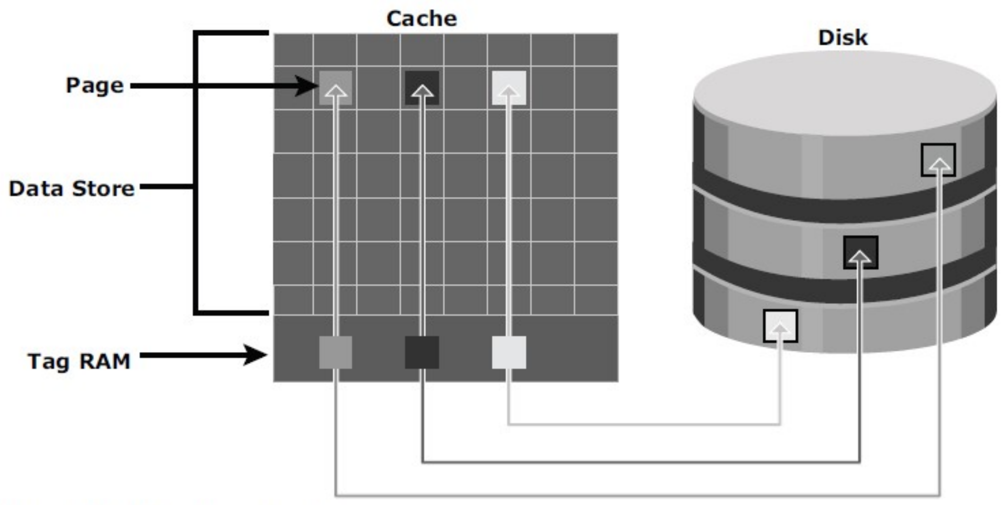


Figure 4-3: Structure of cache

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 or not. 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 front-end controller accesses 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 (see Figure 4-4[a]). 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 ***read cache miss*** *or* ***read miss*** and the data must be read from the disk (see Figure 4-4[b]).

The back-end controller accesses the appropriate disk and retrieves the requested data. Data is then placed in cache and is finally sent to the host through the front-end controller. Cache misses increase I/O response time.

A ***pre-fetch,*** 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 pre-fetch sizes.

In ***fixed pre-fetch***, the intelligent storage system pre-fetches a fixed amount of data. It is most suitable when I/O sizes are uniform.

In ***variable pre-fetch***, the storage system pre-fetches an amount of data in multiples of the size of the host request.

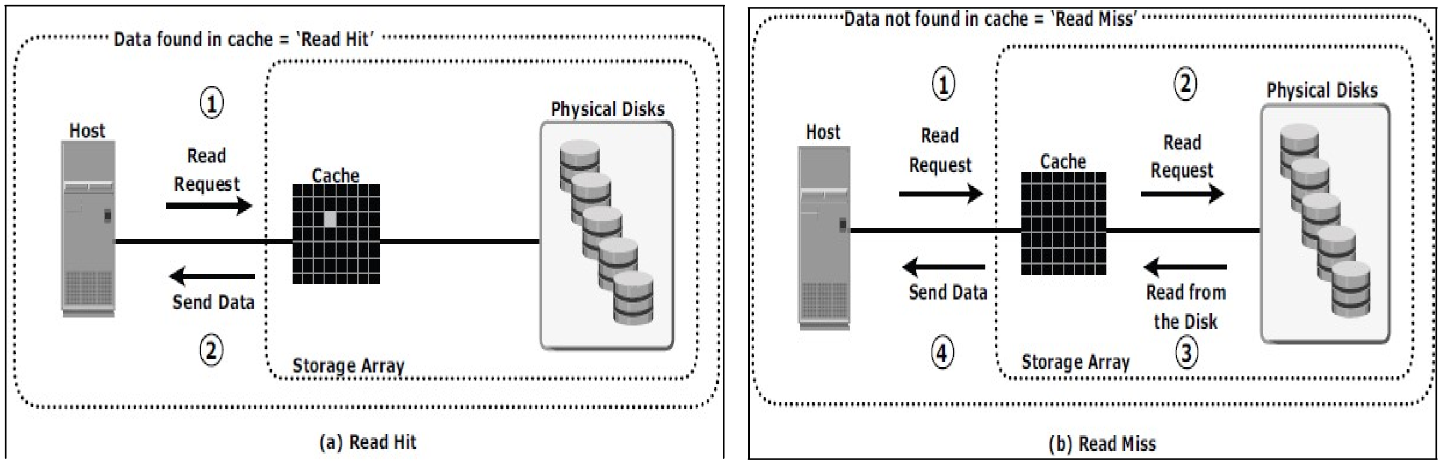


Figure 4-4: Read hit and read miss

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 less time (from the host's perspective) than it would take to write directly to disk.

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

1. 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, as the write operations are isolated from the mechanical delays of the disk. However, uncommitted data is at risk of loss in the event of cache failures.

2. 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 write response time is longer because of the disk operations.

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, as only one global set of addresses has to be managed.

Cache Management: Cache is a finite and expensive resource that needs proper management. 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:

1. 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 which hasn't been accessed for a while will not be requested by the host.

2. Most Recently Used (MRU): In MRU, 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, storage system must take action to **flush dirty pages** (data written into the cahce but not yet written to the disk) to manage availability. Flushing is the process of committing 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 cache utilization level at which the storage system starts highspeed flushing of cache data.

Low watermark (LWM) is the point at which the storage system stops the high-speed or forced flushing and returns to idle flush behavior.

The cache utilization level, as shown in Figure 4-5, drives the mode of flushing to be used:

1. Idle flushing: Occurs continuously, at a modest rate, when the cache utilization level is between the high and low watermark.

2. High watermark flushing: Activated when cache utilization hits the high watermark. The storage system dedicates some additional resources to flushing. This type of flushing has minimal impact on host I/O processing.

3. 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, dirty pages are forcibly flushed to disk.

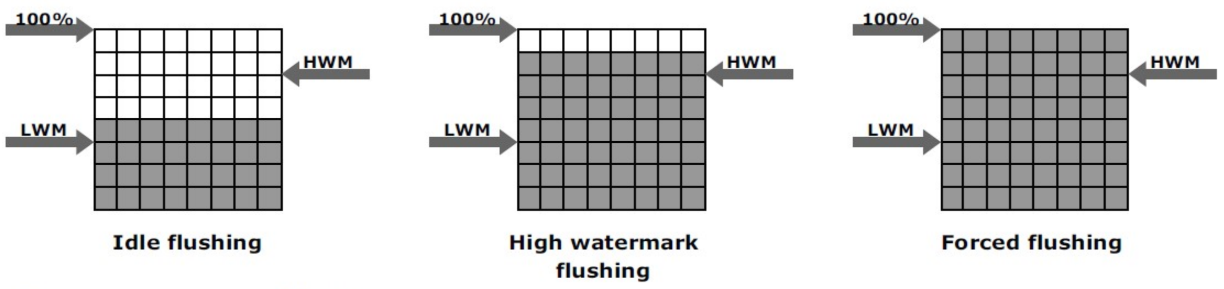


Figure 4-5: Types of flushing

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

1. Cache mirroring: Each write to cache is held in two different memory locations on two independent memory cards. In the event of a cache failure, 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, in the event of a cache failure, the data can still be accessed from the disk. As 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 data in two different cache locations must be identical at all times. It is the responsibility of the array operating environment to ensure coherency.

2. Cache vaulting: In this case, 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 and then written to the intended disks.

Cache is exposed to the risk of uncommitted data loss due to power failure. This problem can be addressed in various ways: powering the memory with a battery until AC power is restored or using battery power to write the cache content to the disk. In the event of extended power failure, using batteries is not a viable option because in intelligent storage systems, large amounts of data may need to be committed to numerous disks.

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.

Physical Disk

A physical disk stores data persistently. Disks are connected to the back-end with either SCSI or a Fibre Channel interface (discussed in subsequent chapters). An intelligent storage system enables the use of a mixture of SCSI or Fibre Channel drives and IDE/ATA drives.

Logical Unit Number: Physical drives or groups of RAID protected drives can be logically split into volumes known as logical volumes, commonly referred to as ***Logical Unit Numbers*** **(LUNs)**. The use of LUNs improves disk utilization.

For example, without the use of LUNs, a host requiring only 200 GB could be allocated an entire 1TB physical disk. Using LUNs, only the required 200 GB would be allocated to the host, allowing the remaining 800 GB to be allocated to other hosts.

In the case of RAID protected drives, these logical units are slices of RAID sets and are spread across all the physical disks belonging to that set. The logical units can also be seen as a logical partition of a RAID set that is presented to a host as a physical disk.

For example, Figure 4-6 shows a RAID set consisting of five disks that have been sliced, or partitioned, into several LUNs. LUNs 0 and 1 are shown in the figure.

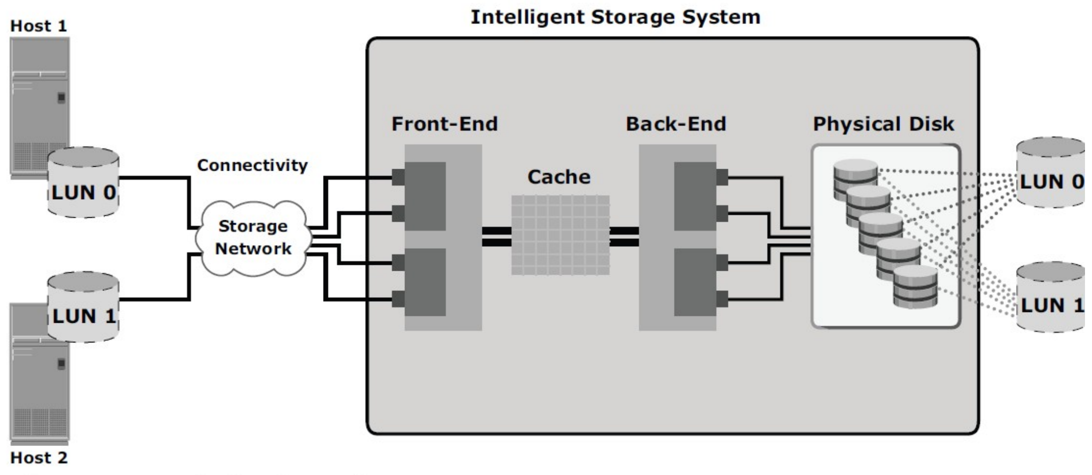


Figure 4-6: Logical unit number

LUNs 0 and 1 are presented to hosts 1 and 2, respectively, as physical volumes for storing and retrieving data. Usable capacity of the physical volumes is determined by the RAID type of the RAID set.

The capacity of a LUN can be expanded by aggregating other LUNs with it. The result of this aggregation is a larger capacity LUN, known as a *meta-LUN*.

LUN Masking: *LUN masking* is a process that provides data access control by defining which LUNs a host can access. LUN masking function is typically implemented at the front end controller. This ensures that volume access by servers is controlled appropriately, preventing unauthorized or accidental use in a distributed 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.

Intelligent Storage Array

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

1. High-end storage systems

2. Midrange storage systems

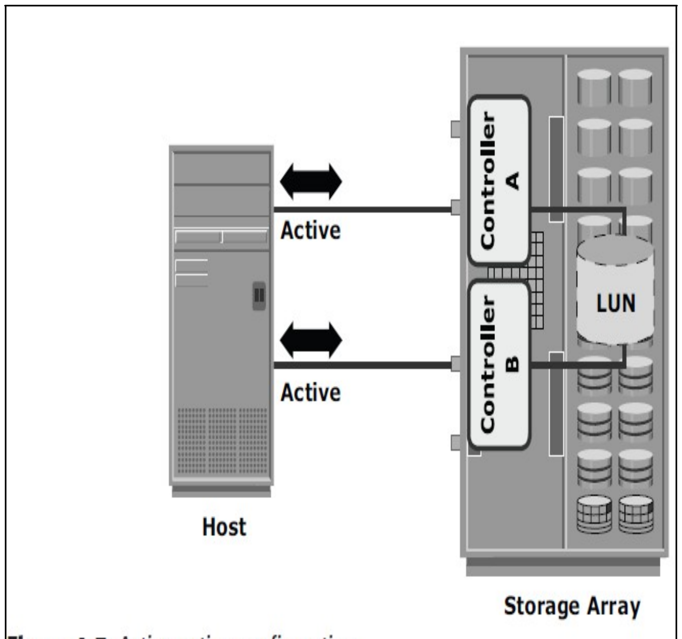


Figure 4-7: Active-active configuration

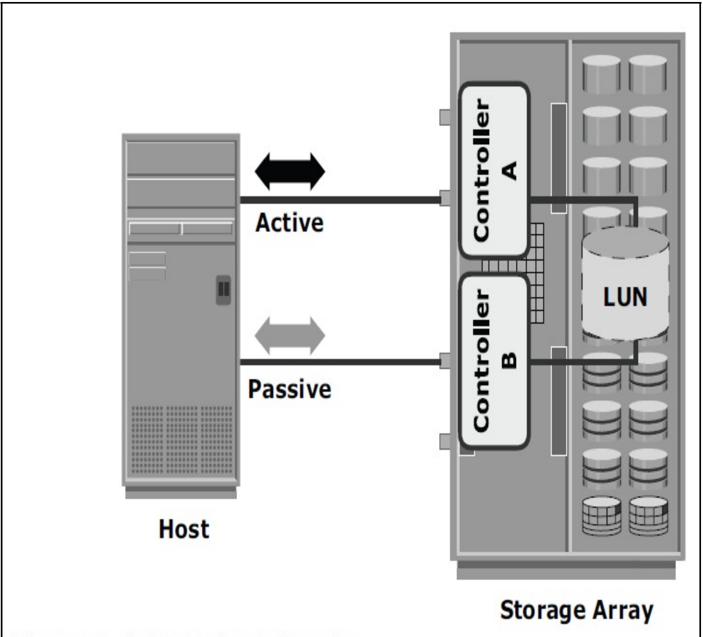


Figure 4-8: Active-passive configuration

High-end Storage Systems

High-end storage systems, referred to as ***active-active arrays,*** are aimed at large enterprises for centralizing corporate data. These arrays 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 across any of the available paths (see Figure).

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

1. Large storage capacity
2. Large amounts of cache to service host I/Os optimally
3. Fault tolerance architecture to improve data availability

4. Connectivity to mainframe computers and open systems hosts
5. Availability of multiple front-end ports and interface protocols to serve a large number of hosts
6. Availability of multiple back-end Fibre Channel or SCSI RAID controllers to manage disk processing
7. Scalability to support increased connectivity, performance, and storage capacity requirements
8. Ability to handle large amounts of concurrent I/Os from a number of servers and applications
9. Support for array-based local and remote replication

Midrange Storage System

Midrange storage systems are also referred to as ***active-passive arrays*** and they are best suited for small- and medium-sized enterprises. In an active-passive array, a host can perform I/Os to a LUN only through the paths to the owning controller of that LUN. These paths are called ***active paths.*** The other paths are passive with respect to this LUN.

As shown in Figure 4-8, the host can perform reads or writes to the LUN only through the path to controller A, as 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 arrays are designed to meet the requirements of small and medium enterprises; therefore, they host less storage capacity and global cache than active-active arrays. There are also fewer front-end ports for connection to servers. However, they ensure high redundancy and high performance for applications with predictable workloads. They also support array-based local and remote replication.

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:

1. **Just-in-time information to business users:** Information must be available to business users when they need it. The explosive growth in online storage, proliferation of new servers and applications, spread of mission-critical data throughout enterprises, and demand for 24 × 7 data availability are some of the challenges that need to be addressed.

2. **Integration of information infrastructure with business processes:** The storage infrastructure should be integrated with various business processes without compromising its security and integrity.

3. **Flexible and resilient storage architecture:** The storage infrastructure must provide flexibility and resilience that aligns with changing business requirements. Storage should scale without compromising performance requirements of the applications and, at the same time, the total cost of managing information must be low.

Module III: Topologies

A: Components and Topologies

Chapter Objective: SAN is a high speed, dedicated network of servers and shared storage devices. Traditionally connected over Fibre Channel (FC) networks, a SAN forms a single-storage pool and facilitates data centralization and consolidation. SAN meets the storage demands efficiently with better economies of scale. A SAN also provides effective maintenance and protection of data.

This chapter provides detailed insight into the FC technology on which a SAN is deployed and also reviews SAN design and management fundamentals.

Fibre Channel: Overview

The FC architecture forms the fundamental construct of the SAN infrastructure. *Fibre Channel* is a high-speed network technology that runs on high-speed optical fiber cables (preferred for front-end SAN connectivity) and serial copper cables (preferred for back-end disk connectivity). The FC technology was created to meet the demand for increased speeds of data transfer among computers, servers, and mass storage subsystems.

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 INCITS (International Committee for Information Technology Standards), is responsible for Fibre Channel interfaces. T11 (previously known as X3T9.3) has been producing interface standards for high performance and mass storage applications since the 1970s.

Higher data transmission speeds are an important feature of the FC networking technology. The initial implementation offered throughput of 100 MB/s (equivalent to raw bit rate of 1Gb/s i.e. 1062.5 Mb/s in Fibre Channel), which was greater than the speeds of Ultra SCSI (20 MB/s) commonly used in DAS environments. FC in full-duplex mode could sustain throughput of 200 MB/s. In comparison with Ultra-SCSI, FC is a significant leap in storage networking technology.

Latest FC implementations of 8 GFC (Fibre Channel) offers throughput of 1600 MB/s (raw bit rates of 8.5 Gb/s), whereas Ultra320 SCSI is available with a throughput of 320 MB/s. The FC architecture is highly scalable and theoretically a single FC network can accommodate approximately 15 million nodes.

The SAN and Its Evolution

A *storage area network (SAN)* carries data between servers (also known as *hosts*) and storage devices through fibre channel switches (see Figure 6-1). A SAN enables storage consolidation and allows storage to be shared across multiple servers. It enables organizations to connect geographically dispersed servers and storage.

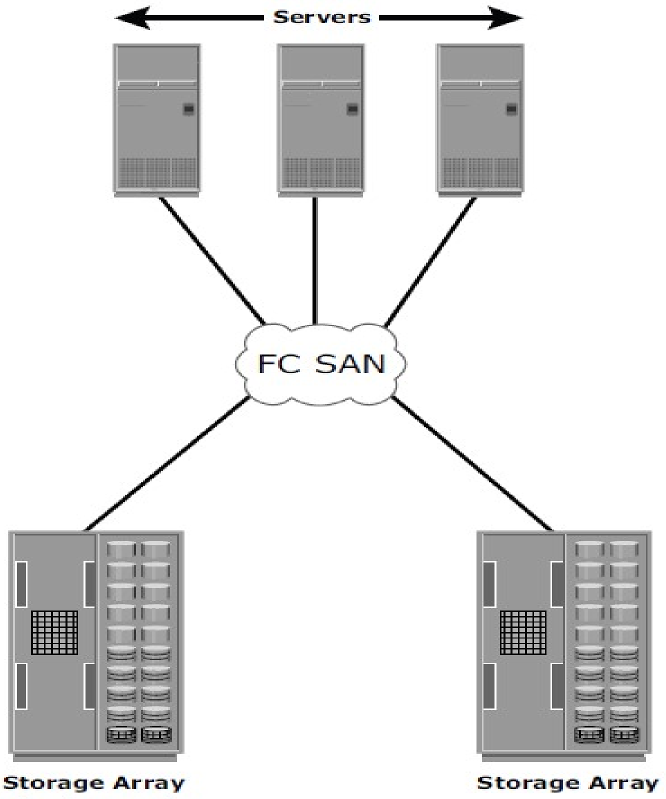


Figure 6-1: SAN implementation

A SAN provides the physical communication infrastructure and enables secure and robust communication between host and storage devices. The SAN management interface organizes connections and manages storage elements and hosts.

In its earliest implementation, the SAN was a simple grouping of hosts and the associated storage that was connected to a network using a hub as a connectivity device. This configuration of a SAN is known as a *Fibre Channel Arbitrated Loop (FC-AL)*, which is detailed later in the chapter. 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*. The switched fabric topologies improved connectivity and performance, which enabled SANs to be highly scalable. This enhanced data accessibility to applications across the enterprise. FC-AL has been abandoned for SANs due to its limitations, but still survives as a disk-drive interface. Figure 6-2 illustrates the FC SAN evolution from FC-AL to enterprise SANs.

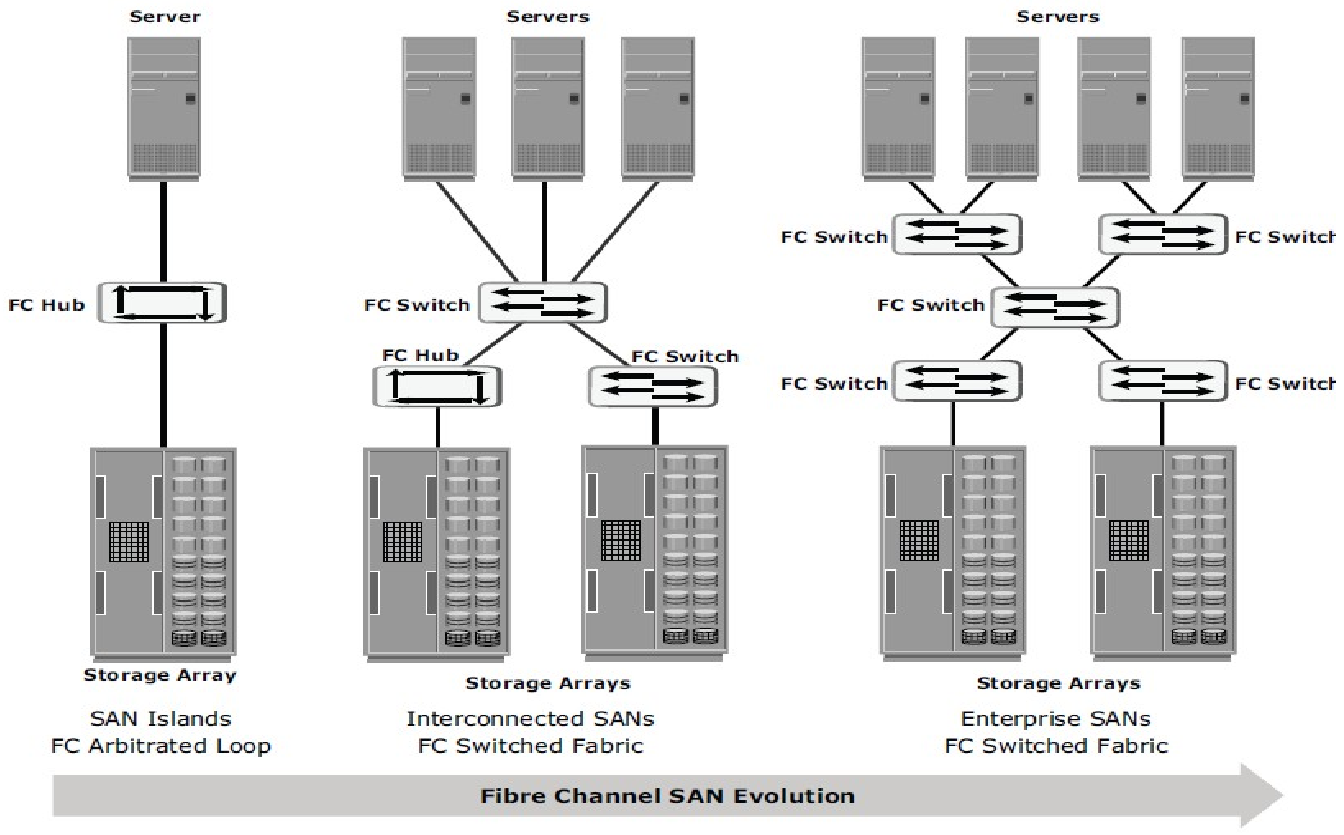


Figure 6-2: FC SAN evolution

Components of SAN

A SAN consists of three basic components: servers, network infrastructure, and storage. These components can be further broken down into the following key elements: node ports, cabling, interconnecting devices (such as FC switches or hubs), storage arrays, and SAN management software.

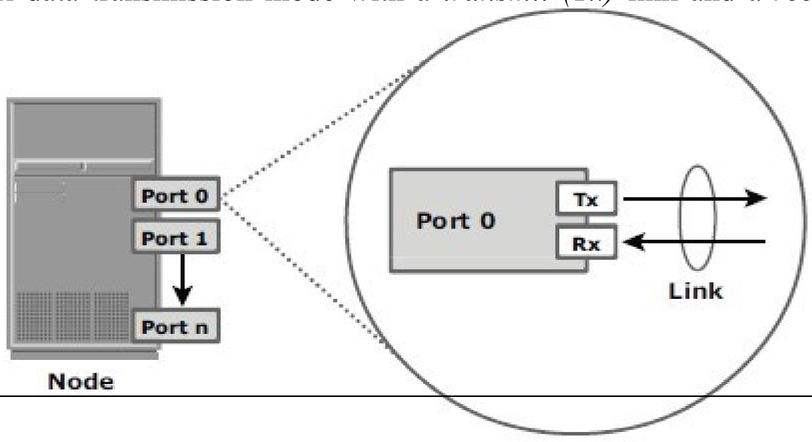
Node Ports

In fibre channel, devices such as hosts, storage and tape libraries are all referred to as *nodes.*

Each node is a source or destination of information for one or more nodes.

Each node requires one or more ports to provide a physical interface for communicating with other nodes. These ports are integral components of an HBA and the storage front-end adapters.

A port operates in full-duplex data transmission mode with a *transmit (Tx)* link and a *receive (Rx)* link (see Figure 6-3).



Cabling

SAN implementations use optical fiber cabling. Copper can be used for shorter distances for back-end connectivity, as it provides a better 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, multi-mode and single-mode. Multi-mode fiber (MMF) cable carries multiple beams of light projected at different angles simultaneously onto the core of the cable (see Figure 6-4 (a)).

Based on the bandwidth, multi-mode fibers are classified as OM1 (62.5µm), OM2 (50µm) and laseroptimized OM3 (50µm). 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 usually used for distances of up to 500 meters 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 (see Figure 6-4 (b)). These cables are available in diameters of 7–11 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 limits modal dispersion. Among all types of fibre cables, single-mode provides minimum signal attenuation over maximum distance (up to 10 km).

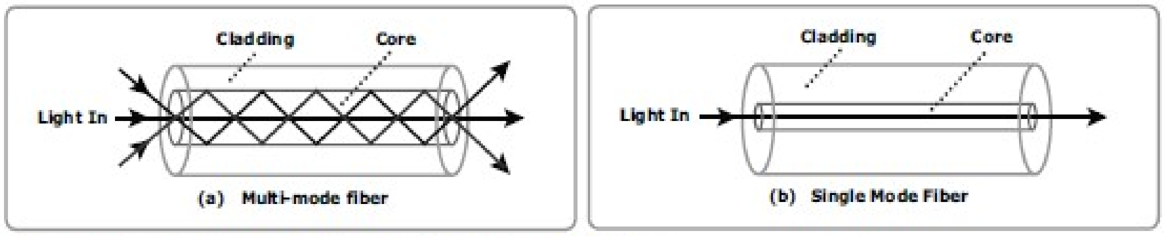


Figure 6-4: Multi-mode fiber and single-mode fiber

A Standard connector (SC) (see Figure 6-5 (a)) and a Lucent connector (LC) (see Figure 6-5 (b)) are two commonly used connectors for fiber optic cables. An SC is used for data transmission speeds up to 1 Gb/s, whereas an LC is used for speeds up to 4 Gb/s. Figure 6-6 depicts a Lucent connector and a Standard connector. A *Straight Tip (ST)* is a fiber optic connector with a plug and a socket that is locked with a half-twisted bayonet lock (see Figure 6-5 (c)). In the early days of FC deployment, fiber optic cabling predominantly used ST connectors. This connector is often used with Fibre Channel patch panels. The Small Form-factor Pluggable (SFP) is an optical transceiver used in optical communication. The standard SFP+ transceivers support data rates up to 10 Gb/s.

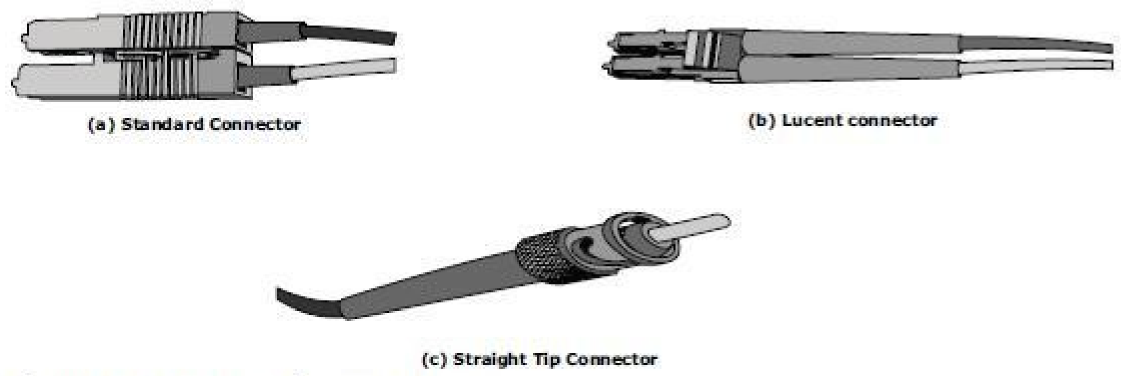


Figure 6-5: SC, LC, and ST connectors

Interconnect Devices

Hubs, switches, and directors are the interconnect devices commonly used in 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 bandwidth because data travels through all the connection points. Because of availability of low cost and high performance switches, hubs are no longer used in 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, resulting in bandwidth aggregation.

Directors are larger than switches and are deployed for data center implementations. The function of directors is similar to that of FC switches, but directors have higher port count and fault tolerance capabilities.

Storage Arrays

The fundamental purpose of a SAN is to provide host access to storage resources. The large storage capacities offered by modern storage arrays have been exploited in SAN environments for storage consolidation and centralization. SAN implementations complement the standard features of storage arrays by providing high availability and redundancy, improved performance, business continuity, and multiple host connectivity.

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 logical partitioning of the SAN, called *zoning*. In addition, the software provides management of typical SAN components such as HBAs, storage components, and interconnecting devices.

FC Connectivity

The FC architecture supports three basic interconnectivity options: point-to-point, arbitrated loop (FC-AL), and fabric connect.

Point-to-Point

Point-to-point is the simplest FC configuration — two devices are connected directly to each other, as shown in Figure 6-6.

This configuration provides a dedicated connection for data transmission between nodes. However, the point-to-point configuration offers limited connectivity, as only two devices can communicate with each other at a given time.

Moreover, it cannot be scaled to accommodate a large number of network devices.

Standard DAS uses point- to- point connectivity.

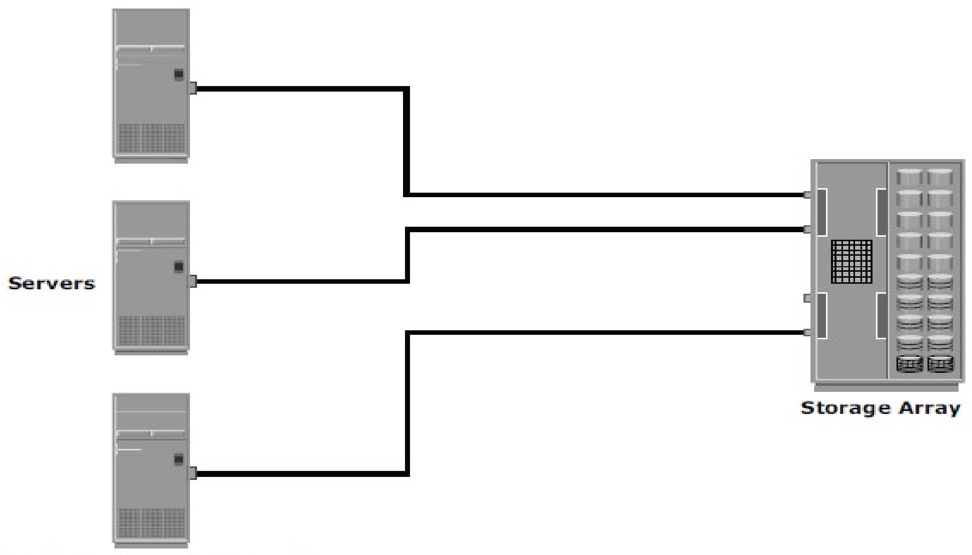


Figure 6-6: Point-to-point topology

Fibre Channel Arbitrated Loop

In the FC-AL configuration, devices are attached to a shared loop, as shown in Figure 6-7.

FC-AL has the characteristics of a token ring topology and a physical star topology.

In FC-AL, each device contends with other devices to perform I/O operations.

Devices on the loop must ―arbitrate‖ to gain control of the loop.

At any given time, only one device can perform I/O operations on the loop.

As a loop configuration, FC-AL can be implemented without any interconnecting devices by directly connecting one device to another in a ring through cables.

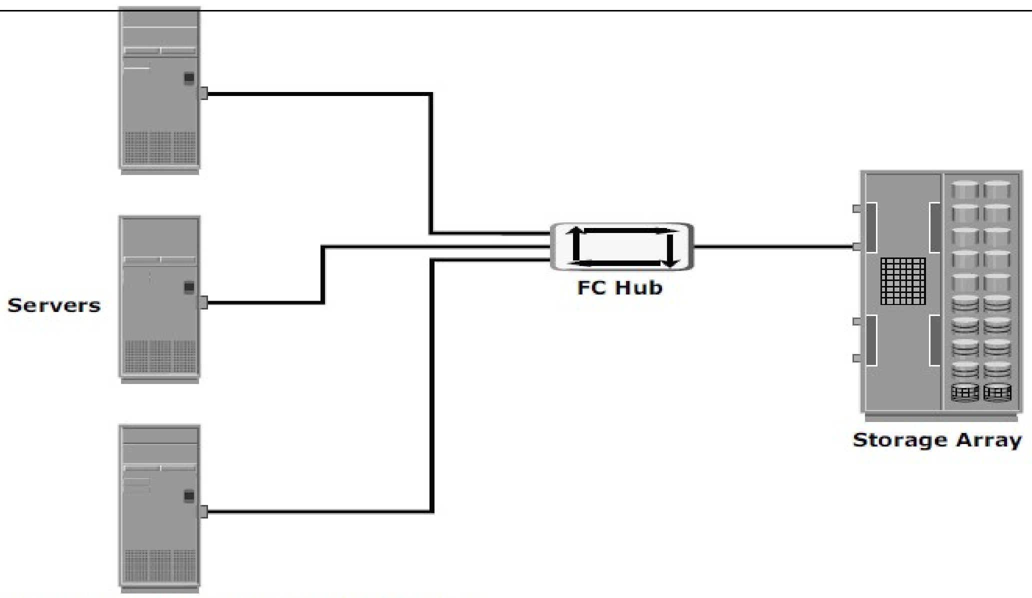


Figure 6-7: Fibre Channel arbitrated loop

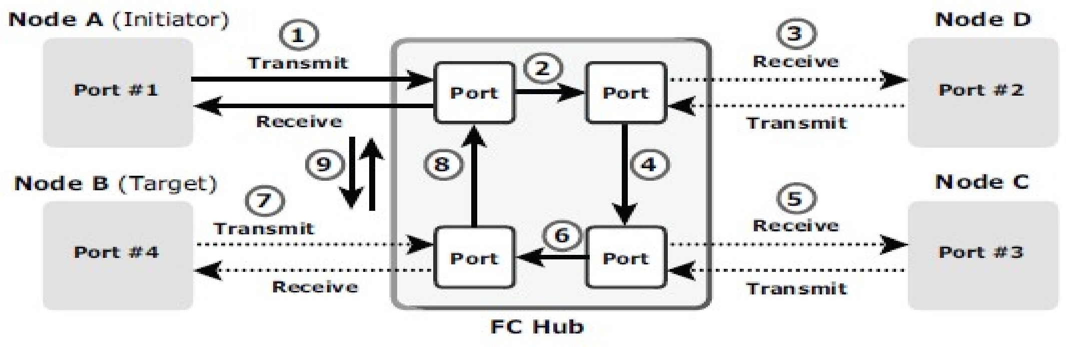
The FC-AL configuration has the following limitations in terms of scalability:

1. FC-AL shares the bandwidth in the loop. Only one device can perform I/O operations at a time. Because each device in a loop has to wait for its turn to process I/O request, the speed of data transmission is low in an FC-AL topology.
2. FC-AL uses 8-bit addressing. It can support up to 127 devices on a loop.
3. Adding or removing a device results in loop re-initialization, which can cause a momentary pause in loop traffic.

FC-AL Transmission

When a node in the FC-AL topology attempts to transmit data, the node sends an *arbitration (ARB)* frame to each node on the loop. If two nodes simultaneously attempt to gain control of the loop, the node with the highest priority is allowed to communicate with another node. This priority is determined on the basis of Arbitrated Loop Physical Address (AL-PA) and Loop ID.

When the initiator node receives the ARB request it sent, it gains control of the loop. The initiator then transmits data to the node with which it has established a virtual connection. Figure 6-8 illustrates the process of data transmission in an FC-AL configuration.



Node A want to communicate with Node B

1. High priority initiator, Node A inserts the ARB frame in the loop.
2. ARB frame is passed to the next node (Node D) in the loop.
3. Node D receives high priority ARB, therefore remains idle.
4. ARB is forwarded to next node (Node C) in the loop.
5. Node C receives high priority ARB, therefore remains idle.
6. ARB is forwarded to next node (Node B) in the loop.
7. Node B receives high priority ARB, therefore remains idle and
8. ARB is forwarded to next node (Node A) in the loop.
9. Node A receives ARB back; now it gains control of the loop and can start communicating with target Node B.

Figure 6-8: Data transmission in FC-AL

Fibre Channel Switched Fabric

Unlike a loop configuration, a Fibre Channel switched fabric (FC-SW) network provides interconnected devices, dedicated bandwidth, and scalability. The addition or removal of a device in a switched fabric is minimally disruptive; it does not affect the ongoing traffic between other devices. FC-SW is also referred to as *fabric connect*. A fabric is a logical space in which all nodes communicate with one another in a network. This virtual space can be created with a switch or a network of switches. Each switch in a fabric contains a unique domain identifier, which is part of the fabric's addressing scheme. In FC-SW, nodes do not share a loop; instead, data is transferred through a dedicated path between the nodes. Each port in a fabric has a unique 24-bit fibre channel address for communication. Figure 6-9 shows an example of FC-SW.

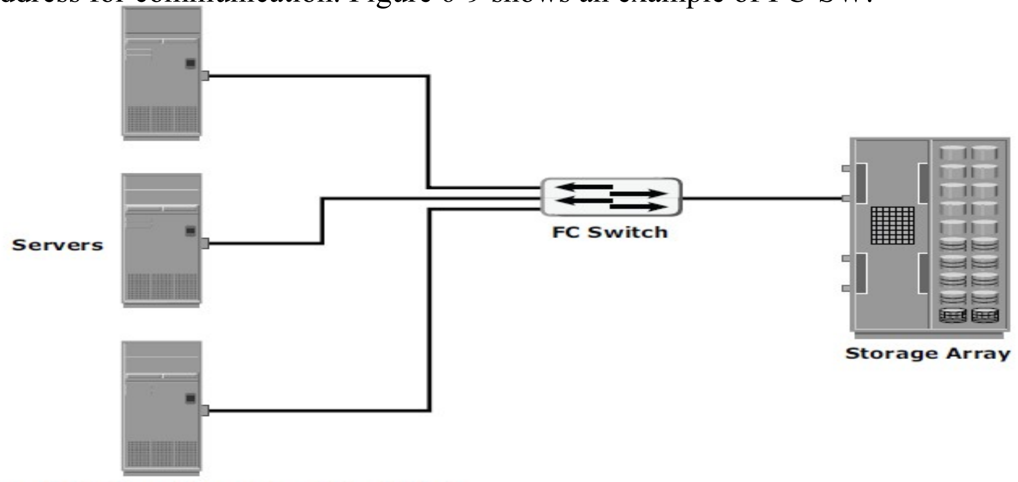


Figure 6-9: Fibre Channel switched fabric

When the number of tiers in a fabric increases, the distance that a fabric management message must travel to reach each switch in the fabric also increases. The increase in the distance also increases the time taken to propagate and complete a fabric reconfiguration event, such as the addition of a new switch, or a zone set propagation event (detailed later in this chapter). Figure 6-10 illustrates two-tier and three-tier fabric architecture.

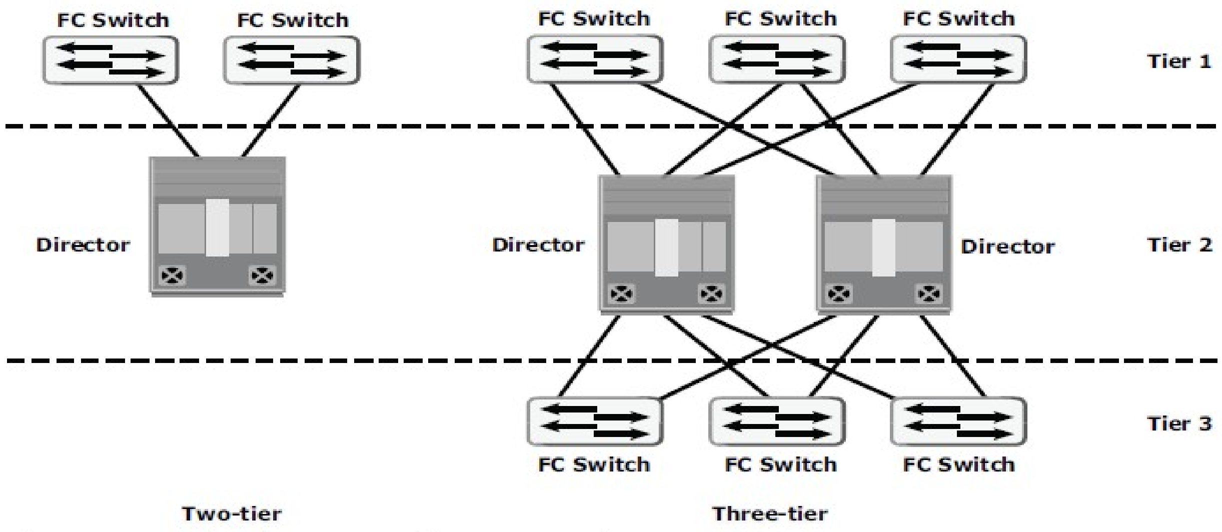


Figure 6-10: Tiered structure of FC-SW topology

FC-SW Transmission

FC-SW uses switches that are intelligent devices. They can switch data traffic from an initiator node to a target node directly through switch ports. Frames are routed between source and destination by the fabric. As shown in Figure 6-11, if node B wants to communicate with node D, Nodes should individually login first and then transmit data via the FC-SW. This link is considered a dedicated connection between the initiator and the target.

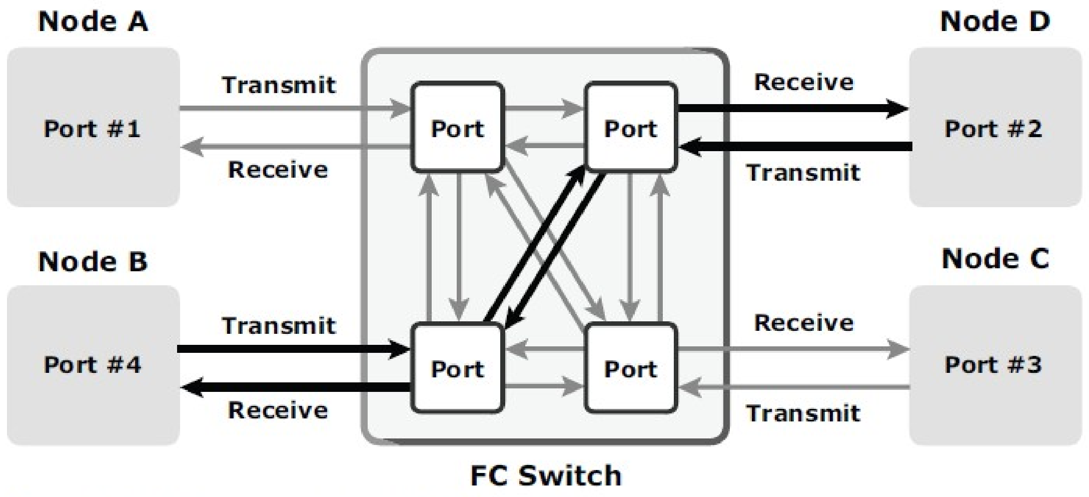


Figure 6-11: Data tansmission in FC-SW topology

Fibre Channel Ports

Ports are the basic building blocks of an FC network. Ports on the switch can be one of the following types:

1. N_port: An end point in the fabric. This port is also known as the *node port*. Typically, it is a host port (HBA) or a storage array port that is connected to a switch in a switched fabric.

2. NL_port: A node port that supports the arbitrated loop topology. This port is also known as the *node loop port*.

3. E_port: An FC port that forms the connection between two FC switches. This port is also known as the *expansion port*. The E_port on an FC switch connects to the E_port of another FC switch in the fabric through a link, which is called an *Inter-Switch Link (ISL)*. ISLs are used to transfer host-to-storage data as well as the fabric management traffic from one switch to another. ISL is also one of the scaling mechanisms in SAN connectivity.

4. F_port: A port on a switch that connects an N_port. It is also known as a *fabric port* and cannot participate in FC-AL.

5. FL_port: A fabric port that participates in FC-AL. This port is connected to the NL_ports on an FC-AL loop. A FL_port also connects a loop to a switch in a switched fabric. As a result, all NL_ports in the loop can participate in FC-SW. This configuration is referred to as a *public loop*. In contrast, an arbitrated loop without any switches is referred to as a *private loop*. A private loop contains nodes with NL_ports, and does not contain FL_port.

6. G_port: A generic port that can operate as an E_port or an F_port and determines its functionality automatically during initialization.

Figure 6-12 shows various FC ports located in the fabric.

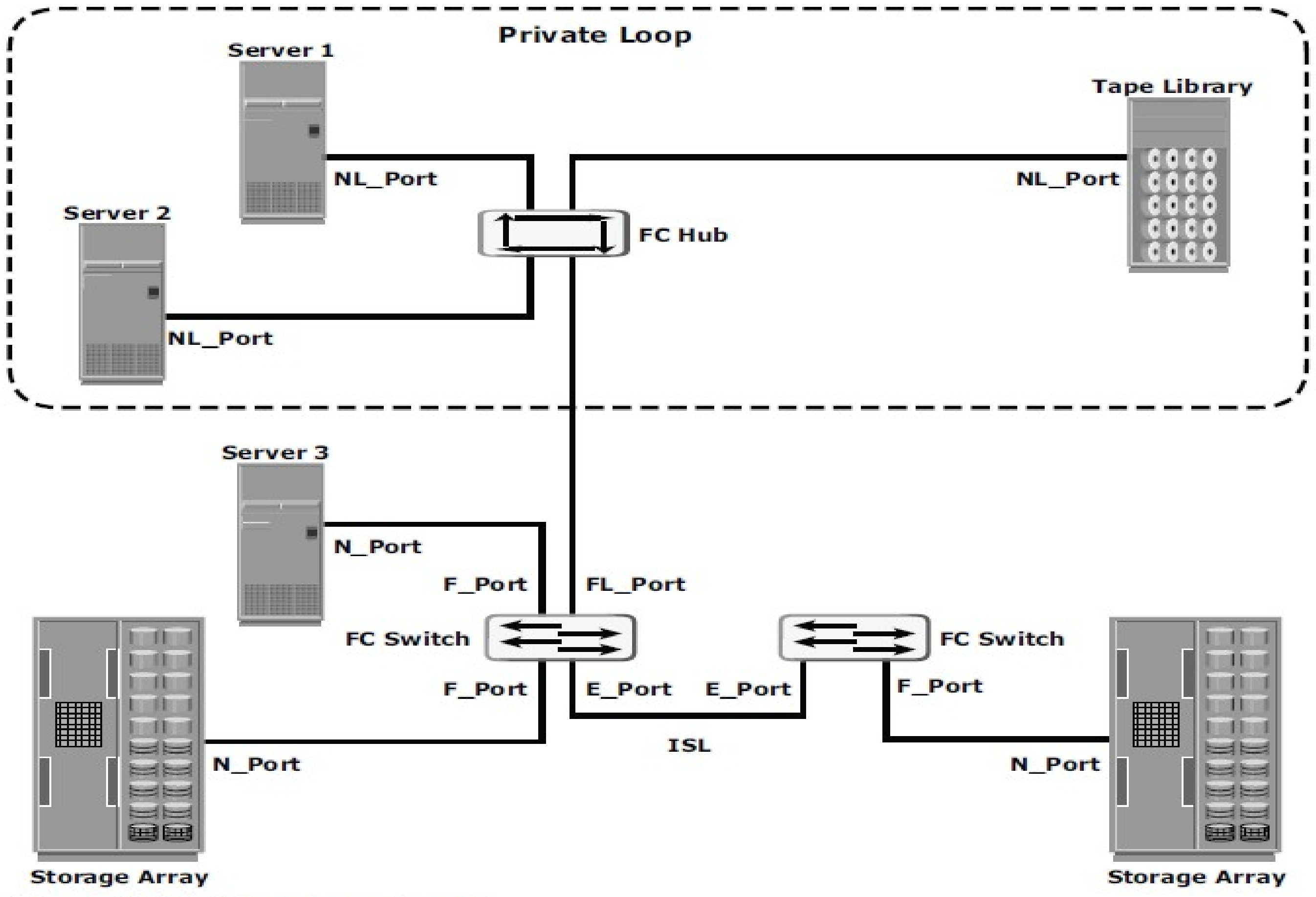


Figure 6-12: Fibre channel ports

Fibre Channel Architecture

The FC architecture represents true channel/network integration with standard interconnecting devices. Connections in a SAN are accomplished using FC. Channel technologies provide high levels of performance with low protocol overheads. Such performance is due to the static nature of channels and the high level of hardware and software integration provided by the channel technologies.

Fibre Channel Protocol (FCP) is the implementation of serial SCSI-3 over an FC network. In the FCP architecture, all external and remote storage devices attached to the SAN appear as local devices to the host operating system.

The key advantages of FCP are as follows:

1. Sustained transmission bandwidth over long distances.
2. Support for a larger number of addressable devices over a network. Theoretically, FC can support over 15 million device addresses on a network.
3. Exhibits the characteristics of channel transport and provides speeds up to 8.5 Gb/s (8 GFC).

Fibre Channel Protocol Stack

It is easier to understand a communication protocol by viewing it as a structure of independent layers. FCP defines the communication protocol in five layers: FC-0 through FC-4 (except FC-3 layer, which is not implemented). In a layered communication model, the peer layers on each node talk to each other through defined protocols. Figure 6-13 illustrates the fibre channel protocol stack.

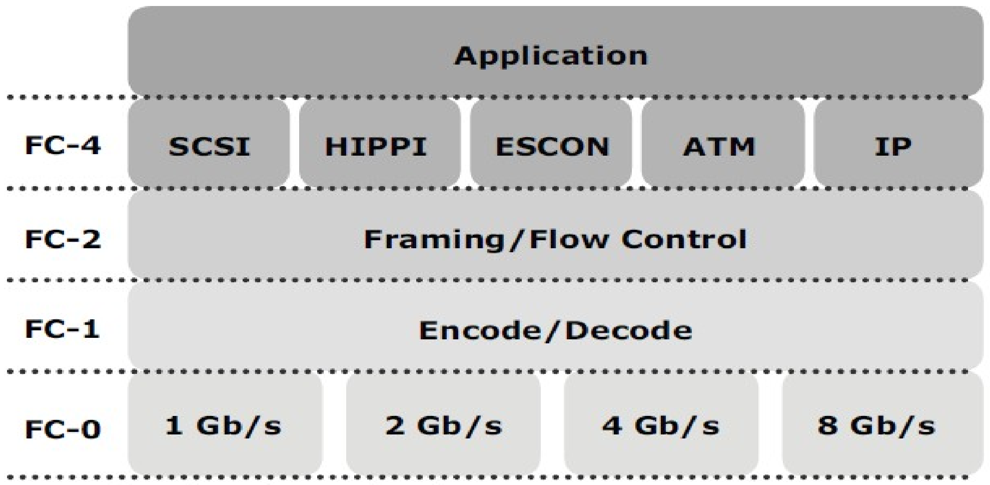


Figure 6-13: Fibre channel protocol stack

FC-4 Upper Layer Protocol: FC-4 is the uppermost layer in the FCP stack. This layer defines the application interfaces and the way Upper Layer Protocols (ULPs) are mapped to the lower FC layers. The FC standard defines several protocols that can operate on the FC-4 layer (see Figure 6-7). Some of the protocols include SCSI, HIPPI Framing Protocol, Enterprise Storage Connectivity (ESCON), ATM, and IP.

***FC-2 Transport Layer**:* The FC-2 is the transport layer that contains the payload, addresses of the source and destination ports, and link control information. The FC-2 layer provides Fibre Channel addressing, structure, and organization of data (frames, sequences, and exchanges). It also defines fabric services, classes of service, flow control, and routing.

FC-1 Transmission Protocol: This layer defines the transmission protocol that includes serial encoding and decoding rules, special characters used, and error control. At the transmitter node, an 8-bit character is encoded into a 10-bit transmissions character. This character is then transmitted to the receiver node. At the receiver node, the 10-bit character is passed to the FC-1 layer, which decodes the 10-bit character into the original 8-bit character.

***FC-0 Physical Interface**:* FC-0 is the lowest layer in the FCP stack. This layer defines the physical interface, media, and transmission of raw bits. The FC-0 specification includes cables, connectors, and optical and electrical parameters for a variety of data rates. The FC transmission can use both electrical and optical media.

Fibre Channel Addressing

An FC address is dynamically assigned when a port logs on to the fabric. The FC address has a distinct format that varies according to the type of node port in the fabric. These ports can be an N_port and an NL_port in a public loop, or an NL_port in a private loop.

The first field of the FC address of an N_port contains the domain ID of the switch (see Figure 6-14). This is an 8-bit field. Out of the possible 256 domain IDs, 239 are available for use; the remaining 17 addresses are reserved for specific services. For example, FFFFFC is reserved for the name server, and FFFFFE is reserved for the fabric login service. The maximum possible number of N_ports in a switched fabric is calculated as 239 domains × 256 areas × 256 ports = 15,663,104 Fibre Channel addresses.

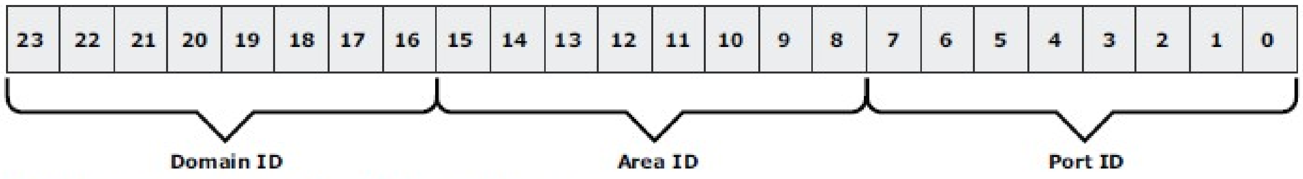


Figure 6-14: 24-bit FC address of N_port

FC Address of an NL_port

The FC addressing scheme for an NL_port differs from other ports. The two upper bytes in the FC addresses of the NL_ports in a private loop are assigned zero values. However, when an arbitrated loop is connected to a fabric through an FL_port, it becomes a public loop. In this case, an NL_port supports a fabric login. The two upper bytes of this NL_port are then assigned a positive value, called a *loop identifier*, by the switch. The loop identifier is the same for all NL_ports on a given loop.

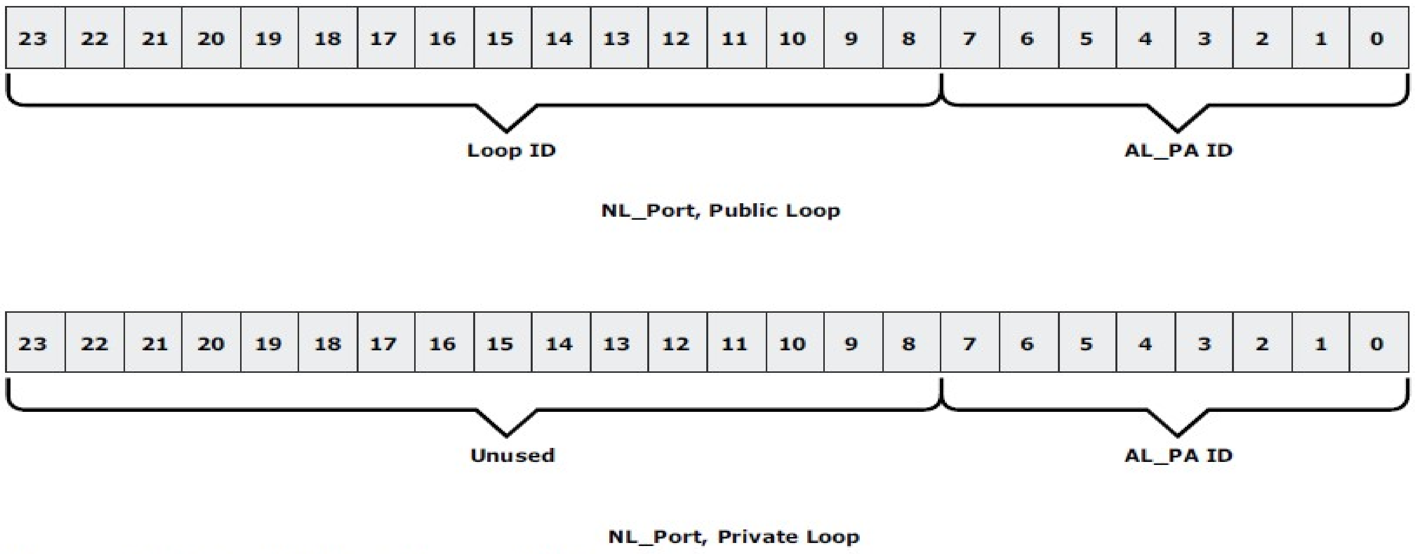


Figure 6-15: 24-bit FC address of NL_port

World Wide Names

Each device in the FC environment is assigned a 64-bit unique identifier called the *World Wide Name* (WWN). The Fibre Channel environment uses two types of WWNs: World Wide Node Name (WWNN) and World Wide Port Name (WWPN). Unlike an FC address, which is assigned dynamically, a WWN is a static name for each device on an FC network. WWNs are similar to the Media Access Control (MAC) addresses used in IP networking. WWNs are *burned* into the hardware or assigned through software. Several configuration definitions in a SAN use WWN for identifying storage devices and HBAs. The name server in an FC environment keeps the association of WWNs to the dynamically created FC addresses for nodes. Figure 6-16 illustrates the WWN structure for an array and the HBA.

World Wide Name - Array

5	0	0	6	0	1	6	0	0	0	6	0	0	1	B	2
0101	0000	0000	0110	0000	0001	0110	0000	0000	0000	0110	0000	0000	0001	1011	0010
	Company ID 24 bits						Port	Model Seed 32 bits							

World Wide Name - HBA

1	0	0	0	0	0	0	0	c	9	2	0	d	c	4	0
	Reserved 12 bits			Company ID 24 bits						Company Specific 24 bits					

Figure 6-16: World Wide Names

FC Frame

An FC frame (Figure 6-17) consists of five parts: *start of frame (SOF)*, *frame header*, *data field*, *cyclic redundancy check (CRC)*, and *end of frame (EOF)*. The SOF and EOF act as delimiters. In addition to this role, the SOF is a flag that indicates whether the frame is the first frame in a sequence of frames. The frame header is 24 bytes long and contains addressing information for the frame. It includes the following information: Source ID (S_ID), Destination ID (D_ID), Sequence ID (SEQ_ID), Sequence Count (SEQ_CNT), Originating Exchange ID (OX_ID), and Responder Exchange ID (RX_ID), in addition to some control fields.

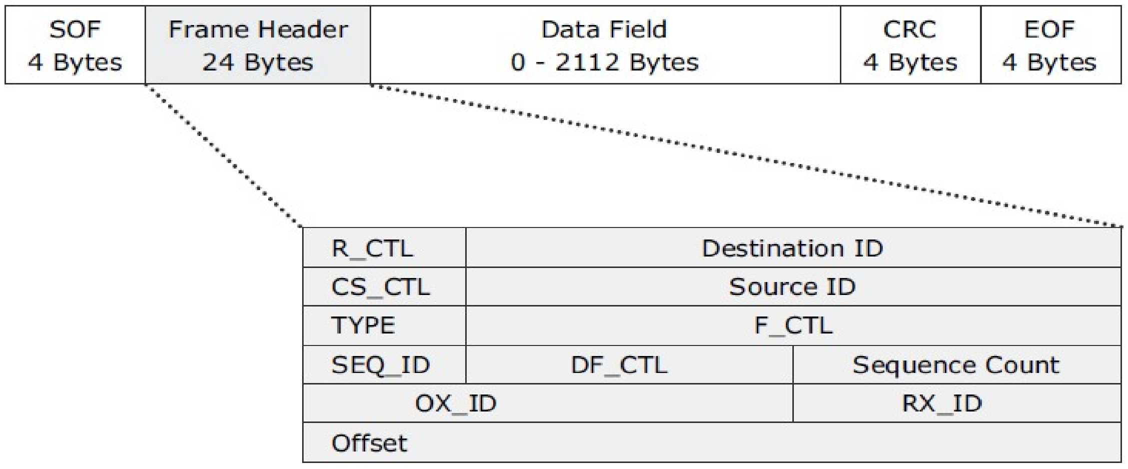


Figure 6-17: FC frame

The frame header also defines the following fields:

1. **Routing Control (R_CTL)**: This field denotes whether the frame is a link control frame or a data frame. Link control frames are nondata frames that do not carry any payload. These frames are used for setup and messaging. In contrast, data frames carry the payload and are used for data transmission.
2. **Class Specific Control (CS_CTL)**: This field specifies link speeds for class 1 and class 4 data transmission.
3. **TYPE**: This field describes the upper layer protocol (ULP) to be carried on the frame if it is a data frame. However, if it is a link control frame, this field is used to signal an event such as ―fabric busy.‖ For example, if the TYPE is 08, and the frame is a data frame, it means that the SCSI will be carried on an FC.
4. **Data Field Control (DF_CTL)**: A 1-byte field that indicates the existence of any optional headers at the beginning of the data payload. It is a mechanism to extend header information into the payload.
5. **Frame Control (F_CTL)**: A 3-byte field that contains control information related to frame content. For example, one of the bits in this field indicates whether this is the first sequence of the exchange.

Structure and Organization of FC Data

In an FC network, data transport is analogous to a conversation between two people, whereby a frame represents a word, a sequence represents a sentence, and an exchange represents a conversation.

1. **Exchange operation:** An exchange operation enables two N_ports to identify and manage a set of information units. This unit maps to a sequence. Sequences can be both unidirectional and bidirectional depending upon the type of data sequence exchanged between the initiator and the target.
2. **Sequence:** A sequence refers to a contiguous set of frames that are sent from one port to another. A sequence corresponds to an information unit, as defined by the ULP.
3. **Frame:** A frame is the fundamental unit of data transfer at Layer 2. Each frame can contain up to 2,112 bytes of payload.

Flow Control

Flow control defines the pace of the flow of data frames during data transmission. FC technology uses two flow-control mechanisms: buffer-to-buffer credit (BB_Credit) and end-to-end credit (EE_Credit).

1. BB_Credit: FC uses the *BB_Credit* mechanism for hardware-based flow control. BB_Credit controls the maximum number of frames that can be present over the link at any given point in time. In a switched fabric, BB_Credit management may take place between any two FC ports. The transmitting port maintains a count of free receiver buffers and continues to send frames if the count is greater than 0. The BB_Credit mechanism provides frame acknowledgment through the *Receiver Ready (R_RDY)* primitive.

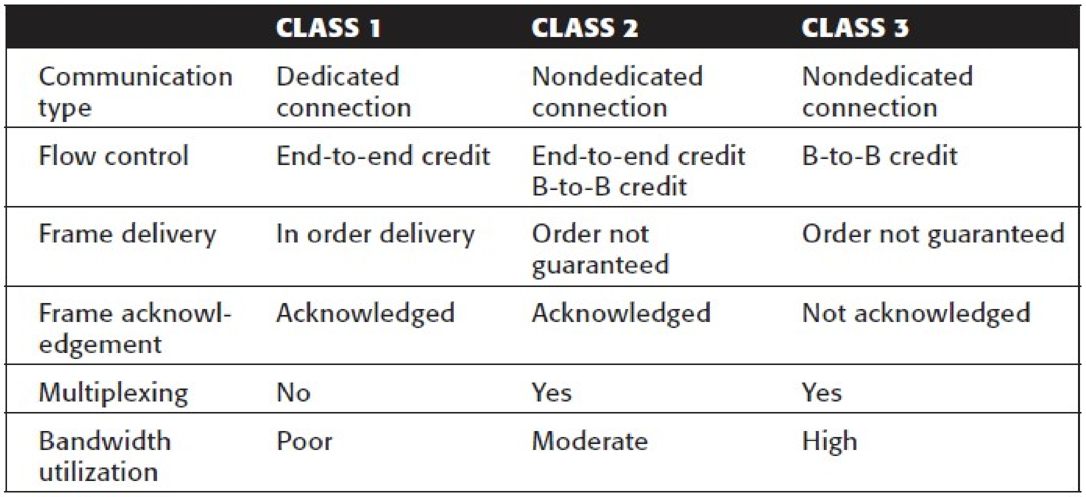
2. EE_Credit: The function of end-to-end credit, known as EE_Credit, is similar to that of BB_ Credit. When an initiator and a target establish themselves as nodes communicating with each other, they exchange the

EE_Credit parameters (part of Port Login). The EE_Credit mechanism affects the flow control for class 1 and class 2 traffic only.

Classes of Service

The FC standards define different classes of service to meet the requirements of a wide range of applications. The table below shows three classes of services and their features (Table 6-1).

Table 6-1: FC Class of Services



	CLASS 1	CLASS 2	CLASS 3
Communication type	Dedicated connection	Nondedicated connection	Nondedicated connection
Flow control	End-to-end credit	End-to-end credit B-to-B credit	B-to-B credit
Frame delivery	In order delivery	Order not guaranteed	Order not guaranteed
Frame acknowl-edgement	Acknowledged	Acknowledged	Not acknowledged
Multiplexing	No	Yes	Yes
Bandwidth utilization	Poor	Moderate	High

Another class of services is *class F,* which is intended for use by the switches communicating through ISLs. Class F is similar to Class 2, and it provides notification of non delivery of frames. Other defined Classes 4, 5, and 6 are used for specific applications.

Zoning

Zoning is an FC switch function that enables nodes within the fabric to be logically segmented into groups that can communicate with each other (see Figure 6-18). When a device (host or storage array) logs onto a fabric, it is registered with the name server. When a port logs onto the fabric, it goes through a device discovery process with other devices registered in the name server. The zoning function controls this process by allowing only the members in the same zone to establish these link-level services.

Multiple zone sets may be defined in a fabric, but only one zone set can be active at a time. A zone set is a set of zones and a zone is a set of members. A member may be in multiple zones. Members, zones, and zone sets form the hierarchy defined in the zoning process (see Figure 6-19). *Members* are nodes within the SAN that can be included in a zone. *Zones* comprise a set of members that have access to one another. A port or a node can be a member of multiple zones. *Zone sets* comprise a group of zones that can be activated or deactivated as a single entity in a fabric. Only one zone set per fabric can be active at a time. Zone sets are also referred to as *zone configurations*.

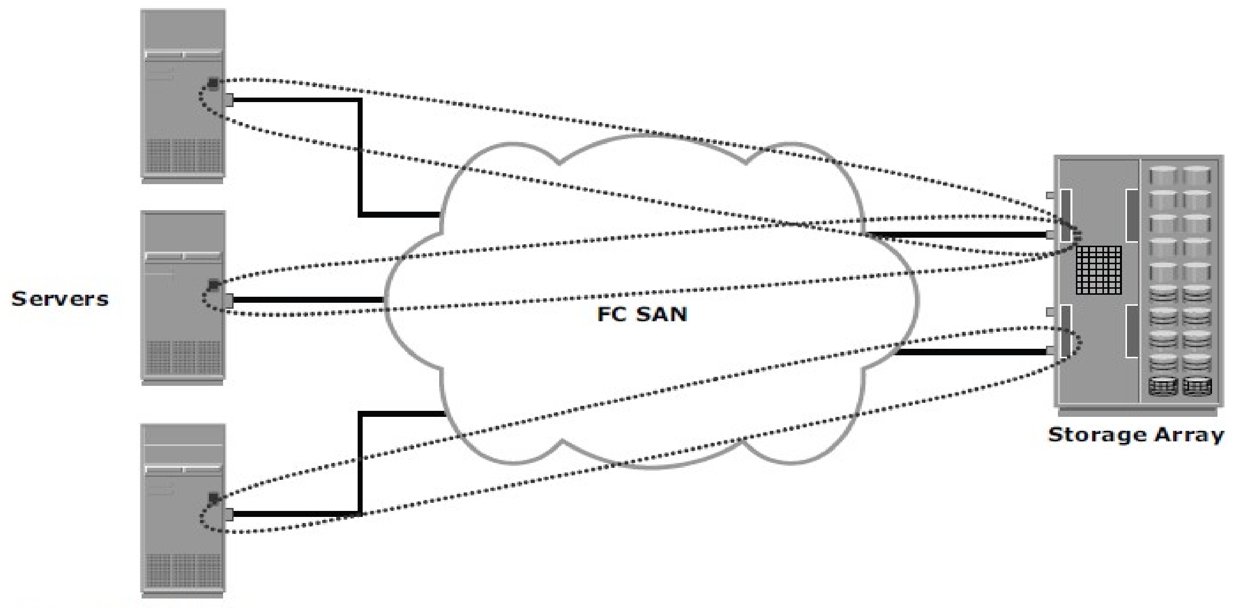


Figure 6-18: Zoning

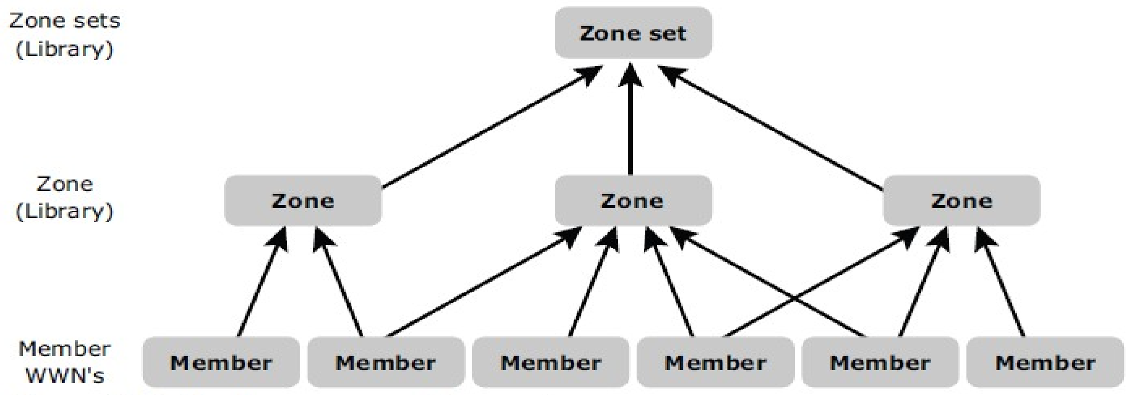


Figure 6-19: Members, zones, and zone sets

Types of Zoning

Zoning can be categorized into three types:

1. Port zoning: It uses the FC addresses of the physical ports to define zones. In port zoning, access to data is determined by the physical switch port to which a node is connected. The FC address is dynamically assigned when the port logs on to the fabric. Therefore, any change in the fabric configuration affects zoning. Port zoning is also called *hard zoning*. Although this method is secure, it requires updating of zoning configuration information in the event of fabric reconfiguration.

2. WWN zoning: It uses World Wide Names to define zones. WWN zoning is also referred to as *soft zoning*. A major advantage of WWN zoning is its flexibility. It allows the SAN to be recabled without reconfiguring the zone information. This is possible because the WWN is static to the node port.

3. Mixed zoning: It combines the qualities of both WWN zoning and port zoning. Using mixed zoning enables a specific port to be tied to the WWN of a node.

Figure 6-20 shows the three types of zoning on an FC network.

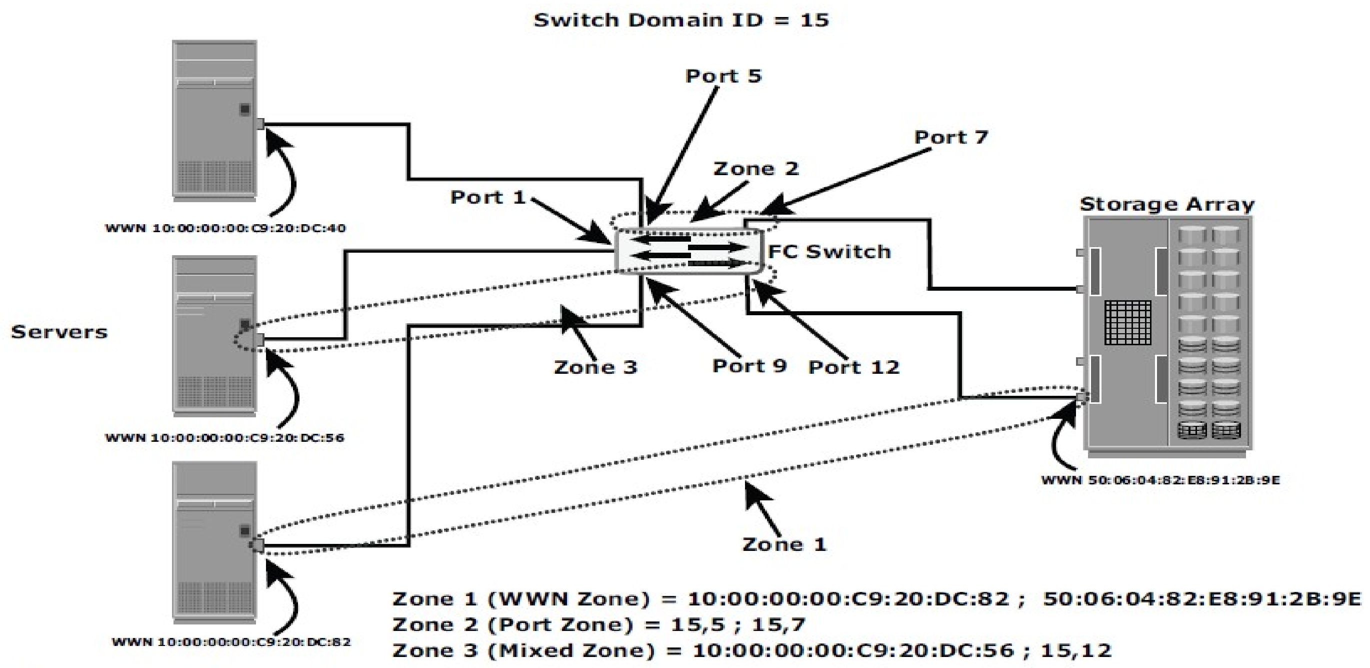

Switch Domain ID = 15
Port 5
Port 7
Zone 2
Port 1
Storage Array
FC Switch
WWN 10:00:00:00:C9:20:DC:40
Servers
Zone 3
Port 9
Port 12
WWN 10:00:00:00:C9:20:DC:56
WWN 50:06:04:82:E8:91:2B:9E
Zone 1
WWN 10:00:00:00:C9:20:DC:82
Zone 1 (WWN Zone) = 10:00:00:00:C9:20:DC:82 ; 50:06:04:82:E8:91:2B:9E
Zone 2 (Port Zone) = 15,5 ; 15,7
Zone 3 (Mixed Zone) = 10:00:00:00:C9:20:DC:56 ; 15,12

Figure 6-20: Types of zoning

Fibre Channel Login Types

Fabric services define three login types:

1. **Fabric login (FLOGI)** is performed between an N_port and an F_port. To log on to the fabric, a device sends a FLOGI frame with the World Wide Node Name (WWNN) and World Wide Port Name (WWPN) parameters to the login service at the well-known FC address FFFFFE. In turn, the switch accepts the login and returns an Accept (ACC) frame with the assigned FC address for the device. Immediately after the FLOGI, the N_port registers itself with the local name server on the switch, indicating its WWNN, WWPN, and assigned FC address.
2. **Port login (PLOGI)** is performed between an N_port and another N_port to establish a session. The initiator N_port sends a PLOGI request frame to the target N_port, which accepts it. The target N_port returns an ACC to the initiator N_port. Next, the N_ports exchange service parameters relevant to the session.

Process login (PRLI) is also performed between an N_port and another N_ port. This login relates to the FC-4 ULPs such as SCSI. N_ports exchange SCSI-3-related service parameters. N_ports share information about the FC-4 type in use, the SCSI initiator, or the target.

FC Topologies

Fabric design follows standard topologies to connect devices. Core-edge fabric is one of the popular topology designs.

Core-Edge Fabric

In the *core-edge fabric* topology, there are two types of switch tiers in this fabric. The *edge tier* usually comprises switches and offers an inexpensive approach to adding more hosts in a fabric. The tier at the edge fans out from the tier at the core. The nodes on the edge can communicate with each other.

The *core tier* usually comprises enterprise directors that ensure high fabric availability. Additionally all traffic has to either traverse through or terminate at this tier. In a two-tier configuration, all storage devices are connected to the core tier, facilitating fan-out.

The core-edge fabric topology increases connectivity within the SAN while conserving overall port utilization. If expansion is required, an additional edge switch can be connected to the core. This topology can have different variations. In a *single-core topology*, all hosts are connected to the edge tier and all storage is connected to the core tier. Figure 6-21 depicts the core and edge switches in a single-core topology.

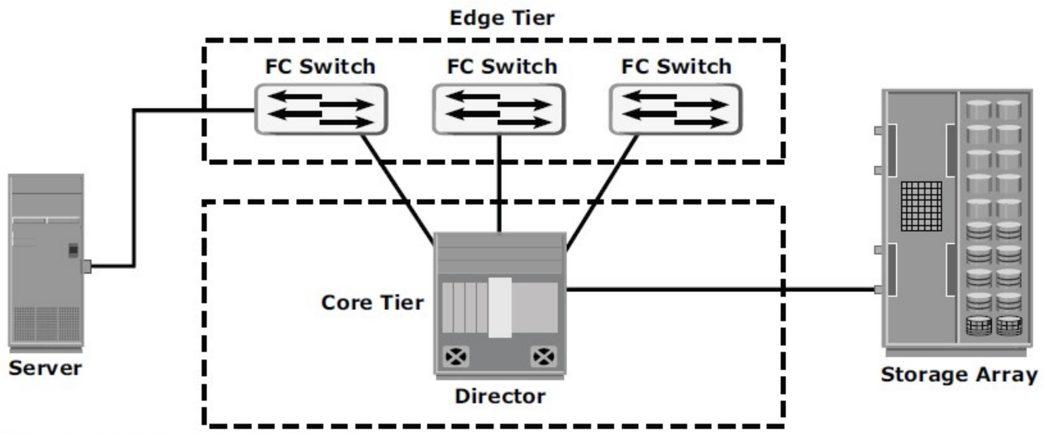


Figure 6-21: Single core topology

A ***dual-core topology*** can be expanded to include more core switches. However, to maintain the topology, it is essential that new ISLs are created to connect each edge switch to the new core switch that is added. Figure 6- 22 illustrates the core and edge switches in a dual-core topology.

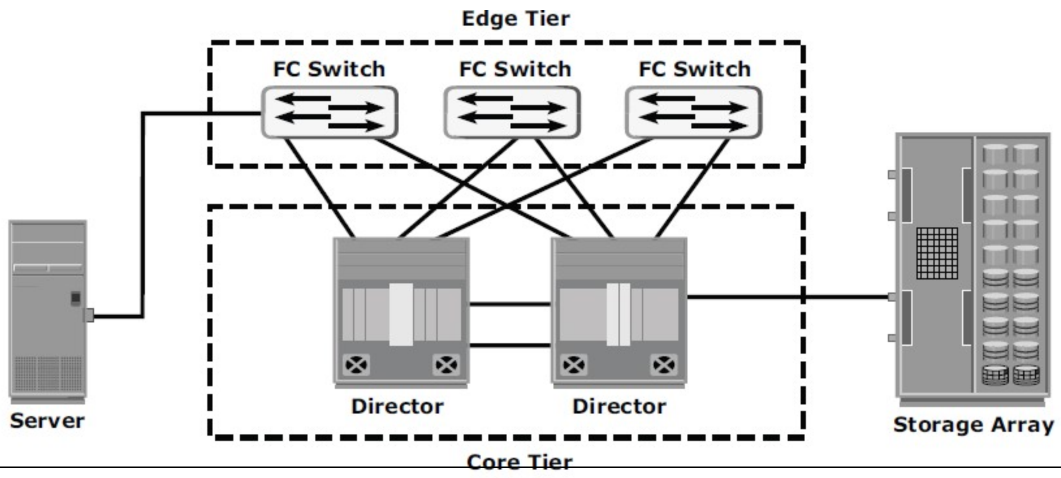


Figure 6-22: Dual-core topology

Benefits and Limitations of Core-Edge Fabric

Benefits

1. The core-edge fabric provides one-hop storage access to all storage in the system.

Because each tier's switch is used for either storage or hosts, one can easily identify which resources are approaching their capacity, making it easier to develop a set of rules for scaling and apportioning.

2. A well-defined, easily reproducible building-block approach makes rolling out new fabrics easier.

Core-edge fabrics can be scaled to larger environments by linking core switches, adding more core switches, or adding more edge switches.

3. This method can be used to extend the existing simple core-edge model or to expand the fabric into a compound or complex core-edge model.

Limitations

1. The core-edge fabric may lead to some performance-related problems because scaling a core-edge topology involves increasing the number of ISLs in the fabric. As more edge switches are added, the domain count in the fabric increases.

2. A common best practice is to keep the number of host-to-storage hops unchanged, at one hop, in a core-edge. Hop count represents the total number of devices a given piece of data (packet) passes through. Generally a large hop count means greater the transmission delay between data traverse from its source to destination.

3. As the number of cores increases, it may be prohibitive to continue to maintain ISLs from each core to each edge switch. When this happens, the fabric design can be changed to a compound or complex core-edge design.

Mesh Topology

In a *mesh topology*, each switch is directly connected to other switches by using ISLs. This topology promotes enhanced connectivity within the SAN. When the number of ports on a network increases, the number of nodes that can participate and communicate also increases.

A mesh topology may be one of the two types: full mesh or partial mesh. In a *full mesh*, every switch is connected to every other switch in the topology. Full mesh topology may be appropriate when the number of switches involved is small.

In a *partial mesh* topology, several hops or ISLs may be required for the traffic to reach its destination. Hosts and storage can be located anywhere in the fabric, and storage can be localized to a director or a switch in both mesh topologies.

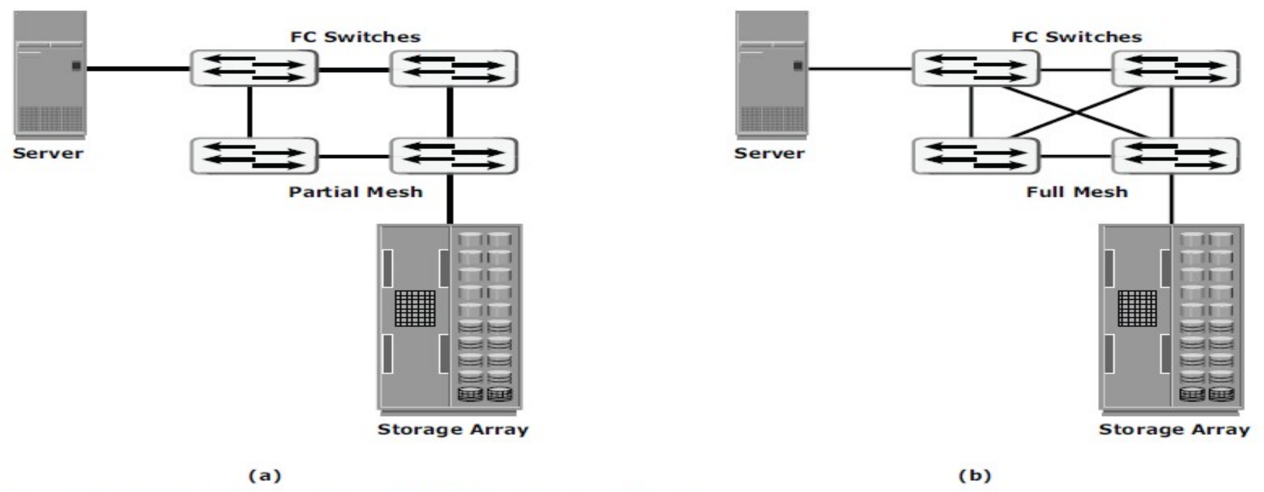


Figure 6-23: Partial mesh and full mesh topologies

Network-Attached Storage

NAS is a preferred storage solution that enables clients to share files quickly and directly with minimum storage management overhead. NAS also helps to eliminate bottlenecks that users face when accessing files from a general-purpose server. NAS uses network and file-sharing protocols to perform filing and storage functions. These protocols include TCP/IP for data transfer and CIFS and NFS for remote file service. NAS enables both UNIX and Microsoft Windows users to share the same data seamlessly. To enable data sharing, NAS typically uses NFS for UNIX, CIFS for Windows, and File Transfer Protocol (FTP) and other protocols for both environments. Recent advancements in networking technology have enabled NAS to scale up to enterprise requirements for improved performance and reliability in accessing data.

General-Purpose Servers vs. NAS Devices A NAS device is optimized for file-serving functions such as storing, retrieving, and accessing files for applications and clients. As shown in Figure 7-1, a general-purpose server can be used to host any application, as it runs a generic operating system. Unlike a general-purpose server, a NAS device is dedicated to file-serving. It has a real-time operating system dedicated to file serving by using open-standard protocols. Some NAS vendors support features such as native clustering for high availability.

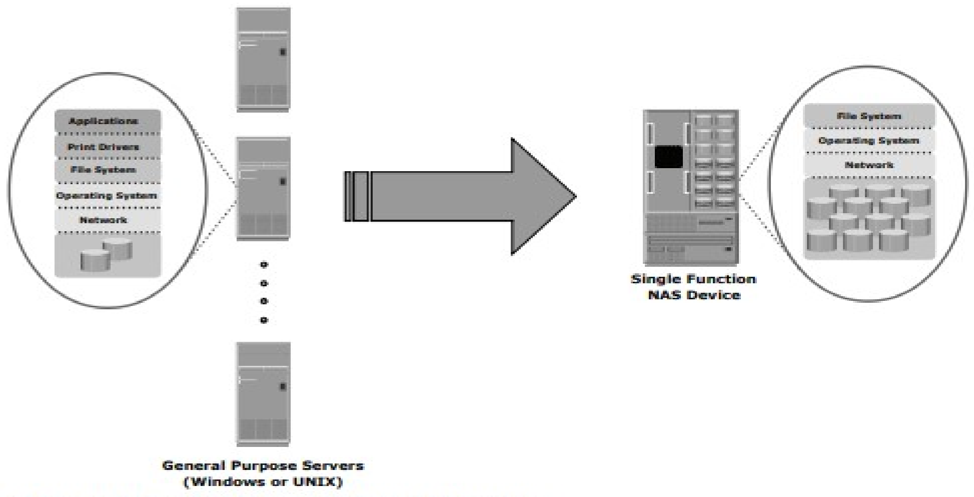


Figure 7-1: General purpose server vs. NAS device

<u>Benefits of NAS NAS offers the following benefits</u>:

■ Supports comprehensive access to information: Enables efficient file sharing and supports many-to-one and one-to-many configurations. The many-to-one configuration enables a NAS device to serve many clients simultaneously. The one-to-many configuration enables one client to connect with many NAS devices simultaneously.

■ Improved efficiency: Eliminates bottlenecks that occur during file access from a general-purpose file server because NAS uses an operating system specialized for file serving. It improves the utilization of general-purpose servers by relieving them of file-server operations.

■ Improved flexibility: Compatible for clients on both UNIX and Windows platforms using industry-standard protocols. NAS is flexible and can serve requests from different types of clients from the same source.

■ Centralized storage: Centralizes data storage to minimize data duplication on client workstations, simplify data management, and ensures greater data protection.

■ Simplified management: Provides a centralized console that makes it possible to manage file systems efficiently. ■ Scalability: Scales well in accordance with different utilization profiles and types of business applications because of the high performance and low-latency design.

■ High availability: Offers efficient replication and recovery options, enabling high data availability. NAS uses redundant networking components that provide maximum connectivity options. A NAS device can use clustering technology for failover

■ Security: Ensures security, user authentication, and file locking in conjunction with industry-standard security schemas. 7.3 NAS File I/O NAS uses file-level access for all of its I/O operations. File I/O is a high-level request that specifies the file to be accessed, but does not specify its logical block address. For example, a file I/O request from a client may specify reading 256 bytes from byte number 1152 onward in a specific file. Unlike block I/O, there is no disk volume or disk sector information in a file I/O request. The NAS operating system keeps track of the location of files on the disk volume and converts client file I/O into block-level I/O to retrieve data. The NAS operating system issues a block I/O request to fulfill the file read and write requests that it receives. The retrieved data is again converted to filelevel I/O for applications and clients.

File Systems and Remote File Sharing A file system is a structured way of storing and organizing data files. Many file systems maintain a file access table to simplify the process of finding and accessing files.

Accessing a File System A file system must be mounted before it can be used. In most cases, the operating system mounts a local file system during the boot process. The mount process creates a link between the file system and the operating system. When mounting a file system, the operating system organizes files and directories in a tree-like structure and grants the user the privilege of accessing this structure. The tree is rooted at a mount point that is named using operating system conventions. Users and applications can traverse the entire tree from the root to the leaf nodes. Files are located at leaf nodes, and directories and subdirectories are located at intermediate roots.

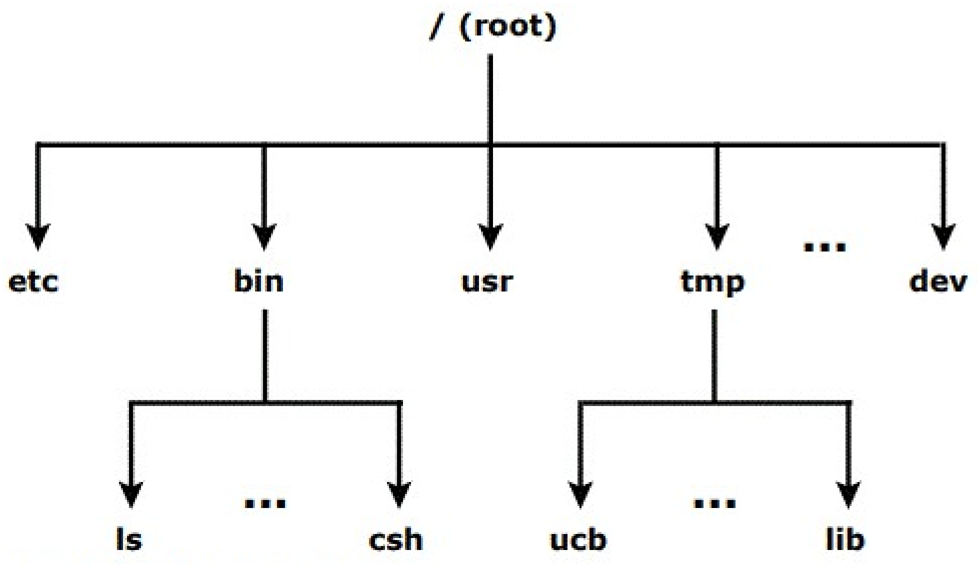


Figure 7-2: UNIX directory structure

Components of NAS A NAS device has the following components (see Figure 7-3)

: ■ NAS head (CPU and Memory)

■ One or more network interface cards (NICs), which provide connectivity to the network. Examples of NICs include Gigabit Ethernet, Fast Ethernet, ATM, and Fiber Distributed Data Interface (FDDI).

■ An optimized operating system for managing NAS functionality

■ NFS and CIFS protocols for file sharing

■ Industry-standard storage protocols to connect and manage physical disk resources, such as ATA, SCSI, or FC

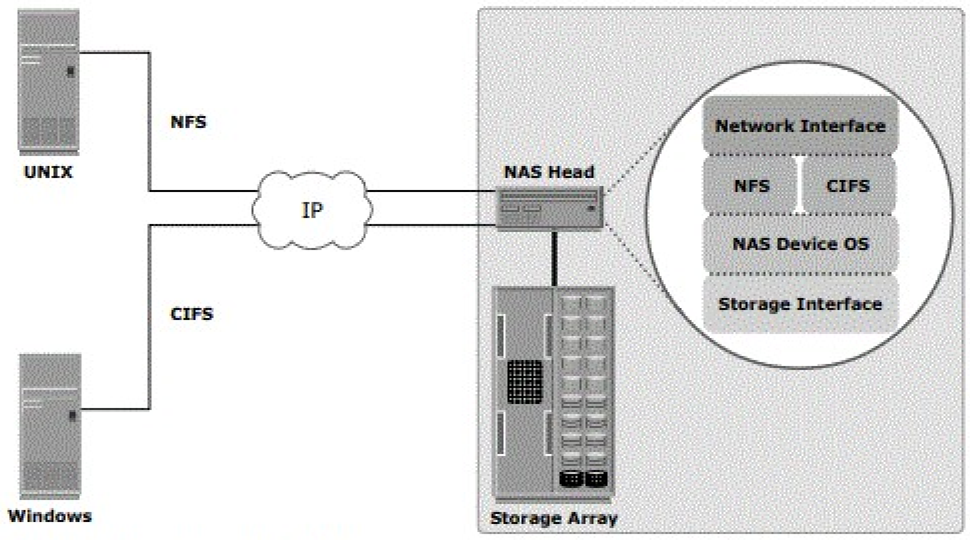


Figure 7-3: Components of NAS

IP SAN

IP SAN technologies can be used in a variety of situations. Figure 8-1 illustrates the co-existence of FC and IP storage technologies in an organization where mission-critical applications are serviced through FC, and businesscritical applications and remote office applications make use of IP SAN. Disaster recovery solutions can also be implemented using both of these technologies. Two primary protocols that leverage IP as the transport mechanism are iSCSI and Fibre Channel over IP (FCIP).

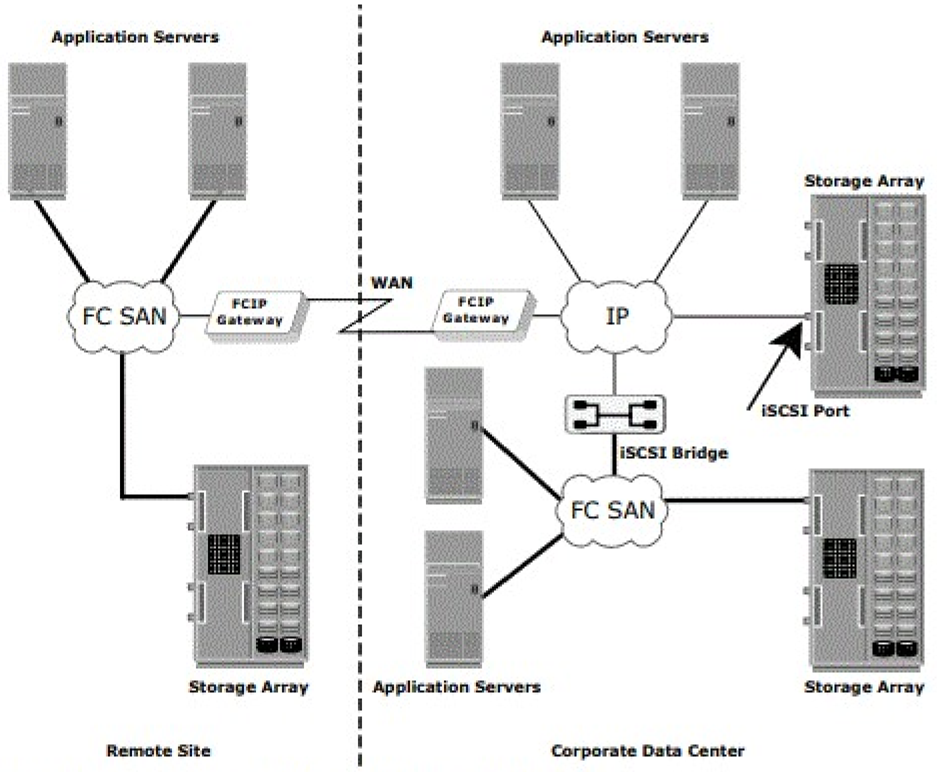


Figure 8-1: Co-existance of FC and IP storage technologies

Content-Addressed Storage

CAS is an object-based system that has been purposely built for storing fixed content data. It is designed for secure online storage and retrieval of fixed con- tent. Unlike file-level and block-level data access that use file names and the physical location of data for storage and retrieval, CAS stores user data and its attributes as separate objects. The stored object is assigned a globally unique address known as a content address (CA). This address is derived from the object's binary representation. CAS provides an optimized and centrally managed stor- age solution that can support single-instance storage (SiS) to eliminate multiple copies of the same data.

Features and Benefits of CAS

CAS has emerged as an alternative to tape and optical solutions because it over- comes many of their obvious deficiencies. CAS also meets the demand to improve data accessibility and to properly protect, dispose of, and ensure service-level agree- ments for archived data. The features and benefits of CAS include the following:

■ Content authenticity: It assures the genuineness of stored content. This is achieved by generating a unique content address and automating the process of continuously checking and recalculating the content address for stored objects.

Content integrity: Refers to the assurance that the stored content has not been altered. Use of hashing algorithm for content authenticity also ensures content integrity in CAS. If the fixed content is altered, CAS assigns a new address to the altered content, rather than overwrite the original fixed content

Location independence: CAS uses a unique identifier that applications can leverage to retrieve data rather than a centralized directory

Single-instance storage (SiS): The unique signature is used to guarantee the storage of only a single instance of an object. This signature is derived from the binary representation of the object.

Retention enforcement: Protecting and retaining data objects is a core requirement of an archive system. CAS creates two immutable compo- nents: a data object and a meta-object for every object stored. The metaobject stores object's attributes and data handling policies.

Record-level protection and disposition: All fixed content is stored in CAS once and is backed up with a protection scheme. The array is com- posed of one or more storage clusters

Technology independence: The CAS system interface is impervious to technology changes. As long as the application server is able to map the original content address the data remains accessible. Although hardware changes are inevitable, the goal of CAS hardware vendors is to ensure compatibility across platforms.

■ Fast record retrieval: CAS maintains all content on disks that provide subsecond "time to first byte" (200 ms–400 ms) in a single cluster. Random disk access in CAS enables fast record retrieval.

Object Storage and Retrieval in CAS

The process of storing and retrieving objects in CAS is explained in Figures 9-3 and 9-4, respectively. This process requires an understanding of the following CAS terminologies:

■ Application programming interface (API): A high-level implementation of an interface that specifies the details of how clients can make service requests. The CAS API resides on the application server and is responsible for storing and retrieving the objects in a CAS system.

■ Access profile: Used by access applications to authenticate to a CAS cluster and by CAS clusters to authenticate themselves to each other for replication.

■ Virtual pools: Enable a single logical cluster to be broken up into multiple logical groupings of data.

■ Binary large object (BLOB): The actual data without the descriptive information (metadata). The distinct bit sequence of user data repre- sents the actual content of a file and is independent of the name and physical location.

■ Content address (CA): An object's address, which is created by a hash algorithm run across the binary representation of the object. While gen- erating a CA, the hash algorithm considers all aspects of the content, returning a unique content address to the user's application. A unique number is calculated from the sequence of bits that constitutes file content. If even a single character changes in the file, the resulting CA is different. A hash output, also called a digest, is a type of fingerprint for a variable-length data file. This output represents the file contents and is used to locate the file in a CAS system. The digest can be used to verify whether the data is authentic or has changed because of equip- ment failure or human intervention. When a user tries to retrieve or open a file, the server sends the CA to the CAS system with the appropriate function to read the file. The CAS system uses the CA to locate the file and passes it back to the application server.

■ C-Clip: A virtual package that contains data (BLOB) and its associated CDF. The C-Clip ID is the CA that the system returns to the client applica- tion. It is also referred as a C-Clip handle or C-Clip reference.

■ C-Clip Descriptor File (CDF): An XML file that the system creates while making a C-Clip. This file includes CAs for all referenced BLOBs and associated metadata. Metadata includes characteristics of CAS objects such as size, format, and expiration date.

Module IV: Transactions of Data

A: San Transactions of data

Continuous access to information is a must for the smooth functioning of business operations today, as the cost of business disruption could be catastrophic. There are many threats to information avail- ability, such as natural disasters (e.g., flood, fire, earthquake), unplanned occurrences (e.g., cybercrime, human error, network and com- puter failure), and planned occurrences (e.g., upgrades, backup, restore) that result in the inaccessibility of information. It is critical for businesses to define appropriate plans that can help them overcome these crises. Business continuity is an important process to define and implement these plans.

Business continuity (BC) is an integrated and enterprise wide process that includes all activities (internal and external to IT) that a business must perform to mitigate the impact of planned and unplanned downtime. BC entails pre- paring for, responding to, and recovering from a system outage that adversely affects business operations. It involves proactive measures, such as business impact analysis and risk assessments, data protection, and security, and reactive countermeasures, such as disaster recovery and restart, to be invoked in the event of a failure. The goal of a business continuity solution is to ensure the "information availability" required to conduct vital business operations.

Information Availability

Information availability (IA) refers to the ability of the infrastructure to func- tion according to business expectations during its specified time of operation. Information availability ensures that people (employees, customers, suppliers, and partners) can access information whenever they need it. Information avail- ability can be defined with the help of reliability, accessibility and timeliness.

n **Reliability:** This reflects a component's ability to function without failure, under stated conditions, for a specified amount of time.

Accessibility: This is the state within which the required information is accessible at the right place, to the right user. The period of time during which the system is in an accessible state is termed *system uptime;* when it is not accessible it is termed *system downtime*.

Timeliness: Defines the exact moment or the time window (a particular time of the day, week, month, and/or year as specified) during which information must be accessible. this time slot are not considered to affect timeliness.

Causes of Information Unavailability

Various planned and unplanned incidents result in data unavailability. *Planned outages* include installation/integration/maintenance of new hardware, soft- ware upgrades or patches, taking backups, application and data restores, facility operations (renovation and construction), and refresh/migration of the testing to the production environment. *Unplanned outages* include failure caused by database corruption, component failure, and human errors.

Another type of incident that may cause data unavailability is natural or man-made disasters such as flood, fire, earthquake, and contamination. As illustrated in Figure 11-1, the majority of outages are planned. Planned outages are expected and scheduled, but still cause data to be unavailable. Statistically, less than 1 percent is likely to be the result of an unforeseen disaster.

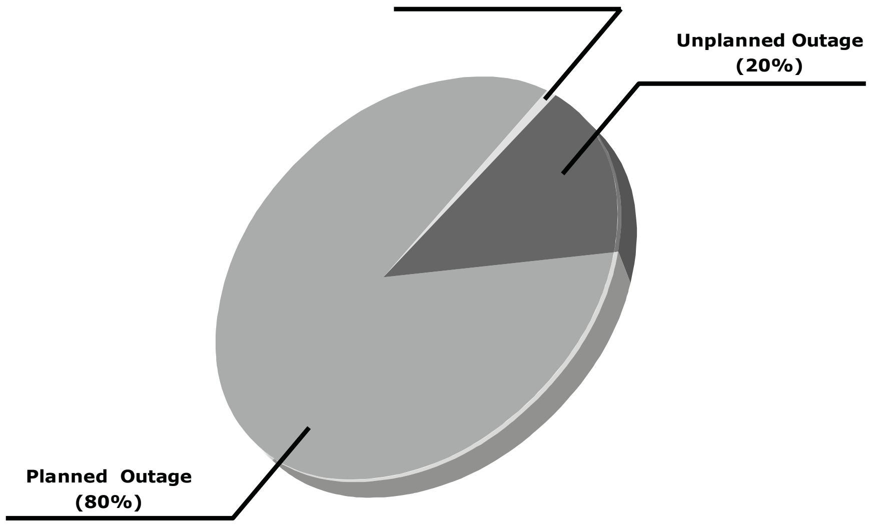


Figure : Disruptors of data availability

Measuring Information Availability

Information availability relies on the availability of the hardware and software components of a data center. Failure of these components might disrupt informa- tion availability. A failure is the termination of a component's ability to perform a required function. The component's ability can be restored by performing an external corrective action, such as a manual reboot, a repair, or replacement of the failed component(s).

Repair involves restoring a component to a condition that enables it to perform a required function within a specified time by using procedures and resources. Proactive risk analysis performed as part of the BC planning process considers the component failure rate and average repair time, which are measured by MTBF and MTTR:

Mean Time Between Failure (MTBF): It is the average time available for a system or component to perform its normal operations between failures.

Mean Time To Repair (MTTR): It is the average time required to repair a failed component. While calculating MTTR, it is assumed that the fault responsible for the failure is correctly identified and that the required spares and personnel are available. Note that a fault is a physical defectat the component level, which may result in data unavailability. MTTR includes the time required to do the following: detect the fault, mobilize the maintenance team, diagnose the fault, obtain the spare parts, repair, test, and resume normal operations.

IA is the fraction of a time period that a system is in a condition to perform its intended function upon demand. It can be expressed in terms of system uptime and downtime and measured as the amount or percentage of system uptime:

IA = system uptime / (system uptime + system downtime)

In terms of MTBF and MTTR, IA could also be expressed as

IA = MTBF / (MTBF + MTTR)

Uptime per year is based on the exact timeliness requirements of the service, this calculation leads to the number of “9s” representation for availability met- rics. Table 11-1 lists the approximate amount of downtime allowed for a service to achieve certain levels of 9s availability.

For example, a service that is said to be “five 9s available” is available for 99.999 percent of the scheduled time in a year (24 × 7 × 365).

Table : Availability Percentage and Allowable Downtime

UpTIme (%)	**DownTIme (%)**	**DownTIme per Year**	**DownTIme per week**
98	2	7.3 days	3 hr 22 minutes
99	1	3.65 days	1 hr 41 minutes
99.8	0.2	17 hr 31 minutes	20 minutes 10 sec
99.9	0.1	8 hr 45 minutes	10 minutes 5 sec
99.99	0.01	52.5 minutes	1 minute
99.999	0.001	5.25 minutes	6 sec
99.9999	0.0001	31.5 sec	0.6 sec

Consequences of Downtime

Data unavailability, or downtime, results in loss of productivity, loss of rev- enue, poor financial performance, and damages to reputation. Loss of produc- tivity reduces the output per unit of labor, equipment, and capital. Loss of revenue includes direct loss, compensatory payments, future revenue losses, billing losses, and investment losses. Poor financial performance affects revenue recognition, cash flow, discounts, payment guarantees, credit rating, and stock price. Damages to reputation may result in a loss of confidence or credibility with customers, suppliers, financial markets, banks, and business partners. Other possible consequences of downtime include the cost of additional equip- ment rental, overtime, and extra shipping.

The business impact of downtime is the sum of all losses sustained as a result of a given disruption. An important metric, *average cost of downtime per hour*, provides a key estimate in determining the appropriate BC solutions. It is calculated as follows:

Average cost of downtime per hour = average productivity loss per hour + average revenue loss per hour

Where:

Productivity loss per hour = (total salaries and benefits of all employees per week) / (average number of working hours per week)

Average revenue loss per hour = (total revenue of an organization per week) / (average number of hours per week that an organization is open for business)

The average downtime cost per hour may also include estimates of projected revenue loss due to other consequences such as damaged reputations and the additional cost of repairing the system.

BC Terminology

This section introduces and defines common terms related to BC operations and are used in the next few chapters to explain advanced concepts:

- **Disaster recovery:** This is the coordinated process of restoring systems, data, and the infrastructure required to support key ongoing business operations in the event of a disaster. It is the process of restoring a previ- ous copy of the data and applying logs or other necessary processes to that copy to bring it to a known point of consistency. Once all recoveries are completed, the data is validated to ensure that it is correct.

- **Disaster restart:** This is the process of restarting business operations with mirrored consistent copies of data and applications.

- **Recovery-Point Objective (RPO):** This is the point in time to which sys- tems and data must be recovered after an outage. It defines the amount of data loss that a business can endure. A large RPO signifies high tolerance to information loss in a business. Based on the RPO, organizations plan for the minimum frequency with which a backup or replica must be made. Forexample, if the RPO is six hours, backups or replicas must be made at least once in 6 hours. Figure 11-2 shows various RPOs and their corresponding ideal recovery strategies. An organization can plan for an appropriate BC technology solution on the basis of the RPO it sets. For example:

- **RPO of 24 hours:** This ensures that backups are created on an offsite tape drive every midnight. The corresponding recovery strategy is to restore data from the set of last backup tapes.

- **RPO of 1 hour:** This ships database logs to the remote site every hour. The corresponding recovery strategy is to recover the database at the point of the last log shipment.

- **RPO of zero:** This mirrors mission-critical data synchronously to a remote site.

a) Recovery-point objective (b) Recovery-time objective

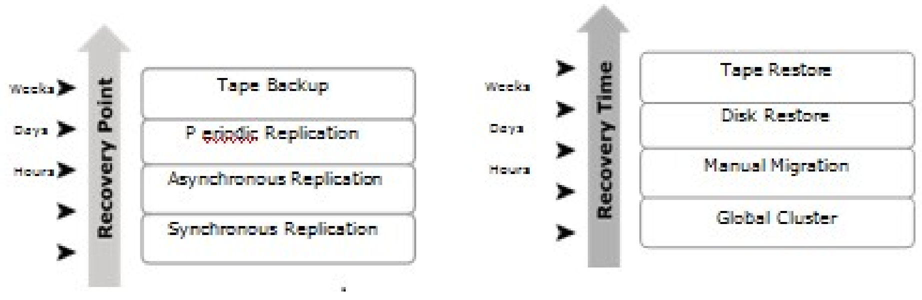


Figure: Strategies to meet RPO and RTO target

- **Recovery-Time Objective (RTO):** The time within which systems, appli- cations, or functions must be recovered after an outage. It defines the amount of downtime that a business can endure and survive. Businesses can optimize disaster recovery plans after defining the RTO for a given data center or network. For example, if the RTO is two hours, then use a disk backup because it enables a faster restore than a tape backup. However, for an RTO of one week, tape backup will likely meet require- ments. Some examples of RTOs and the recovery strategies to ensure data availability are listed below (refer to Figure):

  - **RTO of 72 hours:** Restore from backup tapes at a cold site.

  - **RTO of 12 hours:** Restore from tapes at a hot site.

  - **RTO of 4 hours:** Use a data vault to a hot site.

  - **RTO of 1 hour:** Cluster production servers with controller-based disk mirroring.

  - **RTO of a few seconds:** Cluster production servers with bidirectional mirroring, enabling the applications to run at both sites simultaneously.

BC Planning Lifecycle

BC planning must follow a disciplined approach like any other planning pro- cess. Organizations today dedicate specialized resources to develop and main- tain BC plans. From the conceptualization to the realization of the BC plan, a lifecycle of activities can be defined for the BC process. The BC planning lifecycle includes five stages (see Figure 11-3):

1. Establishing objectives
2. Analyzing
3. Designing and developing
4. Implementing
5. Training, testing, assessing, and maintaining

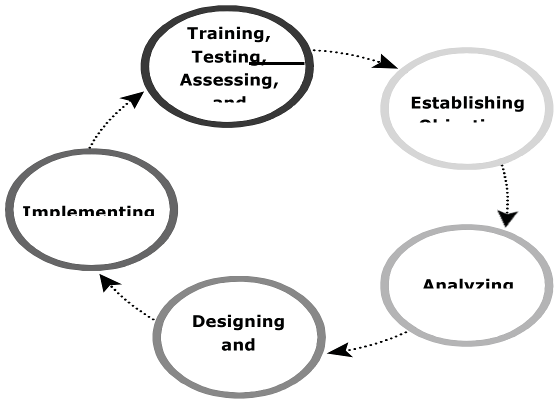


Figure 11-3: BC planning lifecycle

Several activities are performed at each stage of the BC planning lifecycle, including the following key activities:

1. Establishing objectives

   n Determine BC requirements.

   n Estimate the scope and budget to achieve requirements.

   n Select a BC team by considering subject matter experts from all areas of the business, whether internal or external.

   n Create BC policies.

2. Analyzing

   n Collect information on data profiles, business processes, infra- structure support, dependencies, and frequency of using business infrastructure.

   n Identify critical business needs and assign recovery priorities. n Create a risk analysis for critical areas and mitigation strategies. n Conduct a Business Impact Analysis (BIA).

   n Create a cost and benefit analysis based on the consequences of data unavailability.

   n Evaluate options.

3. Designing and developing

   n Define the team structure and assign individual roles and responsi- bilities. For example, different teams are formed for activities such as emergency response, damage assessment, and infrastructure and application recovery.

   n Design data protection strategies and develop infrastructure.

   n Develop contingency scenarios.

n Develop emergency response procedures.

n Detail recovery and restart procedures.

4. Implementing

n Implement risk management and mitigation procedures that include backup, replication, and management of resources.

n Prepare the disaster recovery sites that can be utilized if a disaster affects the primary data center.

n Implement redundancy for every resource in a data center to avoid single points of failure.

5. Training, testing, assessing, and maintaining

n Train the employees who are responsible for backup and replication of business-critical data on a regular basis or whenever there is a modi- fication in the BC plan.

n Train employees on emergency response procedures when disasters are declared.

n Train the recovery team on recovery procedures based on contingency scenarios.

n Perform damage assessment processes and review recovery plans.

n Test the BC plan regularly to evaluate its performance and identify its limitations.

n Assess the performance reports and identify limitations.

n Update the BC plans and recovery/restart procedures to reflect regular changes within the data center.

Failure Analysis

Failure analysis involves analyzing the data center to identify systems that are susceptible to a single point of failure and implementing fault-tolerance mechanisms such as redundancy.

Single Point of Failure

A *single point of failure* refers to the failure of a component that can terminate the availability of the entire system or IT service. Figure 11-4 illustrates the pos- sibility of a single point of failure in a system with various components: server, network, switch, and storage array. The figure depicts a system setup in which an application running on the server provides an interface to the client and performs I/O operations. The client is connected to the server through an IP network, the server is connected to the storage array through a FC connection, an HBA installed at the server sends or receives data to and from a storage array, and an FC switch connects the HBA to the storage port.

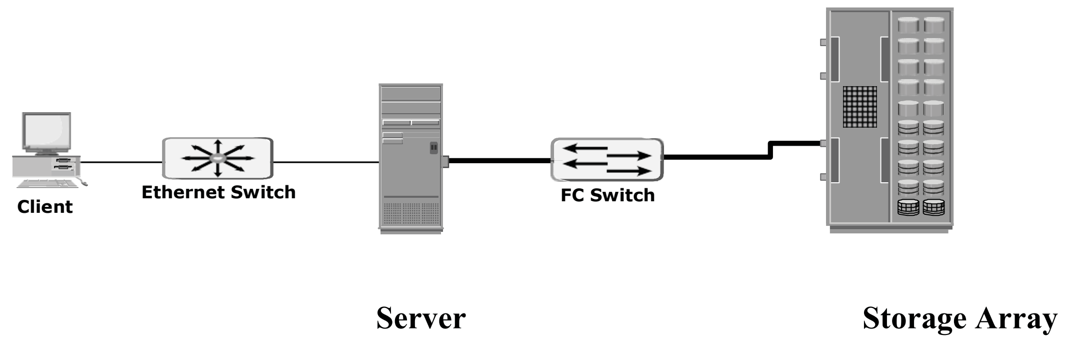


Figure : Single point of failure

In a setup where each component must function as required to ensure data availability, the failure of a single component causes the failure of the entire data center or an application, resulting in disruption of business operations. In this example, several single points of failure can be identified. The single HBA on the server, the server itself, the IP network, the FC switch, the storage array ports, or even the storage array could become potential single points of failure. To avoid single points of failure, it is essential to implement a fault-tolerant mechanism.

Fault Tolerance

To mitigate a single point of failure, systems are designed with redundancy, such that the system will fail only if all the components in the redundancy group fail. This ensures that the failure of a single component does not affect data availability. Figure 11-5 illustrates the fault-tolerant implementation of the system just described (and shown in Figure).

Data centers follow stringent guidelines to implement fault tolerance. Careful analysis is performed to eliminate every single point of failure. In the example shown in Figure 11-5, all enhancements in the infrastructure to mitigate single points of failures are emphasized:

n Configuration of multiple HBAs to mitigate single HBA failure.

n Configuration of multiple fabrics to account for a switch failure.

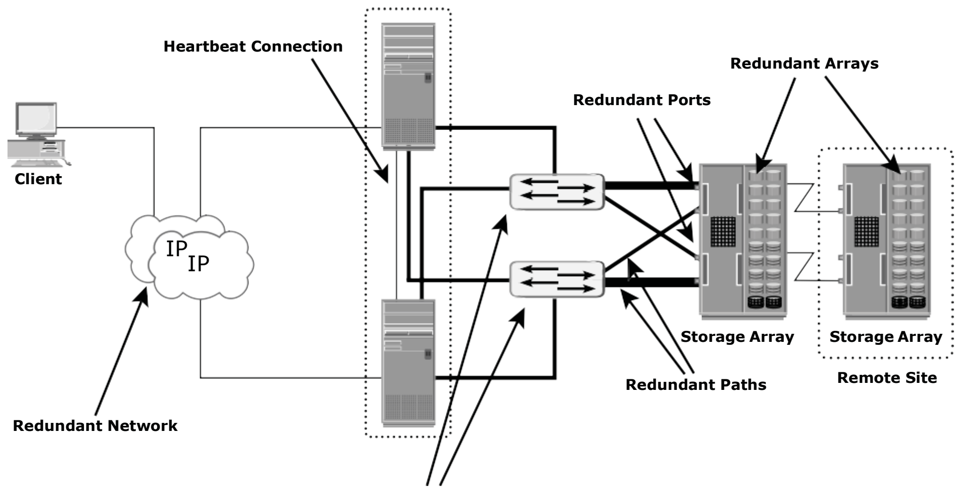
n Configuration of multiple storage array ports to enhance the storage array's availability.

n RAID configuration to ensure continuous operation in the event of disk failure.

n Implementing a storage array at a remote site to mitigate local site failure.

n Implementing server (host) clustering, a fault-tolerance mechanism whereby two or more servers in a cluster access the same set of volumes. Clustered servers exchange *heartbeats* to inform each other about their health. If one of the servers fails, the other server takes up the complete workload.

Clustered Servers



Redundant FC Switches

Figure : Implementation of fault tolerance

Multipathing Software

Configuration of multiple paths increases the data availability through path failover. If servers are configured with one I/O path to the data there will be no access to the data if that path fails. Redundant paths eliminate the path to become single points of failure. Multiple paths to data also improve I/O performance through load sharing and maximize server, storage, and data path utilization.

In practice, merely configuring multiple paths does not serve the purpose. Even with multiple paths, if one path fails, I/O will not reroute unless the system recognizes that it has an alternate path. Multipathing software provides the functionality to recognize and utilize alternate I/O path to data. Multipathing software also manages the load balancing by distributing I/Os to all available, active paths.

Business Impact Analysis

A *business impact analysis (BIA)* identifies and evaluates financial, operational, and service impacts of a disruption to essential business processes. Selected functional areas are evaluated to determine resilience of the infrastructure to support information availability. The BIA process leads to a report detailing the incidents and their impact over business functions. The impact may be specified in terms of money or in terms of time. Based on the potential impacts associated with downtime, businesses can prioritize and implement countermeasures to mitigate the likelihood of such disruptions. These are detailed in the BC plan. A BIA includes the following set of tasks:

- Identify the key business processes critical to its operation.
- Determine the attributes of the business process in terms of applications, databases, and hardware and software requirements.
- Estimate the costs of failure for each business process.
- Calculate the maximum tolerable outage and define RTO and RPO for each business process.
- Establish the minimum resources required for the operation of business processes.
- Determine recovery strategies and the cost for implementing them.
- Optimize the backup and business recovery strategy based on business priorities.
- Analyze the current state of BC readiness and optimize future BC planning.

BC Technology Solutions

After analyzing the business impact of an outage, designing appropriate solu- tions to recover from a failure is the next important activity. One or more copies of the original data are maintained using any of the following strategies, so that data can be recovered and business operations can be restarted using an alternate copy:

- **Backup and recovery:** Backup to tape is the predominant method of ensuring data availability. These days, low-cost, high-capacity disks are used for backup, which considerably speeds up the backup and recovery process. The frequency of backup is determined based on RPO, RTO, and the frequency of data changes.

- **Storage array-based replication (local):** Data can be replicated to a sepa- rate location within the same storage array. The replica is used inde- pendently for BC operations. Replicas can also be used for restoring operations if data corruption occurs.

- **Storage array-based replication (remote):** Data in a storage array can be replicated to another storage array located at a remote site. If the storage array is lost due to a disaster, BC operations start from the remote stor- age array.

- **Host-based replication:** The application software or the LVM ensures that a copy of the data managed by them is maintained either locally or at a remote site for recovery purposes.

RTO AND RPO

Recovery Point Objective (RPO) and Recovery Time Objective (RTO) are two of the most important parameters of a disaster recovery or data protection plan. These are objectives which can guide enterprises to choose an optimal data backup plan.

The RPO/RTO, along with a business impact analysis, provides the basis for identifying and analyzing viable strategies for inclusion in the business continuity plan. Viable strategy options include any which would enable resumption of a business process in a time frame at or near the RPO/RTO.

RPO: Recovery Point Objective

Recovery Point Objective (RPO) describes the interval of time that might pass during a disruption before the quantity of data lost during that period exceeds the Business Continuity Plan’s maximum allowable threshold or “tolerance.”

Example: If the last available good copy of data upon an outage is from 18 hours ago, and the RPO for this business is 20 hours then we are still within the parameters of the Business Continuity Plan’s RPO. In other words it the answers the question – “Up to what point in time could the Business Process’s recovery proceed tolerably given the volume of data lost during that interval?”

RTO: Recovery Time Objective

The Recovery Time Objective (RTO) is the duration of time and a service level within which a business process must be restored after a disaster in order to avoid unacceptable consequences associated with a break in continuity. In other words, the RTO is the answer to the question: “How much time did it take to recover after notification of business process disruption?“

RPO designates the variable amount of data that will be lost or will have to be re-entered during network downtime. RTO designates the amount of “real time” that can pass before the disruption begins to seriously and unacceptably impede the flow of normal business operations.

There is always a gap between the actuals (RTA/RPA) and objectives introduced by various manual and automated steps to bring the business application up. These actuals can only be exposed by disaster and business disruption rehearsals.

Architecture of Backup and Recovery

A backup is a copy of production data, created and retained for the sole purpose of recovering deleted or corrupted data. With growing business and regulatory demands for data storage, retention, and availability, organizations are faced with the task of backing up an ever-increasing amount of data.

Organizations must ensure that the right data is in the right place at the right time. Evaluating backup technologies, recovery, and retention requirements for data and applications is an essential step to ensure successful implementation of the backup and recovery solution. The solution must facilitate easy recovery and retrieval from backups and archives as required by the business.

Backup Purpose

Backups are performed to serve three purposes: disaster recovery, operational backup, and archival.

Disaster Recovery

Backups can be performed to address disaster recovery needs. The backup copies are used for restoring data at an alternate site when the primary site is incapacitated due to a disaster. Based on RPO and RTO requirements, organizations use different backup strategies for disaster recovery. When a tape-based backup method is used as a disaster recovery strategy, the backup tape media is shipped and stored at an offsite location. These tapes can be recalled for restoration at the disaster recovery site.

Operational Backup

Data in the production environment changes with every business transaction and operation. *Operational backup* is a backup of data at a point in time and is used to restore data in the event of data loss or logical corruptions that may occur during routine processing. The majority of restore requests in most organizations fall in this category. For example, it is

common for a user to accidentally delete an important e-mail or for a file to become corrupted, which can be restored from operational backup.

Operational backups are created for the active production information by using incremental or differential backup techniques, detailed later in this chapter. An example of an operational backup is a backup performed for a production database just before a bulk batch update. This ensures the availability of a clean copy of the production database if the batch update corrupts the production database.

Archival

Backups are also performed to address archival requirements. Although CAS has emerged as the primary solution for archives, traditional backups are still used by small and medium enterprises for long-term preservation of transaction

records, e-mail messages, and other business records required for regulatory compliance.

Apart from addressing disaster recovery, archival, and operational requirements, backups serve as a protection

against
data loss due to physical damage of a storage device, software failures, or virus attacks. Backups can also be used to protect against accidents such as a deletion or intentional data destruction.

Backup Considerations

The amount of data loss and downtime that a business can endure in terms of RTO and RPO are the primary considerations in selecting and implementing a specific backup strategy. Another consideration is the retention period, which defines the duration for which a business needs to retain the backup copies. Some data is retained for years and some only for a few days. For example, data backed up for archival is retained for a longer period than data backed up for operational recovery.

It is also important to consider the backup media type, based on the retention period and data accessibility. Organizations must also consider the granularity of backups, explained later in this chapter. The development of a backup strategy must include a decision about the most appropriate time for performing a backup in order to minimize any disruption to production operations. Similarly, the location and time of the restore operation must be considered, along with file characteristics and data compression that influences the backup process.

Location, size, and number of files should also be considered, as they may affect the backup process. Location is an important consideration for the data to be backed up. Many organizations have dozens of heterogeneous platforms supporting complex solutions. Consider a data warehouse environment that uses backup data from many sources. The backup process must address these sources in terms of transactional and content integrity. This process must be coordinated with all heterogeneous platforms on which the data resides.

File size also influences the backup process. Backing up large-size files (example: ten 1 MB files) may use less system resources than backing up an equal amount of data comprising a large number of small-size files (example: ten thousand 1 KB files). The backup and restore operation takes more time when a file system contains many small files.

Like file size, the number of files to be backed up also influences the backup process. For example, in incremental backup, a file system containing one million files with a 10 percent daily change rate will have to create 100,000 entries in the backup catalog, which contains the table of contents for the backed up data set and information about the backup session. This large number of entries in the file system affects the performance of the backup and restore process because it takes a long time to search through a file system.

Backup performance also depends on the media used for the backup. The time-consuming operation of starting and stopping in a tape-based system affects backup performance, especially while backing up a large number of small files.

Data compression is widely used in backup systems because compression saves space on the media. Many backup devices, such as tape drives, have built-in support for hardware-based data compression. To effectively use this, it is important to understand the characteristics of the data.

Backup Granularity

Backup granularity depends on business needs and required RTO/RPO. Based on granularity, backups can be categorized as full, cumulative, and incremental.

Most organizations use a combination of these three backup types to meet their backup and recovery requirements. Figure 12-1 depicts the categories of backup granularity.

1. Full backup is a backup of the complete data on the production volumes at a certain point in time. A full backup copy is created by copying the data on the production volumes to a secondary storage device.

Synthetic (or *constructed*) *full backup* is another type of backup that is used in implementations where the production volume resources cannot be exclusively reserved for a backup process for extended periods to perform a full backup.

It is usually created from the most recent full backup and all the incremental backups performed after that full backup. A synthetic full backup enables a full backup copy to be created offline without disrupting the I/O operation on the production volume. This also frees up network resources from the backup process, making them available for other production uses.

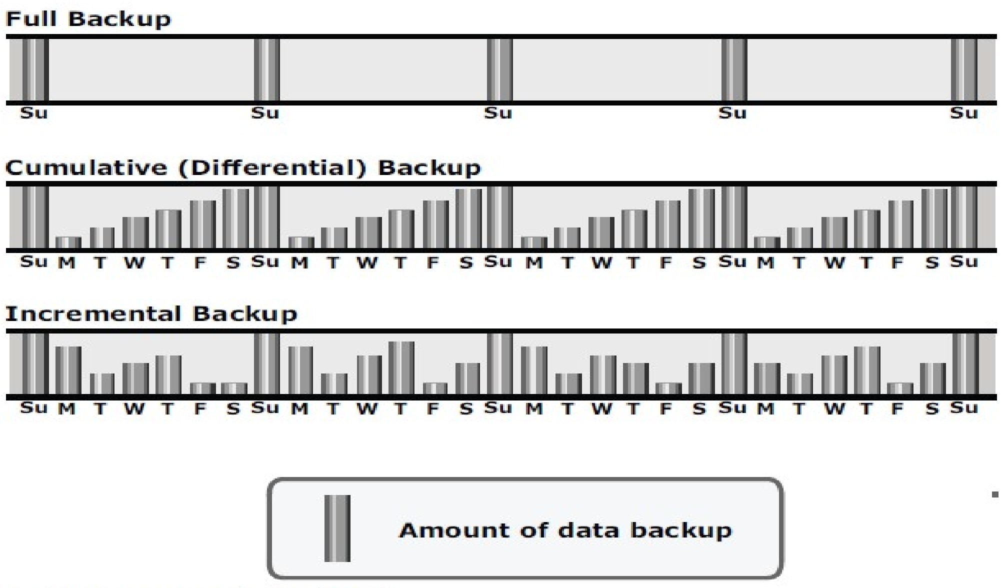


Figure 12-1: Backup granularity levels

2. Incremental backup copies the data that has changed since the last full or incremental backup, whichever has occurred more recently. This is much faster (because the volume of data backed up is restricted to changed data), but it takes longer to restore.

3. Cumulative **(or *differential*)** ***backup*** copies the data that has changed since the last full backup. This method takes longer than incremental backup but is faster to restore.

Restore operations vary with the granularity of the backup.

A full backup provides a single repository from which data can be easily restored.

The process of restoration from an incremental backup requires the last full backup and all the incremental backups available until the point of restoration.

A restore from a cumulative backup requires the last full backup and the most recent cumulative backup.

Restoring from an incremental backup

Figure 12-2 illustrates an example of an incremental backup and restoration.

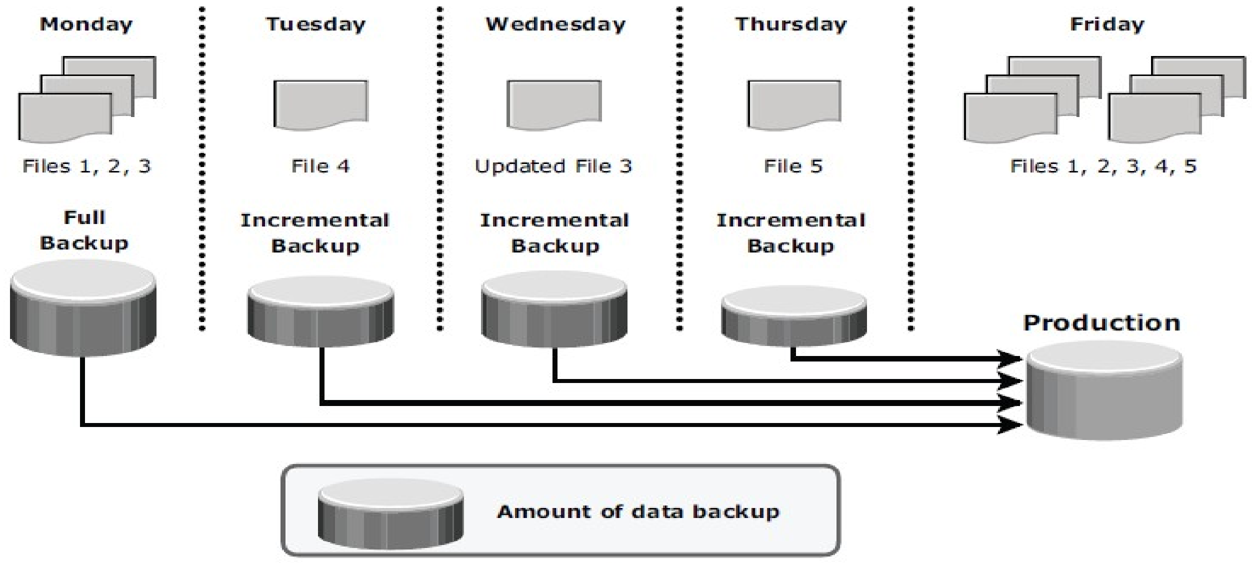


Figure 12-2: Restoring from an incremental backup

In this example, a full backup is performed on Monday evening. Each day after that, an incremental backup is performed. On Tuesday, a new file (File 4 in the figure) is added, and no other files have changed. Consequently, only File 4 is copied during the incremental backup performed on Tuesday evening. On Wednesday, no new files are added, but File 3 has been modified. Therefore, only the modified File 3 is copied during the incremental backup on Wednesday evening. Similarly, the incremental backup on Thursday copies only File 5. On Friday morning, there is data corruption, which requires data restoration from the backup. The first step toward data restoration is restoring all data from the full backup of Monday evening. The next step is applying the incremental backups of Tuesday, Wednesday, and Thursday. In this manner, data can be successfully restored to its previous state, as it existed on Thursday evening.

Restoring a cumulative backup

Figure 12-3 illustrates an example of cumulative backup and restoration.

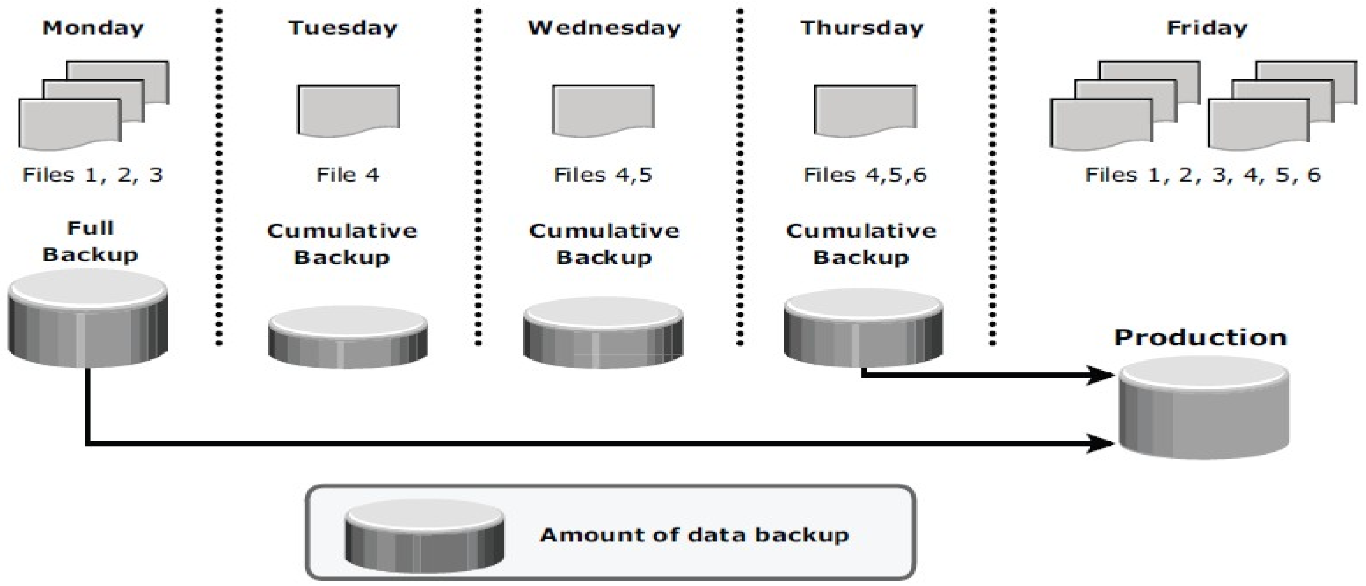


Figure 12-3: Restoring a cumulative backup

In this example, a full backup of the business data is taken on Monday evening. Each day after that, a cumulative backup is taken. On Tuesday, File 4 is added and no other data is modified since the previous full backup of Monday evening. Consequently, the cumulative backup on Tuesday evening copies only File 4. On Wednesday, File 5 is added. The cumulative backup taking place on Wednesday evening copies both File 4 and File 5 because these files have been added or modified since the last full backup. Similarly, on Thursday, File 6 is added. Therefore, the cumulative backup on Thursday evening copies all three files: File 4, File 5, and File 6.

On Friday morning, data corruption occurs that requires data restoration using backup copies. The first step in restoring data from a cumulative backup is restoring all data from the full backup of Monday evening. The next step is to apply only the latest cumulative backup — Thursday evening.

Recovery Considerations

RPO and RTO are major considerations when planning a backup strategy. RPO defines the tolerable limit of data loss for a business and specifies the time interval between two backups. In other words, the RPO determines backup frequency. For example, if application A requires an RPO of one day, it would need the data to be backed up at least once every day. The retention period for a backup is also derived from an RPO specified for operational recovery. For example, users of application "A" may request to restore the application data from its operational backup copy, which was created a month ago. This determines the retention period for the backup. The RPO for application A can therefore range from one day to one month based on operational recovery needs. However, the organization may choose to retain the backup for a longer period of time because of internal policies or external factors, such as regulatory directives.

If short retention periods are specified for backups, it may not be possible to recover all the data needed for the requested recovery point, as some data may be older than the retention period. Long retention periods can be defined for all backups, making it possible to meet any RPO within the defined retention periods. However, this requires a large storage space, which translates into higher cost. Therefore, it is important to define the retention period based on an analysis of all the restore requests in the past and the allocated budget.

RTO relates to the time taken by the recovery process. To meet the defined RTO, the business may choose to use a combination of different backup solutions to minimize recovery time. In a backup environment, RTO influences the type of backup media that should be used. For example, recovery from data streams multiplexed in tape takes longer to complete than recovery from tapes with no multiplexing.

Organizations perform more full backups than they actually need because of recovery constraints. Cumulative and incremental backups depend on a previous full backup. When restoring from tape media, several tapes are needed to fully recover the system. With a full backup, recovery can be achieved with a lower RTO and fewer tapes.

Backup Methods

Hot backup and cold backup are the two methods deployed for backup. They are based on the state of the application when the backup is performed.

In a *hot backup*, the application is up and running, with users accessing their data during the backup process. In a *cold backup*, the application is not active during the backup process.

The backup of online *production data* becomes more challenging because data is actively being used and changed. An open file is locked by the operating system and is not copied during the backup process until the user closes it. The backup application can back up open files by retrying the operation on files that were opened earlier in the backup process. During the backup process, it may be possible that files opened earlier will be closed and a retry will be successful.

The maximum number of retries can be configured depending on the backup application. However, this method is not considered robust because in some environments certain files are always open.

In such situations, the backup application provides *open file agents*. These agents interact directly with the operating system and enable the creation of consistent copies of open files. In some environments, the use of open file agents is not enough. For example, a database is composed of many files of varying sizes, occupying several file systems. To ensure a consistent database backup, all files need to be backed up in the same state. That does not necessarily mean that all files need to be backed up at the same time, but they all must be synchronized so that the database can be restored with consistency.

Consistent backups of databases can also be done by using a cold backup. This requires the database to remain inactive during the backup. Of course, the disadvantage of a cold backup is that the database is inaccessible to users during the backup process.

Hot backup is used in situations where it is not possible to shut down the database. This is facilitated by *database backup agents* that can perform a backup while the database is active. The disadvantage associated with a hot backup is that the agents usually affect overall application performance.

A *point-in-time (PIT)* copy method is deployed in environments where the impact of downtime from a cold backup or the performance resulting from a hot backup is unacceptable. A pointer-based PIT copy consumes only a fraction of the storage space and can be created very quickly. A pointer-based PIT copy is implemented in a disk-based solution whereby a virtual LUN is created and holds pointers to the data stored on the production LUN or save location. In this method of backup, the database is stopped or frozen momentarily while the PIT copy is created. The PIT copy is then mounted on a secondary server and the backup occurs on the primary server. This technique is detailed in Chapter 13. To ensure consistency, it is not enough to back up only production data for recovery. Certain attributes and properties attached to a file, such as permissions, owner, and other metadata, also need to be backed up. These attributes are as important as the data itself and must be backed up for consistency. Backup of boot sector and partition layout information is also critical for successful recovery.

In a disaster recovery environment, *bare-metal recovery (BMR)* refers to a backup in which all metadata, system information, and application configurations are appropriately backed up for a full system recovery. BMR builds the base system, which includes partitioning, the file system layout, the operating system, the applications, and all the relevant configurations. BMR recovers the base system first, before starting the recovery of data files. Some BMR technologies can recover a server onto dissimilar hardware.

Backup Process

A backup system uses client/server architecture with a backup server and multiple backup clients. The backup server manages the backup operations and maintains the backup catalog, which contains information about the backup process and backup metadata. The backup server depends on backup clients to gather the data to be backed up. The backup clients can be local to the server or they can reside on another server, presumably to back up the data visible to that server. The backup server receives backup metadata from the backup clients to perform its activities.

Figure 12-4 illustrates the backup process.

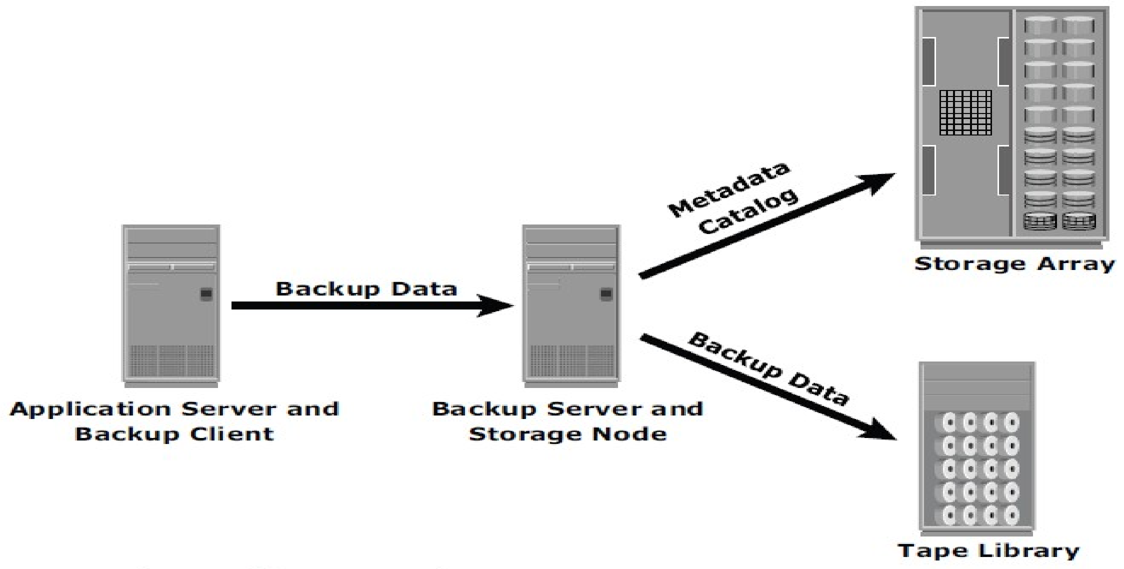

Application Server and Backup Client
Backup Data
Backup Server and Storage Node
Metadata Catalog
Storage Array
Backup Data
Tape Library

Figure 12-4: Backup architecture and process

The storage node is responsible for writing data to the backup device (in a backup environment, a storage node is a host that controls backup devices). Typically, the storage node is integrated with the backup server and both are hosted on the same physical platform. A backup device is attached directly to the storage node's host platform. Some backup architecture refers to the storage node as the *media server* because it connects to the storage device. Storage nodes play an important role in backup planning because they can be used to consolidate backup servers.

The backup process is based on the policies defined on the backup server, such as the time of day or completion of an event. The backup server then initiates the process by sending a request to a backup client (backups can also be initiated by a client). This request instructs the backup client to send its metadata to the backup server, and the data to be backed up to the appropriate storage node. On receiving this request, the backup client sends the metadata to the backup server. The backup server writes this metadata on its metadata catalog.

The backup client also sends the data to the storage node, and the storage node writes the data to the storage device.

After all the data is backed up, the storage node closes the connection to the backup device. The backup server writes backup completion status to the metadata catalog.

Backup and Restore Operations

When a backup process is initiated, significant network communication takes place between the different components of a backup infrastructure. The backup server initiates the backup process for different clients based on the backup schedule configured for them. For example, the backup process for a group of clients may be scheduled to start at 3:00 am every day.

The backup server coordinates the backup process with all the components in a backup configuration (see Figure 12-5). The backup server maintains the information about backup clients to be contacted and storage nodes to be used in a backup operation. The backup server retrieves the backup-related information from the backup catalog and, based on this information, instructs the storage node to load the appropriate backup media into the backup devices. Simultaneously, it instructs the backup clients to start scanning the data, package it, and send it over the network to the assigned storage node. The storage node, in turn, sends metadata to the backup server to keep it updated about the media being used in the backup process. The backup server continuously updates the backup catalog with this information.

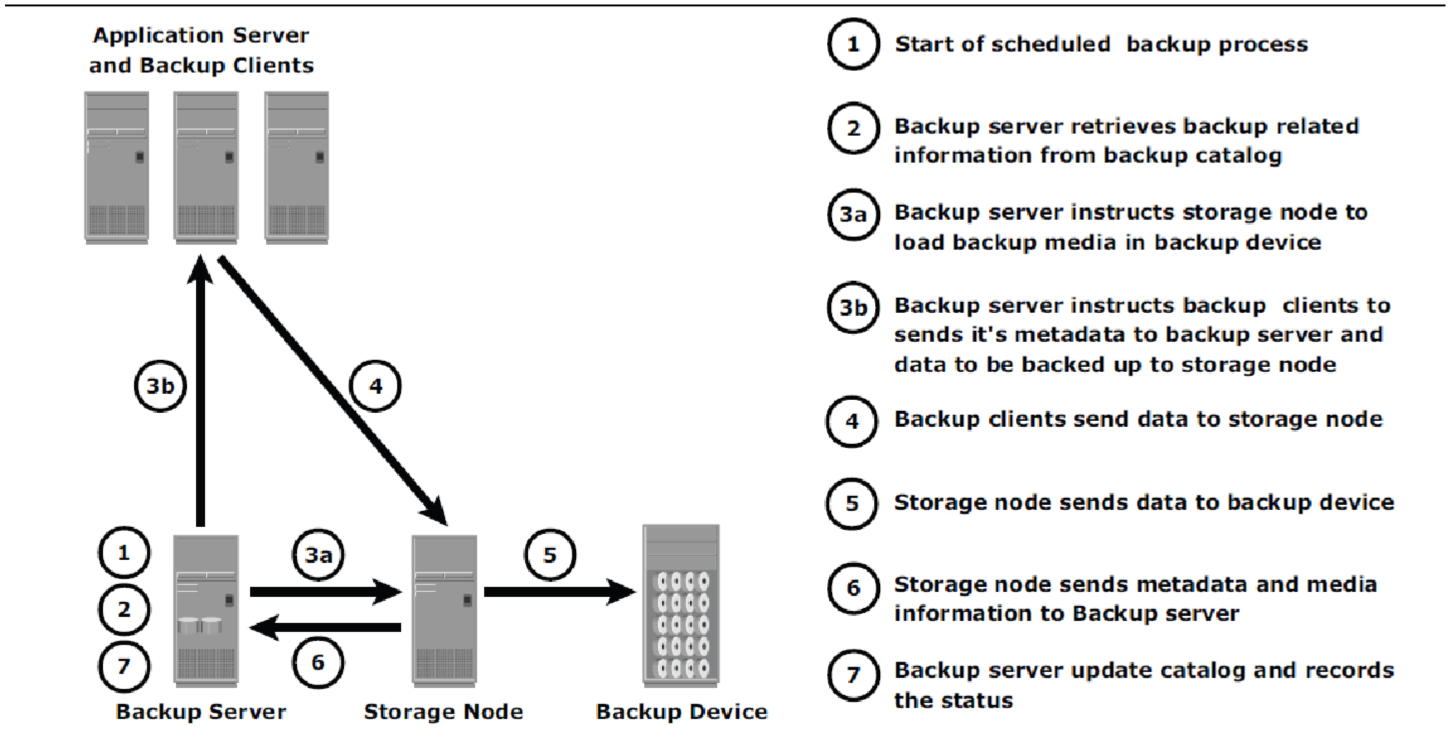

Application Server and Backup Clients
3b
4
1
2
7
3a
6
5
Backup Server
Storage Node
Backup Device
1 Start of scheduled backup process
2 Backup server retrieves backup related information from backup catalog
3a Backup server instructs storage node to load backup media in backup device
3b Backup server instructs backup clients to sends it's metadata to backup server and data to be backed up to storage node
4 Backup clients send data to storage node
5 Storage node sends data to backup device
6 Storage node sends metadata and media information to Backup server
7 Backup server update catalog and records the status

Figure 12-5: Backup operation

After the data is backed up, it can be restored when required. A restore process must be manually initiated. Some backup software has a separate application for restore operations. These restore applications are accessible only to the administrators.

Figure 12-6 depicts a restore process.

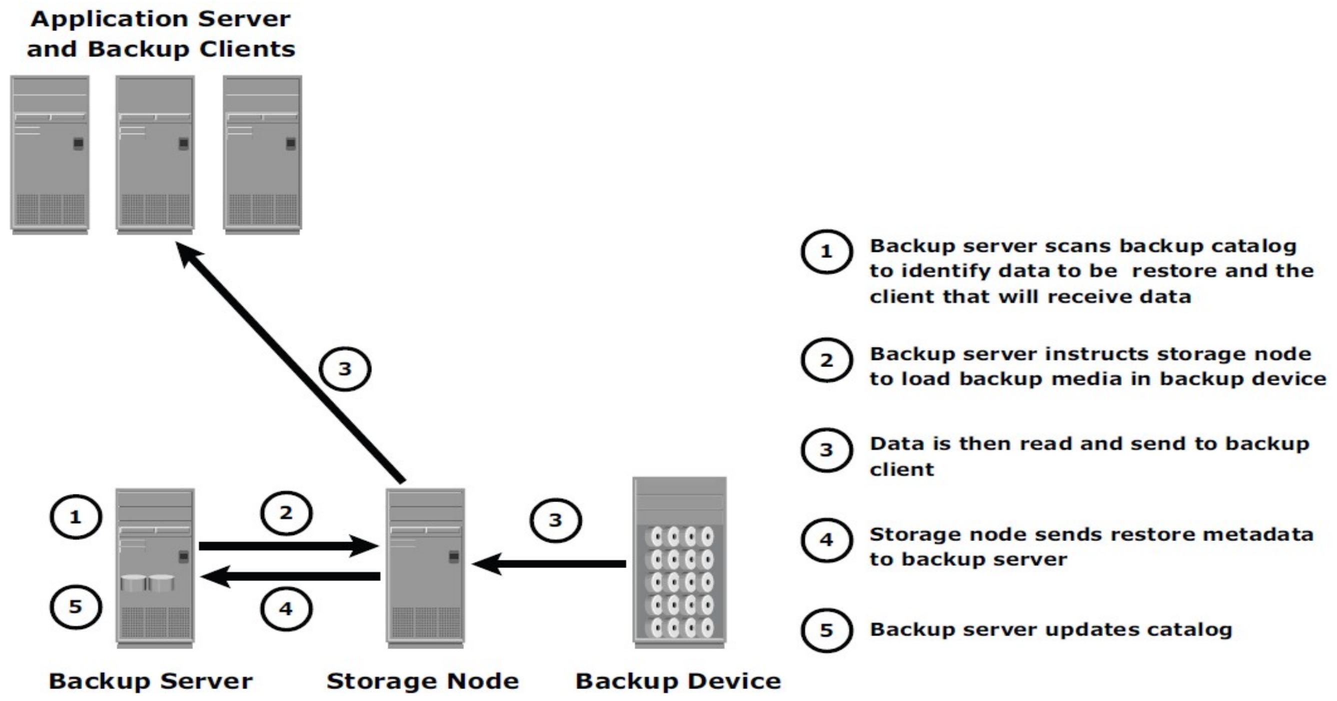

Application Server and Backup Clients
1 Backup server scans backup catalog to identify data to be restore and the client that will receive data
2 Backup server instructs storage node to load backup media in backup device
3 Data is then read and send to backup client
4 Storage node sends restore metadata to backup server
5 Backup server updates catalog
Backup Server
Storage Node
Backup Device

Figure 12-6: Restore operation

Upon receiving a restore request, an administrator opens the restore application to view the list of clients that have been backed up. While selecting the client for which a restore request has been made, the administrator also needs to identify the client that will receive the restored data. Data can be restored on the same client for whom the restore request has been made or on any other client.

The administrator then selects the data to be restored and the specified point in time to which the data has to be restored based on the RPO. Note that because all of this information comes from the backup catalog, the restore application must also communicate to the backup server.

The administrator first selects the data to be restored and initiates the restore process. The backup server, using the appropriate storage node, then identifies the backup media that needs to be mounted on the backup devices. Data is then read and sent to the client that has been identified to receive the restored data.

Backup Topologies

Three basic topologies are used in a backup environment: direct attached backup, LAN based backup, and SAN based backup. A mixed topology is also used by combining LAN based and SAN based topologies.

1. In a *direct-attached backup*, a backup device is attached directly to the client. Only the metadata is sent to the backup server through the LAN. This configuration frees the LAN from backup traffic. The example shown in Figure 12-7 depicts use of a backup device that is not shared. As the environment grows, however, there will be a need for central management of all backup devices and to share the resources to optimize costs.

2. In *LAN-based backup*, all servers are connected to the LAN and all storage devices are directly attached to the storage node (see Figure 12-8). The data to be backed up is transferred from the backup client (source), to the backup device (destination) over the LAN, which may affect network performance. Streaming across the LAN also affects network performance of all systems connected to the same segment as the backup server. Network resources are severely constrained when multiple clients access and share the same tape library unit (TLU).

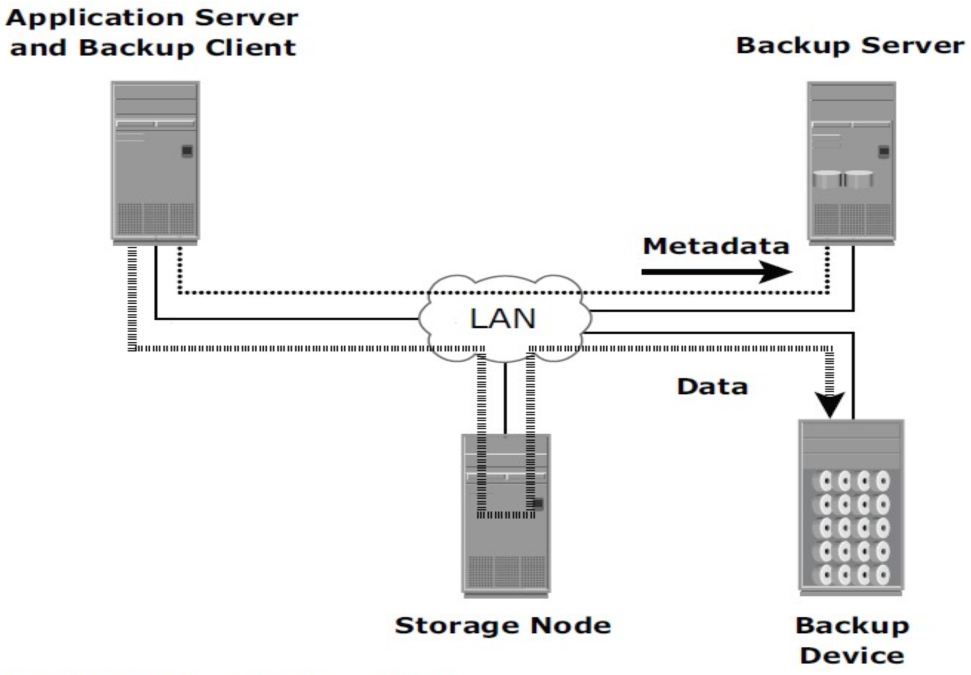


Figure 12-8: LAN-based backup topology

3. The *SAN-based backup* is also known as the *LAN-free backup*. Figure 12-9 illustrates a SAN-based backup. The SAN- based backup topology is the most appropriate solution when a backup device needs to be shared among the clients. In this case the backup device and clients are attached to the SAN.

In this example, clients read the data from the mail servers in the SAN and write to the SAN attached backup device. The backup data traffic is restricted to the SAN, and backup metadata is transported over the LAN. However, the volume of metadata is insignificant when compared to production data. LAN performance is not degraded in this configuration.

4. The *mixed topology* uses both the LAN-based and SAN-based topologies, as shown in Figure 12-10. This topology might be implemented for several reasons, including cost, server location, reduction in administrative overhead, and performance considerations.

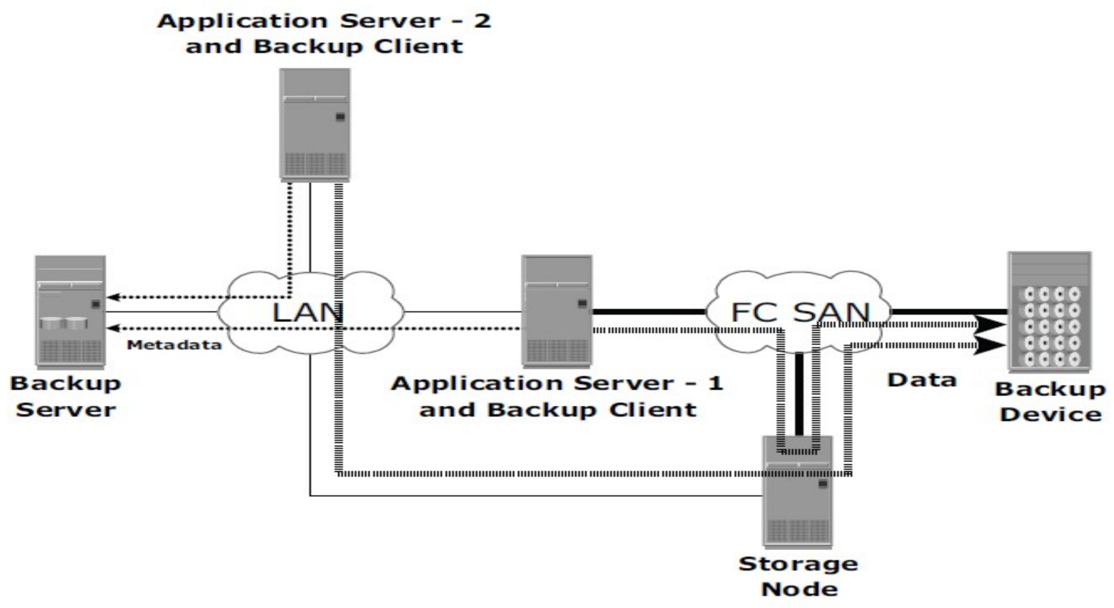

Application Server - 2 and Backup Client
LAN
Metadata
Backup Server
Application Server - 1 and Backup Client
FC SAN
Data
Backup Device
Storage Node

Figure 12-10: Mixed backup topology

Serverless Backup

Serverless backup is a LAN-free backup methodology that does not involve a backup server to copy data. The copy may be created by a network-attached controller, utilizing a SCSI extended copy or an appliance within the SAN. These backups are called serverless because they use SAN resources instead of host resources to transport backup data from its source to the backup device, reducing the impact on the application server.

Backup in NAS Environments

The use of NAS heads imposes a new set of considerations on the backup and recovery strategy in NAS environments. NAS heads use a proprietary operating system and file system structure supporting multiple file-sharing protocols.

In the NAS environment, backups can be implemented in four different ways: Server based, Serverless, Network Data Management Protocol (NDMP) in either NDMP 2-way and NDMP 3-way.

1. In *application server-based backup*, the NAS head retrieves data from storage over the network and transfers it to the backup client running on the application server. The backup client sends this data to a storage node, which in turn writes the data to the backup device. This results in overloading the network with the backup data and the use of production (application) server resources to move backup data. Figure 12-11 illustrates server-based backup in the NAS environment.

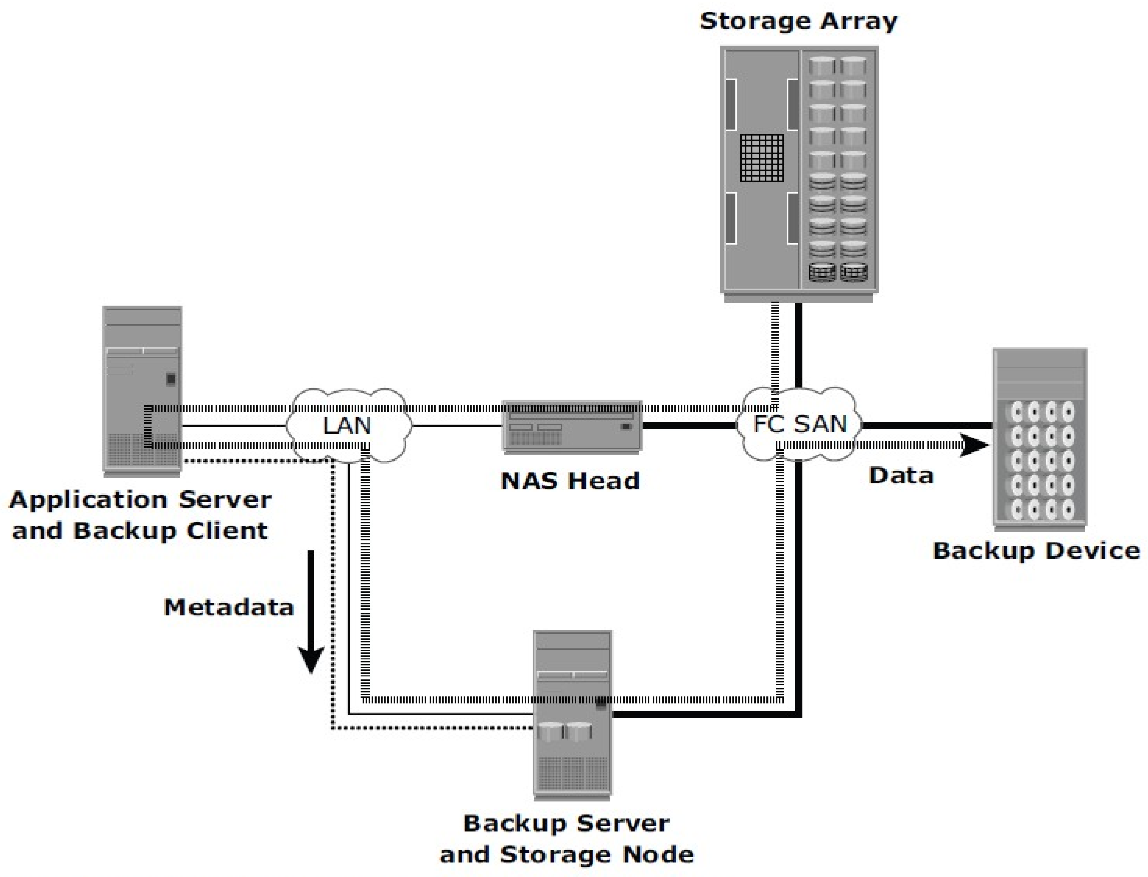

Storage Array
LAN
FC SAN
Application Server and Backup Client
NAS Head
Data
Backup Device
Metadata
Backup Server and Storage Node

Figure 12-11: Server-based backup in NAS environment

2. In *serverless backup*, the network share is mounted directly on the storage node. This avoids overloading the network during the backup process and eliminates the need to use resources on the production server. Figure 12-12 illustrates serverless backup in the NAS environment. In this scenario, the storage node, which is also a backup client, reads the data from the NAS head and writes it to the backup device without involving the application server. Compared to the previous solution, this eliminates one network hop.

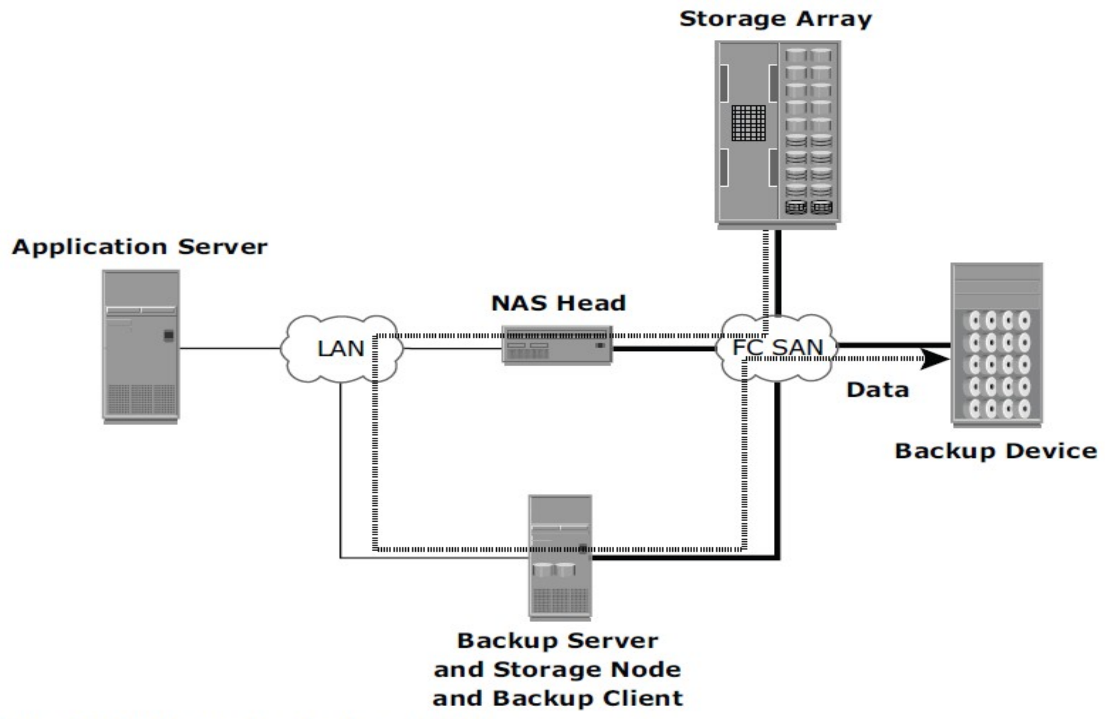


Figure 12-12: Serverless backup in NAS environment

3. In NDMP, backup data is sent directly from the NAS head to the backup device, while metadata is sent to the backup server. Figure 12-13 illustrates backup in the NAS environment using NDMP 2-way. In this model, network traffic is minimized by isolating data movement from the NAS head to the locally attached tape library. Only metadata is transported on the network. This backup solution meets the strategic need to centrally manage and control distributed data while minimizing network traffic.

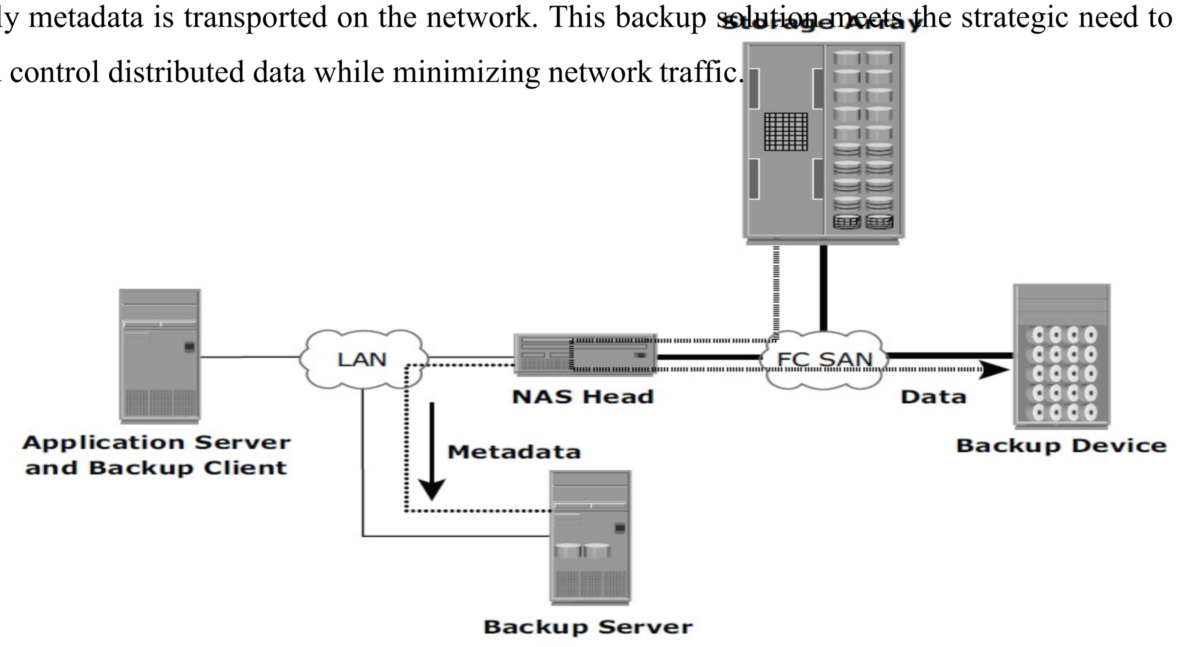


Figure 12-13: NDMP 2-way in NAS environment

4. In an ***NDMP 3-way*** file system, data is not transferred over the public network. A separate private backup network must be established between all NAS heads and the "backup" NAS head to prevent any data transfer on the public network in order to avoid any congestion or affect production operations. Metadata and NDMP control data is still transferred across the public network. Figure 12-14 depicts NDMP 3-way backup.

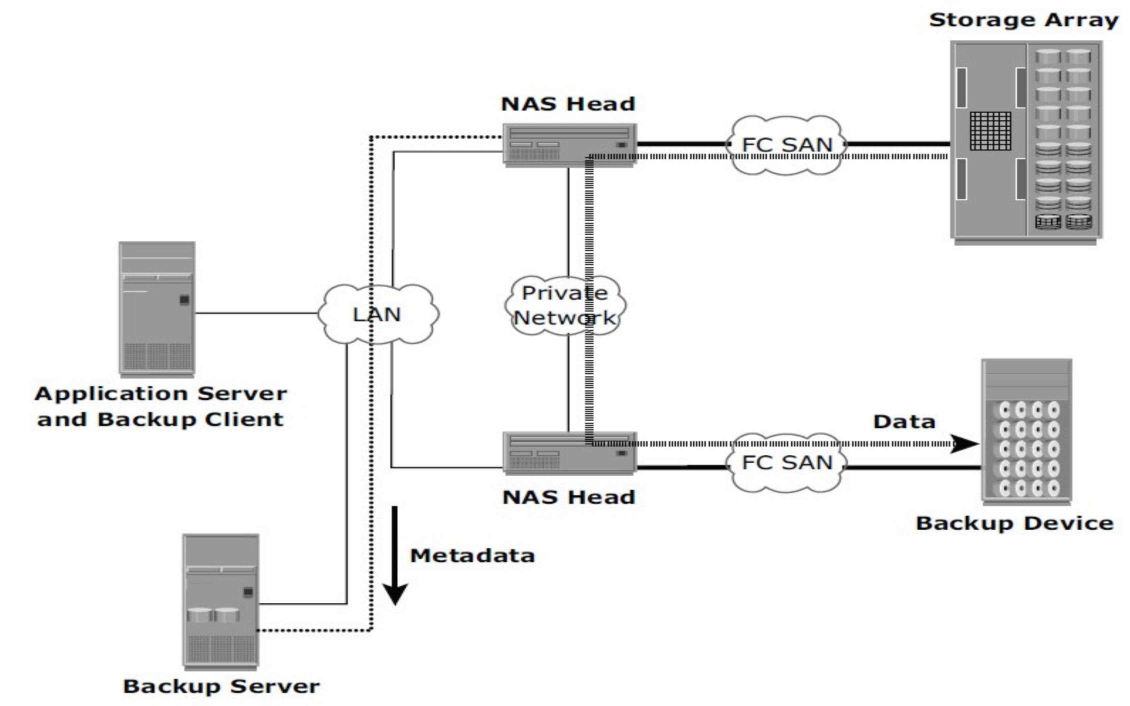


Figure 12-14: NDMP 3-way in NAS environment

Backup Technologies

A wide range of technology solutions are currently available for backup. Tapes and disks are the two most commonly used backup media. The tape technology has matured to scale to enterprise demands, whereas backup to disk is emerging as a viable option with the availability of low-cost disks. Virtual tape libraries use disks as backup medium emulating tapes, providing enhanced backup and recovery capabilities.

Backup to Tape

Tapes, a low-cost technology, are used extensively for backup. Tape drives are used to read/write data from/to a tape cartridge. Tape drives are referred to as sequential, or linear, access devices because the data is written or read sequentially.

Tape Mounting is the process of inserting a tape cartridge into a tape drive. The tape drive has motorized controls to move the magnetic tape around, enabling the head to read or write data.

Several types of tape cartridges are available. They vary in size, capacity, shape, number of reels, density, tape length, tape thickness, tape tracks, and supported speed. Today, a tape cartridge is composed of a magnetic tape with single or dual reels in a plastic enclosure.

Physical Tape Library

The physical tape library provides housing and power for a number of tape drives and tape cartridges, along with a robotic arm or picker mechanism. The backup software has intelligence to manage the robotic arm and entire backup process.

Tape drives read and write data from and to a tape. Tape *cartridges* are placed in the *slots* when not in use by a tape drive. *Robotic arms* are used to move tapes around the library, such as moving a tape drive into a slot. Another type of slot called a *mail* or *import/export slot* is used to add or remove tapes from the library without opening the access doors (refer to Figure 12-15 Front View) because opening the access doors causes a library to go offline. In addition, each physical component in a tape library has an individual *element address* that is used as an addressing mechanism for moving tapes around the library.

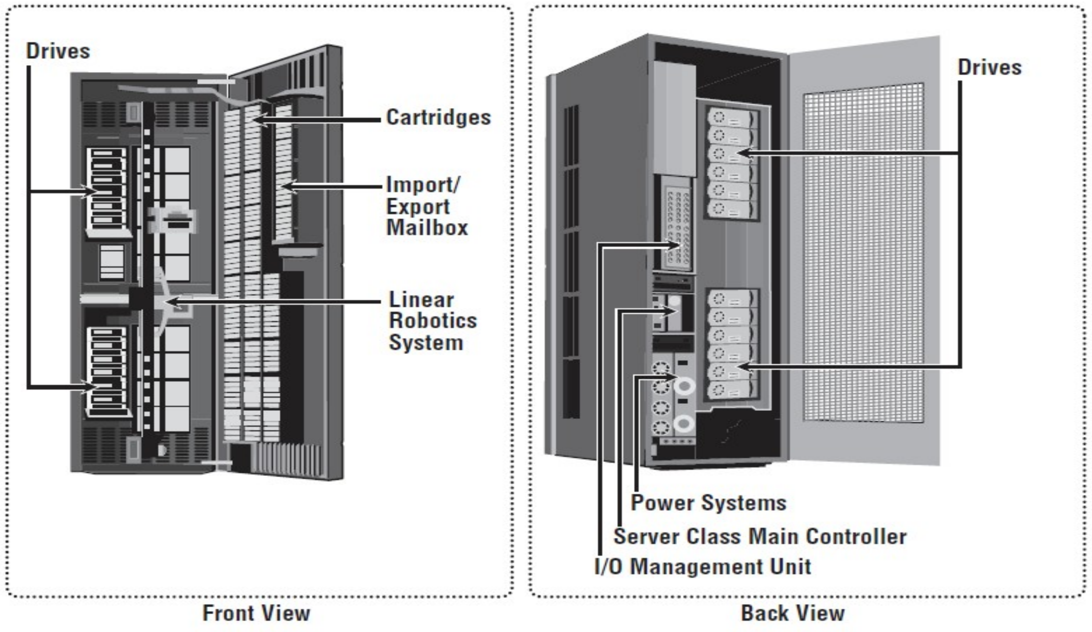


Figure 12-15: Physical tape library

Tape drive *streaming* or *multiple streaming* writes data from multiple streams on a single tape to keep the drive busy. Shown in Figure 12-16, multiple streaming improves media performance, but it has an associated disadvantage. The backup data is interleaved because data from multiple streams is written on it. Consequently, the data recovery time is increased.

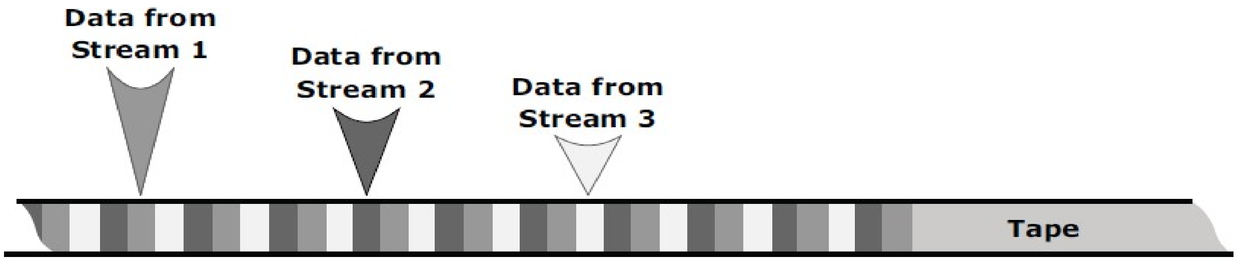


Figure 12-16: Multiple streams on tape media

Limitations of Tape

Tapes are primarily used for long-term offsite storage because of their low cost. Tapes must be stored in locations with a controlled environment to ensure preservation of the media and prevent data corruption. Data access in a tape is sequential, which can slow backup and recovery operations. Physical transportation of the tapes to offsite locations also adds management overhead.

Backup to Disk

Disks have now replaced tapes as the primary device for storing backup data because of their performance advantages. Backup-to-disk systems offer ease of implementation, reduced cost, and improved quality of service. Apart from performance benefits in terms of data transfer rates, disks also offer faster recovery when compared to tapes.

Backing up to disk storage systems offers clear advantages due to their inherent random access and RAID-protection capabilities. In most backup environments, backup to disk is used as a staging area where the data is copied temporarily before transferring or staging it to tapes later. This enhances backup performance. Some backup products allow for backup images to remain on the disk for a period of time even after they have been staged. This enables a much faster restore.

Virtual Tape Library

A *virtual tape library (VTL)* has the same components as that of a physical tape library except that the majority of the components are presented as virtual resources. For backup software, there is no difference between a physical tape library and a virtual tape library. Figure 12-18 shows a virtual tape library. Virtual tape libraries use disks as backup media. Emulation software has a database with a list of virtual tapes, and each virtual tape is assigned a portion of a LUN on the disk. A virtual tape can span multiple LUNs if required.

Similar to a physical tape library, a robot mount is performed when a backup process starts in a virtual tape library. However, unlike a physical tape library, where this process involves some mechanical delays, in a virtual tape library it is almost instantaneous. Even the *load to ready* time is much less than in a physical tape library.

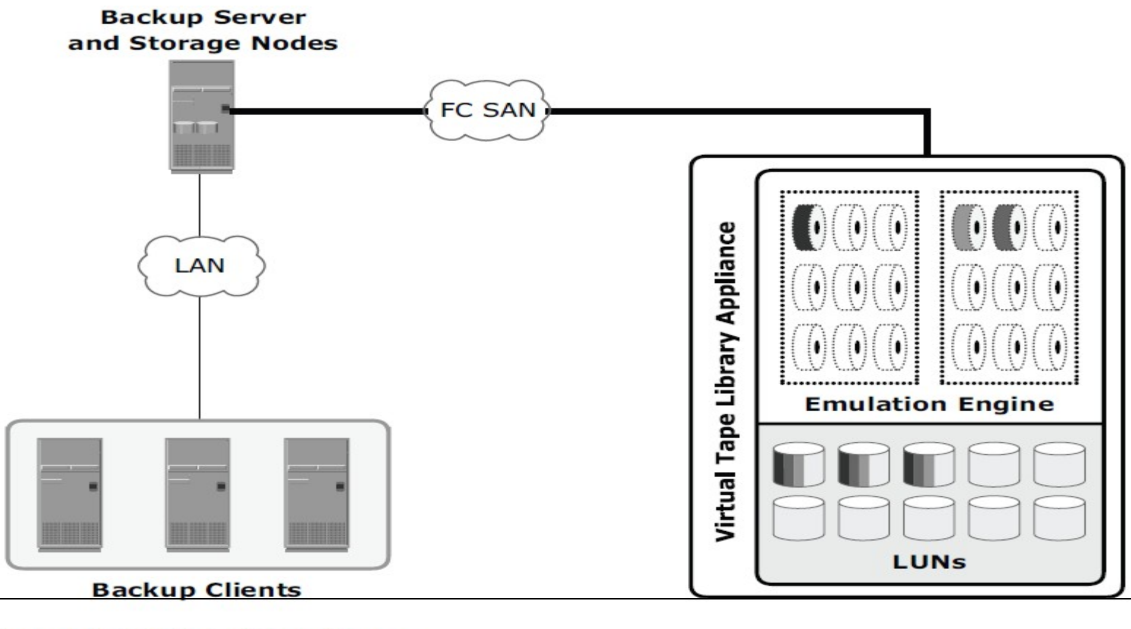

Backup Server and Storage Nodes
FC SAN
LAN
Backup Clients
Virtual Tape Library Appliance
Emulation Engine
LUNs

Figure 12-18: Virtual tape library

After the virtual tape is mounted and the tape drive is positioned, the virtual tape is ready to be used, and backup data can be written to it. Unlike a physical tape library, the virtual tape library is not constrained by the shoe shining effect. In most cases, data is written to the virtual tape immediately. When the operation is complete, the backup software issues a rewind command and then the tape can be unmounted. This rewind is also instantaneous. The virtual tape is then unmounted, and the virtual robotic arm is instructed to move it back to a virtual slot.

Advantages

Using virtual tape offers several advantages over both physical tapes and disks. Compared to physical tape, virtual tape offers better single stream performance, better reliability, and random disk access characteristics. Backup and restore operations are sequential by nature, but they benefit from the disk's random access characteristics because they are always online and ready to be used, improving backup and recovery times. Virtual tape does not require the usual maintenance tasks associated with a physical tape drive, such as periodic cleaning and drive calibration. Compared to backup-to-disk devices, virtual tapes offer easy installation and administration and inherent offsite capabilities. In addition, virtual tapes do not require any additional modules or changes on the backup software.

Local Replication, Remote Replication

Local Replication

Introduction

Replication is the process of creating an exact copy of data. Creating one or more replicas of the production data is one of the ways to provide Business Continuity (BC).

These replicas can be used for recovery and restart operations in the event of data loss.

The primary purpose of replication is to enable users to have designated data at the right place, in a state appropriate to the recovery need. The replica should provide recoverability and restartability.

Recoverability enables restoration of data from the replicas to the production volumes in the event of data loss or data corruption. It must provide minimal RPO and RTO for resuming business operations on the production volumes, while restartability must ensure consistency of data on the replica. This enables restarting business operations using the replicas.

Source and Target

A host accessing data from one or more LUNs on the storage array is called a *production host,* and these LUNs are known as source LUNs (devices/volumes), production LUNs, or simply the *source*.

A LUN (or LUNs) on which the data is replicated is called the target LUN or simply the *target* or replica. Targets can also be accessed by hosts other than production hosts to perform operations such as backup or testing. Target data can be updated by the hosts accessing it without modifying the source. However, the source and the target are not an identical copy of each other anymore. The target can be incrementally resynchronized (copying of data that has changed since the previous synchronization) with the source to make both source and target identical.

Uses of Local Replicas

One or more local replicas of the source data can be created for various purposes, including the following:

1. Alternate source for backup: Under normal backup operations, data is read from the production volumes (LUNs) and written to the backup device. This places additional burden on the production infrastructure, as production LUNs are simultaneously involved in production work. As the local replica contains an exact point-in-time (PIT) copy of the source data, it can be used to perform backup operations. This alleviates the backup I/O workload on the production volumes. Another benefit of using local replicas for backup is that it reduces the *backup window* to zero.

2. Fast recovery: In the event of a partial failure of the source, or data corruption, a local replica can be used to recover lost data. In the event of a complete failure of the source, the replica can be restored to a different set of source devices. In either case, this method provides faster recovery and minimal RTO, compared to traditional restores from tape backups.

In many instances business operations can be started using the source device before the data is completely copied from the replica.

3. Decision-support activities such as reporting: Running the reports using the data on the replicas greatly reduces the I/O burden placed on the production device.

4. Testing platform: A local replica can be used for testing critical business data or applications. For example, when planning an application upgrade, it can be tested using the local replica. If the test is successful, it can be restored to the source volumes.

5. Data migration: Local replication can also be used for data migration. Data migration may be performed for various reasons, such as migrating from a small LUN to a larger LUN.

Data Consistency

Most file systems and databases buffer data in the host before it is written to disk. A consistent replica ensures that data buffered in the host is properly captured on the disk when the replica is created. Ensuring consistency is the primary requirement for all the replication technologies.

Consistency of a Replicated File System

File systems buffer data in host memory to improve application response time. The buffered information is periodically written to disk. In UNIX operating systems, the *sync daemon* is the process that flushes the buffers to disk at set intervals. In some cases, the replica may be created in between the set intervals. Hence, the host memory buffers must be flushed to ensure data consistency on the replica, prior to its creation. Figure 13-1 illustrates flushing of the buffer to its source, which is then replicated. If the host memory buffers are not flushed, data on the replica will not contain the information that was buffered in the host. If the file system is unmounted prior to the creation of the replica, the buffers would be automatically flushed and data would be consistent on the replica. If a mounted file system is replicated, some level of recovery such as *fsck* or *log replay* would be required on the replicated file system. When the file system replication process is completed, the replica file system can be mounted for operational use.

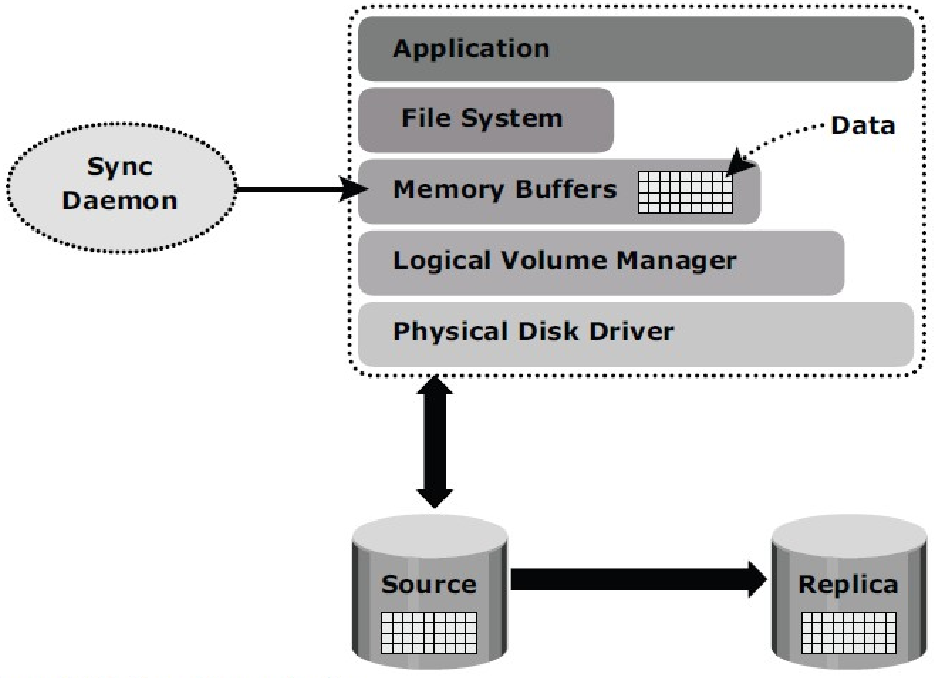


Figure 13-1: File system replication

Consistency of a Replicated Database

A database may be spread over numerous files, file systems, and devices. All of these must be replicated consistently to ensure that the replica is restorable and restartable. Replication can be performed with the database offline or online. If the database is offline, it is not available for I/O operations. Because no updates are occurring, the replica will be consistent.

If the database is online, it is available for I/O operations. Transactions to the database will be updating data continuously. When a database is backed up while it is online, changes made to the database at this time must be applied to the backup copy to make it consistent. Performing an online backup requires additional procedures during backup and restore. Often these procedures can be scripted to automate the process, alleviating administrative work and minimizing human error. Most databases support some form of online or hot backups. There will be increased logging activity during the time when the database is in the hot backup mode.

An alternate approach exploits the *dependent write I/O* principle inherent in any database management system (DBMS). According to this principle, a write I/O is not issued by an application until a prior related write I/O has completed.

For example, a data write is dependent on the successful completion of the prior log write. Dependent write consistency is required for protection against power outages, loss of local channel connectivity, or storage devices. When the failure occurs a dependent write consistent image is created. A restart transforms the dependent write consistent image to a

transactional consistent image — i.e., committed transactions are recovered, and in-flight transactions are discarded.

In order for a transaction to be deemed complete, databases require that a series of writes have to occur in a particular

order. These writes would be recorded on the various devices/file systems. Figure 13-2, illustrates the process of flushing the buffer from host to source; I/Os 1 to 4 must complete, in order for the transaction to be considered complete. I/O 4 is dependent on I/O 3 and will occur only if I/O 3 is complete. I/O 3 is dependent on I/O 2, which in turn depends on I/O

1. Each I/O completes only after completion of the previous I/O(s).

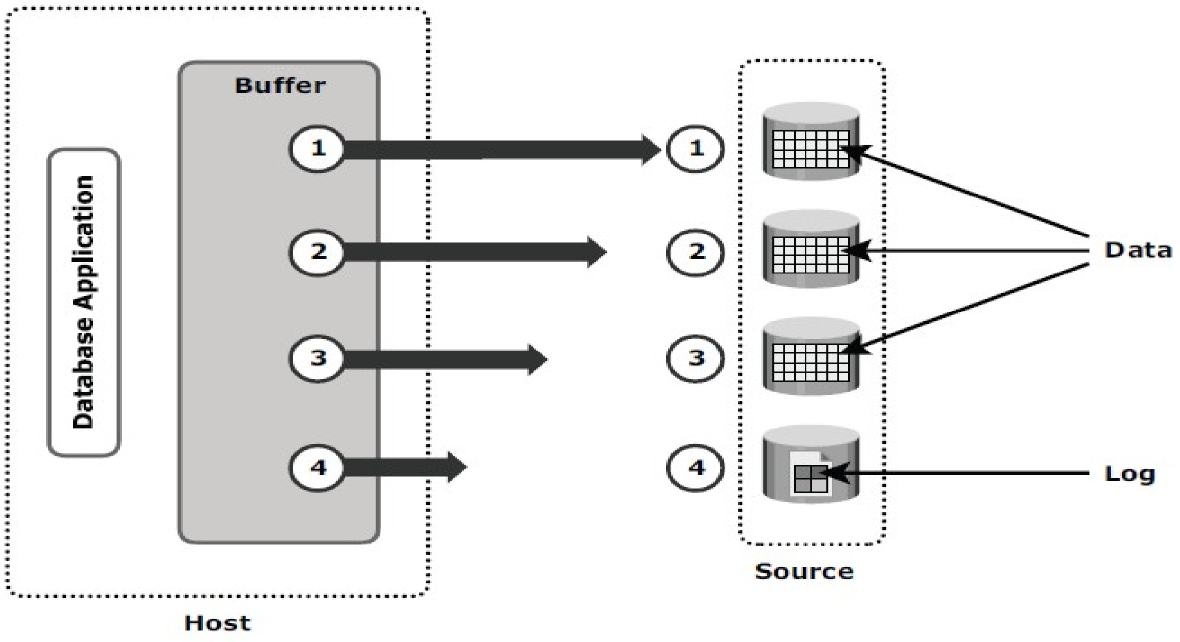


Figure 13-2: Dependent write consistency on sources

At the point in time when the replica is created, all the writes to the source devices must be captured on the replica devices to ensure data consistency. Figure 13-3 illustrates the process of replication from source to replica, I/O transactions 1 to 4 must be carried out in order for the data to be consistent on the replica.

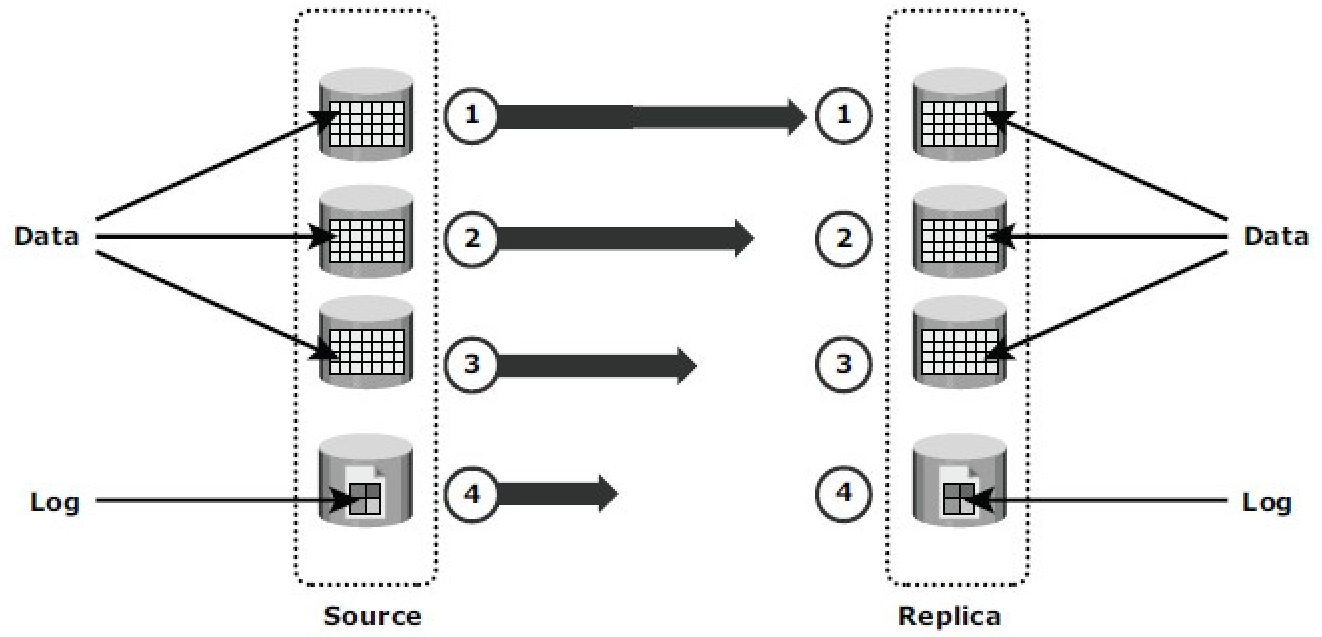


Figure 13-3: Dependent write consistency on replica

Creating a PIT copy for multiple devices happens quickly, but not instantaneously. It is possible that I/O transactions 3 and 4 were copied to the replica devices, but I/O transactions 1 and 2 were not copied. In this case, the data on the replica is inconsistent with the data on the source. If a restart were to be performed on the replica devices, I/O 4, which is available on the replica, might indicate that a particular transaction is complete, but all the data associated with the transaction will be unavailable on the replica, making the replica inconsistent.

Local Replication Technologies

Host-Based Local Replication

In host-based replication, logical volume managers (LVMs) or the file systems perform the local replication process. LVM-based replication and file system (FS) snapshot are examples of host-based local replication.

LVM-Based Replication

In LVM-based replication, logical volume manager is responsible for creating and controlling the host-level logical volume. An LVM has three components: physical volumes (physical disk), volume groups, and logical volumes. A *volume group* is created by grouping together one or more physical volumes. *Logical volumes* are created within a given volume group. A volume group can have multiple logical volumes.

In LVM-based replication, each *logical partition* in a logical volume is mapped to two physical partitions on two

different physical volumes, as shown in Figure 13-4. An application write to a logical partition is written to the two physical partitions by the LVM device driver. This is also known as *LVM mirroring*.
Mirrors can be split and the data contained therein can be independently accessed. LVM mirrors can be added or removed dynamically.

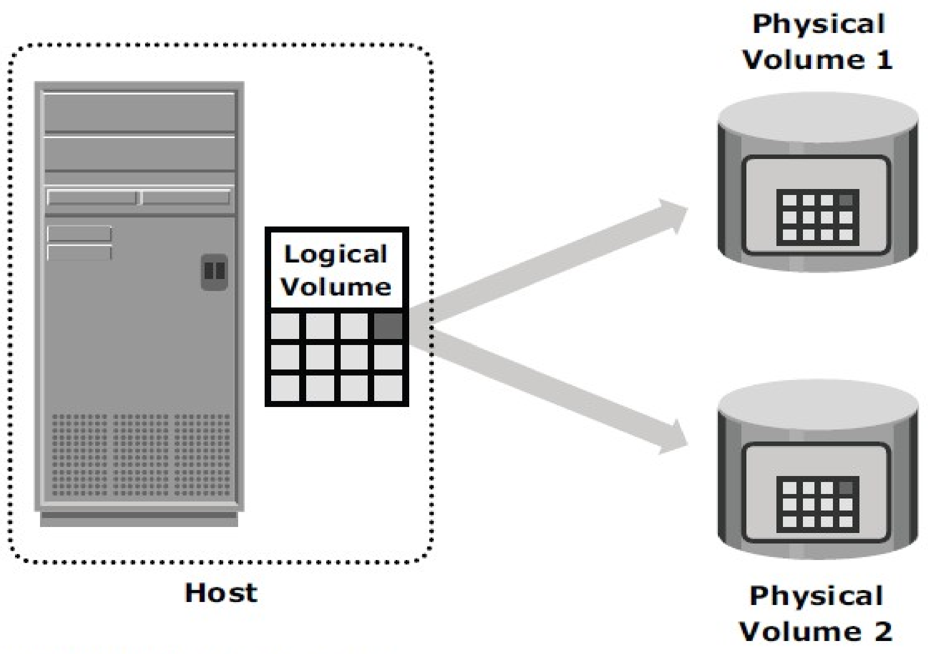


Figure 13-4: LVM-based mirroring

Advantages of LVM-Based Replication
The LVM-based replication technology is not dependent on a vendor-specific storage system. Typically, LVM is part of the operating system and no additional license is required to deploy LVM mirroring.
Limitations of LVM-Based Replication
As every write generated by an application translates into two writes on the disk, an additional burden is placed on the host CPU. This can degrade application performance. Presenting an LVM-based local replica to a second host is usually not possible because the replica will still be part of the volume group, which is usually accessed by one host at any given time.
File System Snapshot
File system (FS) snapshot is a pointer-based replica that requires a fraction of the space used by the original FS. This snapshot can be implemented by either FS itself or by LVM. It uses Copy on First Write (CoFW) principle. CoFW mechanism is discussed later in the chapter.
When the snapshot is created, a bitmap and a blockmap are created in the metadata of the Snap FS. The bitmap is used to keep track of blocks that are changed on the production FS after creation of the snap. The blockmap is used to indicate the exact address from which data is to be read when the data is accessed from the Snap FS.

Immediately after creation of the snapshot all reads from the snapshot will actually be served by reading the production FS.

Storage Array–Based Replication

In *storage array-based local replication*, the array operating environment performs the local replication process. The host

resources such as CPU and memory are not used in the replication process. Consequently, the host is not burdened by the replication operations. The replica can be accessed by an alternate host for any business operations.

In this replication, the required number of replica devices should be selected on the same array and then data is replicated between source-replica pairs. A database could be laid out over multiple physical volumes and in that case all the devices must be replicated for a consistent PIT copy of the database.

Figure 13-5 shows storage array based local replication, where source and target are in the same array and accessed by different hosts.

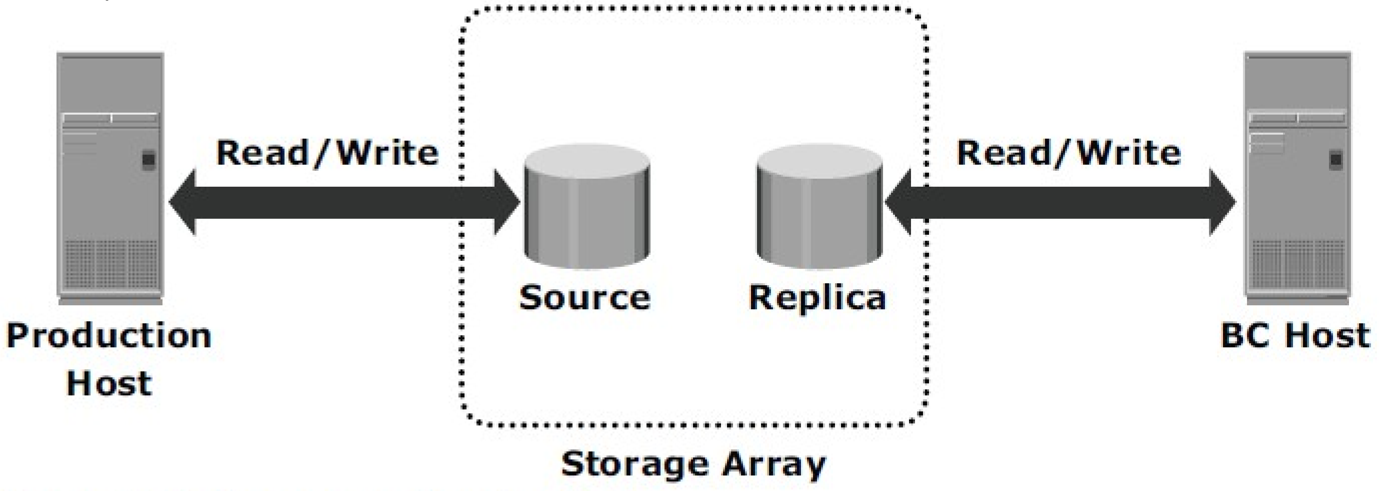


Figure 13-5: Storage array-based replication

Full-Volume Mirroring

In *full-volume mirroring*, the target is attached to the source and established as a mirror of the source (Figure 13-6 [a]). Existing data on the source is copied to the target. New updates to the source are also updated on the target. After all the data is copied and both the source and the target contain identical data, the target can be considered a mirror of the source. While the target is attached to the source and the synchronization is taking place, the target remains unavailable to any other host. However, the production host can access the source.

After synchronization is complete, the target can be detached from the source and is made available for BC operations. Figure 13-6 (b) shows full-volume mirroring when the target is detached from the source. Notice that both the source and the target can be accessed for read and write operations by the production hosts.

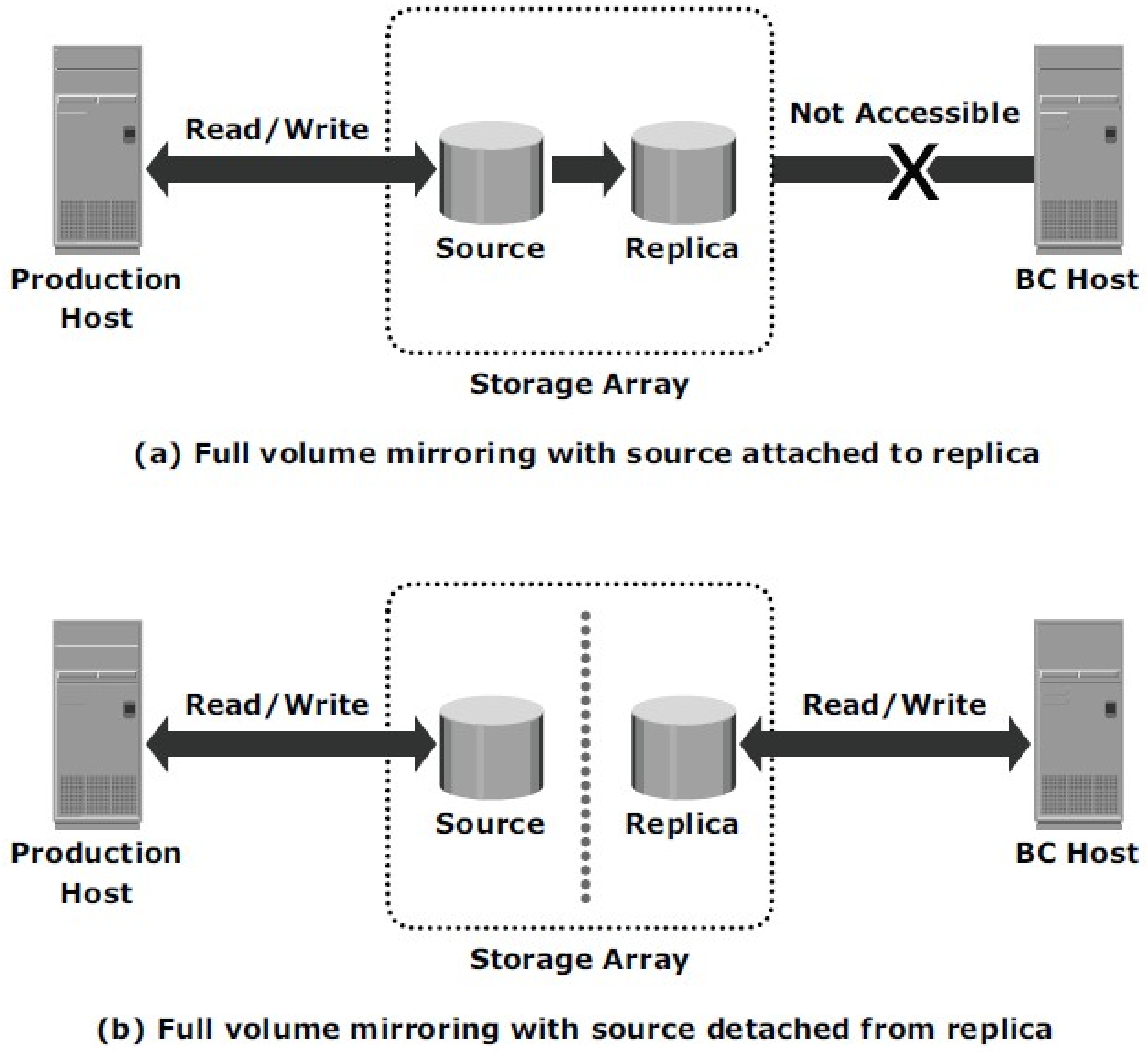


Figure 13-6: Full-volume mirroring

After the split from the source, the target becomes a PIT copy of the source. The point-in-time of a replica is determined by the time when the source is detached from the target. For example, if the time of detachment is 4:00 pm, the PIT for the target is 4:00 pm

After detachment, changes made to both source and replica can be tracked at some predefined granularity. This enables incremental resynchronization (source to target) or incremental restore (target to source). The granularity of the data change can range from 512 byte blocks to 64 KB blocks. Changes are typically tracked using bitmaps, with one bit assigned for each block. If any updates occur to a particular block, the whole block is marked as changed, regardless of the size of the actual update. However, for resynchronization (or restore), only the changed blocks have to be copied, eliminating the need for a full synchronization (or restore) operation. This method reduces the time required for these operations considerably.

In full-volume mirroring, the target is inaccessible for the duration of the synchronization process, until detachment from the source. For large databases, this can take a long time.

Pointer-Based, Full-Volume Replication

An alternative to full-volume mirroring is *pointer-based full-volume replication*. Like full-volume mirroring, this technology can provide full copies of the source data on the targets. Unlike full-volume mirroring, the target is made immediately available at the activation of the replication session. Hence, one need not wait for data synchronization to, and detachment of, the target in order to access it. The time of activation defines the PIT copy of source.

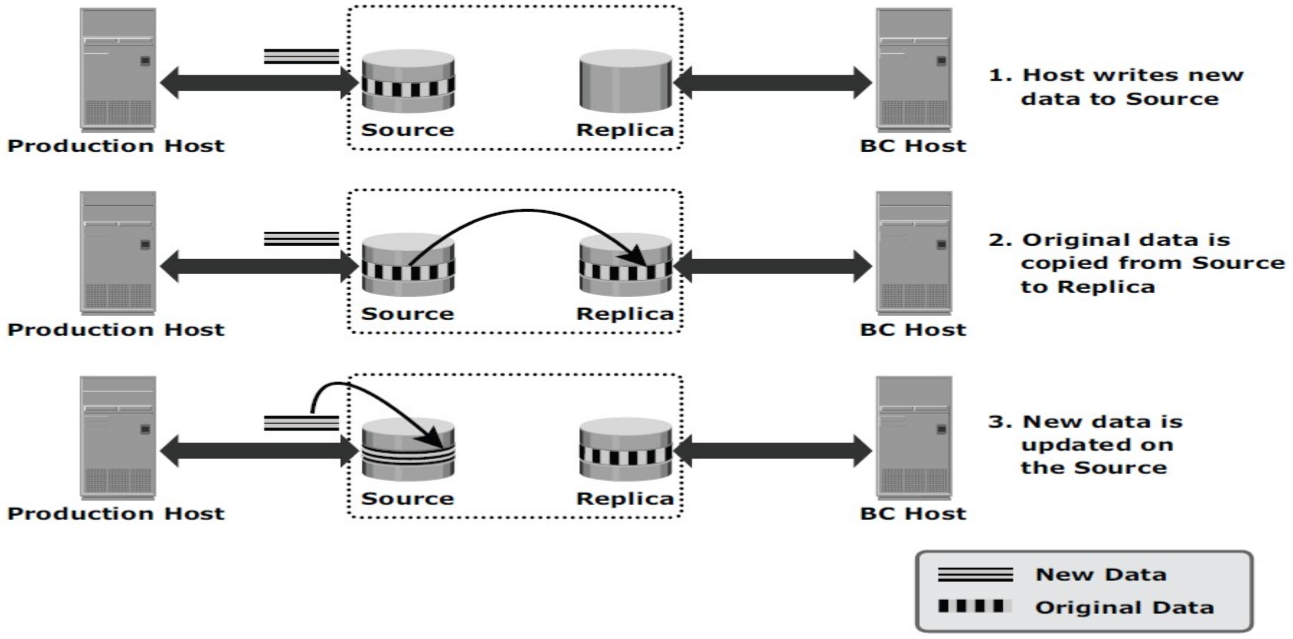
Pointer-based, full-volume replication can be activated in either Copy on First Access (CoFA) mode or Full Copy mode. In either case, at the time of activation, a protection bitmap is created for all data on the source devices. Pointers are initialized to map the (currently) empty data blocks on the target to the corresponding original data blocks on the source. The granularity can range from

512 byte blocks to 64 KB blocks or higher. Data is then copied from the source to the target, based on the mode of activation.

In CoFA, after the replication session is initiated, data is copied from the source to the target when the following occurs:

1. A write operation is issued to a specific address on the source for the first time (see Figure 13-7).
2. A read or write operation is issued to a specific address on the target for the first time (see Figure 13-8 and Figure 13- 9).

When a write is issued to the source for the first time after session activation, original data at that address is copied to the target. After this operation, the new data is updated on the source. This ensures that original data at the point-in-time of activation is preserved on the target. This is illustrated in Figure 13-7.



igure 13-7: Copy on first access (CoFA) — write to source

When a read is issued to the target for the first time after session activation, the original data is copied from the source to the target and is made available to the host. This is illustrated in Figure 13-8.

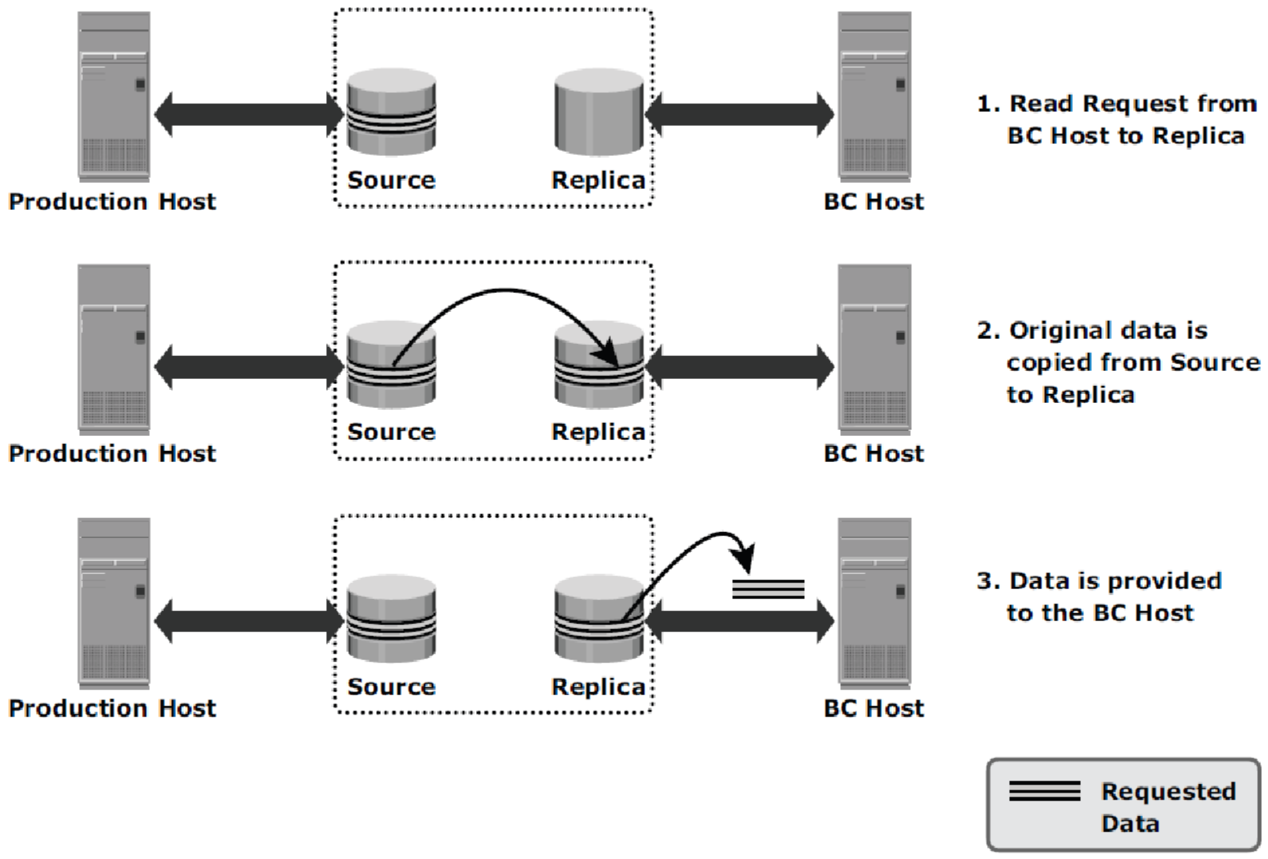

Production Host
Source
Replica
BC Host
1. Read Request from BC Host to Replica
2. Original data is copied from Source to Replica
3. Data is provided to the BC Host
Requested Data

Figure 13-8: Copy on first access (CoFA) — read from target

When a write is issued to the target for the first time after session activation, the original data is copied from the source to the target. After this, the new data is updated on the target. This is illustrated in Figure 13-9.

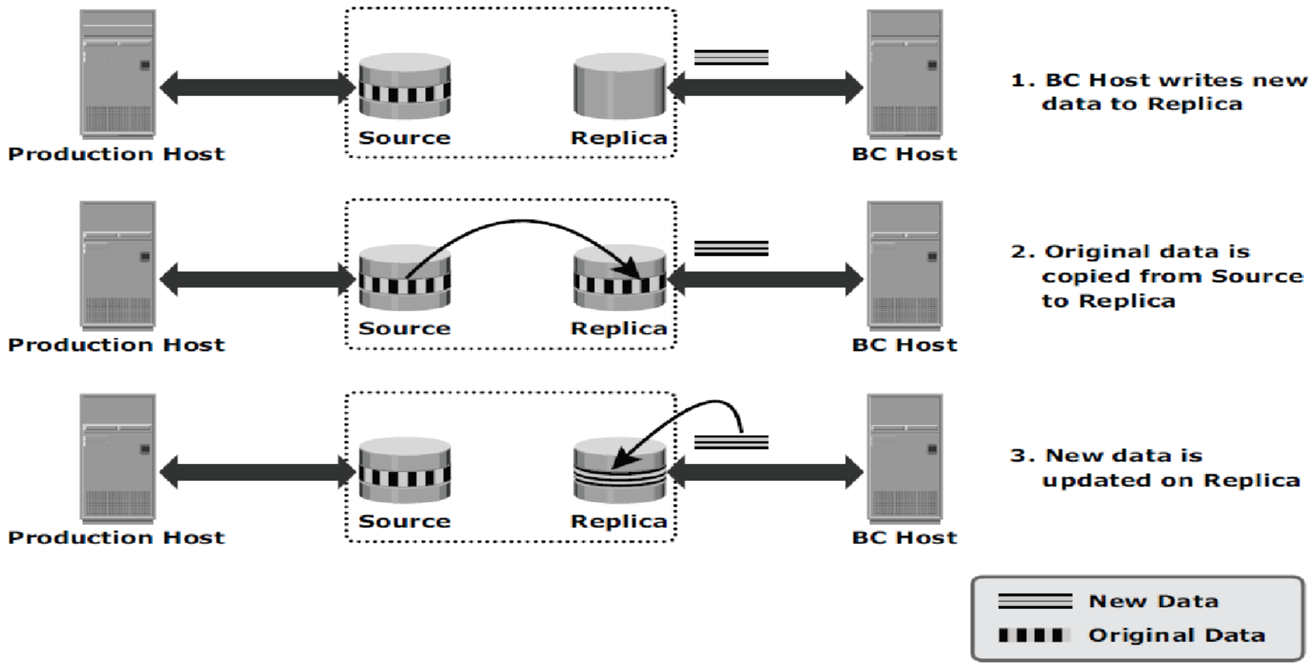

Production Host
Source
Replica
BC Host
1. BC Host writes new data to Replica
2. Original data is copied from Source to Replica
3. New data is updated on Replica
New Data
Original Data

Figure 13-9: Copy on first access (CoFA) — write to target

In all cases, the protection bit for that block is reset to indicate that the original data has been copied over to the target. The pointer to the source data can now be discarded. Subsequent writes to the same data block on the source, and reads or writes to the same data blocks on the target, do not trigger a copy operation (and hence are termed Copy on First Access).

Pointer-Based Virtual Replication

In *pointer-based virtual replication*, at the time of session activation, the target contains pointers to the location of data on the source. The target does not contain data, at any time. Hence, the target is known as a *virtual replica*. Similar to pointer-based full-volume replication, a protection bitmap is created for all data on the source device, and the target is immediately accessible. Granularity can range from 512 byte blocks to 64 KB blocks or greater. When a write is issued to the source for the first time after session activation, original data at that address is copied to a predefined area in the array. This area is generally termed the *save location*. The pointer in the target is updated to point to this data address in the save location. After this, the new write is updated on the source. This process is illustrated in Figure 13-10.

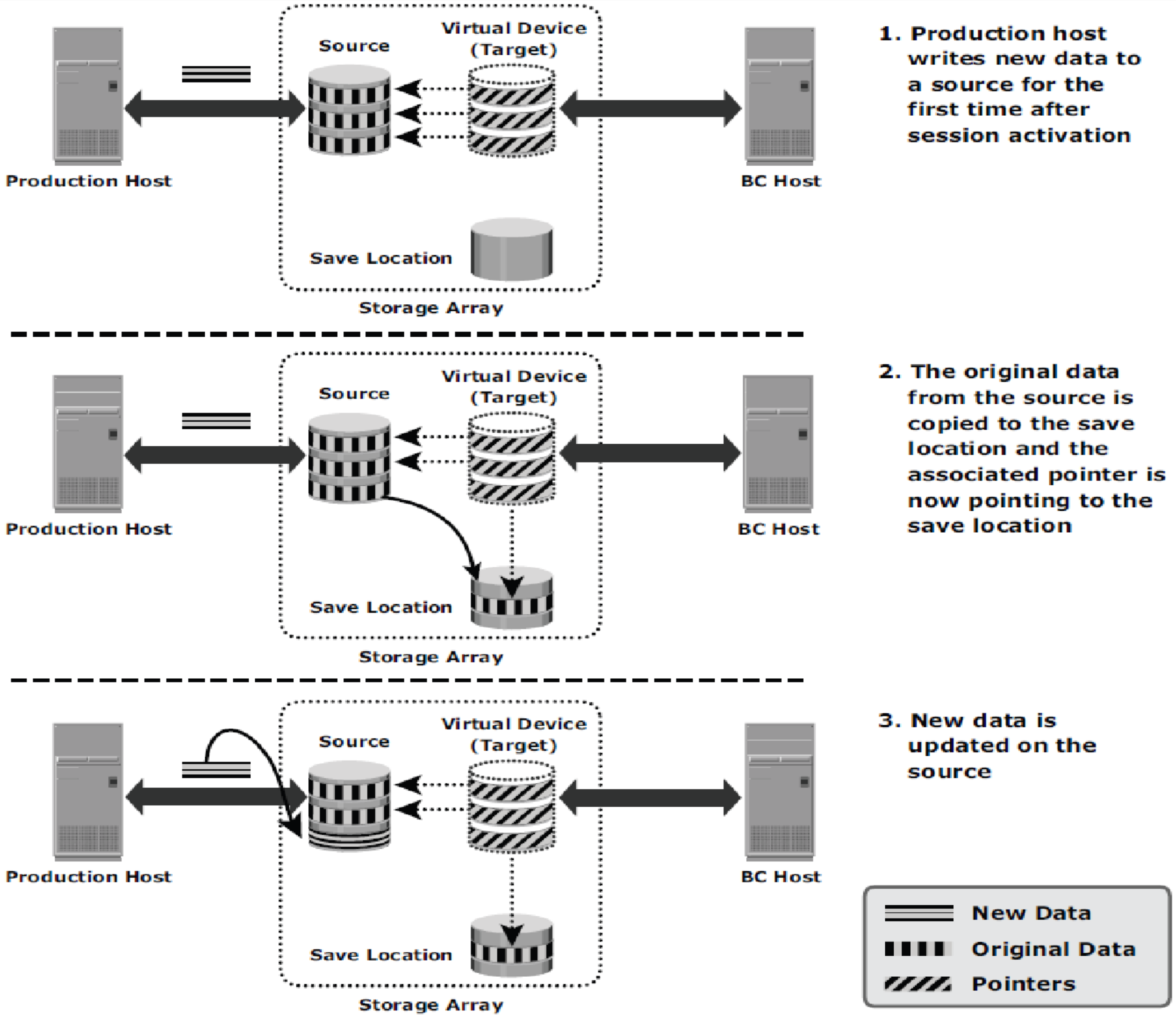

Source
Virtual Device (Target)
Production Host
BC Host
Save Location
Storage Array
1. Production host writes new data to a source for the first time after session activation
2. The original data from the source is copied to the save location and the associated pointer is now pointing to the save location
3. New data is updated on the source
New Data
Original Data
Pointers

Figure 13-10: Pointer-based virtual replication — write to source

When a write is issued to the target for the first time after session activation, original data is copied from the source to the save location and similarly the pointer is updated to data in save location. Another copy of the original data is created in the save location before the new write is updated on the save location. This process is illustrated in Figure 13-11.

When reads are issued to the target, unchanged data blocks since session activation are read from the source. Original data blocks that have changed are read from the save location.

Pointer-based virtual replication uses CoFW technology. Subsequent writes to the same data block on the source or the target do not trigger a copy operation.

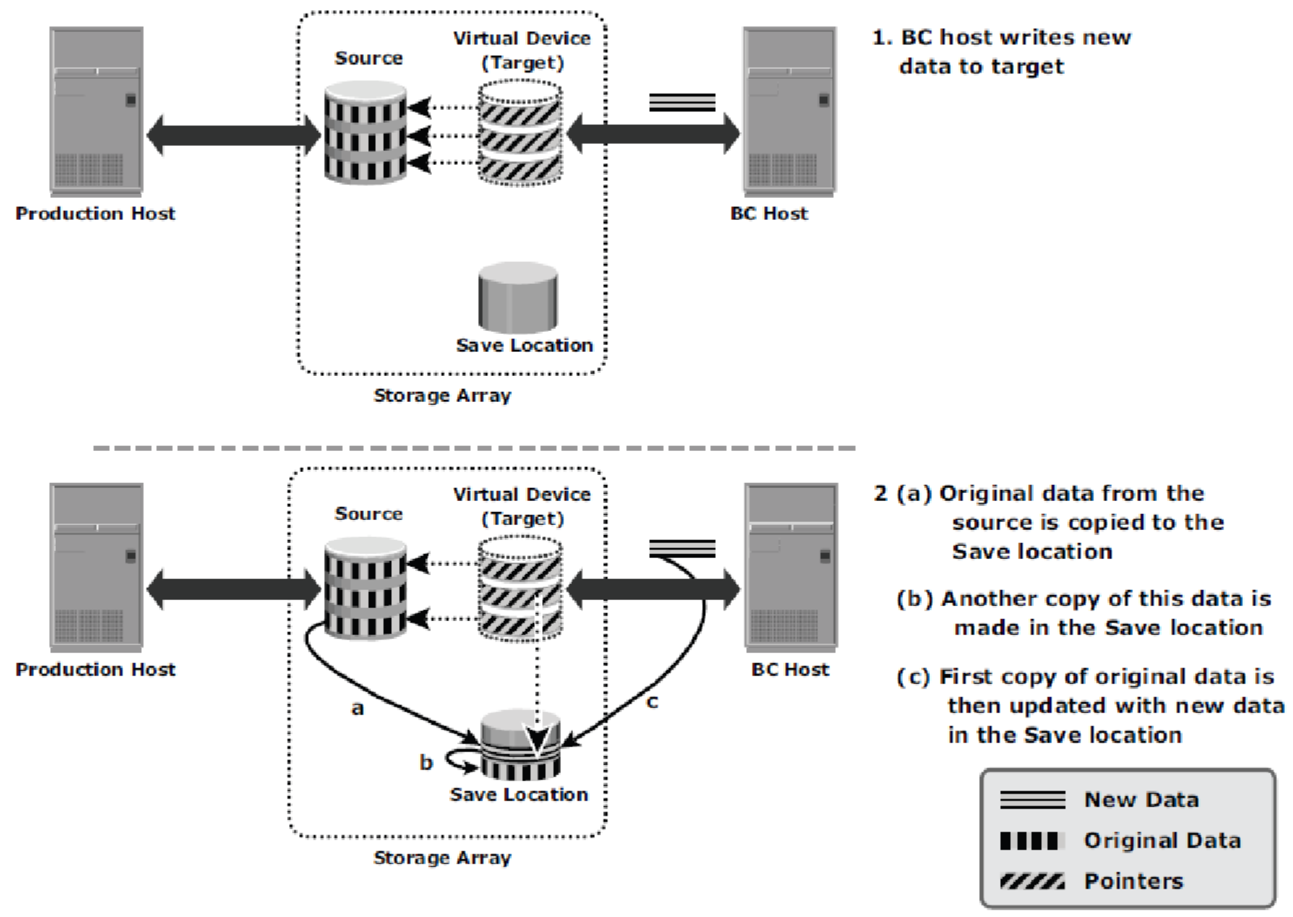

Source
Virtual Device (Target)
Production Host
BC Host
Save Location
Storage Array
1. BC host writes new data to target
Source
Virtual Device (Target)
Production Host
BC Host
a
b
c
Save Location
Storage Array
2 (a) Original data from the source is copied to the Save location
(b) Another copy of this data is made in the Save location
(c) First copy of original data is then updated with new data in the Save location
New Data
Original Data
Pointers

Figure 13-11: Pointer-based virtual replication – write to target

Data on the target is a combined view of unchanged data on the source and data on the save location. Unavailability of the source device invalidates the data on the target. As the target only contains pointers to data, the physical capacity required for the target is a fraction of the source device. The capacity required for the save location depends on the amount of expected data change.

Restore and Restart Considerations

Local replicas can be used to restore data to production devices. Alternatively, applications can be restarted using the consistent point-in-time copy of the data on the replicas.

A replica can be used to restore data to the production devices in the event of logical corruption of production devices —

i.e., the devices are available but the data on them is invalid. Examples of logical corruption include accidental deletion of information (tables or entries in a database), incorrect data entry, and incorrect updating to existing information. Restore operations from a replica are incremental and provide a very small RTO. In some instances, applications can be resumed on the production devices prior to completion of the data copy. Prior to the restore operation, access to production and replica devices should be stopped.

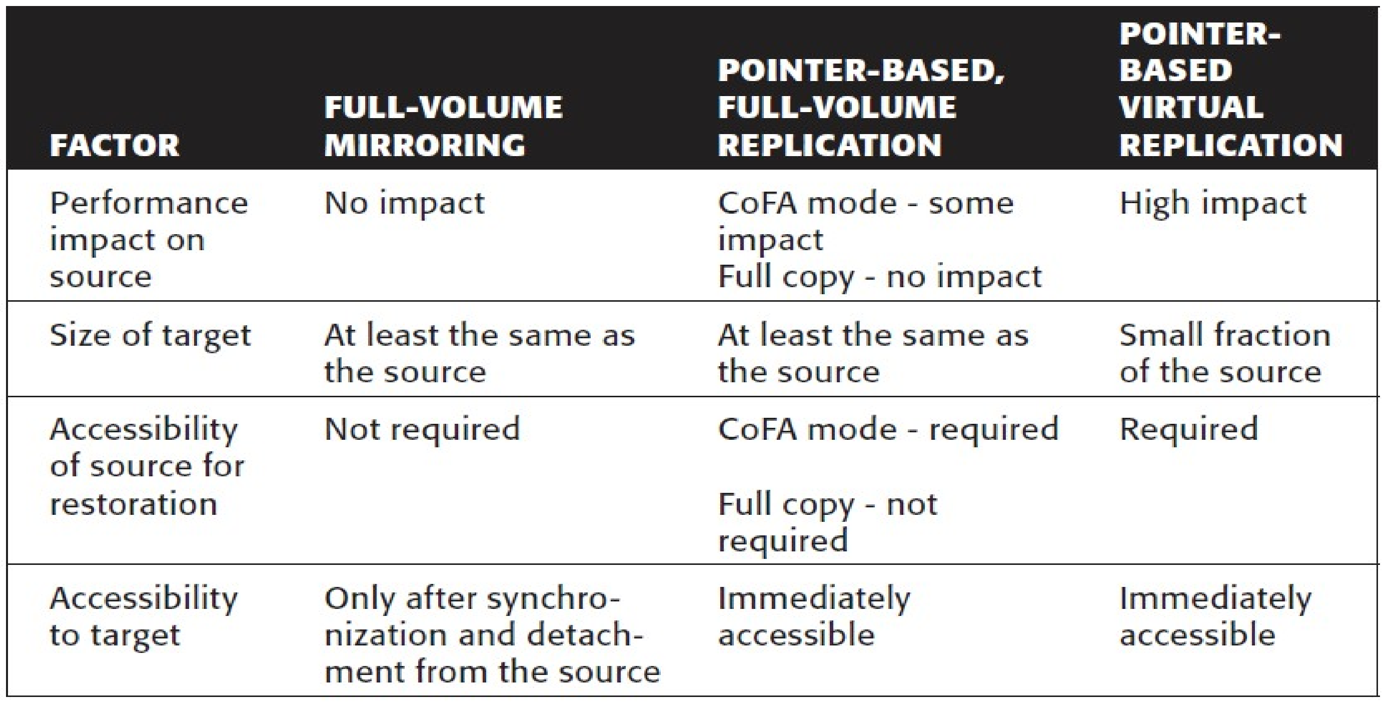
Production devices may also become unavailable due to physical failures, such as production server or physical drive failure. In this case, applications can be restarted using data on the latest replica. If the production server fails, once the issue has been resolved, the latest information from the replica devices can be restored back to the production devices. If the production device(s) fail, applications can continue to run on replica devices. A new PIT copy of the replica devices can be created or the latest information from the replica devices can be restored to a new set of production devices. Prior to restarting applications using the replica devices, access to the replica devices should be stopped. As a protection against further failures, a ―Gold Copy‖ (another copy of replica device) of the replica device should be created to preserve a copy of data in the event of failure or corruption of the replica devices.

Full-volume replicas (both full-volume mirrors and pointer-based in Full Copy mode) can be restored to the original source devices or to a new set of source devices. Restores to the original source devices can be incremental, but restores to a new set of devices are a full-volume copy operation.

In pointer-based virtual and pointer-based full-volume replication in CoFA mode, access to data on the replica is dependent on the health and accessibility of the original source volumes. If the original source volume is inaccessible for any reason, these replicas cannot be used for a restore or a restart.

Table 13-1 presents a comparative analysis of the various storage array–based replication technologies.

Table 13-1: Comparison of Local Replication Technologies



FACTOR	FULL-VOLUME MIRRORING	POINTER-BASED, FULL-VOLUME REPLICATION	POINTER-BASED VIRTUAL REPLICATION
Performance impact on source	No impact	CoFA mode - some impact Full copy - no impact	High impact
Size of target	At least the same as the source	At least the same as the source	Small fraction of the source
Accessibility of source for restoration	Not required	CoFA mode - required Full copy - not required	Required
Accessibility to target	Only after synchro-nization and detach-ment from the source	Immediately accessible	Immediately accessible

Tracking Changes to Source and Target

Updates occur on the source device after the creation of point-in-time local replicas. If the primary purpose of local replication is to have a viable pointin- time copy for data recovery or restore operations, then the target devices should not be modified. Changes can occur on the target device if it is used for non-BC operations. To enable incremental resynchronization or restore operations, changes to both the source and target devices after the point-in-time can be tracked. This is typically done using bitmaps, with one bit per block of data. The block sizes can range from 512 bytes to 64 KB or greater. For example, if the block size is 32 KB, then a 1 GB device would require 32,768 bits. The size of the bitmap would be 4 KB. If any or all of a 32 KB block is changed, the corresponding bit in the bitmap is flagged. If the block size is reduced for tracking purposes, then the bitmap size increases correspondingly.

The bits in the source and target bitmaps are all set to 0 (zero) when the replica is created. Any changes to the source or target are then flagged by setting the appropriate bits to 1 in the bitmap. When resynchronization or a restore is required, a *logical OR* operation between the source bitmap and the target bitmap is performed. The bitmap resulting from this operation (see Figure 13-12) references all blocks that have been modified in either the source or the target.

This enables an optimized resynchronization or a restore operation, as it eliminates the need to copy all the blocks between the source and the target. The direction of data movement depends on whether a resynchronization or a restore operation is performed.

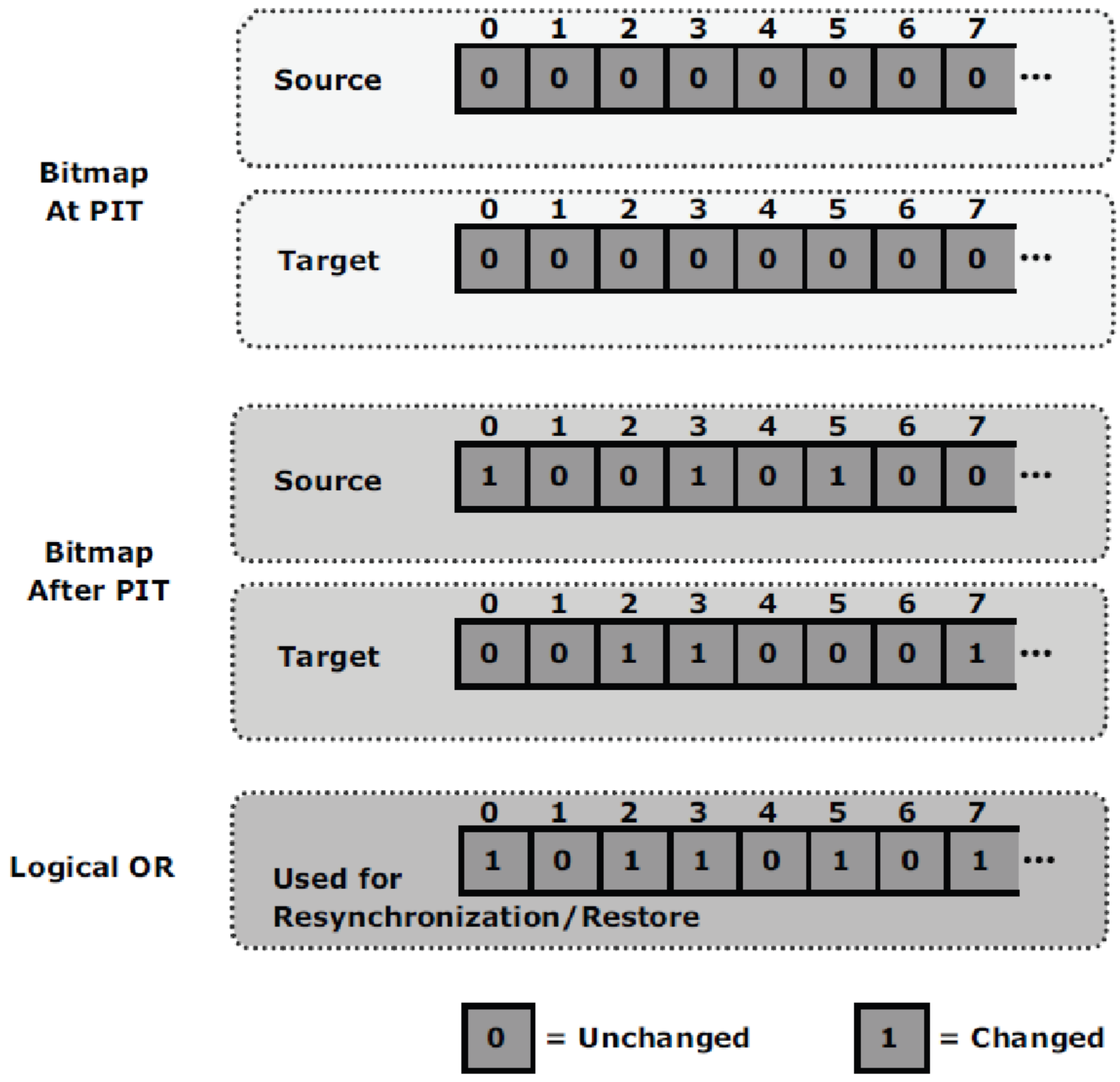


Figure 13-12: Tracking changes

If resynchronization is required, then changes to the target are overwritten with the corresponding blocks from the source. In this example, that would be blocks 3, 4, and 8 on the target (from the left).

Creating Multiple Replicas

Most storage array-based replication technologies enable source devices to maintain replication relationships with multiple targets. Changes made to the source and each of the targets can be tracked. This enables incremental resynchronization of the targets. Each PIT copy can be used for different BC activities and as a restore point. Figure 13-13 shows an example in which a copy is created every six hours from the same source.

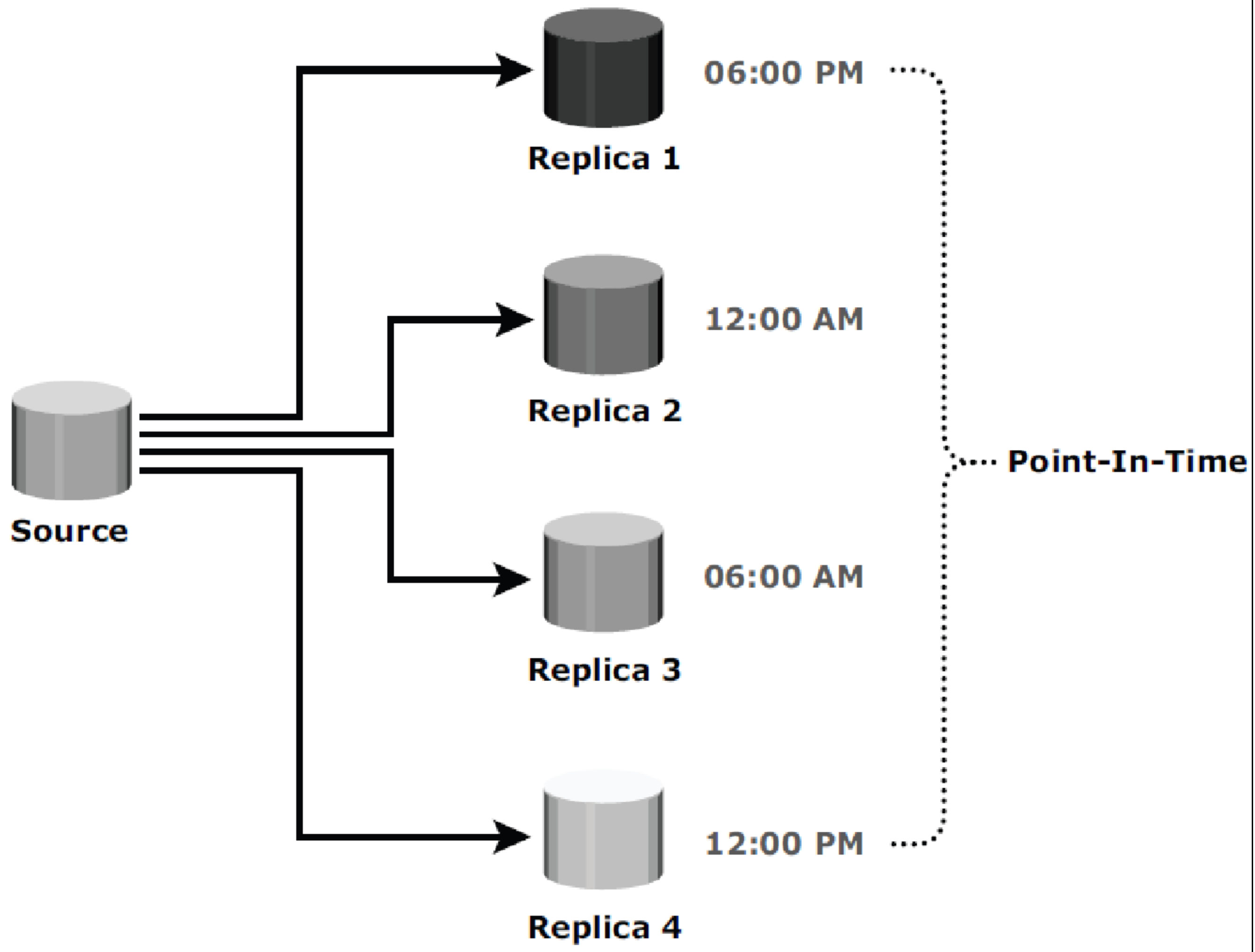


Figure 13-13: Multiple replicas created at different points in time

If the source is corrupted, the data can be restored from the latest PIT copy. The maximum RPO in the example shown in Figure 13-15 is six hours. More frequent replicas will reduce the RPO and the RTO.

Management Interface

The replication management software residing on the storage array provides an interface for smooth and error-free replication. This management software interface provides options for synchronization, resynchronization, splitting, starting and stopping a replication session, and monitoring replication performance. In general, two types of interface are provided:

1. **CLI:** An administrator can directly enter commands against a prompt in a CLI. A user can also perform some command operations based on assigned privileges. CLI scripts are developed to perform specific BC operations in a database or application environment.

2. GUI: This type of interface includes the toolbar and menu bar options. A user can select an operation name and monitor performance in real time. Most of the tools have a browser-based GUI to transfer commands and are often integrated with other storage management suite products.

***Definition - What does* Data Center Monitoring**

Data center monitoring is the process of monitoring, managing and operating a data center to be in compliance with the operating and organizational requirements.

It is the process of using manual and automated tools and techniques to ensure the best operating health of a data center. It ensures that the key functions and services of a data center are delivered without any interruptions or abnormalities.

Data center monitoring is also known as data center management.

Data Center Monitoring

Data center monitoring is a broad process that focuses on monitoring the entire data center infrastructure. Typically, data center monitoring is performed through automated tools that provide statistical insights into data center performance/status. This data is used by data center administrators in identifying irregularities and in fixing them.

Data center monitoring usually incorporates:

- Monitoring data center servers and computers for performance, security uptime and more
- Monitoring and managing network operations and resolving network problems as they arise
- Providing end-to-end visibility across all data center components including computers, storage, network and software

In addition to monitoring IT-specific components, data center monitoring also focuses on monitoring supported elements such as:

- Power availability and consumption
- Data center temperature, and the heating and ventilation systems
- Physical data center security to restrict unauthorized personnel from entering the premises

Monitoring the Storage Infrastructure

Monitoring helps to analyze the status and utilization of various storage infrastructure components. This analysis facilitates optimal use of resources and proactive management. Monitoring supports capacity planning, trend analysis, and root cause/impact analysis. As the business grows, monitoring helps to optimize the storage infrastructure resources. The monitoring process also includes the storage infrastructure's environmental controls and the operating environments for key components such as storage arrays and servers.

Parameters Monitored

Storage infrastructure components should be monitored for accessibility, capacity, performance, and security. Accessibility refers to the availability of a component to perform a desired operation. A component is said to be accessible when it is functioning without any fault at any given point in time. Monitoring hardware components (e.g., a SAN interconnect device, a port, an HBA, or a disk drive) or software components (e.g., a database instance) for accessibility involves checking their availability status by listening to pre-determined alerts from devices. For example, a port may go down resulting in a chain of availability alerts. A storage infrastructure uses redundant components to avoid a single point of failure. Failure of a component may cause an outage that affects application availability, or it may cause serious performance degradation even though accessibility is not compromised. For example, an HBA failure can restrict the server to a few paths for access to data devices in a multipath environment, potentially resulting in degraded performance. In a single-path environment, an HBA failure results in complete accessibility loss between the server and the storage. Continuously monitoring for expected accessibility of each component and reporting any deviations helps the administrator to identify failing components and plan corrective action to maintain SLA requirements. Capacity refers to the amount of storage infrastructure resources available.

Examples of capacity monitoring include examining the free space available on a file system or a RAID group, the mailbox quota allocated to users, or the numbers of ports available on a switch. Inadequate capacity may lead to degraded performance or affect accessibility or even application/service availability. Capacity monitoring ensures uninterrupted data availability and scalability by averting outages before they occur. For example, if a report indicates that 90 percent of the ports are utilized in a particular SAN fabric, a new switch should be added if more arrays and servers need to be installed on the same fabric. Capacity monitoring is preventive and predictive, usually leveraged with advanced analytical tools for trend analysis. These trends help to understand emerging challenges, and can provide an estimation of time needed to meet them.

Components Monitored

Hosts, networks, and storage are components within the storage environment that should be monitored for accessibility, capacity, performance, and security.

Hosts

Mission-critical application hosts should be monitored continuously. The accessibility of a host depends on the status of the hardware components and software processes running on it. For example, an application crash due to host hardware failure can cause instant unavailability of the data to the user.

Servers are used in a cluster to ensure high availability. In a server virtualization environment, multiple virtual machines share a pool of resources. These resources are dynamically reallocated, which ensures application accessibility and ease of management.

File system utilization of hosts also needs to be monitored. Monitoring helps in estimating the file system's growth rate and helps in predicting when it will reach 100 percent. Accordingly, the administrator can extend (manually or automatically) the file system's space proactively to prevent a failure resulting from a file system being full. New provisioning technologies even enable the allocation of storage on demand as the need arises. Alternatively, system administrators can enforce a quota for users, provisioning a fixed amount of space for their files. For example, a quota could be specified at a user level, restricting the maximum space to 10 GB per user, or at a file level that restricts a file to a maximum of 100 MB.

Storage Network

The storage network needs to be monitored to ensure proper communication between the server and the storage array. Uninterrupted access to data over the storage network depends on the accessibility of the physical and logical components in the storage network. The physical components of a storage network include elements such as switches, ports, cables, GBICs, and power supplies. The logical components include constructs, such as zones and fabrics. Any failure in the physical or logical components may cause data unavailability.

Storage

The accessibility of the storage array should be monitored for its hardware components and various processes. Storage arrays configured with redundant components do not affect accessibility in the event of an individual component failure, but failure of any process can disrupt or compromise business continuity operations. For example, the failure of a replication task affects disaster recovery capabilities. Some storage arrays also provide the capability to send a message to the vendor's support center in the event of hardware or process failures, referred to as a call home.

Monitoring

A storage infrastructure requires implementation of an end-to-end solution to actively monitor all the parameters of its critical components. Early detection and instant alerts ensure the protection of critical assets. In addition, the monitoring tool should be able to analyze the impact of a failure and deduce the root cause of symptoms.

Accessibility Monitoring

Failure of any component may affect the accessibility of one or more components due to their interconnections and dependencies, or it may lead to overall performance degradation. Consider an implementation in a storage infrastructure with three servers: H1, H2, and H3. All the servers are configured with two HBAs, each connected to the storage array through two switches, SW1 and SW2, as shown in Figure 16-1. The three servers share two storage ports on the storage array. Path failover software has been installed on all three servers.

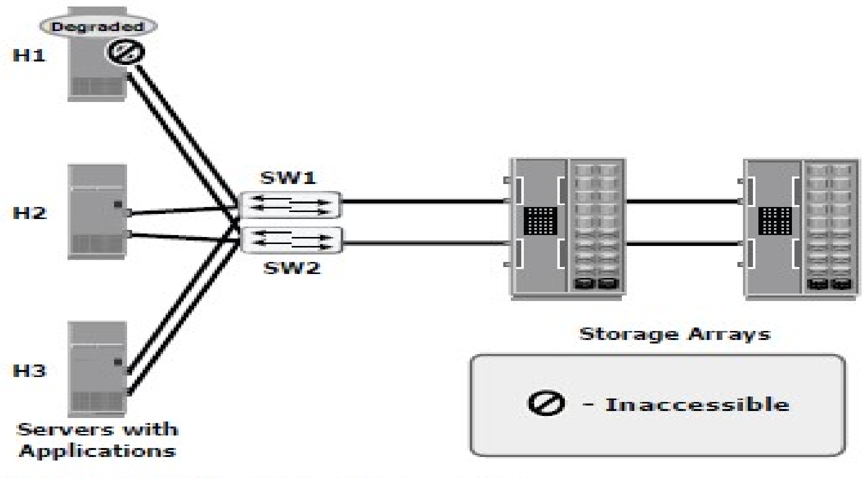


Figure 16-2: HBA failure in a storage infrastructure

Capacity Monitoring

In the scenario shown in Figure 16-4, each of the servers is allocated storage on the storage array. When a new server is deployed in this configuration, the applications on the new servers have to be given access to the storage devices from the array through switches SW1 and SW2. Monitoring the available capacity on the array helps to proactively decide whether the array can provide the required storage to the new server. Other considerations include the availability of ports on SW1 and SW2 to connect to the new server as well as the availability of storage ports to connect to the switches. Proactive monitoring also helps to identify the availability of an alternate fabric or an array to connect to the server.

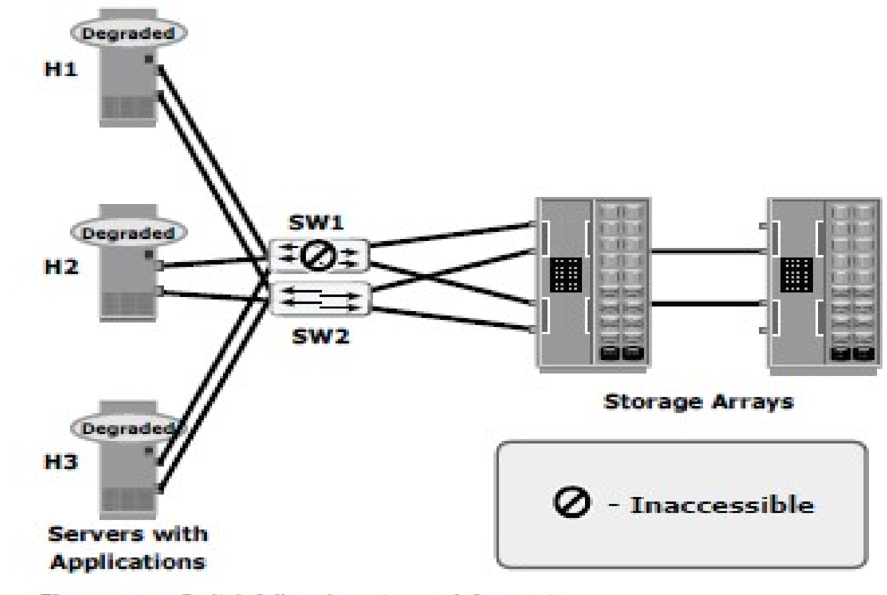


Figure 16-3: Switch failure in a storage infrastructure

Performance Monitoring

The example shown in Figure 16-6 illustrates the importance of monitoring performance on storage arrays. In this example, servers H1, H2, and H3 (with two HBAs each) are connected to the storage array through switches SW1 and SW2. The three servers share the same storage ports on the storage array. A new server, H4 running an application with high work load, has to be deployed to share the same storage ports as H1, H2, and H3.

Monitoring array port utilization ensures that the new server does not adversely affect the performance of the other servers. In this example, utilization for the shared ports is shown by the solid and dotted lines in the line graph for the storage ports. Notice that the port represented by a solid line is close to 100 percent utilization. If the actual utilization of both ports prior to deploying the new server is closer to the dotted line, there is room to add the new server. Otherwise, deploying the new server will affect the performance of all servers.

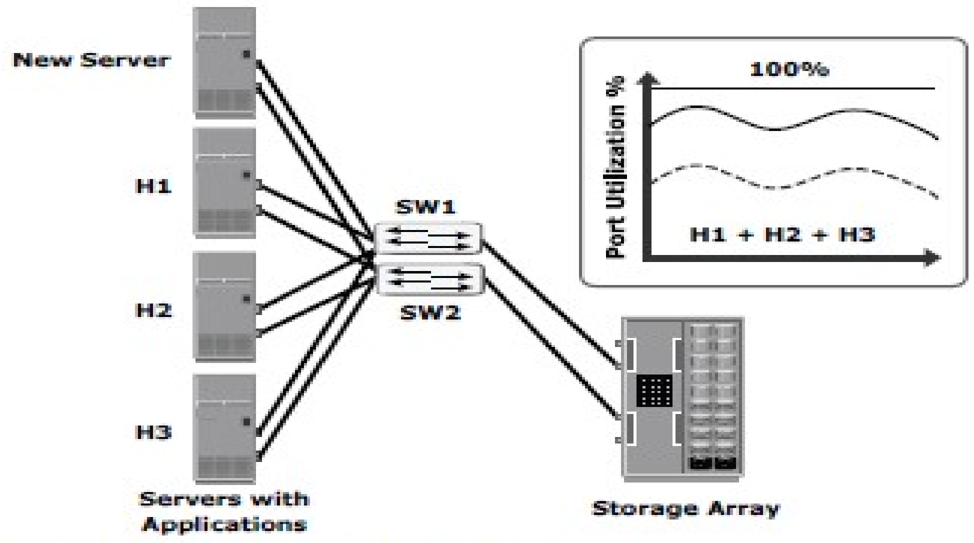


Figure 16-6: Monitoring array port utilization

Security Monitoring

The example shown in Figure 16-8 illustrates the importance of monitoring security breaches in a storage array.

In this example, the storage array is shared between two workgroups, WG1 and WG2. The data of WG1 should not be accessible by WG2. Likewise, WG2 should not be accessible by WG1. A user from WG1 may try to make a local replica of the data that belongs to WG2. Usually, vailable mechanisms prevent such an action. However, if this action is not monitored or recorded, it is difficult to track such a violation of security protocols. Conversely, if this action is monitored, a warning message can be sent to prompt a corrective action or at least enable discovery as part of regular auditing operations.

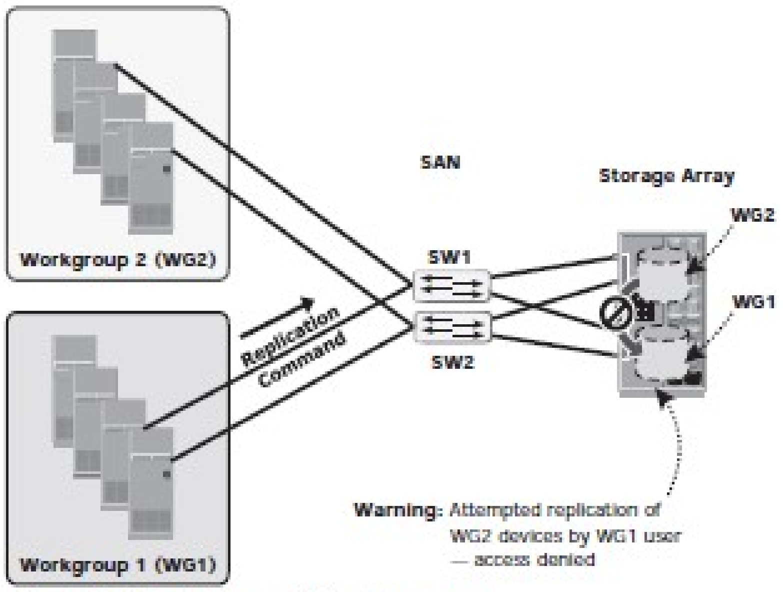


Figure 16-8: Monitoring security in a storage array

Module V: Security

A: Realistic solution and Storage

Storage Security Domains

Storage devices that are not connected to a storage network are less vulnerable because they are not exposed to security threats via networks. However, with increasing use of networking in storage environments, storage devices are becoming highly exposed to security threats from a variety of sources. Specific controls must be implemented to secure a storage networking environment.

This requires a closer look at storage networking security and a clear understanding of the access paths leading to storage resources. If a particular path is unauthorized and needs to be prohibited by technical controls, one must ensure that these controls are not compromised. If each component within the storage network is considered a potential access point, one must analyze the attack surface that each of these access points provides and identify the associated vulnerability.

In order to identify the threats that apply to a storage network, access paths to data storage can be categorized into three security domains: application access, management access, and BURA (backup, recovery, and archive). Figure 15-1 depicts the three security domains of a storage system environment. The first security domain involves application access to the stored data through the storage network. The second security domain includes management access to storage and interconnect devices and to the data residing on those devices. This domain is primarily accessed by storage administrators who configure and manage the environment. The third domain consists of BURA access. Along with the access points in the other two domains, backup media also needs to be secured.

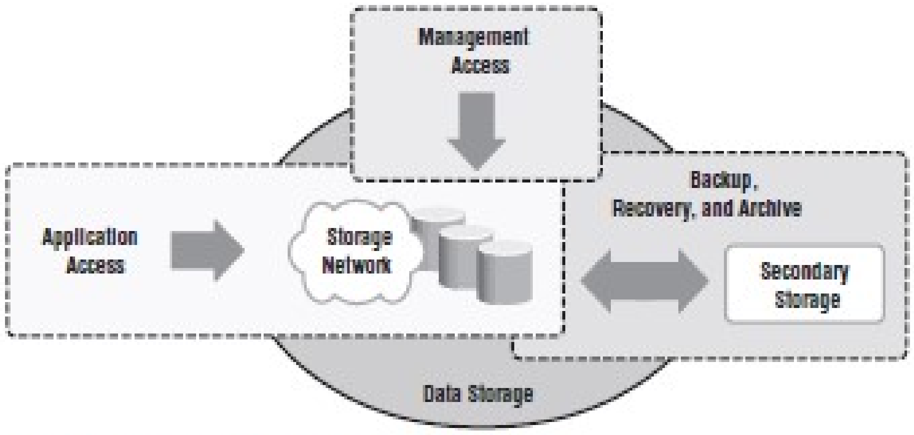


Figure 15-1: Three security domains of data storage

Securing the Application Access Domain

The application access domain may include only those applications that access the data through the file system or a database interface. Figure 15-2 shows application access in a storage networking environment. Host A can access all V1 volumes; host B can access all V2 volumes. These volumes are classified according to access level, such as confidential, restricted, and public.

Some of the possible threat in this scenario could be host A spoofing the identity or elevating the privileges of host B to gain access to host B's resources. Another threat could be an unauthorized host gain access to the network; the attacker on this host may try to spoof the identity of another host and tamper with data, snoop the network, or execute a DoS attack. Also any form of media theft could also compromise security. These threats can pose several serious challenges to
the network security, hence they need to be addressed.

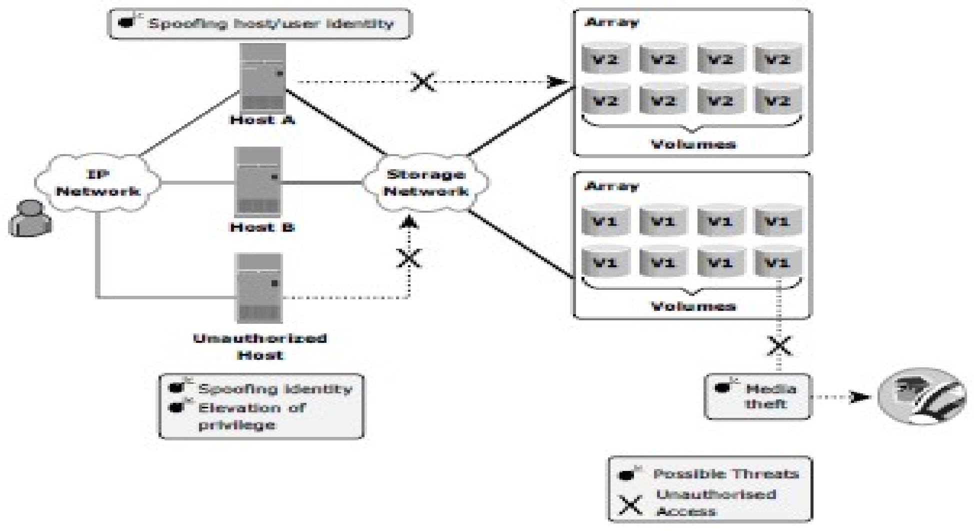


Figure 15-2: Security threats in application access domain

Securing the Management Access Domain

Management access, whether monitoring, provisioning, or managing storage resources, is associated with every device within the storage network. Most management software supports some form of CLI, system management console, or a web-based interface. It is very important to implement appropriate controls for securing storage management applications because the damage that can be caused to the storage system by using these applications can be far more extensive than that caused by vulnerability in a server.

Figure 15-3 depicts a storage networking environment in which production hosts are connected to a SAN fabric and are accessing storage Array A, which is connected to storage Array B for replication purposes. Further, this configuration has a storage management platform on Host B and a monitoring console on Host A. All these hosts are interconnected through an IP network. Some of the possible threats in this system are, unauthorized host may spoof the user or host identity to manage the storage arrays or network. For example, Host A may gain management access to array B. Remote console support for the management software also increases the attack surface. Using remote console support, several other systems in the network may also be used to execute an attack.

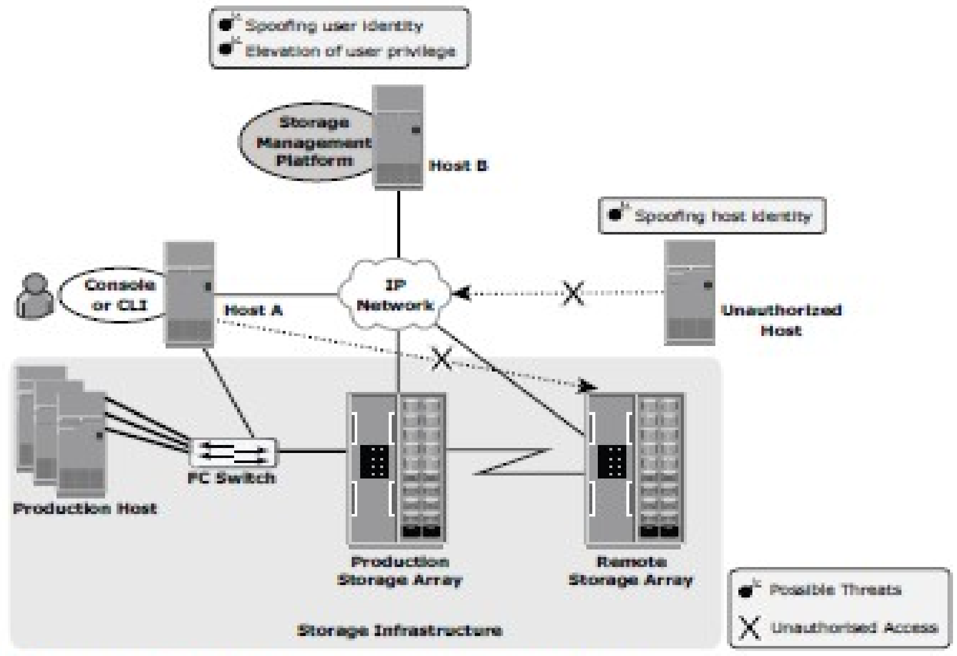


Figure 15-3: Security threats in management access domain

Security Implementations in Storage Networking

The following discussion details some of the basic security implementations in SAN, NAS, and IP-SAN environments.

SAN

Traditional FC SANs enjoy a natural security advantage over IP-based networks. An FC SAN is configured as an isolated private environment with fewer nodes than an IP network. Consequently, FC SANs impose fewer security threats. However, this scenario has changed with storage consolidation, driving rapid growth and necessitating designs for large, complex SANs that span multiple sites across the enterprise. Today, no single comprehensive security solution is available for SANs. Many SAN security mechanisms have evolved from their counterpart in IP networking, thereby bringing in mature security solutions. FC-SP (Fibre Channel Security Protocol) standards (T11 standards), published in 2006, align security mechanisms and algorithms between IP and FC interconnects. These standards describe protocols used to implement security measures in an FC fabric, among fabric elements and N_Ports within the fabric. They also include guidelines for authenticating FC entities, setting up session keys, negotiating the parameters required to ensure frame-by-frame integrity and confidentiality, and establishing and distributing policies across an FC fabric.

The current version of the FC-SP standard is referred to as FC-SP-1.

SAN Security Architecture

Storage networking environments are a potential target for unauthorized access, theft, and misuse because of the vastness and complexity of these environments. Therefore, security strategies are based on the defense in depth concept, which recommends multiple integrated layers of security. This ensures that the failure of one security control will not compromise the assets under protection. Figure 15-5 illustrates various levels (zones) of a storage networking environment that must be secured and the security measures that can be deployed.

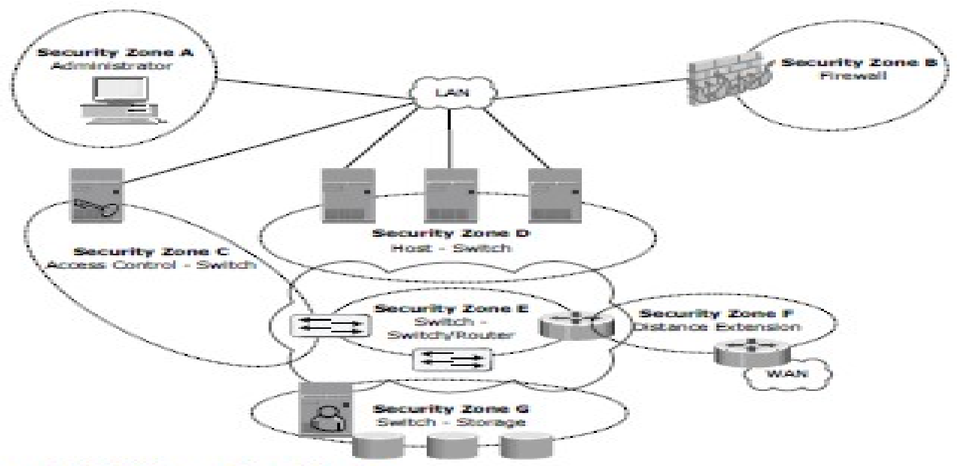


Figure 15-5: SAN security architecture

Types of Storage Virtualization

Virtual storage is about providing logical storage to hosts and applications independent of physical resources. Virtualization can be implemented in both SAN and NAS storage environments. In a SAN, virtualization is applied at the block level, whereas in NAS, it is applied at the file level.

Block-Level Storage Virtualization

Block-level storage virtualization provides a translation layer in the SAN, between the hosts and the storage arrays, as shown in Figure 10-6. Instead of being directed to the LUNs on the individual storage arrays, the hosts are directed to the virtualized LUNs on the virtualization device. The virtualization device translates between the virtual LUNs and the physical LUNs on the individual arrays. This facilitates the use of arrays from different vendors simultaneously, without any interoperability issues. For a host, all the arrays appear like a single target device and LUNs can be distributed or even split across multiple arrays. Block-level storage virtualization extends storage volumes online, resolves application growth requirements, consolidates heterogeneous storage arrays, and enables transparent volume access. It also provides the advantage of non disruptive data migration.

In traditional SAN environments, LUN migration from one array to another was an offline event because the hosts needed to be updated to reflect the new array configuration. In other instances, host CPU cycles were required to migrate data from one array to the other, especially in a multi vendor environment. With a block-level virtualization solution in place, the virtualization
engine handles the back-end migration of data, which enables LUNs to remain online and accessible while data is being migrated. No physical changes are required because the host still points to the same virtual targets on the virtualization device. However, the mappings on the virtualization device should be changed. These changes can be executed dynamically and
are transparent to the end user.

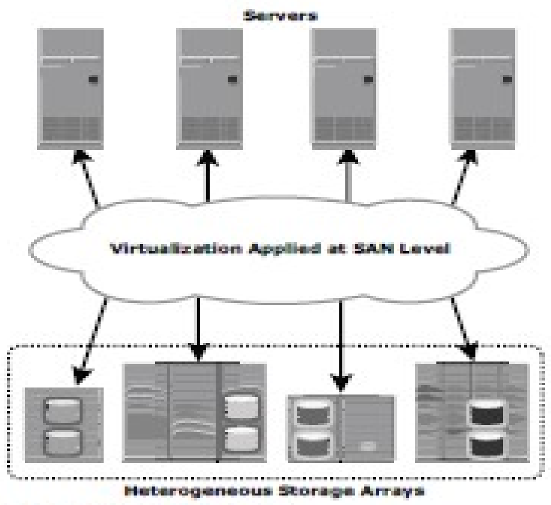

Servers
Virtualization Applied at SAN Level
Heterogeneous Storage Arrays

Figure 10-6: Block-level storage virtualization

File-Level Virtualization

File-level virtualization addresses the NAS challenges by eliminating the dependencies between the data accessed at the file level and the location where the files are physically stored. This provides opportunities to optimize storage utilization and server consolidation and to perform non disruptive file migrations. Figure 10-7 illustrates a NAS environment before and after the implementation of file-level virtualization.

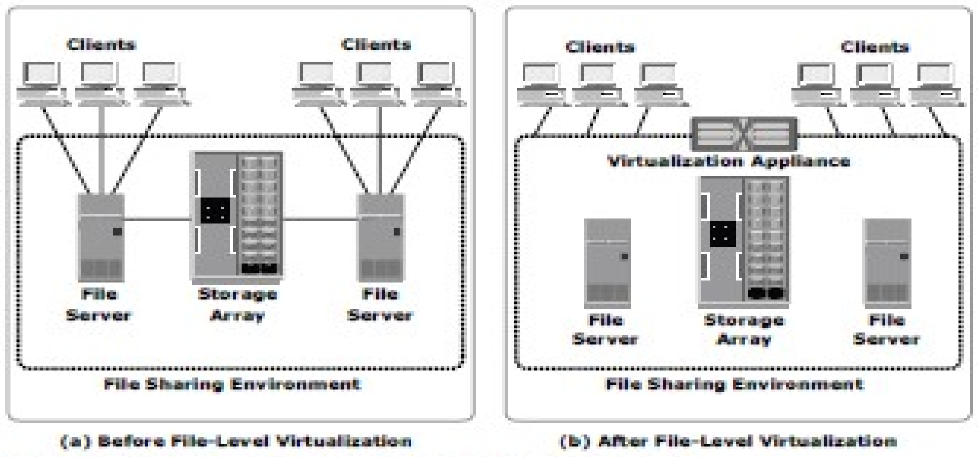


Figure 10-7: NAS device before and after file-level virtualization

Before virtualization, each NAS device or file server is physically and logically independent. Each host knows exactly where its file-level resources are located. Underutilized storage resources and capacity problems result because files are bound to a specific file server. It is necessary to move the files from one server to another because of performance reasons or when the file server fills up. Moving files across the environment is not easy and requires downtime for the file servers. Moreover, hosts and applications need to be reconfigured with the new path, making it difficult for storage administrators to improve storage efficiency while maintaining the required service level.

File-level virtualization simplifies file mobility. It provides user or application independence from the location where the files are stored. File-level virtualization creates a logical pool of storage, enabling users to use a logical path, rather than a physical path, to access files. File-level virtualization facilitates the movement of file systems across the online file servers. This means that while the files are being moved, clients can access their files non disruptively. Clients can also read their files from the old location and write them back to the new location without realizing that the physical location has changed. Multiple clients connected to multiple servers can perform online movement of their files to optimize utilization of their resources. A global namespace can be used to map the logical path of a file to the physical path names. Detailed implementation of functionality and operation of file-level storage virtualization is discussed in the next section.