MODULE-2

Data Protection: RAID

- In 1987, Patterson, Gibson, and Katz at the University of California, Berkeley, published a paper titled "A Case for **Redundant Arrays of Inexpensive Disks (RAID)**."
- RAID is the use of small-capacity, inexpensive disk drives as an alternative to large-capacity drives common on mainframe computers.
- Later RAID has been redefined to refer to ***independent*** disks to reflect advances in the storagetechnology.

RAID Implementation Methods

- The two methods of RAID implementation are:
    1. Hardware RAID.
    2. Software RAID.

Hardware RAID

- In hardware RAID implementations, a specialized hardware controller is implemented either on the *host* or on the *array*.
- **Controller card RAID** is a *host-based hardware RAID* implementation in which a specialized RAID controller is installed in the host, and disk drives are connected to it.
- Manufacturers also integrate RAID controllers on motherboards.
- A host-based RAID controller is not an efficient solution in a data center environment with a large number of hosts.
- The external RAID controller is an *array-based hardware RAID*.
- It acts as an interface between the host and disks.
- It presents storage volumes to the host, and the host manages these volumes as physical drives.
- The key functions of the RAID controllers are as follows:
    - ✓ Management and control of disk aggregations

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

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.
- Advantages when compared to Hardware RAID:
  - ✓ Cost
  - ✓ Simplicity benefits
- Limitations:
  - ✓ **Performance**: Software RAID affects overall system performance. This is due to additional CPU cycles required to perform RAID calculations.
  - ✓ **Supported features**: Software RAID does not support all RAID levels.
  - ✓ **Operating system compatibility**: Software RAID is tied to the host operating system; hence, upgrades to software RAID or to the operating system should be validated for compatibility. This leads to inflexibility in the data-processing environment.

RAID Techniques

- There are three RAID techniques
  1. striping
  2. mirroring
  3. parity

Striping

- **Striping** is a technique to spread data across multiple drives (more than one) to use the drives in parallel.
- All the read-write heads work simultaneously, allowing more data to be processed in a shorter time and increasing performance, compared to reading and writing from a single disk.
- Within each disk in a RAID set, a **predefined number of contiguously addressable** disk blocks are defined as a **strip**.

- The set of aligned strips that spans across all the disks within the RAID set is called a **stripe**.
- The below figure shows physical and logical representations of a striped RAID set.
- **Strip size** (also called **stripe depth**) describes the number of blocks in a strip and is the maximum amount of data that can be written to or read from a single disk in the set.
- All strips in a stripe have the same number of blocks.
  - ✓ Having a smaller strip size means that data is broken into smaller pieces while spread across the disks.
- **Stripe size** is a multiple of strip size by the number of **data** disks in the RAID set.
  - ✓ Eg: In a 5 disk striped RAID set with a strip size of 64 KB, the stripe size is 320KB (64KB x 5).
- **Stripe width** refers to the number of *data* strips in a stripe.
- Striped RAID does not provide any data protection unless parity or mirroring is used.

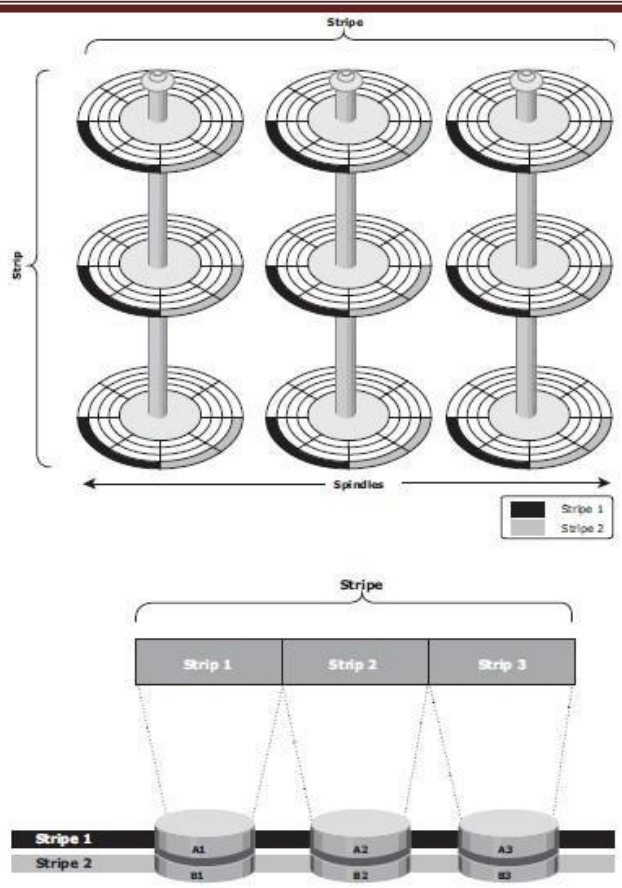

Fig : Striped RAID set

Mirroring

- **Mirroring** is a technique whereby the same data is stored on two different disk drives, yielding two copies of the data.
- If one disk drive failure occurs, the data is intact on the surviving disk drive (see Fig below) and the controller continues to service the host's data requests from the surviving disk of a mirrored pair.
- When the failed disk is replaced with a new disk, the controller copies the data from the surviving disk of the mirrored pair.
- This activity is transparent to the host.

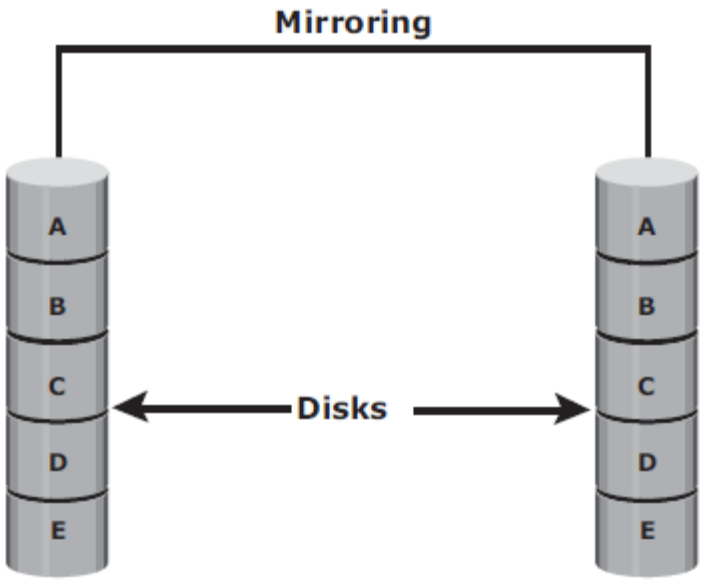

Fig: Mirrored disks in an array

- Advantages:
  - ✓ complete data redundancy,
  - ✓ mirroring enables fast recovery from disk failure.
  - ✓ data protection
- Mirroring is not a substitute for data backup. Mirroring constantly captures changes in the data, whereas a backup captures point-in-time images of the data.
- Disadvantages:
  - ✓ Mirroring involves duplication of data — the amount of storage capacity needed is

twice the amount of data being stored.

- ✓ Expensive

Parity

- **Parity** is a method to protect striped data from disk drive failure without the cost of mirroring.
- *An additional disk drive is added to hold parity*, a mathematical construct that allows re-creation of the missing data.
- Parity is a **redundancy technique** that ensures protection of data without maintaining a full set of duplicate data.
- Calculation of parity is a function of the RAID controller.
- Parity information can be stored on separate, dedicated disk drives or distributed across all the drives in a RAID set.
- Fig shows a parity RAID set.

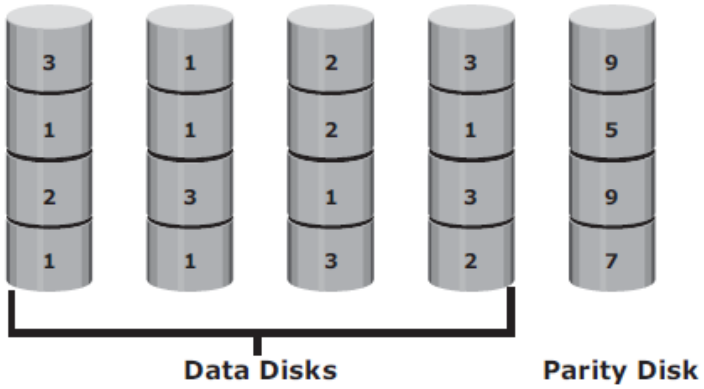

Fig : Parity RAID

- The first four disks, labeled "*Data Disks*," contain the data. The fifth disk, labeled "*Parity Disk*," stores the parity information, which, in this case, is the sum of the elements in each row.
- Now, if one of the data disks fails, the missing value can be calculated by subtracting the sum of the rest of the elements from the parity value.
- Here, computation of parity is represented as an arithmetic sum of the data. However, parity calculation is a bitwise XOR operation.

XOR Operation:

- A bit-by-bit Exclusive -OR (XOR) operation takes two bit patterns of equal length and performs the logical XOR operation on each pair of corresponding bits.
- The result in each position is 1 if the two bits are different, and 0 if they are the same.

- The truth table of the XOR operation is shown below (A and B denote inputs and C, the output the XOR operation).

Table 1.1: Truth table for XOR Operation

A	B	C
0	0	0
0	1	1
1	0	1
1	1	0

- If any of the data from A, B, or C is lost, it can be reproduced by performing an XOR operation on the remaining available data.
- Eg: if a disk containing all the data from A fails, the data can be regenerated by performing an XOR between B and C.
- Advantages:
  - ✓ Compared to mirroring, parity implementation considerably reduces the **cost** associated with data protection.
- Disadvantages:
  - ✓ Parity information is generated from data on the data disk. Therefore, parity is recalculated every time there is a change in data.
  - ✓ This recalculation is time-consuming and affects the performance of the RAID array.
- For parity RAID, the stripe size calculation does not include the parity strip.
- Eg: in a five (4 + 1) disk parity RAID set with a strip size of 64 KB, the stripe size will be 256 KB (64 KB x 4).

RAID Levels

- RAID Level selection is determined by below factors:
  - ✓ Application performance
  - ✓ data availability requirements
  - ✓ cost
- RAID Levels are defined on the basis of:
  - ✓ Striping
  - ✓ Mirroring
  - ✓ Parity techniques
- Some RAID levels use a single technique whereas others use a combination of techniques.
- Table shows the commonly used RAID levels

Table : RAID Levels

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

RAID 0

- **RAID 0** configuration uses *data striping techniques*, where data is striped across all the disks within a RAID set. Therefore it utilizes the full storage capacity of a RAID set.
- To read data, all the strips are put back together by the controller.
- Fig shows RAID 0 in an array in which data is striped across five disks.

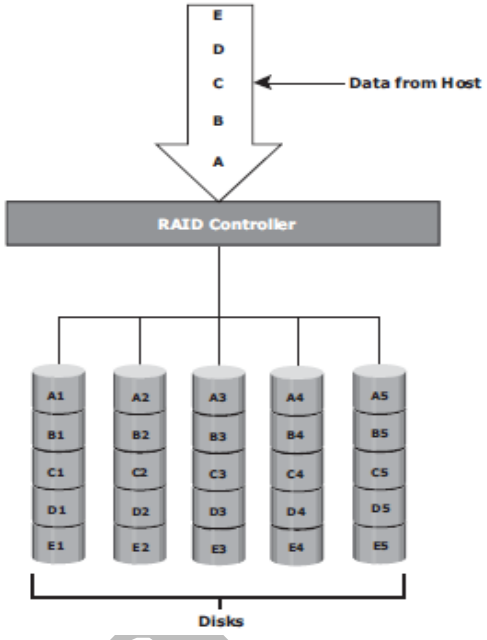

Fig : RAID 0

- When the number of drives in the RAID set increases, performance improves because more data can be read or written simultaneously.
- RAID 0 is a good option for applications that need high I/O throughput.
- However, if these applications require high availability during drive failures, RAID 0 does not provide data protection and availability.

RAID 1

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

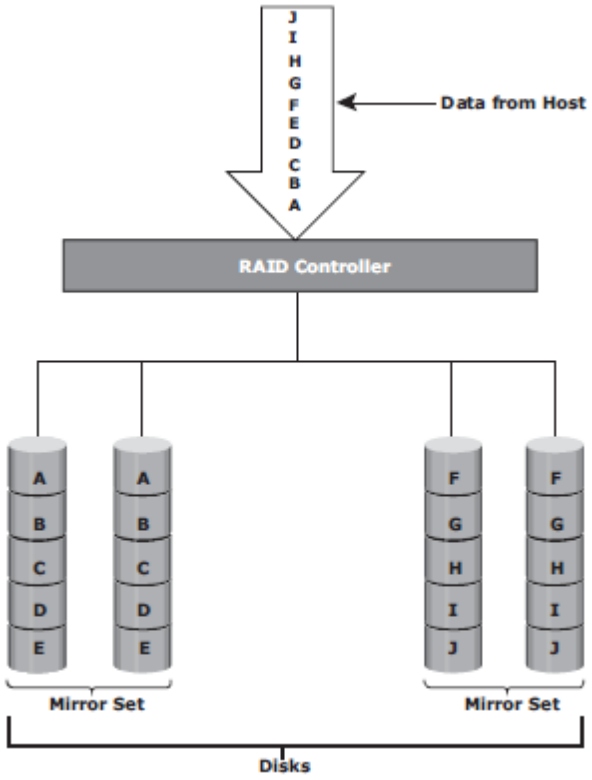

Fig : RAID 1

Nested RAID

- Most data centers require data redundancy and performance from their RAID arrays.
- RAID 1+0 and RAID 0+1 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 Fig 1.16).

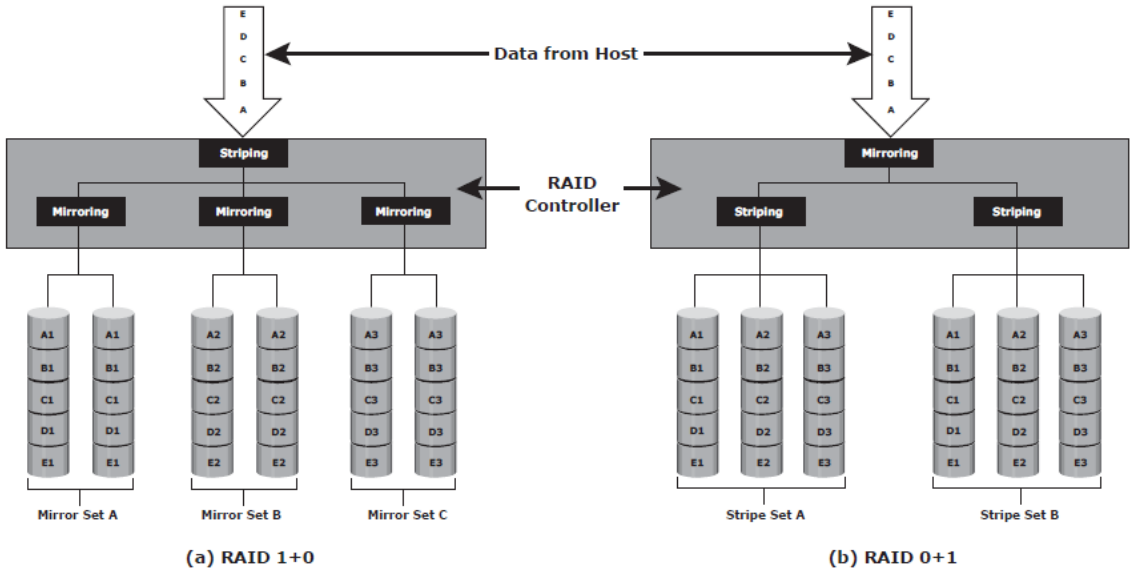

Fig: Nested RAID

RAID 1+0:

- RAID 1+0 is also known as RAID 10 (Ten) or RAID 1/0.
- RAID 1+0 performs well for workloads with small, random, write-intensive I/Os.
- Some applications that benefit from RAID 1+0 include the following:
  - ✓ High transaction rate Online Transaction Processing (OLTP)
  - ✓ Large messaging installations
  - ✓ Database applications with write intensive random access workloads
- **RAID 1+0** is also called striped mirror.
- The basic element of RAID 1+0 is a mirrored pair, which means that data is first mirrored and then both copies of the data are striped across multiple disk drive pairs in a RAID set.
- When replacing a failed drive, only the mirror is rebuilt. The disk array controller uses the surviving drive in the mirrored pair for data recovery and continuous operation.

Working of RAID 1+0:

- Eg: consider an example of six disks forming a RAID 1+0 (RAID 1 first and then RAID 0) set.
- These six disks are paired into three sets of two disks, where each set acts as a RAID 1 set (mirrored pair of disks). Data is then striped across all the three mirrored sets to form RAID 0.

- Following are the steps performed in RAID 1+0 (see Fig 1.16 [a]):
  - ✓ Drives 1+2 = RAID 1 (Mirror Set A)
  - ✓ Drives 3+4 = RAID 1 (Mirror Set B)
  - ✓ Drives 5+6 = RAID 1 (Mirror Set C)
- Now, RAID 0 striping is performed across sets A through C.
- In this configuration, if drive 5 fails, then the mirror set C alone is affected. It still has drive 6 and continues to function and the entire RAID 1+0 array also keeps functioning.
- Now, suppose drive 3 fails while drive 5 was being replaced. In this case the array still continues to function because drive 3 is in a different mirror set.
- So, in this configuration, up to three drives can fail without affecting the array, as long as they are all in different mirror sets.
- **RAID 0+1** is also called a mirrored stripe.
- The basic element of RAID 0+1 is a stripe. This means that the process of striping data across disk drives is performed initially, and then the entire stripe is mirrored.
- In this configuration if one drive fails, then the entire stripe is faulted.

Working of RAID 0+1:

- Eg: Consider the same example of six disks forming a RAID 0+1 (that is, RAID 0 first and then RAID 1).
- Here, six disks are paired into two sets of three disks each.
- Each of these sets, in turn, act as a RAID 0 set that contains three disks and then these two sets are mirrored to form RAID 1.
- Following are the steps performed in RAID 0+1 (see Fig 1.16 [b]):
  - ✓ Drives 1 + 2 + 3 = RAID 0 (Stripe Set A)
  - ✓ Drives 4 + 5 + 6 = RAID 0 (Stripe Set B)
- These two stripe sets are mirrored.
- If one of the drives, say drive 3, fails, the entire stripe set A fails.
- A rebuild operation copies the entire stripe, copying the data from each disk in the healthy stripe to an equivalent disk in the failed stripe.
- This causes increased and unnecessary I/O load on the surviving disks and makes the RAID set more vulnerable to a second disk failure.

RAID 3

- RAID 3 stripes data for 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.
- 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 in a stripe.
- 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.
- Fig shows the RAID 3 implementation

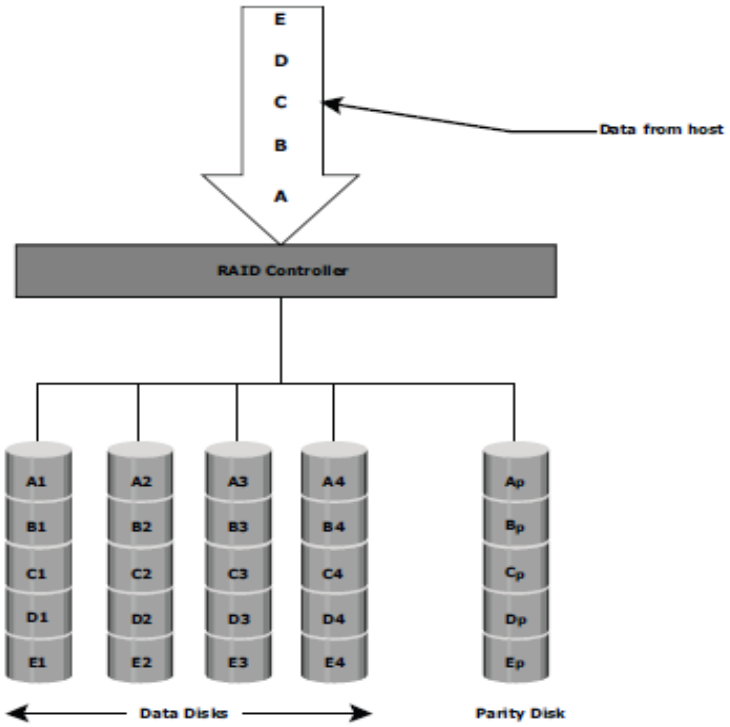

Fig: RAID 3

RAID 4

- RAID 4 stripes data for high performance and uses parity for improved fault tolerance. Data is striped across all disks except the parity disk in the array.
- Parity information is stored 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 versatile RAID implementation.
- It is similar to RAID 4 because it uses striping. The drives (strips) are also independently accessible.
- The difference between RAID 4 and RAID 5 is the 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. Below Figure illustrates the RAID 5 implementation.
- Fig illustrates the RAID 5 implementation.
- RAID 5 is good for random, read-intensive I/O applications and preferred for messaging, data mining, medium-performance media serving, and relational database management system (RDBMS) implementations, in which database administrators (DBAs) optimize data access.

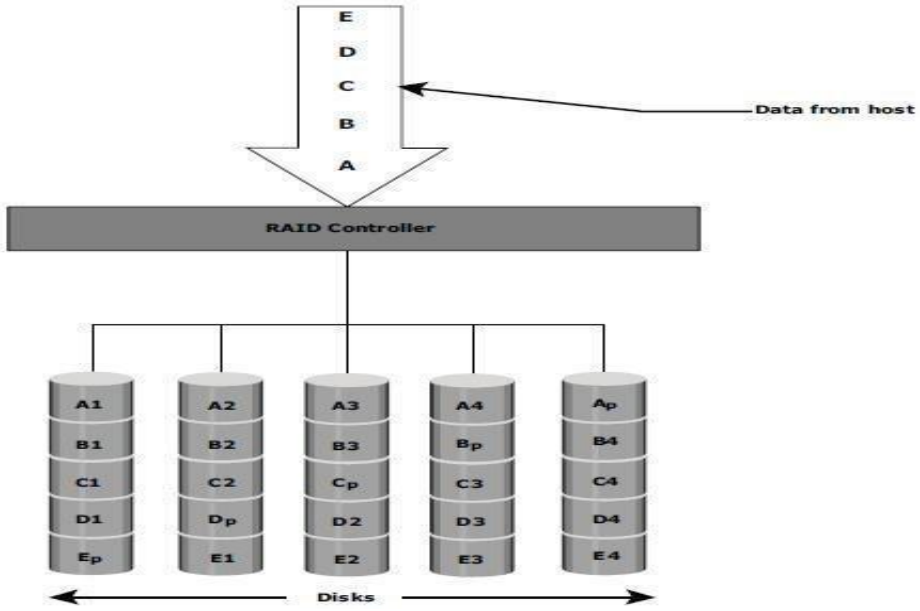

Fig : RAID 5

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. 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.
- Fig illustrates the RAID 6 implementation

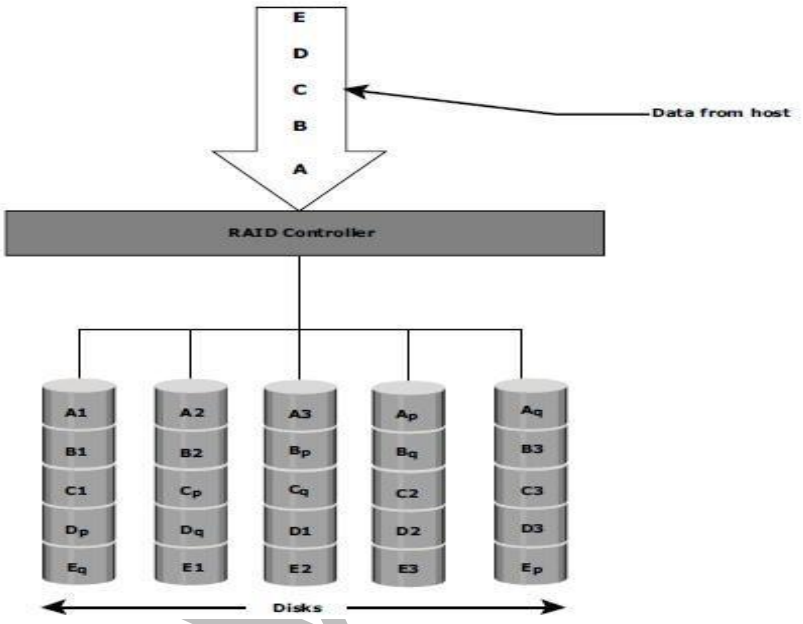

Fig: RAID 6

RAID Impact on Disk Performance

- When choosing a RAID type, it is imperative to consider its impact on disk performance and application IOPS.
- In both mirrored (RAID 1) and parity RAID (RAID 5) configurations, every write operation translates into more I/O overhead for the disks which is referred to as **write penalty**.
- In a RAID 1 implementation, every write operation must be performed on two disks configured as a mirrored pair. **The write penalty is 2.**
- In a RAID 5 implementation, a write operation may manifest as four I/O operations. When performing small I/Os to a disk configured with RAID 5, the controller has to read, calculate, and write a parity segment for every data write operation.
- Fig illustrates a single write operation on RAID 5 that contains a group of five disks.

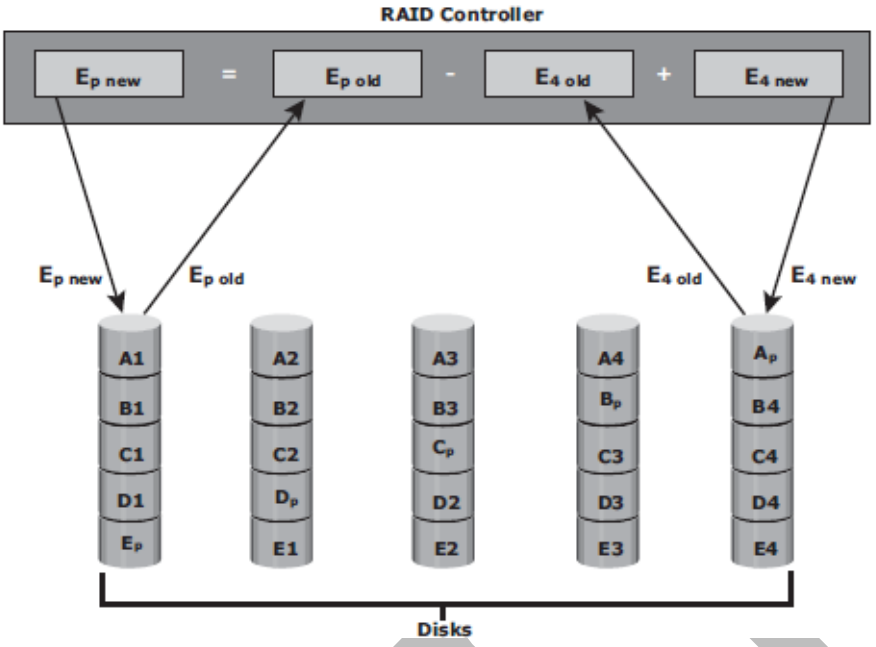

Fig : Write Penalty in RAID 5

- Four of these disks are used for data and one is used for parity.
- The **parity (E_p)** at the controller is calculated as follows:

$$E_p = E_1 + E_2 + E_3 + E_4 \text{ (XOR operations)}$$

- Whenever the controller performs a write I/O, parity must be computed by reading the old parity (E_p old) and the old data (E_4 old) from the disk, which means two read I/Os.
- The new parity (E_p new) is computed as follows:

$$E_p \text{ new} = E_p \text{ old} - E_4 \text{ old} + E_4 \text{ new (XOR operations)}$$

- After computing the new parity, the controller completes the write I/O by doing two write I/Os for the new data and the new parity onto the disks..
- Therefore, the controller performs two disk reads and two disk writes for every write operation, and **the write penalty is 4**.
- In RAID 6, which maintains dual parity, a disk write requires **three read operations**: two parity and one data.
- After calculating both new parities, the controller performs **three write operations**: two parity and an I/O.
- Therefore, in a RAID 6 implementation, the controller performs six I/O operations for each write I/O, and the **write penalty is 6**.

Components of an Intelligent Storage System

- Intelligent Storage Systems are **feature-rich RAID arrays** that provide highly optimized I/O processing capabilities.
- These storage systems are configured with a large amount of memory (called *cache*) and multiple I/O paths and use sophisticated algorithms to meet the requirements of performance-sensitive applications.
- An intelligent storage system consists of **four key components** (Refer Fig):
  - ✓ Front End
  - ✓ Cache
  - ✓ Back end
  - ✓ Physical disks.
- 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.
- In modern intelligent storage systems, front end, cache, and back end are typically integrated on a single board (referred to as a storage processor or storage controller).

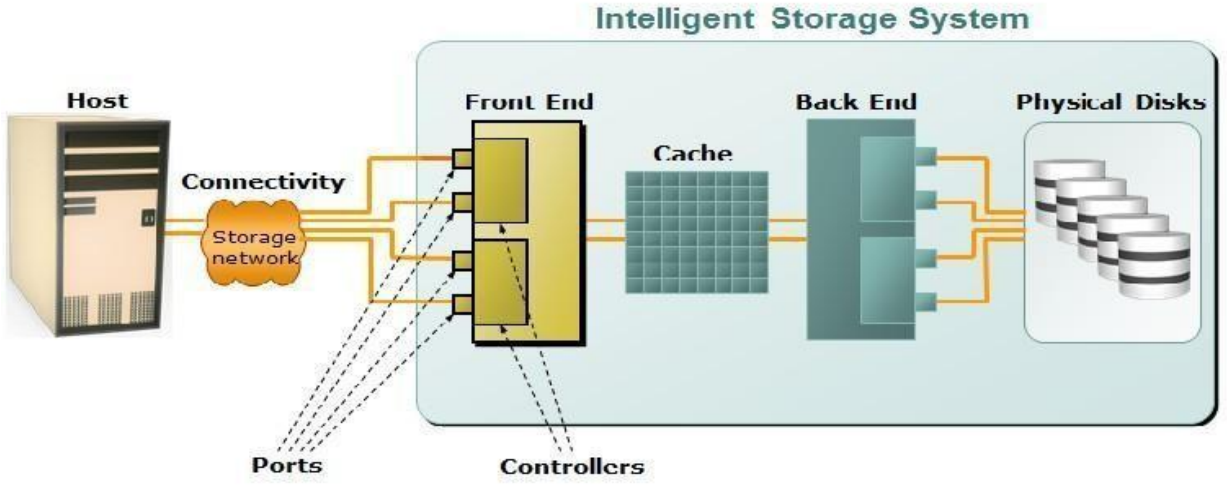

Fig : Components of an Intelligent Storage System

Front End

- The front end provides the interface between the storage system and the host.
- It consists of two components:
  - i. Front-End Ports Front-End Controllers.

- A front end has redundant controllers for high availability, and each controller contains multiple **front-end ports** that enable large numbers of hosts to connect to the intelligent storage system.
- Each front-end controller has processing logic that executes the appropriate transport protocol, such as Fibre Channel, iSCSI, FICON, or FCoE for storage connections.
- **Front-end controllers** route data to and from cache via the internal data bus.
- When the cache receives the write data, the controller sends an acknowledgment message back to the host.

Cache

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

Structure Of Cache

- Cache is organized into pages, which is the smallest unit of cache allocation. The size of a cache page is configured according to the application I/O size.
- Cache consists of the **data store** and **tag RAM**.
- The data store holds the data whereas the tag RAM tracks the location of the data in the data store (see Fig) and in the disk.
- Entries in tag RAM indicate where data is found in cache and where the data belongs on the disk.
- Tag RAM includes a dirty bit flag, which indicates whether the data in cache has been committed to the disk.
- It also contains time-based information, such as the time of last access, which is used to identify cached information that has not been accessed for a long period and may be freed up.

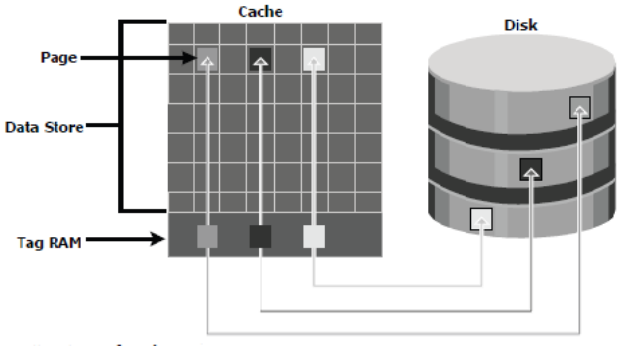

Fig : Structure of cache

Read Operation with Cache

- When a host issues a read request, the storage controller reads the tag RAM to determine whether the required data is available in cache.
- If the requested data is found in the cache, it is called a **read cache hit** or **read hit** and data is sent directly to the host, without any disk operation (see Fig [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 **cache miss** and the data must be read from the disk. 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 prefetch 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.

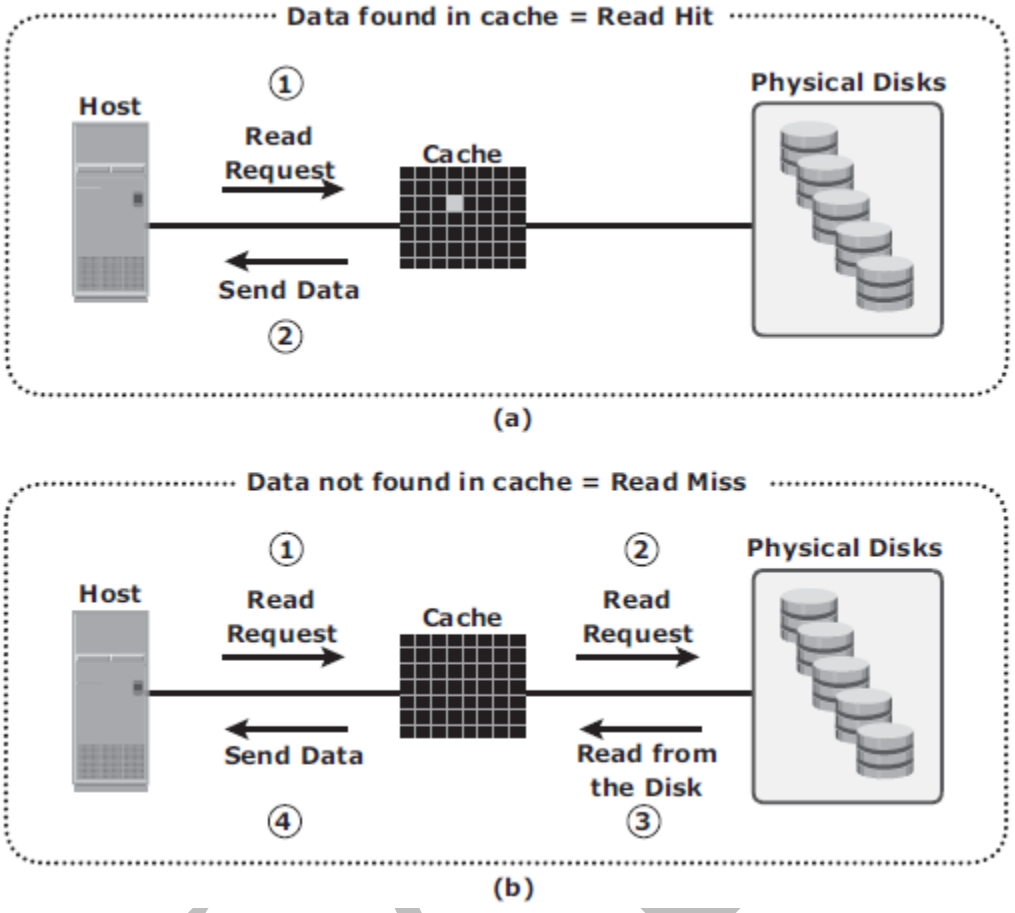

Fig 1.23 : Read hit and read miss

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 far less time (from the host's perspective) than it would take to write directly to disk.
- *Sequential writes* also offer opportunities for optimization because many smaller writes can be coalesced for larger transfers to disk drives with the use of cache.
- **A write operation** with cache is implemented in the following ways:
- **Write-back cache:** Data is placed in cache and an acknowledgment is sent to the host immediately. Later, data from several writes are committed 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.
- **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 can be bypassed under certain conditions, such as large size write I/O.
- In this implementation, if the size of an I/O request exceeds the predefined size, called **write aside size**, writes are sent to the disk directly to reduce the impact of large writes consuming a large cache space.
- This is useful in an environment where cache resources are constrained and cache is required for small random I/Os.

Cache Implementation

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

Cache Management

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

- ✓ **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, the storage system must take action to **flush dirty pages** (data written into the cache but not yet written to the disk) to manage space availability.
- **Flushing** is the process that commits data from cache to the disk.
- On the basis of the I/O access rate and pattern, high and low levels called **watermarks** are set in cache to manage the flushing process.
- **High watermark (HWM)** is the cache utilization level at which the storage system starts high-speed flushing of cache data.
- **Low watermark (LWM)** is the point at which the storage system stops flushing data to the disks.
- The *cache utilization level*, as shown in Fig, drives the mode of flushing to be used:
  - ✓ **Idle flushing:** Occurs continuously, at a modest rate, when the cache utilization level is between the high and low watermark.
  - ✓ **High watermark flushing:** Activated when cache utilization hits the high watermark. The storage system dedicates some additional resources for flushing. This type of flushing has some impact on I/O processing.
  - ✓ **Forced flushing:** Occurs in the event of a large I/O burst when cache reaches 100 percent of its capacity, which significantly affects the I/O response time. In forced flushing, system flushes the cache on priority by allocating more resources.

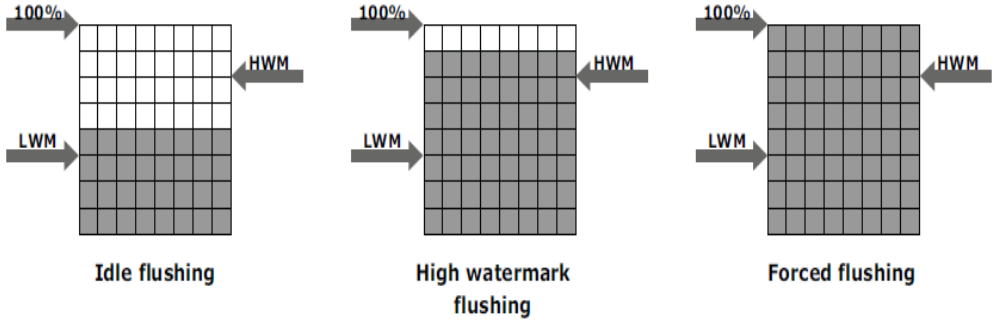

Fig : Types of flushing

Cache Data Protection

- Cache is volatile memory, so a power failure or any kind of cache failure will cause loss of the data that is not yet committed to the disk.

- This risk of losing uncommitted data held in cache can be mitigated using
  i. cache mirroring
  ii. cache vaulting

- **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.
  - ✓ In cache mirroring approaches, the problem of maintaining *cache coherency* is introduced.
  - ✓ Cache coherency means that data in two different cache locations must be identical at all times. It is the responsibility of the array operating environment to ensure coherency.

- **Cache vaulting**
  - ✓ The risk of data loss due to power failure can be addressed in various ways:
    - powering the memory with a battery until the AC power is restored
    - using battery power to write the cache content to the disk.
  - ✓ If an extended power failure occurs, using batteries is not a viable option.
  - ✓ This is because in intelligent storage systems, large amounts of data might need to be committed to numerous disks, and batteries might not provide power for sufficient time to write each piece of data to its intended disk.
  - ✓ 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.

Back End

- The **back end** provides an interface between cache and the physical disks.
- It consists of two components:
  i. Back-end ports
  ii. 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, and also RAID functionality.
- For high data protection and high availability, storage systems are configured with dual controllers with multiple ports.

Physical Disk

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

Types of Intelligent Storage Systems

- An intelligent storage system is divided into following two categories:
  1. High-end storage systems
  2. Midrange storage systems
- High-end storage systems have been implemented with active-active configuration, whereas midrange storage systems have been implemented with active-passive configuration.
- The distinctions between these two implementations are becoming increasingly insignificant.

High-end Storage Systems

- High-end storage systems, referred to as **active-active arrays,** are generally aimed at large 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 Fig).

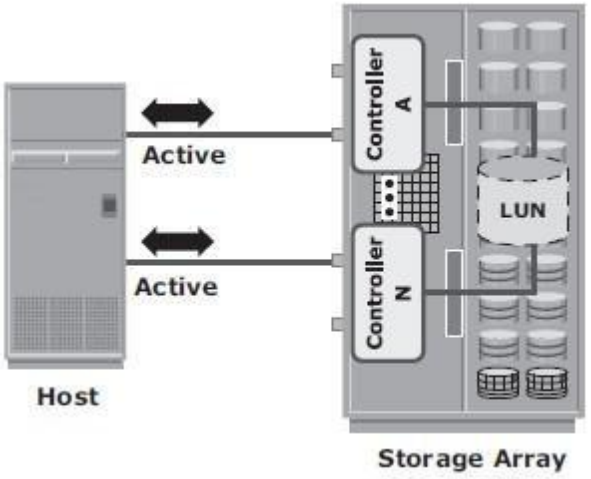

Fig : Active-active configuration

Advantages of High-end storage:

- Large storage capacity
- Large amounts of cache to service host I/Os optimally
- Fault tolerance architecture to improve data availability
- Connectivity to mainframe computers and open systems hosts Availability of multiple front-end ports and interface protocols to serve a large number of hosts
- Availability of multiple back-end Fibre Channel or SCSI RAID controllers to manage disk processing
- Scalability to support increased connectivity, performance, and storage
- capacity requirements
- Ability to handle large amounts of concurrent I/Os from a number of servers and applications
- 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.
- They also provide optimal storage solutions at a *lower cost*.
- In an *active-passive* array, a host can perform I/Os to a LUN only through the paths to the **owning controller** of that LUN. These paths are called *Active Paths*. The other paths are *passive* with respect to this LUN.

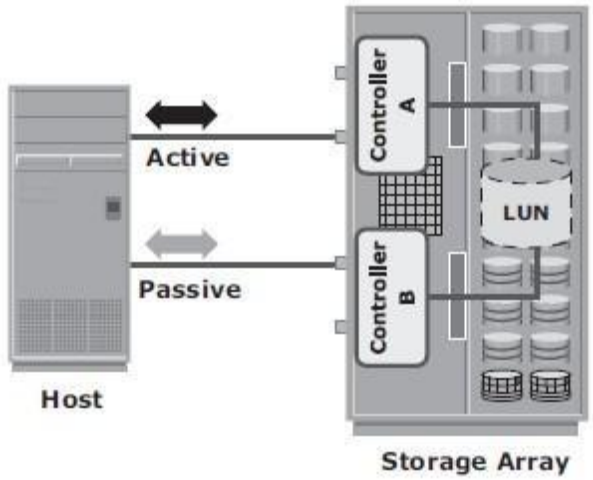

Fig : Active-passive configuration

- As shown in Fig, 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 storage systems are typically designed with two controllers, each of which contains host interfaces, cache, RAID controllers, and disk drive interfaces.
- Midrange arrays are designed to meet the requirements of small and medium enterprise applications; therefore, they host less storage capacity and cache than high-end storage arrays.
- There are also fewer front-end ports for connection to hosts.
- But they ensure high redundancy and high performance for applications with predictable workloads.
- They also support array-based local and remote replication.

FIBRE CHANNEL STORAGE AREA NETWORKS

SAN is a high-speed dedicated network of servers and shared storage. Commom SAN deployments are:

- ✓ FC SAN
- ✓ IP SAN

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.

Components of SAN

- Components of FC SAN infrastructure are:

1) **Node Ports,**
2) **Cabling,**
3) **Connectors,**
4) **Interconnecting Devices (Such As Fc Switches Or Hubs),**
5) **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.
- A port operates in full-duplex data transmission mode with a **transmit (Tx) link and a**

receive (Rx) link (see Fig 2.1).

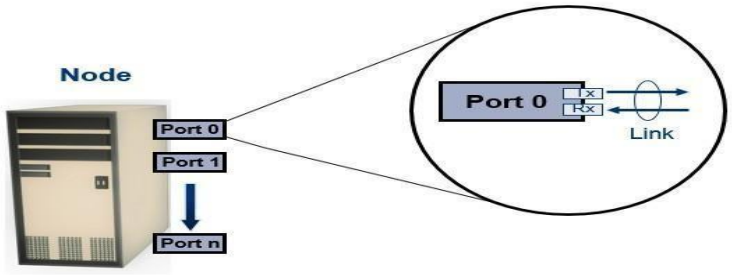

Fig 2.1: Nodes, Ports, links

Cabling

- SAN implementations use optical fiber cabling.
- Copper can be used for shorter distances for back-end connectivity
- Optical fiber cables carry data in the form of light.
- There are two types of optical cables :**Multi-Mode And Single-Mode.**

1) **Multi-mode fiber (MMF)** cable carries multiple beams of light projected at different angles simultaneously onto the core of the cable (see Fig 2.2 (a)).
    - 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*.
    - MMFs are generally used within data centers for shorter distance runs
2) **Single-mode fiber (SMF)** carries a single ray of light projected at the center of the core (see Fig 2.2 (b)).
    - 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).
    - A single-mode cable is used for long-distance cable runs, limited only by the power

of the laser at the transmitter and sensitivity of the receiver.

- SMFs are used for longer distances.

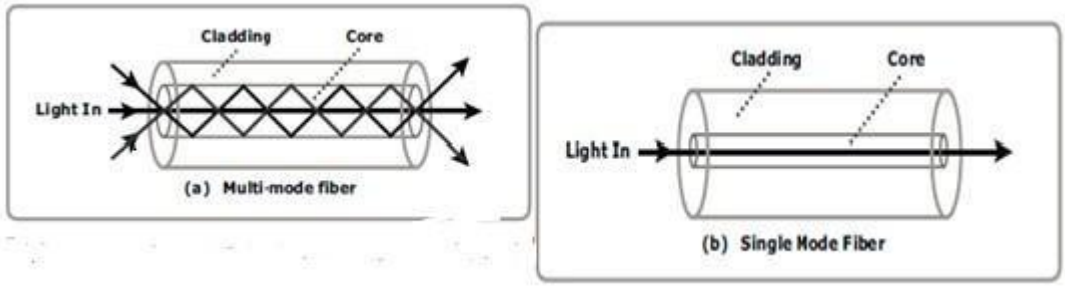

Fig 2.2: Multimode fiber and single-mode fiber

Connectors

- They are attached at the end of the cable to enable swift connection and disconnection of the cable to and from a port.
- A **Standard connector (SC)** (see Fig 2.3 (a)) and a **Lucent connector (LC)** (see Fig 2.3 (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 2.3 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 Fig 2.3 (c)).

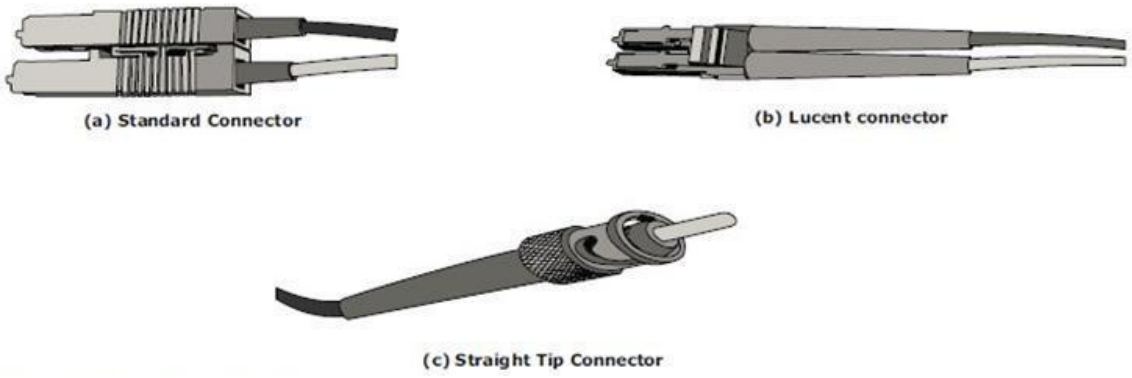

Fig 2.3: SC,LC, and ST connectors

Interconnect Devices

The commonly used interconnecting devices in SAN are

1) **Hubs,**
2) **Switches,**
3) **Directors**

- **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.
- Switches are available with:
  - ✓ Fixed port count
  - ✓ Modular design : port count is increased by installing additional port cards to open slots.
- **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.
- Port card or blade has multiple ports for connecting nodes and other FC switches

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*

FC Connectivity

The FC architecture supports three basic interconnectivity options:

1) **Point-To-point,**
2) **Arbitrated Loop (Fc-AL),**
3) **FC Switched Fabric**

Point-to-Point

- **Point-to-point** is the simplest FC configuration — two devices are connected directly to each other, as shown in Fig 2.4.
- This configuration provides a dedicated connection for data transmission between nodes.
- The point-to-point configuration offers limited connectivity, as only two devices can communicate with each other at a given time.
- It cannot be scaled to accommodate a large number of network devices. Standard DAS uses point to- point connectivity.

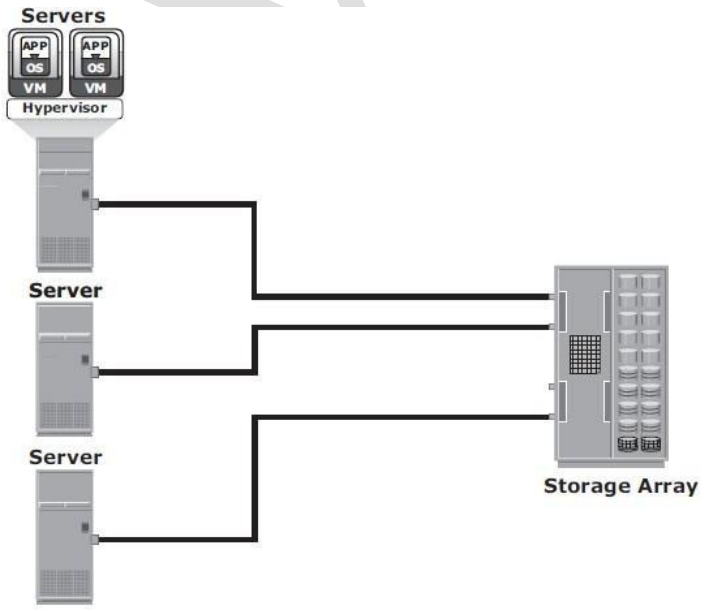

Fig 2.4: Point-to-point connectivity

Fibre Channel Arbitrated Loop

- In the FC-AL configuration, devices are attached to a shared loop, as shown in Fig 2.5.
- 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.
- FC-AL implementations may also use hubs whereby the arbitrated loop is physically connected in a star topology.

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

- 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 an I/O request, the speed of data transmission is low in an FC-AL topology.
- FC-AL uses 8-bit addressing. It can support up to 127 devices on a loop.
- Adding or removing a device results in loop re-initialization, which can cause a momentary pause in loop traffic.

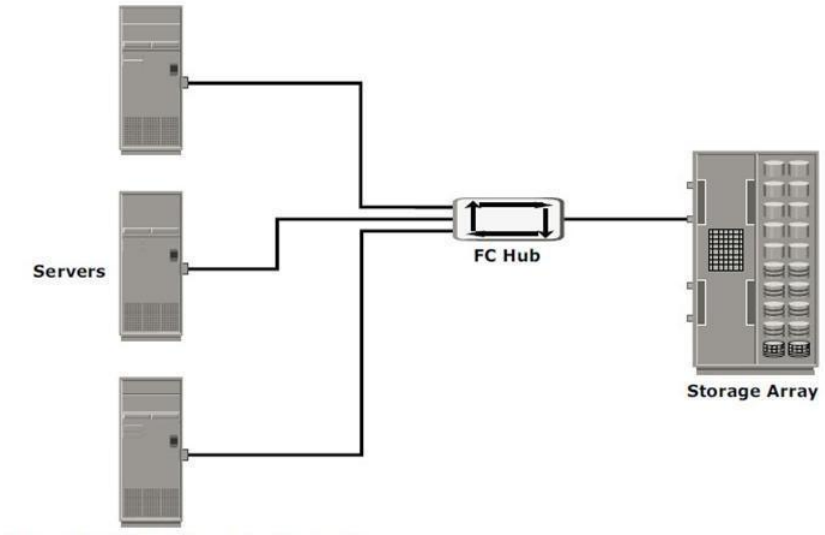

Fig 2.5: Fibre Channel Arbitrated Loop

Fibre Channel Switched Fabric(FC-SW)

- FC-SW provides dedicated data path and scalability.
- The addition and removal of a device doesnot affect the on-going traffic between other devices.
- FC-SW is 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 a unique domain identifi er, which is part of the fabric's addressing scheme.
- In a switched fabric, the link between any two switches is called an *Interswitch link* (ISL).
- ISLs enable switches to be connected together to form a single, larger fabric.
- ISLs are used to transfer host-to-storage data and fabric management traffic from one switch to another.
- By using ISLs, a switched fabric can be expanded to connect a large number of nodes.

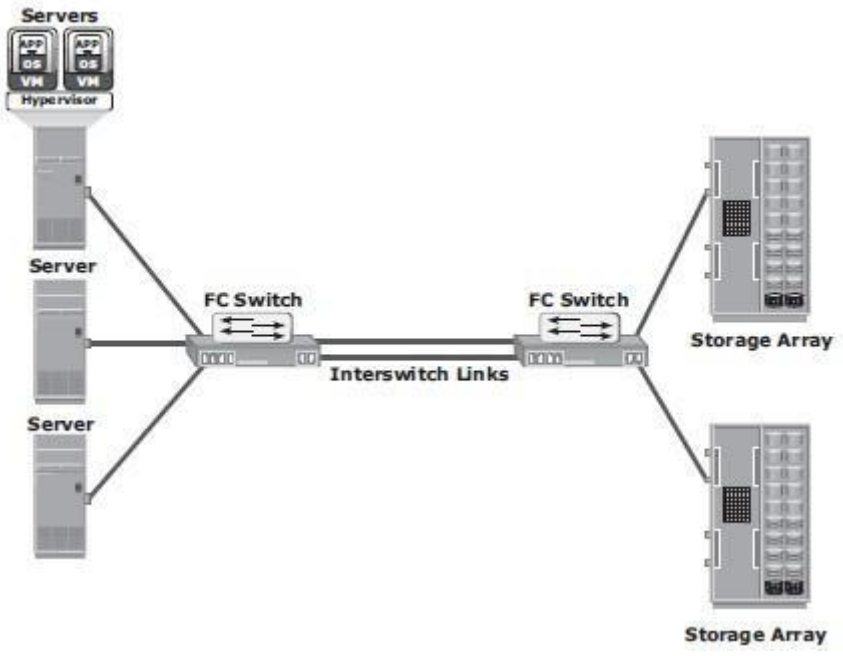

Fig 2.6: Fibre Channel switched Fabric

- A Fabric may contain tiers.
- The number of tiers in a fabric is based on the number of switches between two points that are farthest from each other

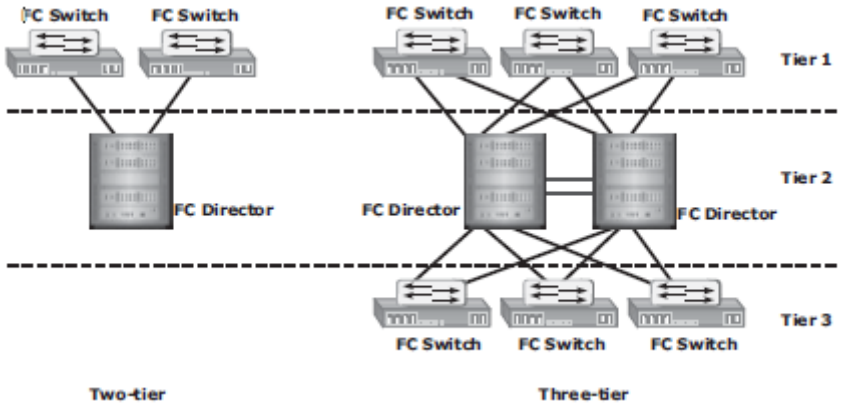

Fig 2.7: Tiered structure of Fibre Channel switched Fabric

FC-SW Transmission

- FC-SW uses switches that can switch data traffi c between nodes directly through switch ports.
- Frames are routed between source and destination by the fabric.

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.

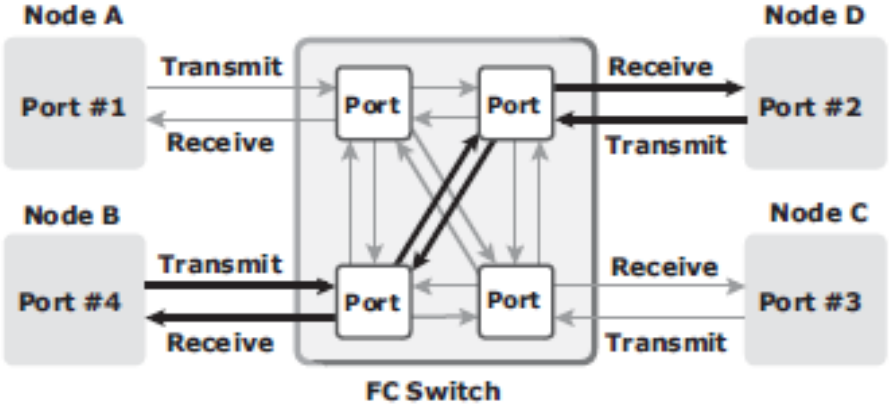

Fig 2.8: Data transmission in fibre channel switched fabric

Fibre Channel Architecture

- Connections in a SAN are accomplished using FC.
- ***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:
  - Sustained transmission bandwidth over long distances.
  - Support for a larger number of addressable devices over a network.
  - Theoretically, FC can support over 15 million device addresses on a network.
  - 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.
- Fig 2.9 illustrates the fibre channel protocol stack.

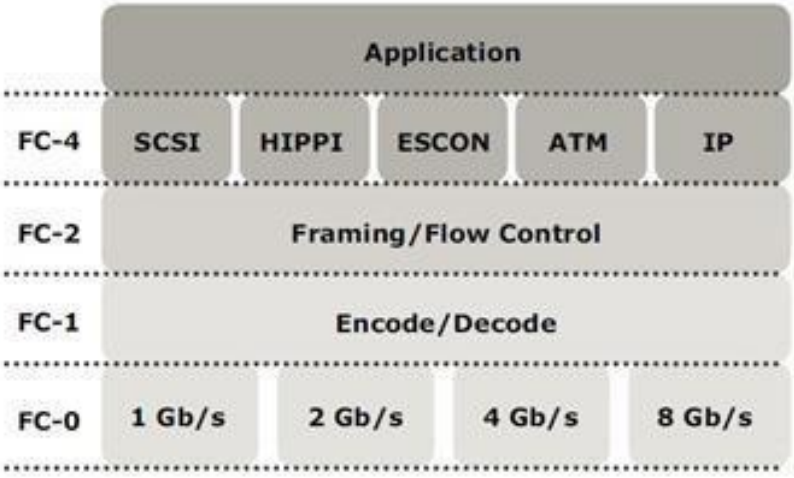

Fig 2.9: Fibre Channel Protocol Stack

- **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 Fig 2.9). 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, as shown in Fig 2.10. The addressing mechanism provided here corresponds to the fabric with the switch as an interconnecting device.
- The first field of the FC address contains the domain ID of the switch (see Fig 2.10).
- A *domain ID* is a unique number provided to each switch in the fabric.
- This is an 8-bit field, there are only 239 available addresses for domain ID because some addresses are deemed special and reserved for fabric management services.
- For example, FFFFFC is reserved for the name server, and FFFFFE is reserved for the fabric login service.
- The *area ID* is used to identify a group of switch ports used for connecting nodes. An example of a group of ports with a common area ID is a port card on the switch.
- The last field, the *port ID*, identifies the port within the group.
- The maximum possible number of node ports in a switched fabric is calculated as:
  239 domains ¥ 256 areas ¥ 256 ports = 15,663,104

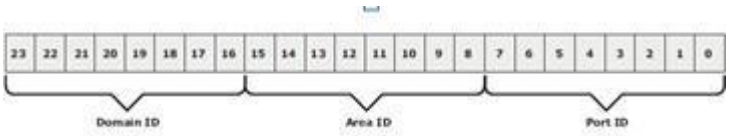

Fig 2.10 24-bit FC address of N_port

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 Fig 2.11).
- Whenever a change takes place in the name server database, the fabric controller sends a Registered State Change Notifi cation (RSCN) to all the nodes impacted by the change.
- If zoning is not configured, the fabric controller sends an RSCN to all the nodes in the fabric. Involving the nodes that are not impacted by the change results in increased fabric-management traffic.
- Zoning helps to limit the number of RSCNs in a fabric. In the presence of zoning, a fabric sends the RSCN to only those nodes in a zone where the change has occurred.

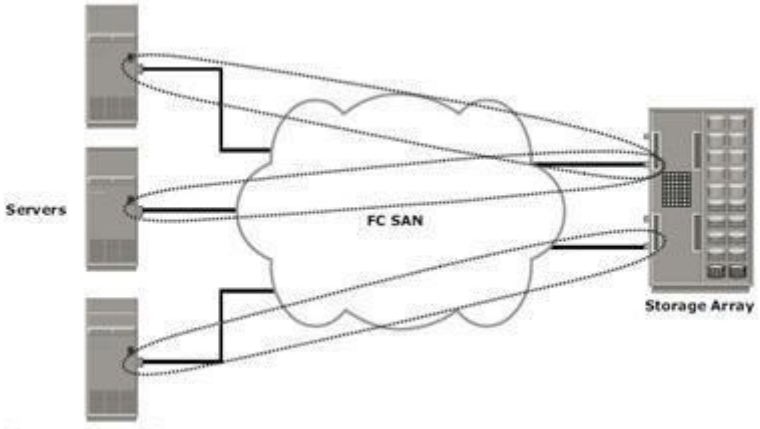

Fig 2.11 Zoning

- 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 Fig 2.12).
- **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*.

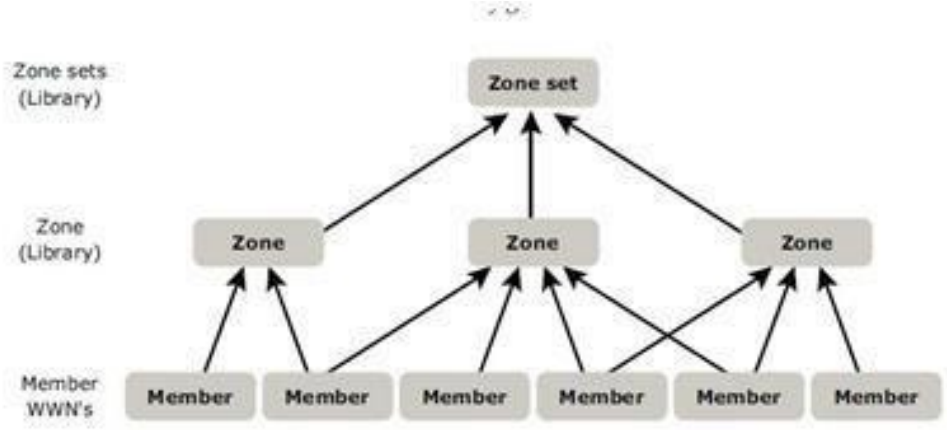

Fig 2.12: Members,Zones, and Zone sets

Types of Zoning

Zoning can be categorized into three types:

1) **Port zoning**
2) **WWN zoning**
3) **Mixed zoning**

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.

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.

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.

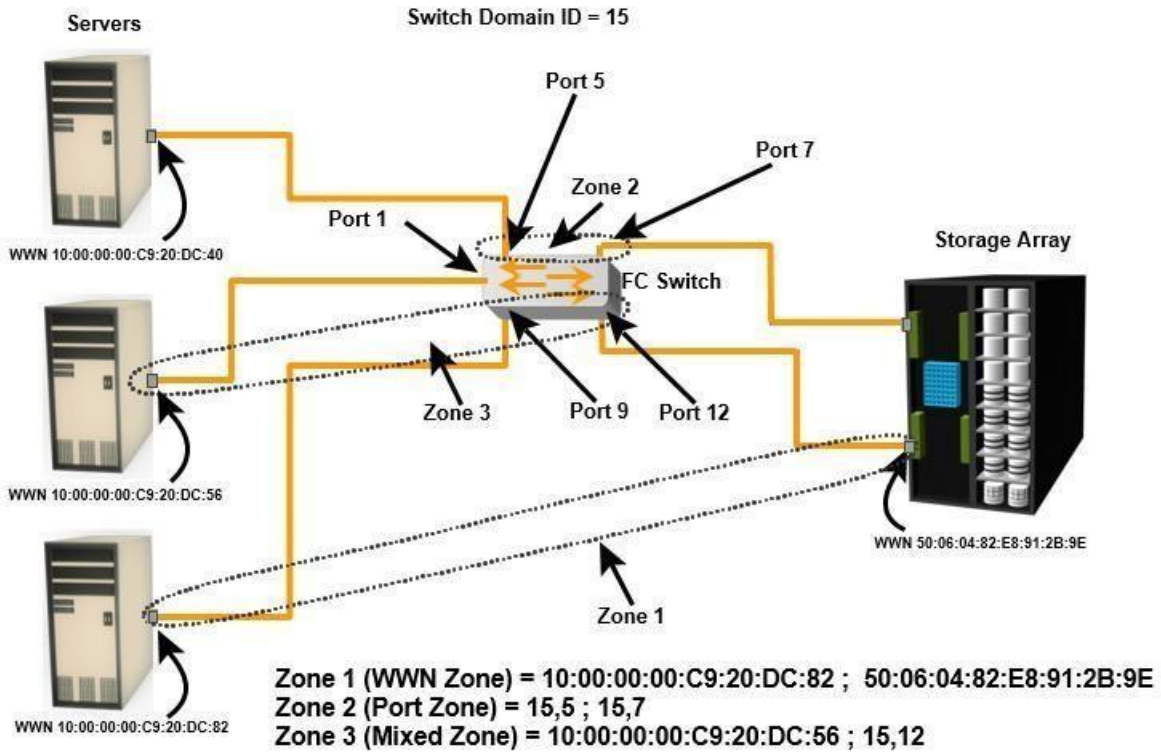

Fig 2.14: Types of Zoning

- Zoning is used in conjunction with LUN masking for controlling server access to storage. However, these are two different activities. Zoning takes place at the fabric level and LUN masking is done at the array level.

FC Topologies

- Fabric design follows standard topologies to connect devices. There are two types of topologies.
  - **Mesh Topology**
  - **Core-Edge Fabric**

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. A typical deployment would involve up to four switches or directors, with each of them servicing highly localized host-to-storage traffic. In a full mesh topology, a maximum of one ISL or hop is required for host-to-storage traffic.
- 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. A full mesh topology with a symmetric design results in an even number of switches, whereas a partial mesh has an asymmetric design and may result in an odd number of switches. Fig 2.15 depicts both a full mesh and a partial mesh topology.

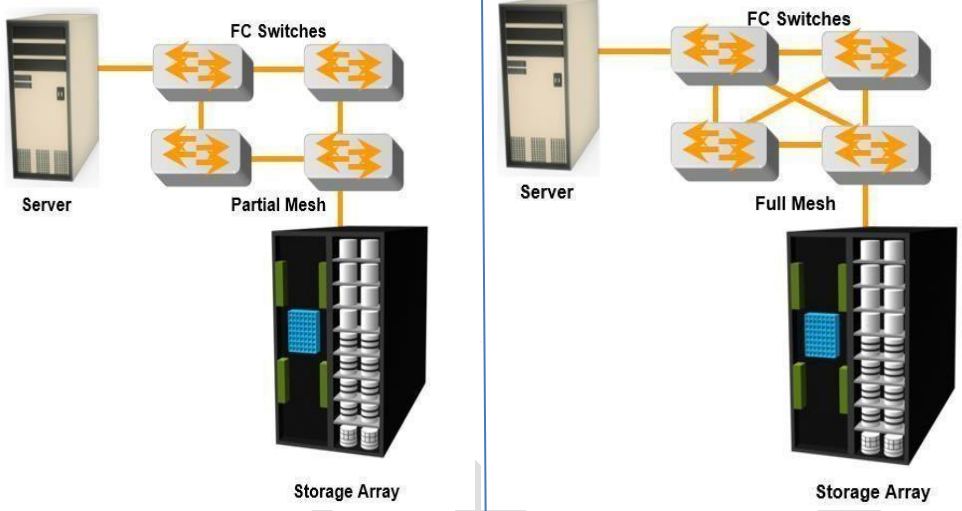

Fig a :Partial **Fig b:Full**

Fig 2.15: Partial and Full mesh Topologies

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 host-to-storage traffic has to traverse one and two ISLs in a two-tier and three-tier configuration, respectively.
- 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. Fig 2.16 depicts the core and edge switches in a 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. Fig 2.17 illustrates the core and edge switches in a dual-core topology.

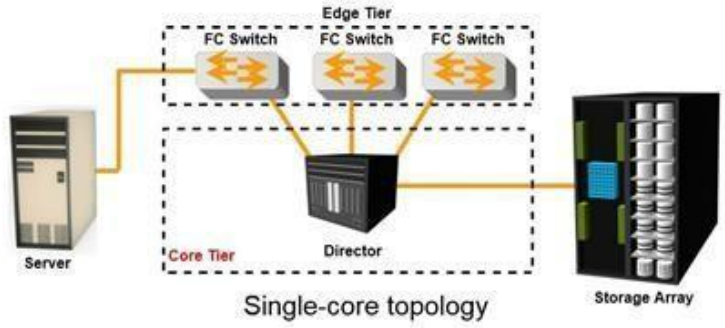

Fig 2.16: Single-core topology

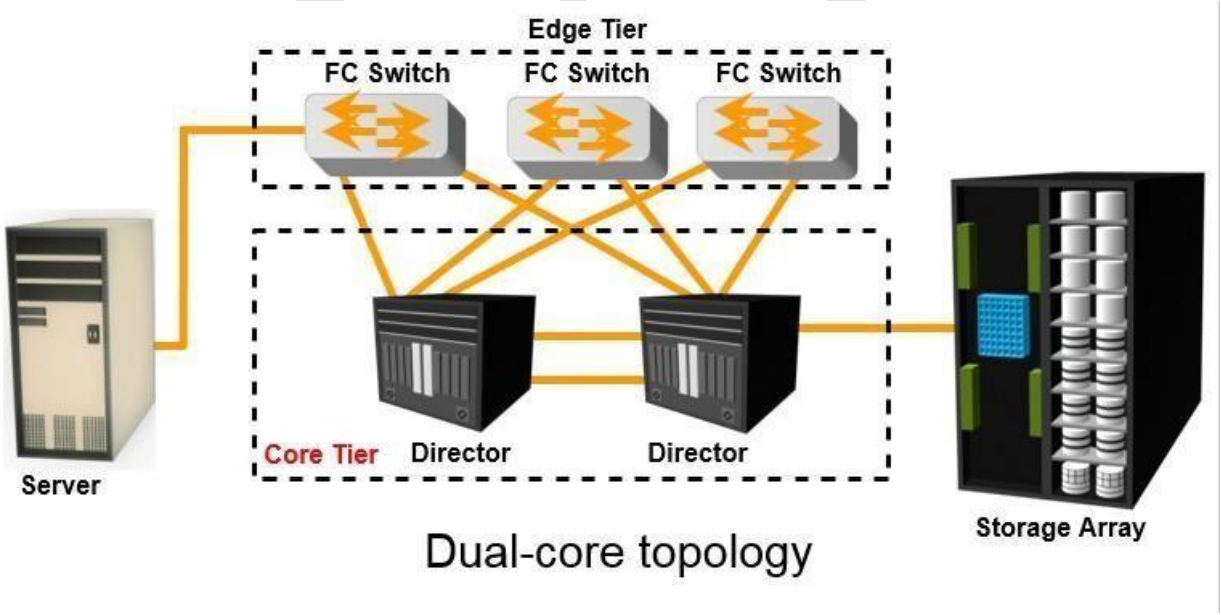

Fig 2.17: multi-core topology

Benefits and Limitations of Core-Edge Fabric

- The core-edge fabric provides one-hop storage access to all storage in the system. Because traffic travels in a deterministic pattern (from the edge to the core), a core-edge provides easier calculation of ISL loading and traffic patterns.
- 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.
- Core-edge fabrics can be scaled to larger environments by linking core switches, adding more core switches, or adding more edge switches.
- However, 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.

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

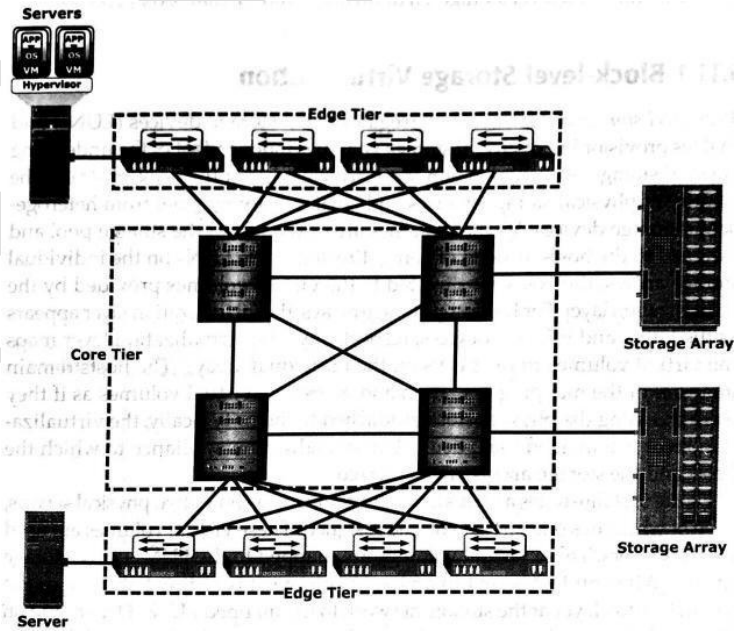

Fig 2.18: Compound core-edge topology

SAN based virtualization and VSAN technology

There are two network-based virtualization techniques in a SAN environment:

- block-level storage virtualization
- virtual SAN (VSAN).

Block-level Storage Virtualization

- *Block-level storage virtualization* aggregates block storage devices (LUNs) and enables provisioning of virtual storage volumes, independent of the underlying physical storage.
- A virtualization layer, which exists at the SAN, abstracts the identity of physical storage devices and creates a storage pool from heterogeneous storage devices.
- Virtual volumes are created from the storage pool and assigned to the hosts.
- Instead of being directed to the LUNs on the individual storage arrays, the hosts are directed to the virtual volumes provided by the virtualization layer.
- For hosts and storage arrays, the virtualization layer appears as the target and initiator devices, respectively.
- The virtualization layer maps the virtual volumes to the LUNs on the individual arrays.
- The hosts remain unaware of the mapping operation and access the virtual volumes as if they were accessing the physical storage attached to them.
- Typically, the virtualization layer is managed via a dedicated virtualization appliance to which the hosts and the storage arrays are connected.
- Fig 2.19 illustrates a virtualized environment. It shows two physical servers, each of which has one virtual volume assigned. These virtual volumes are used by the servers. These virtual volumes are mapped to the LUNs in the storage arrays.
- When an I/O is sent to a virtual volume, it is redirected through the virtualization layer at the storage network to the mapped LUNs.

- Depending on the capabilities of the virtualization appliance, the architecture may allow for more complex mapping between array LUNs and virtual volumes.

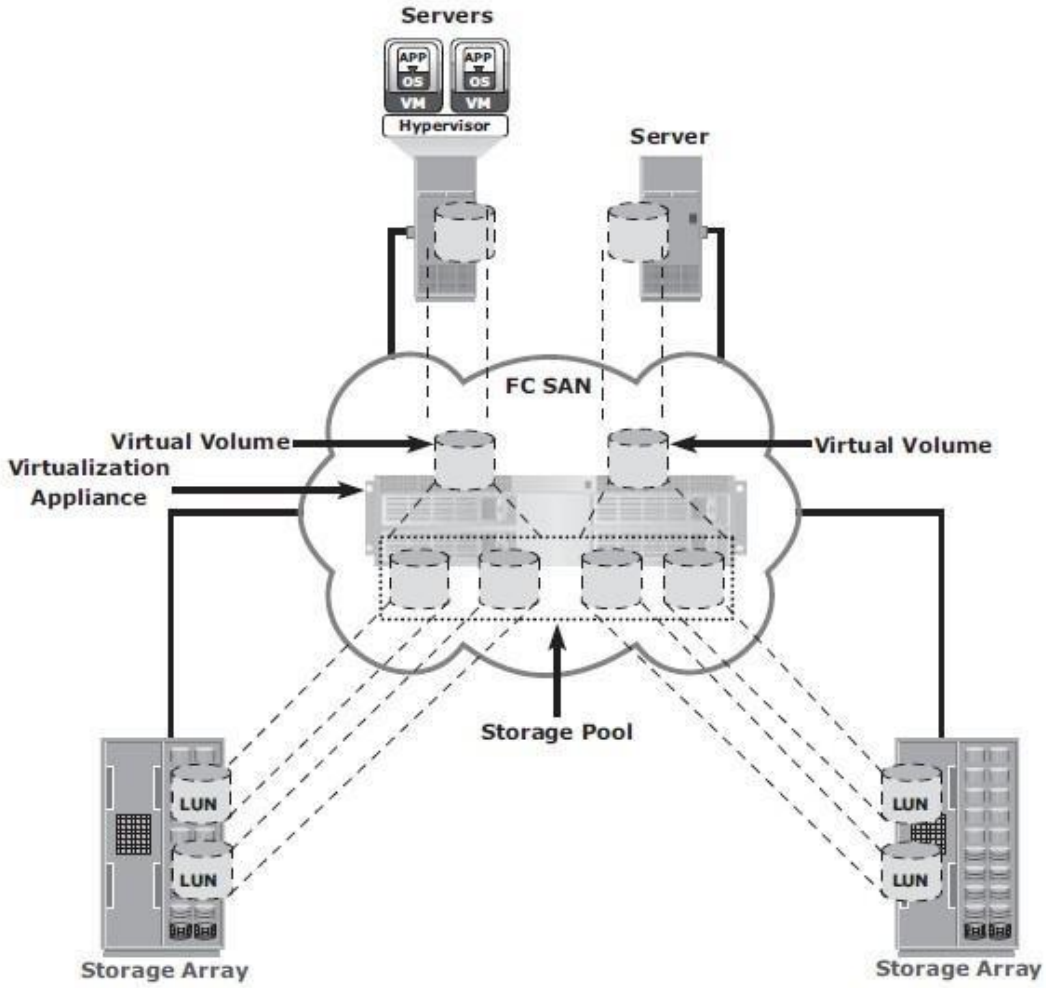

Fig 2.19 Block-level storage virtualization

- Block-level storage virtualization also provides the advantage of nondisruptive data migration.
- In a traditional SAN environment, LUN migration from one array to another is 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 multivendor environment.
- With a block-level virtualization as a solution, the virtualization layer handles the back-end

migration of data, which enables LUNs to remain online and accessible while data is migrating.

- No physical changes are required because the host still points to the same virtual targets on the virtualization layer.
- Previously, block-level storage virtualization provided nondisruptive data migration only within a data center. The new generation of block-level storage virtualization enables nondisruptive data migration both within and between data centers.
- It provides the capability to connect the virtualization layers at multiple data centers. The connected virtualization layers are managed centrally and work as a single virtualization layer stretched across data centers (Fig 2.20). This enables the federation of block-storage resources both within and across data centers. The virtual volumes are created from the federated storage resources.

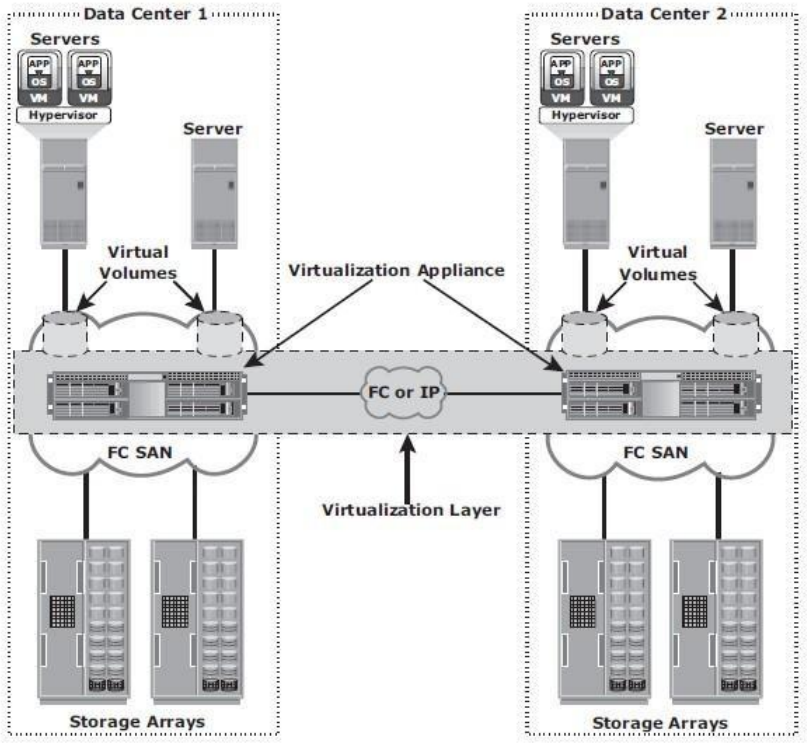

Fig 2.20 Federation of block storage across data centers

Virtual SAN (VSAN)

- *Virtual SAN* (also called *virtual fabric*) is a logical fabric on an FC SAN, which enables communication among a group of nodes regardless of their physical location in the fabric.
- In a VSAN, a group of hosts or storage ports communicate with each other using a virtual topology defined on the physical SAN.
- Multiple VSANs may be created on a single physical SAN.
- Each VSAN acts as an independent fabric with its own set of fabric services, such as name server, and zoning.
- Fabric-related configurations in one VSAN do not affect the traffic in another.
- VSANs improve SAN security, scalability, availability, and manageability.
- VSANs facilitate an easy, flexible, and less expensive way to manage networks.
- Configuring VSANs is easier and quicker compared to building separate physical FC SANs for various node groups.
- To regroup nodes, an administrator simply changes the VSAN configurations without moving nodes and recabling.