Unit 2 ISM - class pdf

Information Storage And Management (SRM Institute of Science and Technology)

Chapter 5
Fibre Channel Storage Area Networks

KEY CONCEPTS

- Fibre Channel (FC) Architecture
- Fibre Channel Protocol Stack
- Ports in Fibre Channel SAN
- Fibre Channel Addressing
- World Wide Names
- Zoning
- Fibre Channel SAN Topologies
- Block-level Storage Virtualization
- Virtual SAN

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

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

requirements of applications and, at the same time, the total cost of managing information must be low.

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

Common SAN deployments are Fibre Channel (FC) SAN and IP SAN. Fibre Channel SAN uses Fibre Channel protocol for the transport of data, commands, and status information between servers (or hosts) and storage devices. IP SAN uses IP-based protocols for communication.

This chapter provides detailed insight into the FC technology on which an FC SAN is deployed. It also covers FC SAN components, topologies, and block storage virtualization.

5.1 Fibre Channel: Overview

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

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

5.2 The SAN and Its Evolution

A SAN carries data between servers (or *hosts*) and storage devices through Fibre Channel network (see Figure 5-1). A SAN enables storage consolidation and enables storage to be shared across multiple servers. This improves the utilization of storage resources compared to direct-attached storage architecture and reduces the total amount of storage an organization needs to purchase and manage. With consolidation, storage management becomes centralized and less complex, which further reduces the cost of managing information. SAN also enables organizations to connect geographically dispersed servers and storage.

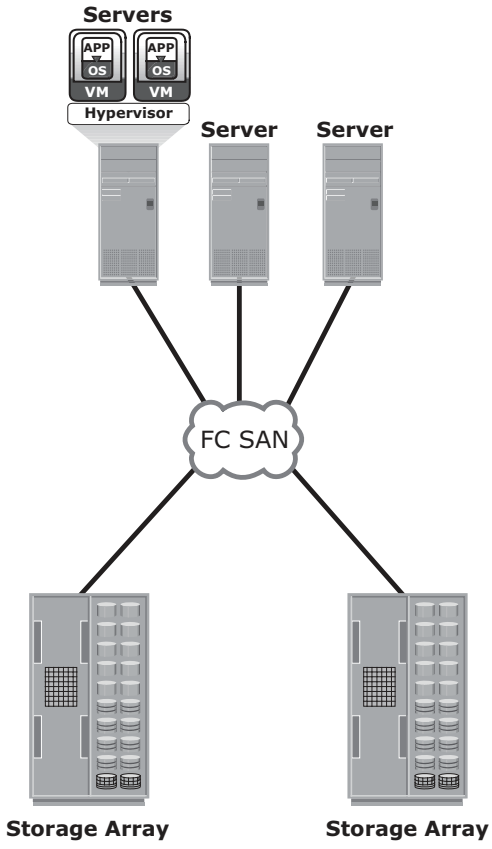


Figure 5-1: FC SAN implementation

In its earliest implementation, the FC SAN was a simple grouping of hosts and storage devices connected to a network using an FC hub as a connectivity device. This configuration of an FC SAN is known as a *Fibre Channel Arbitrated*

Loop (FC-AL). Use of hubs resulted in isolated FC-AL SAN islands because hubs provide limited connectivity and bandwidth.

The inherent limitations associated with hubs gave way to high-performance FC *switches*. Use of switches in SAN improved connectivity and performance and enabled FC SANs to be highly scalable. This enhanced data accessibility to applications across the enterprise. Now, FC-AL has been almost abandoned for FC SANs due to its limitations but still survives as a back-end connectivity option to disk drives. Figure 5-2 illustrates the FC SAN evolution from FC-AL to enterprise SANs.

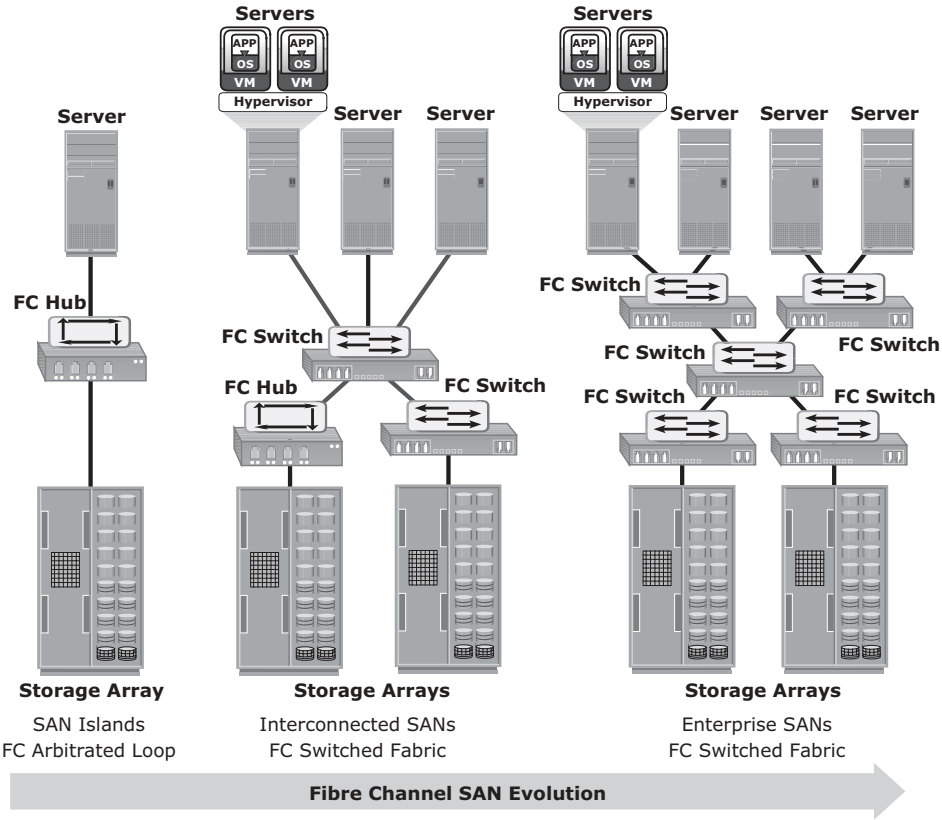


Figure 5-2: FC SAN evolution

5.3 Components of FC SAN

FC SAN is a network of servers and shared storage devices. Servers and storage are the end points or devices in the SAN (called *nodes*). FC SAN infrastructure consists of node ports, cables, connectors, and interconnecting devices (such as FC switches or hubs), along with SAN management software.

5.3.1 Node Ports

In a Fibre Channel network, the end devices, such as hosts, storage arrays, and tape libraries, are all referred to as *nodes*. Each node is a source or destination of information. Each node requires one or more ports to provide a physical interface for communicating with other nodes. These ports are integral components of host adapters, such as HBA, and storage front-end controllers or adapters. In an FC environment a port operates in full-duplex data transmission mode with a *transmit* (Tx) link and a *receive* (Rx) link (see Figure 5-3).

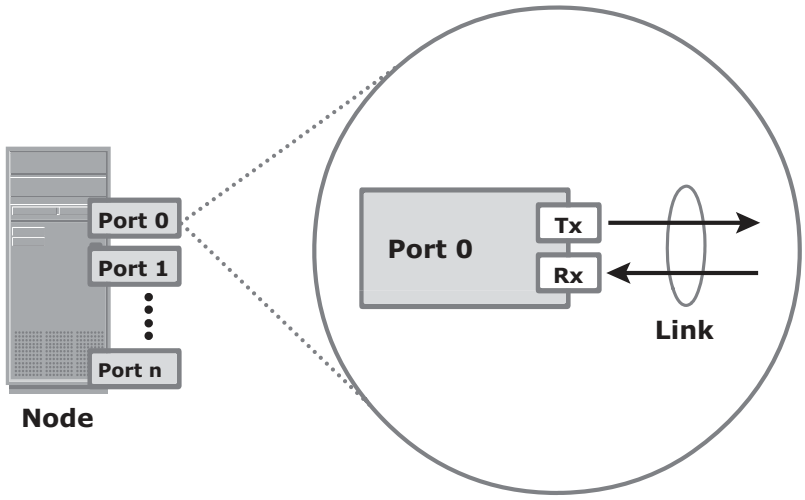


Figure 5-3: Nodes, ports, and links

5.3.2 Cables and Connectors

SAN implementations use optical fiber cabling. Copper can be used for shorter distances for back-end connectivity because it provides an acceptable signal-to-noise ratio for distances up to 30 meters. Optical fiber cables carry data in the form of light. There are two types of optical cables: multimode and single-mode. *Multimode fiber* (MMF) cable carries multiple beams of light projected at different angles simultaneously onto the core of the cable (see Figure 5-4 [a]). Based on the bandwidth, multimode fibers are classified as OM1 (62.5µm core), OM2 (50µm core), and laser-optimized OM3 (50µm core). In an MMF transmission, multiple light beams traveling inside the cable tend to disperse and collide. This collision weakens the signal strength after it travels a certain distance — a process known as *modal dispersion*. An MMF cable is typically used for short distances because of signal degradation (attenuation) due to modal dispersion.

Single-mode fiber (SMF) carries a single ray of light projected at the center of the core (see Figure 5-4 [b]). These cables are available in core diameters of 7 to 11 microns;

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

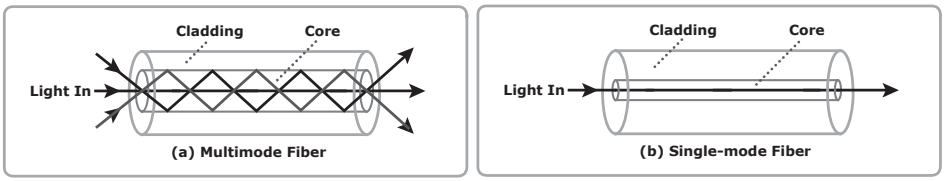


Figure 5-4: Multimode fiber and single-mode fiber

MMFs are generally used within data centers for shorter distance runs, whereas SMFs are used for longer distances.

A connector is attached at the end of a cable to enable swift connection and disconnection of the cable to and from a port. A *Standard connector* (SC) (see Figure 5-5 [a]) and a *Lucent connector* (LC) (see Figure 5-5 [b]) are two commonly used connectors for fiber optic cables. *Straight Tip* (ST) is another fiber-optic connector, which is often used with fiber patch panels (see Figure 5.5 [c]).

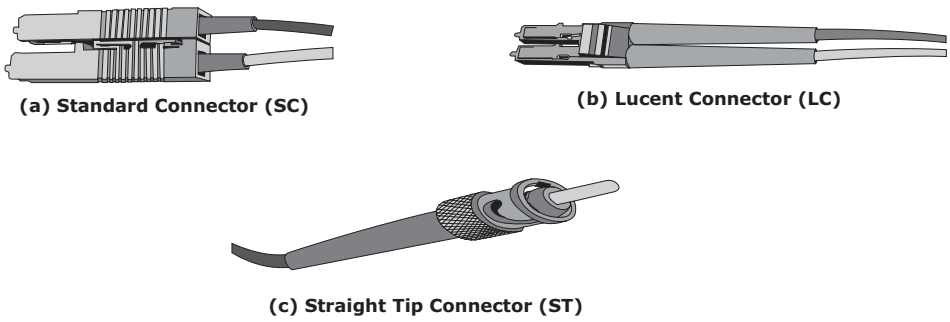


Figure 5-5: SC, LC, and ST connectors

5.3.3 Interconnect Devices

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

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

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

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

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

A port card or blade has multiple ports for connecting nodes and other FC switches. Typically, a Fibre Channel transceiver is installed at each port slot that houses the transmit (Tx) and receive (Rx) link. In a transceiver, the Tx and Rx links share common circuitry. Transceivers inside a port card are connected to an application specific integrated circuit, also called port ASIC. Blades in a director usually have more than one ASIC for higher throughput.

5.3.4 SAN Management Software

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

It provides key management functions, including mapping of storage devices, switches, and servers, monitoring and generating alerts for discovered devices, and *zoning* (discussed in section 5.9 "Zoning" later in this chapter).

FC SWITCH VERSUS FC HUB

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

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

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

5.4 FC Connectivity

The FC architecture supports three basic interconnectivity options: point-to-point, arbitrated loop, and Fibre Channel switched fabric.

5.4.1 Point-to-Point

Point-to-point is the simplest FC configuration — two devices are connected directly to each other, as shown in Figure 5-6. This configuration provides a dedicated connection for data transmission between nodes. However, the point-to-point configuration offers limited connectivity, because only two devices can communicate with each other at a given time. Moreover, it cannot be scaled to accommodate a large number of nodes. Standard DAS uses point-to-point connectivity.

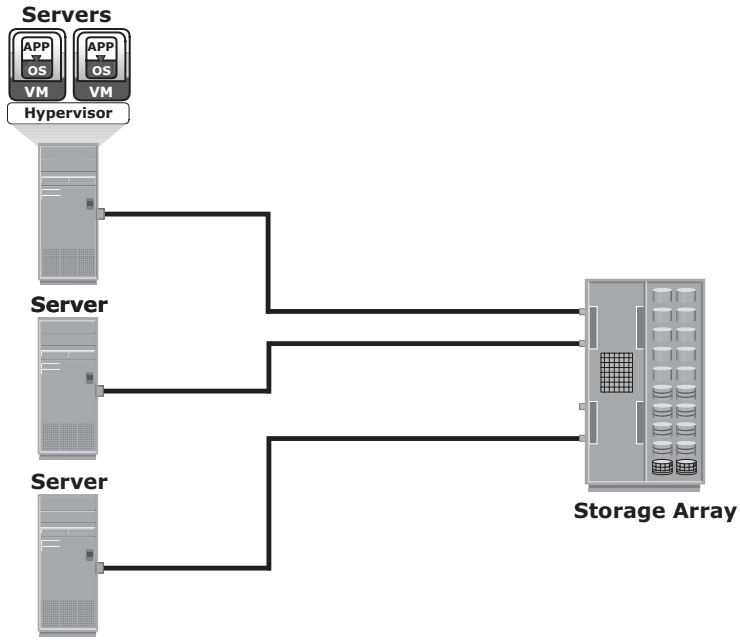


Figure 5-6: Point-to-point connectivity

5.4.2 Fibre Channel Arbitrated Loop

In the FC-AL configuration, devices are attached to a shared loop. 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 (see Figure 5-7).

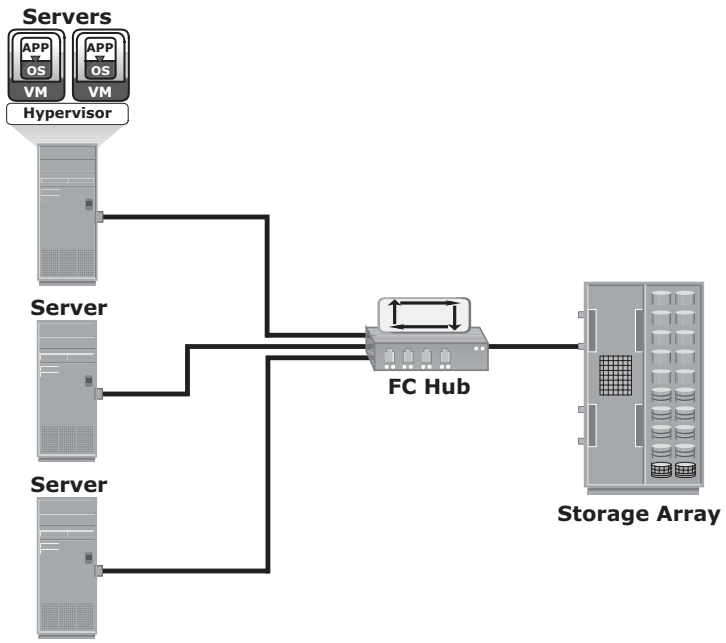


Figure 5-7: Fibre Channel Arbitrated Loop

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

However, 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 loop and only one device can perform I/O operations at a time. Because each device in a loop must wait for its turn to process an I/O request, the overall performance in FC-AL environments is low.
- FC-AL uses only 8-bits of 24-bit Fibre Channel addressing (the remaining 16-bits are masked) and enables the assignment of 127 valid addresses to the ports. Hence, it can support up to 127 devices on a loop. One address is reserved for optionally connecting the loop to an FC switch port. Therefore, up to 126 nodes can be connected to the loop.
- Adding or removing a device results in loop re-initialization, which can cause a momentary pause in loop traffic.

5.4.3 Fibre Channel Switched Fabric

Unlike a loop configuration, a Fibre Channel switched fabric (FC-SW) network provides dedicated data path 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 5-8 shows an example of the FC-SW fabric.

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.

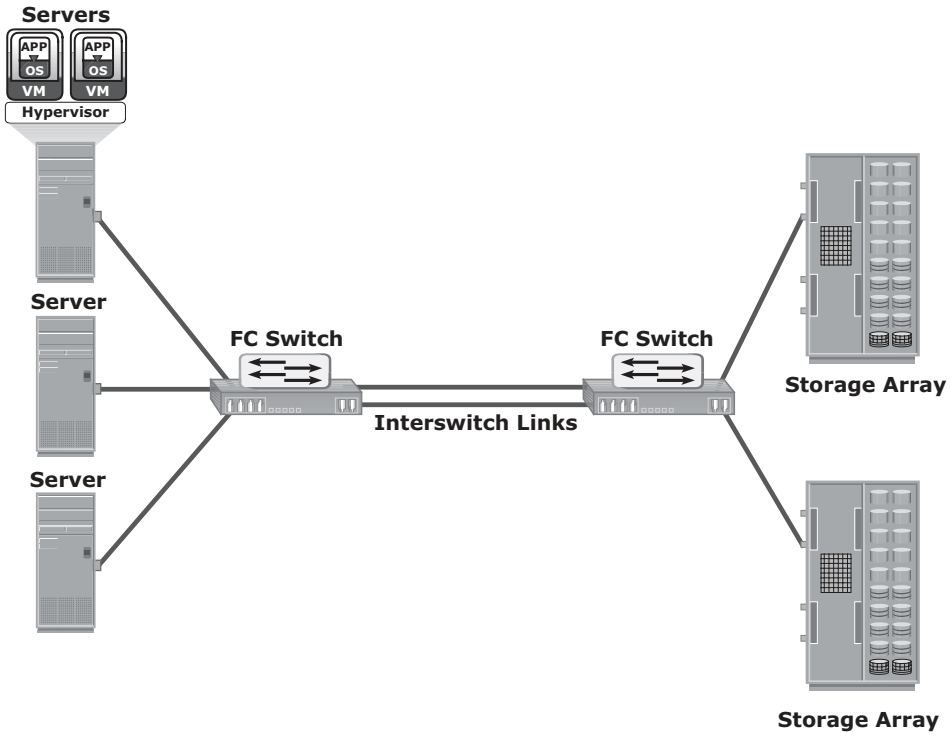


Figure 5-8: Fibre Channel switched fabric

A fabric can be described by the number of tiers it contains. The number of tiers in a fabric is based on the number of switches traversed between two points that are farthest from each other. This number is based on the infrastructure

constructed by the fabric instead of how the storage and server are connected across the switches.

When the number of tiers in a fabric increases, the distance that the fabric management traffic must travel to reach each switch also increases. This 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. Figure 5-9 illustrates two-tier and three-tier fabric architecture.

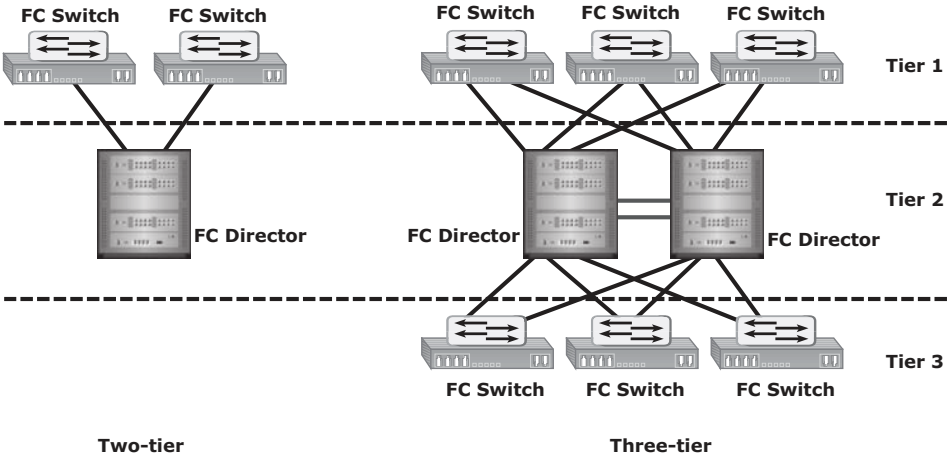


Figure 5-9: Tiered structure of Fibre Channel switched fabric

FC-SW Transmission

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

As shown in Figure 5-10, if node B wants to communicate with node D, the 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.

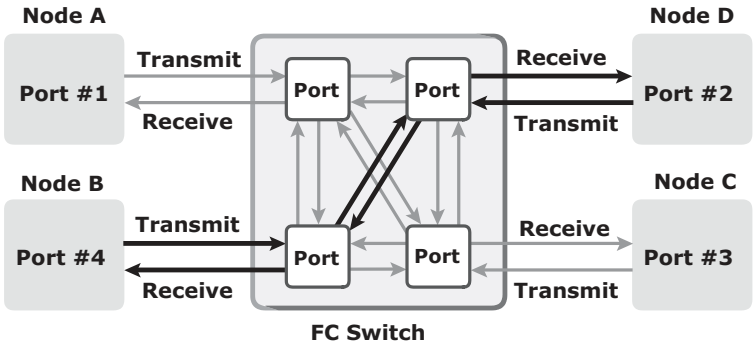


Figure 5-10: Data transmission in Fibre Channel switched fabric

5.5 Switched Fabric Ports

Ports in a switched fabric can be one of the following types:

- **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 connected to a switch in a switched fabric.
- **E_Port:** A 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 ISLs.
- **F_Port:** A port on a switch that connects an N_Port. It is also known as a *fabric port*.
- **G_Port:** A generic port on a switch that can operate as an E_Port or an F_Port and determines its functionality automatically during initialization.

Figure 5-11 shows various FC ports located in a switched fabric.

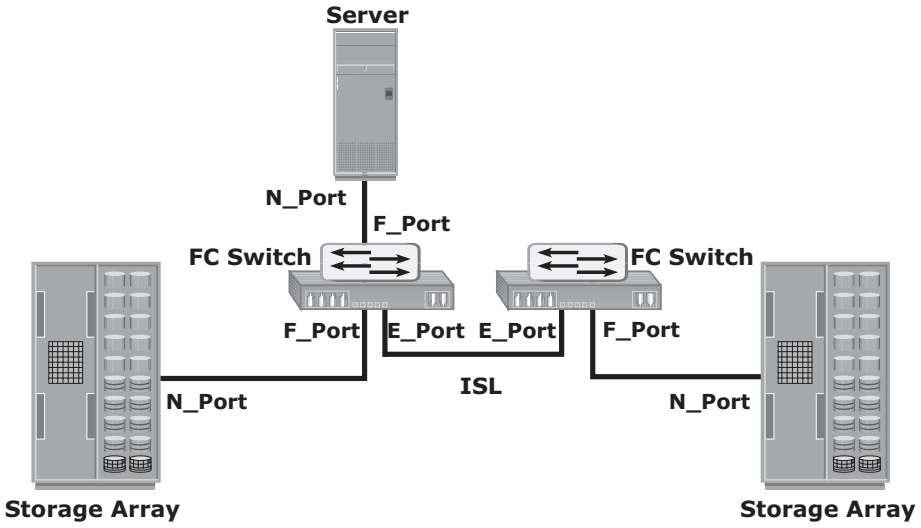


Figure 5-11: Switched fabric ports

5.6 Fibre Channel Architecture

Traditionally, host computer operating systems have communicated with peripheral devices over channel connections, such as ESCON and SCSI. Channel technologies provide high levels of performance with low protocol overheads. Such performance is achievable due to the static nature of channels and the high level of hardware and software integration provided by the channel technologies.

However, these technologies suffer from inherent limitations in terms of the number of devices that can be connected and the distance between these devices.

In contrast to channel technology, network technologies are more flexible and provide greater distance capabilities. Network connectivity provides greater scalability and uses shared bandwidth for communication. This flexibility results in greater protocol overhead and reduced performance.

The FC architecture represents true channel/network integration and captures some of the benefits of both channel and network technology. FC SAN uses the *Fibre Channel Protocol* (FCP) that provides both channel speed for data transfer with low protocol overhead and scalability of network technology.

FCP forms the fundamental construct of the FC SAN infrastructure. Fibre Channel provides a serial data transfer interface that operates over copper wire and optical fiber. FCP is the implementation of serial SCSI over an FC network. In 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 more than 15 million device addresses on a network.
- Support speeds up to 16 Gbps (16 GFC).

5.6.1 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 5-12 illustrates the Fibre Channel protocol stack.

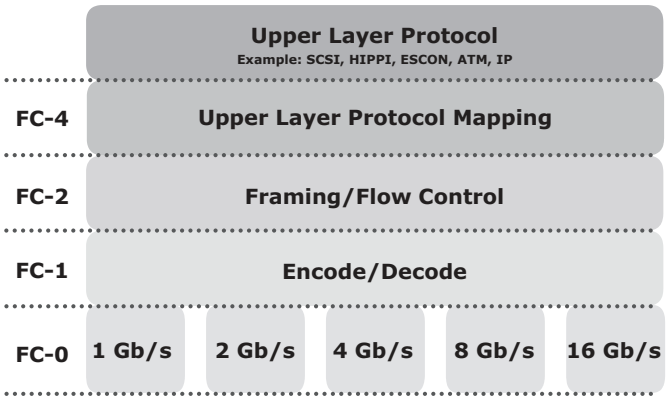


Figure 5-12: Fibre Channel protocol stack

FC-4 Layer

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 5-12). Some of the protocols include SCSI, High Performance Parallel Interface (HIPPI) Framing Protocol, Enterprise Storage Connectivity (ESCON), Asynchronous Transfer Mode (ATM), and IP.

FC-2 Layer

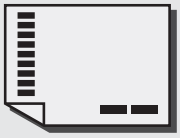
The FC-2 layer provides Fibre Channel addressing, structure, and organization of data (frames, sequences, and exchanges). It also defines fabric services, classes of service, flow control, and routing.

FC-1 Layer

The FC-1 layer defines how data is encoded prior to transmission and decoded upon receipt. 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 links with speeds of 10 Gbps and above use 64-bit to 66-bit encoding algorithms. The FC-1 layer also defines the transmission words, such as FC frame delimiters, which identify the start and end of a frame and primitive signals that indicate events at a transmitting port. In addition to these, the FC-1 layer performs link initialization and error recovery.

FC-0 Layer

FC-0 is the lowest layer in the FCP stack. This layer defines the physical interface, media, and transmission of 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.



Mainframe SANs use *Fibre Connectivity* (FICON) for a low-latency, high-bandwidth connection to the storage controller. FICON was designed as a replacement for *Enterprise System Connection* (ESCON) to support mainframe-attached storage systems.

5.6.2 Fibre Channel Addressing

An FC address is dynamically assigned when a node port logs on to the fabric. The FC address has a distinct format, as shown in Figure 5-13. The addressing mechanism provided here corresponds to the fabric with the switch as an interconnecting device.

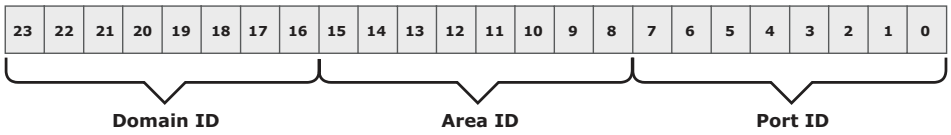


Figure 5-13: 24-bit FC address of N_Port

The first field of the FC address contains the domain ID of the switch. A *domain ID* is a unique number provided to each switch in the fabric. Although 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.

Therefore, the maximum possible number of node ports in a switched fabric is calculated as:

239 domains × 256 areas × 256 ports = 15,663,104

N_PORT ID VIRTUALIZATION (NPIV)

***NPIV* is a Fibre Channel configuration that enables multiple N_Port IDs to share a single physical N_Port. A typical use of NPIV would be for SAN storage provisioning to virtual machines in a virtualized server environment. With NPIV, several virtual machines on a host may share a common physical N_Port in the host, with each virtual machine using its own N_Port_ID for that physical node port. For this to work, the FC switch must be NPIV-enabled.**

5.6.3 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 node 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 5-14 illustrates the WWN structure examples for an array and an 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
Format Type	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
Format Type	Reserved 12 bits			Company ID 24 bits						Company Specific 24 bits					

Figure 5-14: World Wide Names

5.6.4 FC Frame

An FC frame (Figure 5-15) consists of five parts: *start of frame* (SOF), *frame header*, *data field*, *cyclic redundancy check* (CRC), and *end of frame* (EOF).

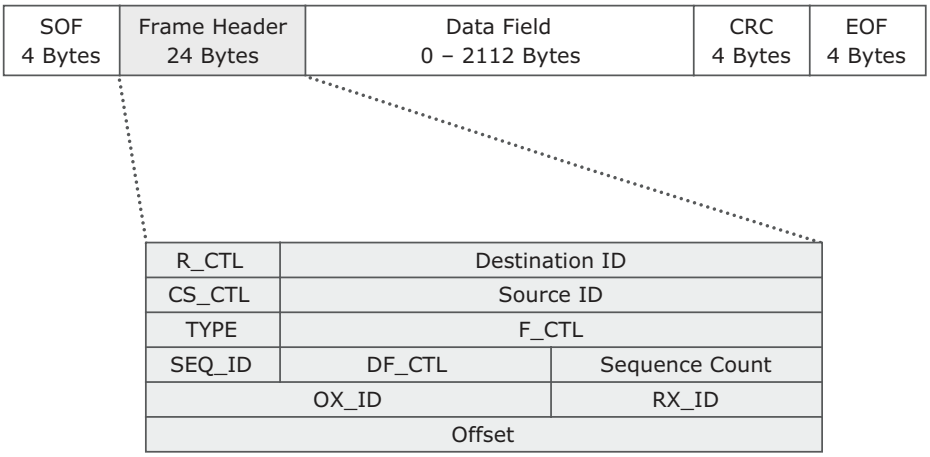


Figure 5-15: FC frame

The SOF and EOF act as delimiters. In addition to this role, the SOF also 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.

The S_ID and D_ID are FC addresses for the source port and the destination port, respectively. The SEQ_ID and OX_ID identify the frame as a component of a specific sequence and exchange, respectively.

The frame header also defines the following fields:

- **Routing Control (R_CTL):** This field denotes whether the frame is a link control frame or a data frame. Link control frames are frames that do not carry any user data. These frames are used for setup and messaging. In contrast, data frames carry the user data.
- **Class Specific Control (CS_CTL):** This field specifies link speeds for class 1 and class 4 data transmission. (Class of service is discussed in section 5.6.7 "Classes of Service" later in the chapter.)
- **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.
- **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.
- **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.

The data field in an FC frame contains the data payload, up to 2,112 bytes of actual data with 36 bytes of fixed overhead.

The CRC checksum facilitates error detection for the content of the frame. This checksum verifies data integrity by checking whether the content of the frames are received correctly. The CRC checksum is calculated by the sender before encoding at the FC-1 layer. Similarly, it is calculated by the receiver after decoding at the FC-1 layer.

5.6.5. 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.

- **Exchange:** An exchange operation enables two node ports to identify and manage a set of information units. Each upper layer protocol has its protocol-specific information that must be sent to another port to perform certain operations. This protocol-specific information is called an information unit. The structure of these information units is defined in the FC-4 layer. This unit maps to a sequence. An exchange is composed of one or more sequences.
- **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.
- **Frame:** A frame is the fundamental unit of data transfer at Layer 2. Each frame can contain up to 2,112 bytes of payload.

5.6.6 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).

BB_Credit

FC uses the *BB_Credit* mechanism for 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 uses *Receiver Ready* (R_RDY) primitive that indicates a buffer has been freed on the port that transmitted the R_RDY.

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 provides the flow control for class 1 and class 2 traffic only.

5.6.7 Classes of Service

The FC standards define different classes of service to meet the requirements of a wide range of applications. Table 5-1 shows three classes of services and their features.

Table 5-1: 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 acknowledgment	Acknowledged	Acknowledged	Not acknowledged
Multiplexing	No	Yes	Yes
Bandwidth utilization	Poor	Moderate	High

Another class of service is *class F*, which is used for fabric management. Class F is similar to Class 2 and provides notification of nondelivery of frames.

5.7 Fabric Services

All FC switches, regardless of the manufacturer, provide a common set of services as defined in the Fibre Channel standards. These services are available at certain predefined addresses. Some of these services are Fabric Login Server, Fabric Controller, Name Server, and Management Server (see Figure 5-16).

The *Fabric Login Server* is located at the predefined address of FFFFFE and is used during the initial part of the node's fabric login process.

The *Name Server* (formally known as *Distributed Name Server*) is located at the predefined address FFFFFC and is responsible for name registration and management of node ports. Each switch exchanges its Name Server information with other switches in the fabric to maintain a synchronized, distributed name service.

Each switch has a *Fabric Controller* located at the predefined address FFFFFD. The Fabric Controller provides services to both node ports and other switches. The Fabric Controller is responsible for managing and distributing Registered State Change Notifications (RSCNs) to the node ports registered with the

Chapter 6
IP SAN and FCoE

KEY CONCEPTS

- iSCSI Protocol
- Native and Bridged iSCSI
- FCIP Protocol
- FCoE Protocol

Traditional SAN enables the transfer of block I/O over Fibre Channel and provides high performance and scalability. These advantages of FC SAN come with the additional cost of buying FC components, such as FC HBA and switches. Organizations typically have an existing Internet Protocol (IP)-based infrastructure, which could be leveraged for storage networking. Advancements in technology have enabled IP to be used for transporting block I/O over the IP network. This technology of transporting block I/Os over an IP is referred to as IP SAN. IP is a mature technology, and using IP as a storage networking option provides several advantages. When block I/O is run over IP, the existing network infrastructure can be leveraged, which is more economical than investing in a new SAN infrastructure. In addition, many robust and mature security options are now available for IP networks. Many long-distance, disaster recovery (DR) solutions are already leveraging IP-based networks. With IP SAN, organizations can extend the geographical reach of their storage infrastructure.

Two primary protocols that leverage IP as the transport mechanism are *Internet SCSI* (iSCSI) and *Fibre Channel over IP* (FCIP). iSCSI is encapsulation of SCSI I/O over IP. FCIP is a protocol in which an FCIP entity such as an FCIP gateway is used to tunnel FC fabrics through an IP network. In FCIP, FC frames are encapsulated onto the IP payload. An FCIP implementation is capable of merging interconnected fabrics into a single fabric. Frequently, only a small subset of nodes at either end require connectivity across fabrics. Thus, the majority of FCIP implementations today use switch-specific features such as IVR (Inter-VSAN Routing) or FCRS

(Fibre Channel Routing Services) to create a tunnel. In this manner, traffic may be routed between specific nodes without actually merging the fabrics.

This chapter describes both iSCSI and FCIP protocols, components, and topologies in detail. This chapter also covers an emerging protocol, *Fibre Channel over Ethernet* (FCoE). FCoE converges Ethernet and FC traffic over a single physical link. Therefore, it eliminates the complexity of managing two separate networks in the data center.

6.1 iSCSI

iSCSI is an IP based protocol that establishes and manages connections between host and storage over IP, as shown in Figure 6-1. iSCSI encapsulates SCSI commands and data into an IP packet and transports them using TCP/IP. iSCSI is widely adopted for connecting servers to storage because it is relatively inexpensive and easy to implement, especially in environments in which an FC SAN does not exist.

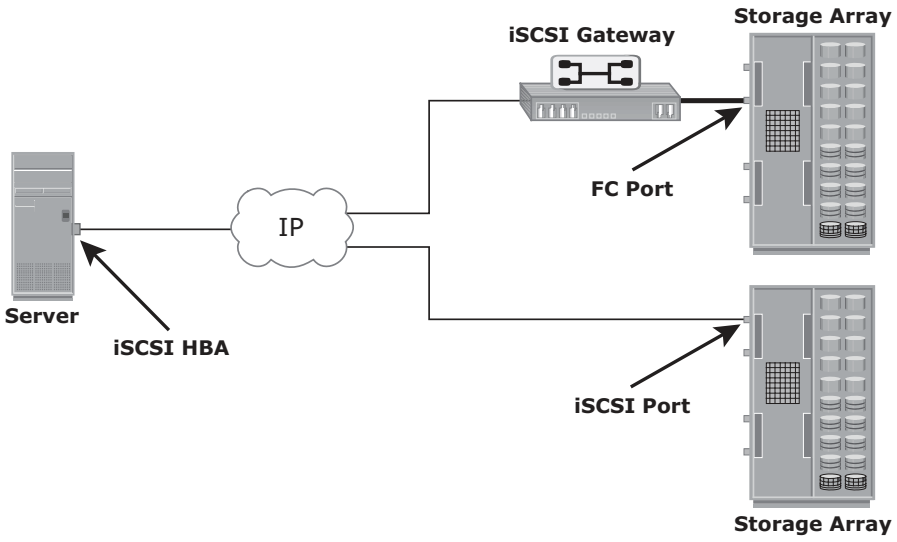


Figure 6-1: iSCSI implementation

6.1.1 Components of iSCSI

An initiator (host), target (storage or iSCSI gateway), and an IP-based network are the key iSCSI components. If an iSCSI-capable storage array is deployed, then a host with the iSCSI initiator can directly communicate with the storage array over an IP network. However, in an implementation that uses an existing FC array for iSCSI communication, an iSCSI gateway is used. These devices perform

the translation of IP packets to FC frames and vice versa, thereby bridging the connectivity between the IP and FC environments.

6.1.2 iSCSI Host Connectivity

A standard NIC with software iSCSI initiator, a TCP offload engine (TOE) NIC with software iSCSI initiator, and an iSCSI HBA are the three iSCSI host connectivity options. The function of the iSCSI initiator is to route the SCSI commands over an IP network.

A standard NIC with a software iSCSI initiator is the simplest and least expensive connectivity option. It is easy to implement because most servers come with at least one, and in many cases two, embedded NICs. It requires only a software initiator for iSCSI functionality. Because NICs provide standard IP function, encapsulation of SCSI into IP packets and decapsulation are carried out by the host CPU. This places additional overhead on the host CPU. If a standard NIC is used in heavy I/O load situations, the host CPU might become a bottleneck. TOE NIC helps alleviate this burden. A TOE NIC offloads TCP management functions from the host and leaves only the iSCSI functionality to the host processor. The host passes the iSCSI information to the TOE card, and the TOE card sends the information to the destination using TCP/IP. Although this solution improves performance, the iSCSI functionality is still handled by a software initiator that requires host CPU cycles.

An iSCSI HBA is capable of providing performance benefits because it offloads the entire iSCSI and TCP/IP processing from the host processor. The use of an iSCSI HBA is also the simplest way to boot hosts from a SAN environment via iSCSI. If there is no iSCSI HBA, modifications must be made to the basic operating system to boot a host from the storage devices because the NIC needs to obtain an IP address before the operating system loads. The functionality of an iSCSI HBA is similar to the functionality of an FC HBA.

6.1.3 iSCSI Topologies

Two topologies of iSCSI implementations are native and bridged. *Native topology* does not have FC components. The initiators may be either directly attached to targets or connected through the IP network. *Bridged topology* enables the coexistence of FC with IP by providing iSCSI-to-FC bridging functionality. For example, the initiators can exist in an IP environment while the storage remains in an FC environment.

Native iSCSI Connectivity

FC components are not required for iSCSI connectivity if an iSCSI-enabled array is deployed. In Figure 6-2 (a), the array has one or more iSCSI ports configured with an IP address and is connected to a standard Ethernet switch.

After an initiator is logged on to the network, it can access the available LUNs on the storage array. A single array port can service multiple hosts or initiators as long as the array port can handle the amount of storage traffic that the hosts generate.

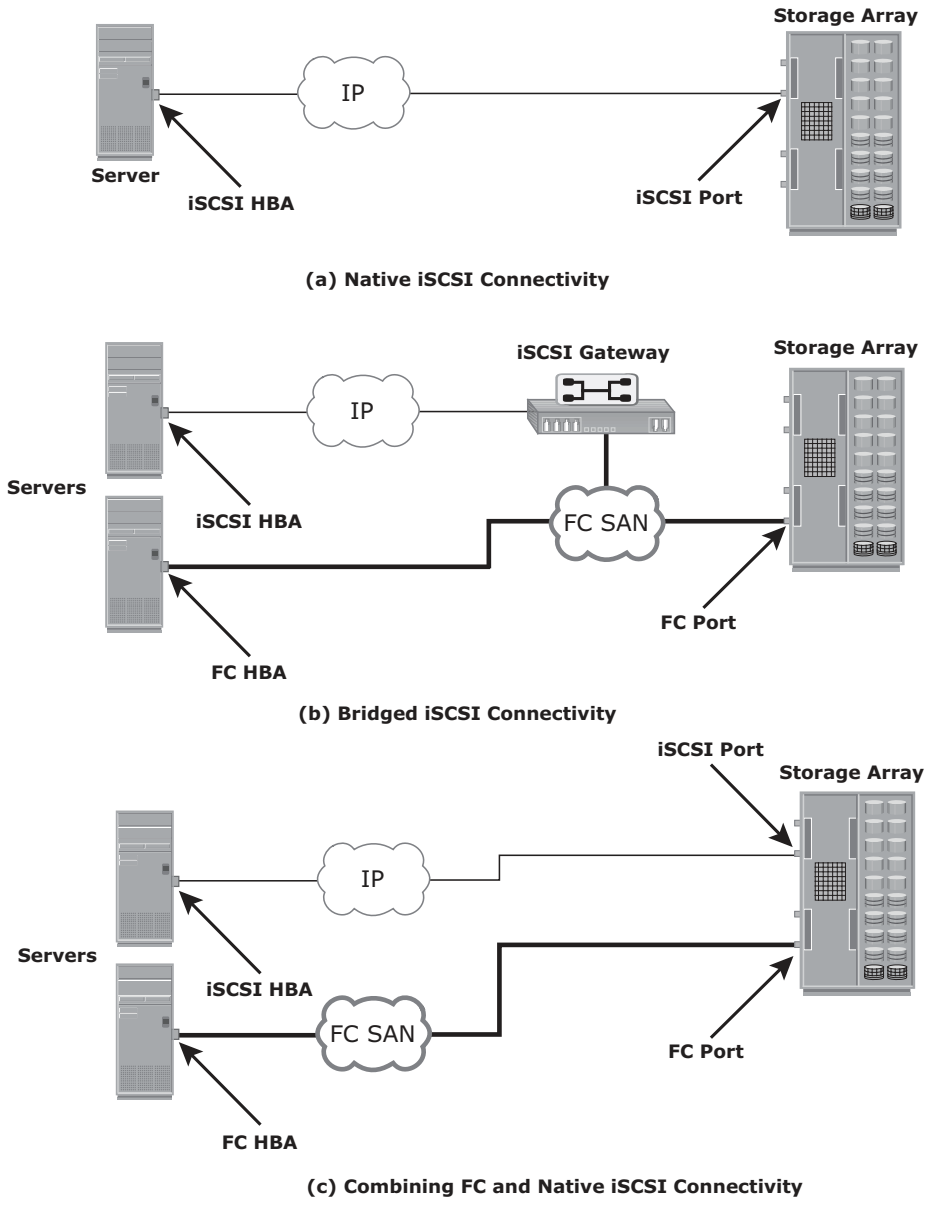


Figure 6-2: iSCSI Topologies

Bridged iSCSI Connectivity

A bridged iSCSI implementation includes FC components in its configuration. Figure 6-2 (b) illustrates iSCSI host connectivity to an FC storage array.

In this case, the array does not have any iSCSI ports. Therefore, an external device, called a gateway or a multiprotocol router, must be used to facilitate the communication between the iSCSI host and FC storage. The gateway converts IP packets to FC frames and vice versa. The bridge devices contain both FC and Ethernet ports to facilitate the communication between the FC and IP environments.

In a bridged iSCSI implementation, the iSCSI initiator is configured with the gateway's IP address as its target destination. On the other side, the gateway is configured as an FC initiator to the storage array.

Combining FC and Native iSCSI Connectivity

The most common topology is a combination of FC and native iSCSI. Typically, a storage array comes with both FC and iSCSI ports that enable iSCSI and FC connectivity in the same environment, as shown in Figure 6-2 (c).

6.1.4 iSCSI Protocol Stack

Figure 6-3 displays a model of the iSCSI protocol layers and depicts the encapsulation order of the SCSI commands for their delivery through a physical carrier.

SCSI is the command protocol that works at the application layer of the Open System Interconnection (OSI) model. The initiators and targets use SCSI commands and responses to talk to each other. The SCSI command descriptor blocks, data, and status messages are encapsulated into TCP/IP and transmitted across the network between the initiators and targets.

iSCSI is the session-layer protocol that initiates a reliable session between devices that recognize SCSI commands and TCP/IP. The iSCSI session-layer interface is responsible for handling login, authentication, target discovery, and session management. TCP is used with iSCSI at the transport layer to provide reliable transmission.

TCP controls message flow, windowing, error recovery, and retransmission. It relies upon the network layer of the OSI model to provide global addressing and connectivity. The Layer 2 protocols at the data link layer of this model enable node-to-node communication through a physical network.

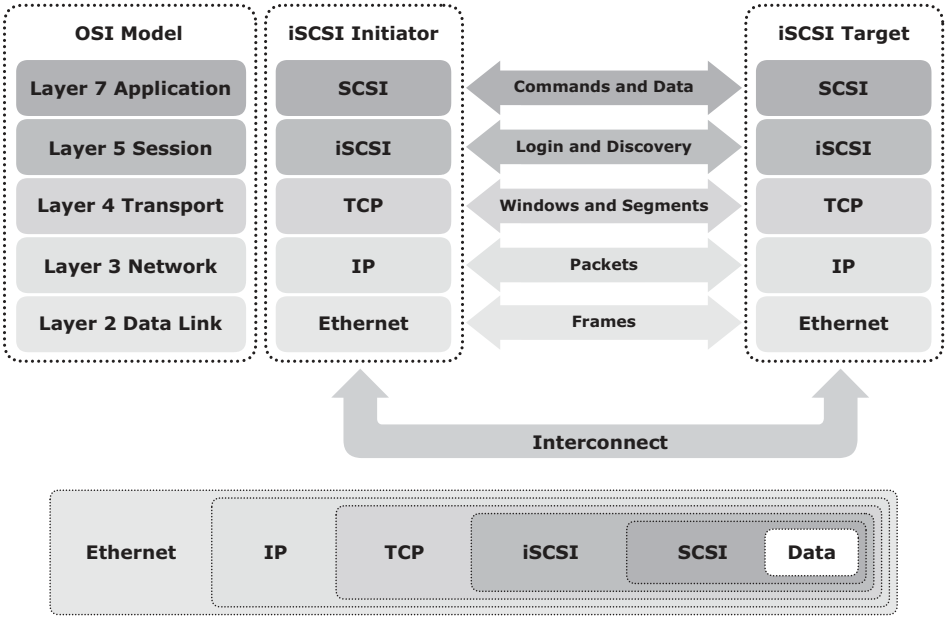


Figure 6-3: iSCSI protocol stack

6.1.5 iSCSI PDU

A *protocol data unit* (PDU) is the basic "information unit" in the iSCSI environment. The iSCSI initiators and targets communicate with each other using iSCSI PDUs. This communication includes establishing iSCSI connections and iSCSI sessions, performing iSCSI discovery, sending SCSI commands and data, and receiving SCSI status. All iSCSI PDUs contain one or more header segments followed by zero or more data segments. The PDU is then encapsulated into an IP packet to facilitate the transport.

A PDU includes the components shown in Figure 6-4. The IP header provides packet-routing information to move the packet across a network. The TCP header contains the information required to guarantee the packet delivery to the target. The iSCSI header (basic header segment) describes how to extract SCSI commands and data for the target. iSCSI adds an optional CRC, known as the *digest*, to ensure datagram integrity. This is in addition to TCP checksum and Ethernet CRC. The header and the data digests are optionally used in the PDU to validate integrity and data placement.

As shown in Figure 6-5, each iSCSI PDU does not correspond in a 1:1 relationship with an IP packet. Depending on its size, an iSCSI PDU can span an IP packet or even coexist with another PDU in the same packet.

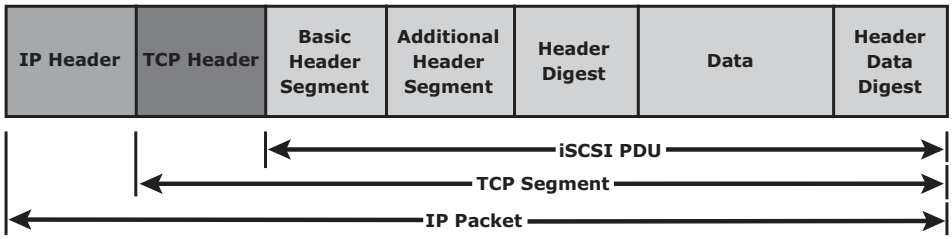
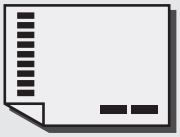


Figure 6-4: iSCSI PDU encapsulated in an IP packet



A message transmitted on a network is divided into a number of packets. If necessary, each packet can be sent by a different route across the network. Packets can arrive in a different order than the order in which they were sent. IP only delivers them; it is up to TCP to organize them in the right sequence. The target extracts the SCSI commands and data on the basis of the information in the iSCSI header.

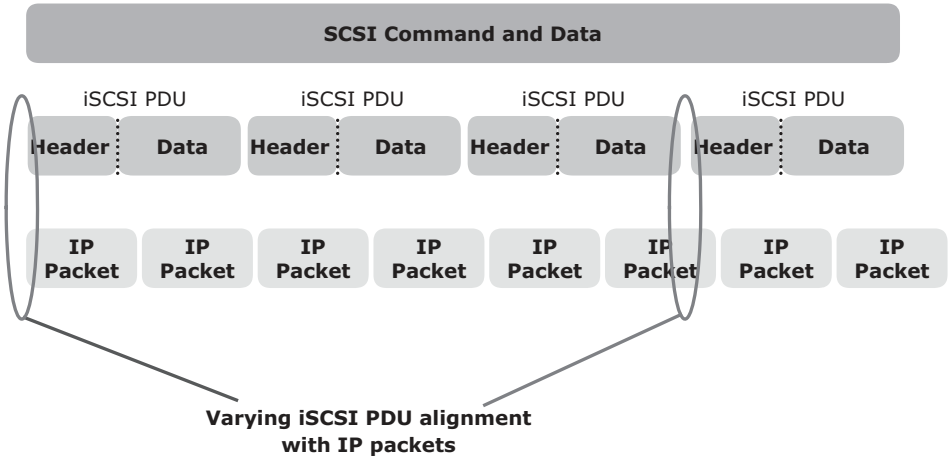


Figure 6-5: Alignment of iSCSI PDUs with IP packets

To achieve the 1:1 relationship between the IP packet and the iSCSI PDU, the maximum transmission unit (MTU) size of the IP packet is modified. This eliminates fragmentation of the IP packet, which improves the transmission efficiency.

6.1.6 iSCSI Discovery

An initiator must discover the location of its targets on the network and the names of the targets available to it before it can establish a session. This discovery can take place in two ways: *SendTargets discovery* or *internet Storage Name Service* (iSNS).

In *SendTargets discovery*, the initiator is manually configured with the target's network portal to establish a discovery session. The initiator issues the `SendTargets` command, and the target network portal responds with the names and addresses of the targets available to the host.

iSNS (see Figure 6-6) enables automatic discovery of iSCSI devices on an IP network. The initiators and targets can be configured to automatically register themselves with the iSNS server. Whenever an initiator wants to know the targets that it can access, it can query the iSNS server for a list of available targets.

The discovery can also take place by using service location protocol (SLP). However, this is less commonly used than `SendTargets` discovery and iSNS.

6.1.7 iSCSI Names

A unique worldwide iSCSI identifier, known as an *iSCSI name*, is used to identify the initiators and targets within an iSCSI network to facilitate communication. The unique identifier can be a combination of the names of the department, application, or manufacturer, serial number, asset number, or any tag that can be used to recognize and manage the devices. Following are two types of iSCSI names commonly used:

- **iSCSI Qualified Name (IQN):** An organization must own a registered domain name to generate iSCSI Qualified Names. This domain name does not need to be active or resolve to an address. It just needs to be reserved to prevent other organizations from using the same domain name to generate iSCSI names. A date is included in the name to avoid potential conflicts caused by the transfer of domain names. An example of an IQN is `iqn.2008-02.com.example:`*`optional_string`*.

  The *optional_string* provides a serial number, an asset number, or any other device identifiers. An iSCSI Qualified Name enables storage administrators to assign meaningful names to iSCSI devices, and therefore, manage those devices more easily.

- **Extended Unique Identifier (EUI):** An EUI is a globally unique identifier based on the IEEE EUI-64 naming standard. An EUI is composed of the eui prefix followed by a 16-character hexadecimal name, such as `eui.0300732A32598D26`.

In either format, the allowed special characters are dots, dashes, and blank spaces.

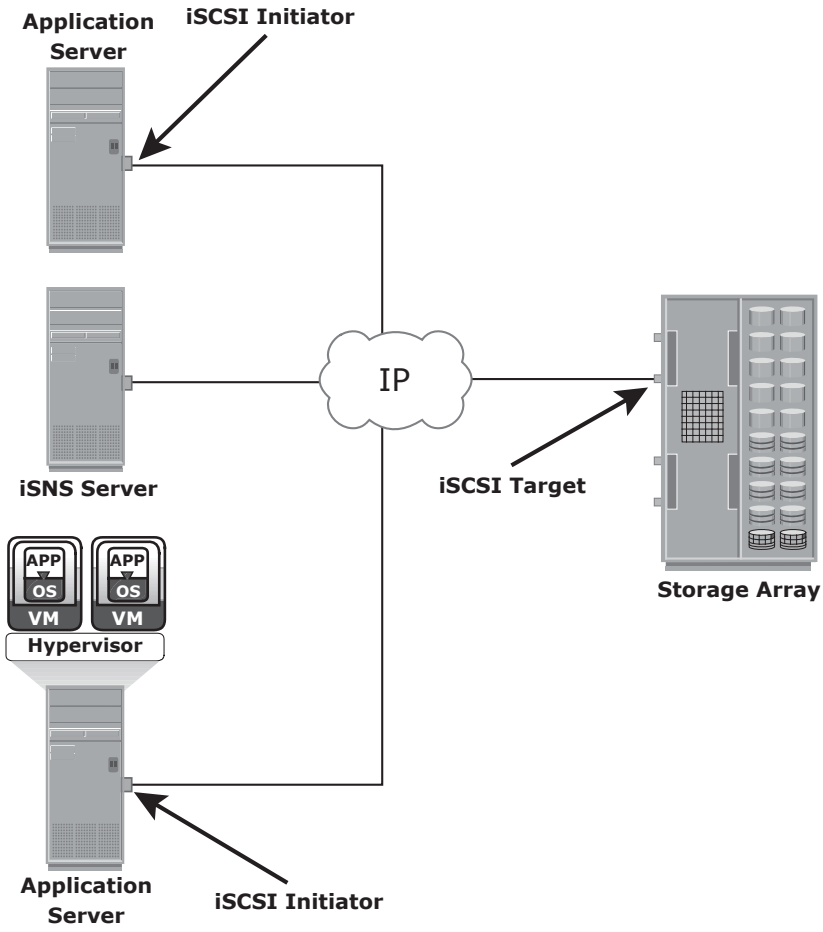


Figure 6-6: Discovery using iSNS

Chapter 7
Network-Attached Storage

KEY CONCEPTS

- NAS Devices
- Network File Sharing
- Unified, Gateway, and Scale-Out NAS
- NAS Connectivity and Protocols
- NAS Performance
- MTU and Jumbo Frames
- TCP Window and Link Aggregation
- File-Level Virtualization

File sharing, as the name implies, enables users to share files with other users. Traditional methods of file sharing involve copying files to portable media such as floppy diskette, CD, DVD, or USB drives and delivering them to other users with whom it is being shared. However, this approach is not suitable in an enterprise environment in which a large number of users at different locations need access to common files.

Network-based file sharing provides the flexibility to share files over long distances among a large number of users. File servers use client-server technology to enable file sharing over a network. To address the tremendous growth of file data in enterprise environments, organizations have been deploying large numbers of file servers. These servers are either connected to direct-attached storage (DAS) or storage area network (SAN)-attached storage. This has resulted in the proliferation of islands of over-utilized and under-utilized file servers and storage. In

addition, such environments have poor scalability, higher management cost, and greater complexity. *Network-attached storage* (NAS) emerged as a solution to these challenges.

NAS is a dedicated, high-performance file sharing and storage device. NAS enables its clients to share files over an IP network. NAS provides the advantages of server consolidation by eliminating the need for multiple file servers. It also consolidates the storage used by the clients onto a single system, making it easier to manage the storage. NAS uses network and file-sharing protocols to provide access to the file data. These protocols include TCP/IP for data transfer, and Common Internet File System (CIFS) and Network File System (NFS) for network file service. NAS enables both UNIX and Microsoft Windows users to share the same data seamlessly.

A NAS device uses its own operating system and integrated hardware and software components to meet specific file-service needs. Its operating system is optimized for file I/O and, therefore, performs file I/O better than a general-purpose server. As a result, a NAS device can serve more clients than general-purpose servers and provide the benefit of server consolidation.

A network-based file sharing environment is composed of multiple file servers or NAS devices. It might be required to move the files from one device to another due to reasons such as cost or performance. File-level virtualization, implemented in the file sharing environment, provides a simple, nondisruptive file-mobility solution. It enables the movement of files across NAS devices, even if the files are being accessed.

This chapter describes the components of NAS, different types of NAS implementations, and the file-sharing protocols used in NAS implementations. The chapter also explains factors that affect NAS performance, and file-level virtualization.

7.1 General-Purpose Servers versus 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 because it runs a general-purpose operating system. Unlike a general-purpose server, a NAS device is dedicated to file-serving. It has specialized operating system dedicated to file serving by using industry-standard protocols. Some NAS vendors support features, such as native clustering for high availability.

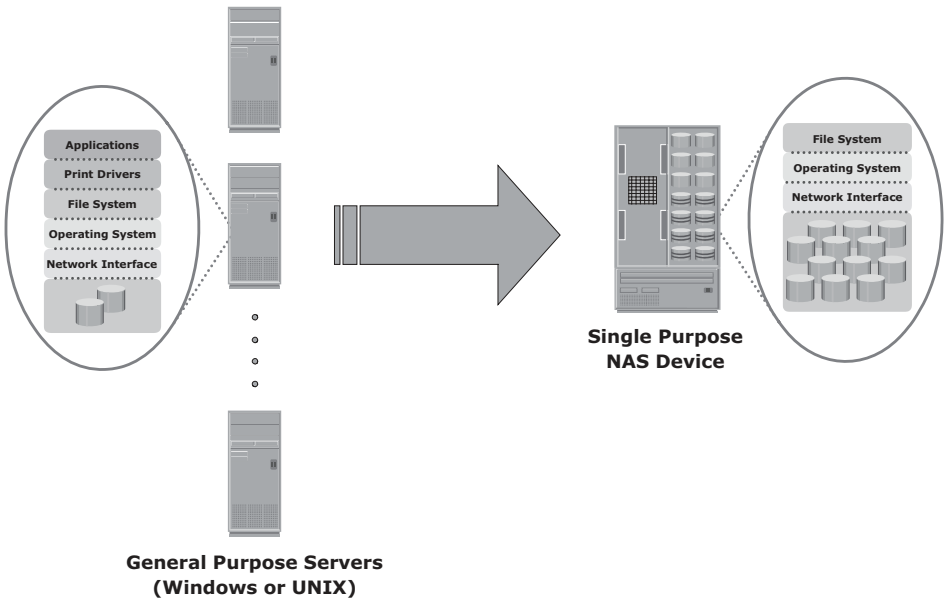


Figure 7-1: General purpose server versus NAS device

7.2 Benefits of NAS

NAS offers the following benefits:

- **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:** NAS delivers better performance compared to a general-purpose file server because NAS uses an operating system specialized for file serving.
- **Improved flexibility:** Compatible with 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, and ensure greater data protection
- **Simplified management:** Provides a centralized console that makes it possible to manage file systems efficiently

- **Scalability:** Scales well 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 components that provide maximum connectivity options. A NAS device supports clustering technology for failover.
- **Security:** Ensures security, user authentication, and file locking with industry-standard security schemas
- **Low cost:** NAS uses commonly available and inexpensive Ethernet components.
- **Ease of deployment:** Configuration at the client is minimal, because the clients have required NAS connection software built in.

7.3 File Systems and Network File Sharing

A *file system* is a structured way to store and organize data files. Many file systems maintain a file access table to simplify the process of searching and accessing files.

7.3.1 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 on the NAS and the operating system on the client. When mounting a file system, the operating system organizes files and directories in a tree-like structure and grants the privilege to the user to access this structure. The tree is rooted at a mount point. The mount point is named using operating system conventions. Users and applications can traverse the entire tree from the root to the leaf nodes as file system permissions allow. Files are located at leaf nodes, and directories and subdirectories are located at intermediate roots. The access to the file system terminates when the file system is unmounted. Figure 7-2 shows an example of a UNIX directory structure.

7.3.2 Network File Sharing

Network file sharing refers to storing and accessing files over a network. In a file-sharing environment, the user who creates a file (the creator or owner of a file) determines the type of access (such as read, write, execute, append, and

delete) to be given to other users and controls changes to the file. When multiple users try to access a shared file at the same time, a locking scheme is required to maintain data integrity and, at the same time, make this sharing possible.

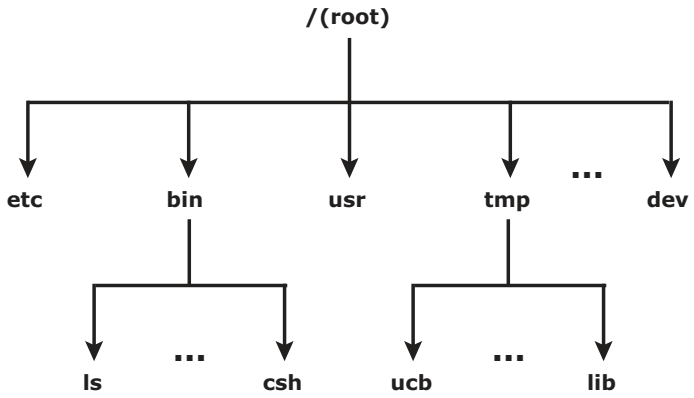


Figure 7-2: UNIX directory structure

Some examples of file-sharing methods are file transfer protocol (FTP), Distributed File System (DFS), client-server models that use file-sharing protocols such as NFS and CIFS, and the peer-to-peer (P2P) model

FTP is a client-server protocol that enables data transfer over a network. An FTP server and an FTP client communicate with each other using TCP as the transport protocol. FTP, as defined by the standard, is not a secure method of data transfer because it uses unencrypted data transfer over a network. FTP over Secure Shell (SSH) adds security to the original FTP specification. When FTP is used over SSH, it is referred to as Secure FTP (SFTP).

A *distributed file system* (DFS) is a file system that is distributed across several hosts. A DFS can provide hosts with direct access to the entire file system, while ensuring efficient management and data security. Standard client-server file-sharing protocols, such as NFS and CIFS, enable the owner of a file to set the required type of access, such as read-only or read-write, for a particular user or group of users. Using this protocol, the clients mount remote file systems that are available on dedicated file servers.

A name service, such as Domain Name System (DNS), and directory services such as Microsoft Active Directory, and Network Information Services (NIS), helps users identify and access a unique resource over the network. A *name service protocol* such as the Lightweight Directory Access Protocol (LDAP) creates a namespace, which holds the unique name of every network resource and helps recognize resources on the network.

A *peer-to-peer* (P2P) file sharing model uses a peer-to-peer network. P2P enables client machines to directly share files with each other over a

network. Clients use a file sharing software that searches for other peer clients. This differs from the client-server model that uses file servers to store files for sharing.

7.4 Components of NAS

A NAS device has two key components: NAS head and storage (see Figure 7-3). In some NAS implementations, the storage could be external to the NAS device and shared with other hosts. The NAS head includes the following components:

- CPU and memory
- One or more network interface cards (NICs), which provide connectivity to the client network. Examples of network protocols supported by NIC include Gigabit Ethernet, Fast Ethernet, ATM, and Fiber Distributed Data Interface (FDDI).
- An optimized operating system for managing the NAS functionality. It translates file-level requests into block-storage requests and further converts the data supplied at the block level to file data.
- NFS, CIFS, and other protocols for file sharing
- Industry-standard storage protocols and ports to connect and manage physical disk resources

The NAS environment includes clients accessing a NAS device over an IP network using file-sharing protocols.

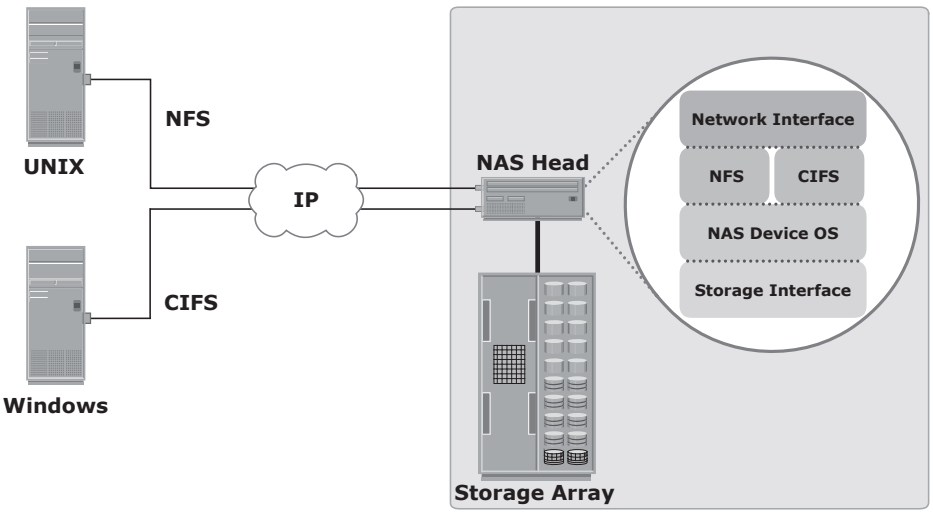


Figure 7-3: Components of NAS

7.5 NAS I/O Operation

NAS provides file-level data access to its clients. File I/O is a high-level request that specifies the file to be accessed. For example, a client may request a file by specifying its name, location, or other attributes. 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 process of handling I/Os in a NAS environment is as follows:

1. The requestor (client) packages an I/O request into TCP/IP and forwards it through the network stack. The NAS device receives this request from the network.
2. The NAS device converts the I/O request into an appropriate physical storage request, which is a block-level I/O, and then performs the operation on the physical storage.
3. When the NAS device receives data from the storage, it processes and repackages the data into an appropriate file protocol response.
4. The NAS device packages this response into TCP/IP again and forwards it to the client through the network.

Figure 7-4 illustrates this process.

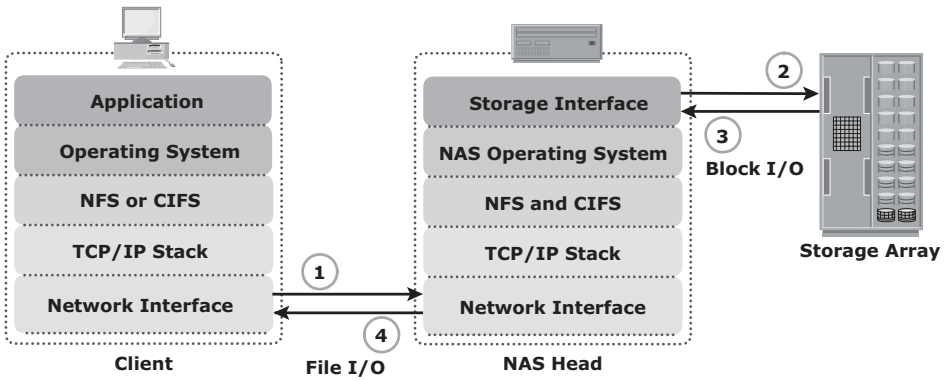


Figure 7-4: NAS I/O operation

7.6 NAS Implementations

Three common NAS implementations are unified, gateway, and scale-out. The *unified* NAS consolidates NAS-based and SAN-based data access within a unified storage platform and provides a unified management interface for managing both the environments.

In a *gateway* implementation, the NAS device uses external storage to store and retrieve data, and unlike unified storage, there are separate administrative tasks for the NAS device and storage.

The *scale-out* NAS implementation pools multiple nodes together in a cluster. A node may consist of either the NAS head or storage or both. The cluster performs the NAS operation as a single entity.

7.6.1 Unified NAS

Unified NAS performs file serving and storing of file data, along with providing access to block-level data. It supports both CIFS and NFS protocols for file access and iSCSI and FC protocols for block level access. Due to consolidation of NAS-based and SAN-based access on a single storage platform, unified NAS reduces an organization's infrastructure and management costs.

A unified NAS contains one or more NAS heads and storage in a single system. NAS heads are connected to the storage controllers (SCs), which provide access to the storage. These storage controllers also provide connectivity to iSCSI and FC hosts. The storage may consist of different drive types, such as SAS, ATA, FC, and flash drives, to meet different workload requirements.

7.6.2 Unified NAS Connectivity

Each NAS head in a unified NAS has front-end Ethernet ports, which connect to the IP network. The front-end ports provide connectivity to the clients and service the file I/O requests. Each NAS head has back-end ports, to provide connectivity to the storage controllers.

iSCSI and FC ports on a storage controller enable hosts to access the storage directly or through a storage network at the block level. Figure 7-5 illustrates an example of unified NAS connectivity.

7.6.3 Gateway NAS

A gateway NAS device consists of one or more NAS heads and uses external and independently managed storage. Similar to unified NAS, the storage is shared with other applications that use block-level I/O. Management functions in this type of solution are more complex than those in a unified NAS environment because there are separate administrative tasks for the NAS head and the storage. A gateway solution can use the FC infrastructure, such as switches and directors for accessing SAN-attached storage arrays or direct-attached storage arrays.

The gateway NAS is more scalable compared to unified NAS because NAS heads and storage arrays can be independently scaled up when required.

For example, NAS heads can be added to scale up the NAS device performance. When the storage limit is reached, it can scale up, adding capacity on the SAN, independent of NAS heads. Similar to a unified NAS, a gateway NAS also enables high utilization of storage capacity by sharing it with the SAN environment.

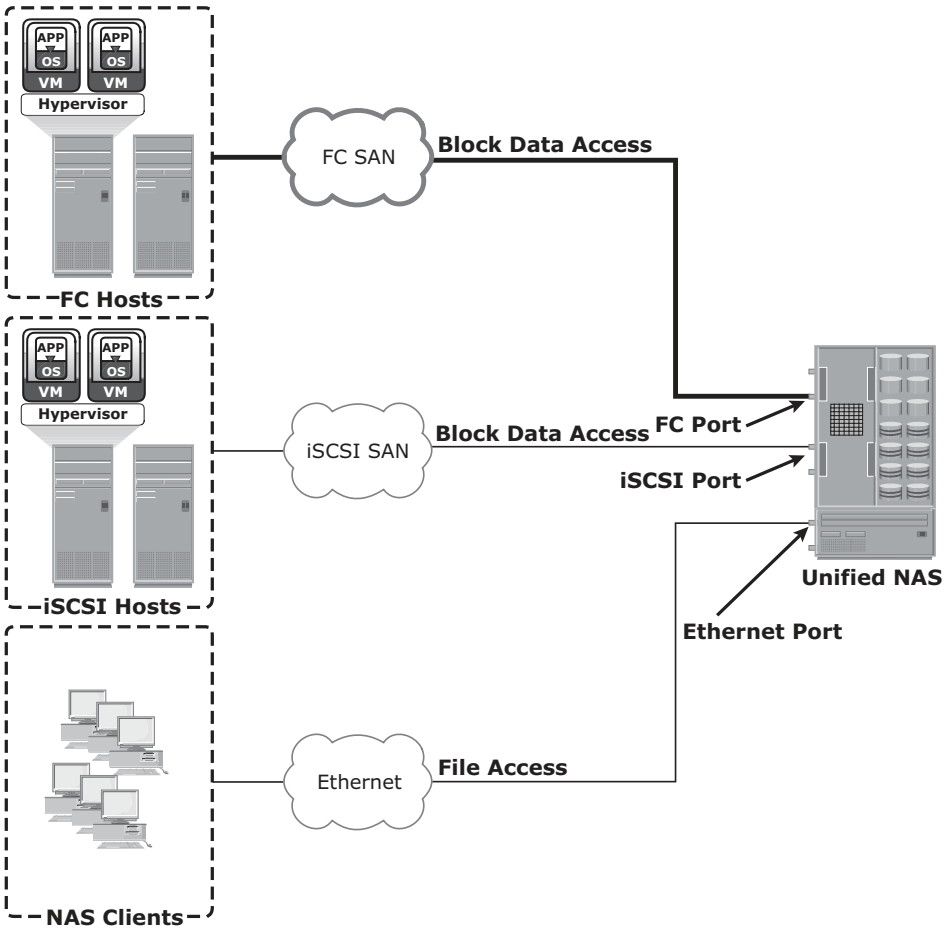


Figure 7-5: Unified NAS connectivity

7.6.4 Gateway NAS Connectivity

In a gateway solution, the front-end connectivity is similar to that in a unified storage solution. Communication between the NAS gateway and the storage system in a gateway solution is achieved through a traditional FC SAN. To deploy a gateway NAS solution, factors, such as multiple paths for data, redundant

fabrics, and load distribution, must be considered. Figure 7-6 illustrates an example of gateway NAS connectivity.

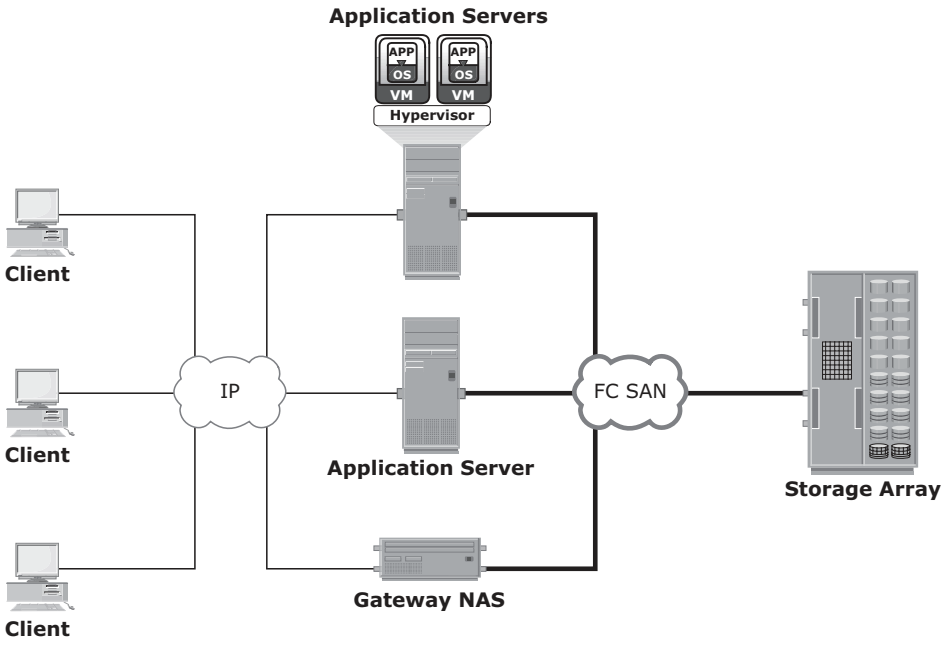


Figure 7-6: Gateway NAS connectivity

Implementation of both unified and gateway solutions requires analysis of the SAN environment. This analysis is required to determine the feasibility of combining the NAS workload with the SAN workload. Analyze the SAN to determine whether the workload is primarily read or write, and if it is random or sequential. Also determine the predominant I/O size in use. Typically, NAS workloads are random with small I/O sizes. Introducing sequential workload with random workloads can be disruptive to the sequential workload. Therefore, it is recommended to separate the NAS and SAN disks. Also, determine whether the NAS workload performs adequately with the configured cache in the storage system.

7.6.5 Scale-Out NAS

Both unified and gateway NAS implementations provide the capability to scale-up their resources based on data growth and rise in performance requirements. Scaling up these NAS devices involves adding CPUs, memory, and storage to

the NAS device. Scalability is limited by the capacity of the NAS device to house and use additional NAS heads and storage.

Scale-out NAS enables grouping multiple nodes together to construct a clustered NAS system. A scale-out NAS provides the capability to scale its resources by simply adding nodes to a clustered NAS architecture. The cluster works as a single NAS device and is managed centrally. Nodes can be added to the cluster, when more performance or more capacity is needed, without causing any downtime. Scale-out NAS provides the flexibility to use many nodes of moderate performance and availability characteristics to produce a total system that has better aggregate performance and availability. It also provides ease of use, low cost, and theoretically unlimited scalability.

Scale-out NAS creates a single file system that runs on all nodes in the cluster. All information is shared among nodes, so the entire file system is accessible by clients connecting to any node in the cluster. Scale-out NAS stripes data across all nodes in a cluster along with mirror or parity protection. As data is sent from clients to the cluster, the data is divided and allocated to different nodes in parallel. When a client sends a request to read a file, the scale-out NAS retrieves the appropriate blocks from multiple nodes, recombines the blocks into a file, and presents the file to the client. As nodes are added, the file system grows dynamically and data is evenly distributed to every node. Each node added to the cluster increases the aggregate storage, memory, CPU, and network capacity. Hence, cluster performance also increases.

Scale-out NAS is suitable to solve the "Big Data" challenges that enterprises and customers face today. It provides the capability to manage and store large, high-growth data in a single place with the flexibility to meet a broad range of performance requirements.

7.6.6 Scale-Out NAS Connectivity

Scale-out NAS clusters use separate internal and external networks for back-end and front-end connectivity, respectively. An internal network provides connections for intracluster communication, and an external network connection enables clients to access and share file data. Each node in the cluster connects to the internal network. The internal network offers high throughput and low latency and uses high-speed networking technology, such as InfiniBand or Gigabit Ethernet. To enable clients to access a node, the node must be connected to the external Ethernet network. Redundant internal or external networks may be used for high availability. Figure 7-7 illustrates an example of scale-out NAS connectivity.

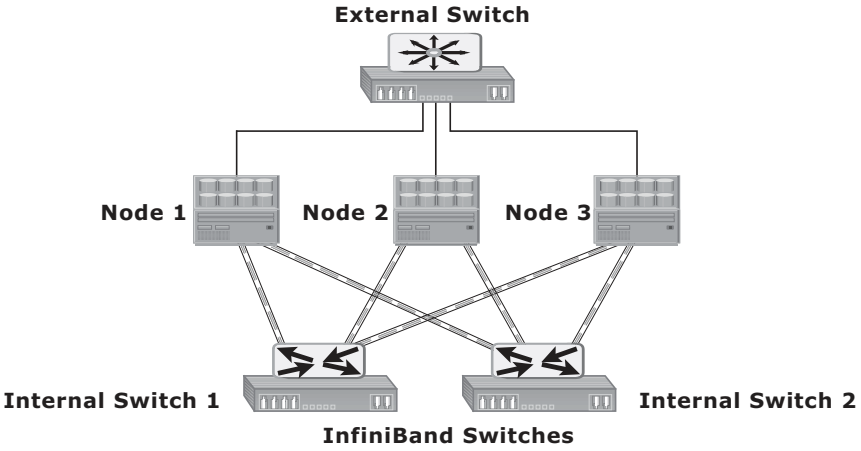


Figure 7-7: Scale-out NAS with dual internal and single external networks

INFINIBAND

InfiniBand is a networking technology that provides a low-latency, high-bandwidth communication link between hosts and peripherals. It provides serial connection and is often used for inter-server communications in high-performance computing environments. InfiniBand enables remote direct memory access (RDMA) that enables a device (host or peripheral) to access data directly from the memory of a remote device. InfiniBand also enables a single physical link to carry multiple channels of data simultaneously using a multiplexing technique. The InfiniBand networking infrastructure consists of host channel adapters (HCAs), target channel adapters (TCAs), and InfiniBand switches. HCAs are located within hosts. HCAs provide the mechanism to connect CPUs and memory of the hosts to the InfiniBand network. Similarly, TCAs enable storage and other peripheral devices to connect to the InfiniBand network. InfiniBand switches provide connectivity among HCAs and TCAs.

7.7 NAS File-Sharing Protocols

Most NAS devices support multiple file-service protocols to handle file I/O requests to a remote file system. As discussed earlier, NFS and CIFS are the common protocols for file sharing. NAS devices enable users to share file data across different operating environments and provide a means for users to migrate transparently from one operating system to another.

7.7.1 NFS

NFS is a client-server protocol for file sharing that is commonly used on UNIX systems. NFS was originally based on the connectionless *User Datagram Protocol* (UDP). It uses a machine-independent model to represent user data. It also uses Remote Procedure Call (RPC) as a method of inter-process communication between two computers. The NFS protocol provides a set of RPCs to access a remote file system for the following operations:

- Searching files and directories
- Opening, reading, writing to, and closing a file
- Changing file attributes
- Modifying file links and directories

NFS creates a connection between the client and the remote system to transfer data. NFS (NFSv3 and earlier) is a *stateless protocol*, which means that it does not maintain any kind of table to store information about open files and associated pointers. Therefore, each call provides a full set of arguments to access files on the server. These arguments include a file handle reference to the file, a particular position to read or write, and the versions of NFS.

Currently, three versions of NFS are in use:

- **NFS version 2 (NFSv2):** Uses UDP to provide a stateless network connection between a client and a server. Features, such as locking, are handled outside the protocol.
- **NFS version 3 (NFSv3):** The most commonly used version, which uses UDP or TCP, and is based on the stateless protocol design. It includes some new features, such as a 64-bit file size, asynchronous writes, and additional file attributes to reduce refetching.
- **NFS version 4 (NFSv4):** Uses TCP and is based on a stateful protocol design. It offers enhanced security. The latest NFS version 4.1 is the enhancement of NFSv4 and includes some new features, such as session model, parallel NFS (pNFS), and data retention.

PNFS AND MPFS

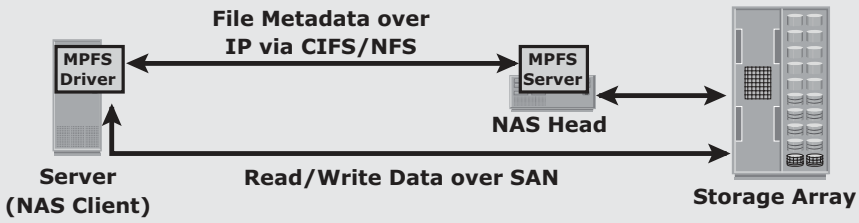
pNFS, as part of NFSv4.1, separates the file system protocol processing into two parts: metadata processing and data processing. The metadata includes information about a file system object, such as its name, location within the namespace, owner, access control list (ACL), and other attributes. The pNFS server, also called a metadata server,

(Continued)

PNFS AND MPFS *(continued)*

does the metadata processing and is kept out of the data path. pNFS clients send the metadata information to the pNFS server. The pNFS clients access storage devices directly using multiple parallel data paths. The pNFS client uses a storage network protocol, such as iSCSI or FC, to perform I/O to storage devices. The pNFS clients get information about the storage devices from the metadata server. Because the pNFS server is relieved of data processing and pNFS clients can access the storage devices directly using parallel paths, the pNFS mechanism significantly improves the pNFS client performance.

The EMC-patented Multi-Path File System (MPFS) protocol works similar to pNFS. The MPFS driver software, installed at the NAS clients, sends the file's metadata to the NAS device (MPFS server) via the IP network. The MPFS driver obtains information about the location of the data from the NAS device over the IP network. After knowing the data location, the MPFS driver communicates directly to the storage devices and enables the NAS clients to access the data over SAN. The following Figure shows the MPFS architecture that provides different paths for transferring a file's metadata and data.



7.7.2 CIFS

CIFS is a client-server application protocol that enables client programs to make requests for files and services on remote computers over TCP/IP. It is a public, or open, variation of Server Message Block (SMB) protocol.

The CIFS protocol enables remote clients to gain access to files on a server. CIFS enables file sharing with other clients by using special locks. Filenames in CIFS are encoded using unicode characters. CIFS provides the following features to ensure data integrity:

- It uses file and record locking to prevent users from overwriting the work of another user on a file or a record.
- It supports fault tolerance and can automatically restore connections and reopen files that were open prior to an interruption. The fault tolerance features of CIFS depend on whether an application is written to take advantage of these features. Moreover, CIFS is a stateful protocol because the CIFS server maintains connection information regarding every connected

client. If a network failure or CIFS server failure occurs, the client receives a disconnection notification. User disruption is minimized if the application has the embedded intelligence to restore the connection. However, if the embedded intelligence is missing, the user must take steps to reestablish the CIFS connection.

Users refer to remote file systems with an easy-to-use file-naming scheme: `\\server\share` or `\\servername.domain.suffix\share`.

The file naming scheme in an NFS environment is:
`Server:/export` or `Server.domain.suffix:/export`.

7.8 Factors Affecting NAS Performance

NAS uses IP network; therefore, bandwidth and latency issues associated with IP affect NAS performance. Network congestion is one of the most significant sources of latency (Figure 7-8) in a NAS environment. Other factors that affect NAS performance at different levels follow:

1. **Number of hops:** A large number of hops can increase latency because IP processing is required at each hop, adding to the delay caused at the router.
2. **Authentication with a directory service such as Active Directory or NIS:** The authentication service must be available on the network with enough resources to accommodate the authentication load. Otherwise, a large number of authentication requests can increase latency.
3. **Retransmission:** Link errors and buffer overflows can result in retransmission. This causes packets that have not reached the specified destination to be re-sent. Care must be taken to match both speed and duplex settings on the network devices and the NAS heads. Improper configuration might result in errors and retransmission, adding to latency.
4. **Overutilized routers and switches:** The amount of time that an overutilized device in a network takes to respond is always more than the response time of an optimally utilized or underutilized device. Network administrators can view utilization statistics to determine the optimum utilization of switches and routers in a network. Additional devices should be added if the current devices are overutilized.

Chapter 8
Object-Based and Unified Storage

KEY CONCEPTS

- Object-Based Storage
- Content Addressed Storage
- Unified Storage

Recent studies have shown that more than 90 percent of data generated is unstructured. This growth of unstructured data has posed new challenges to IT administrators and storage managers. With this growth, traditional NAS, which is a dominant solution for storing unstructured data, has become inefficient. Data growth adds high overhead to the network-attached storage (NAS) in terms of managing a large number of permissions and nested directories. In an enterprise environment, NAS also manages large amounts of metadata generated by hosts, storage systems, and individual applications. Typically this metadata is stored as part of the file and distributed throughout the environment. This adds to the complexity and latency in searching and retrieving files. These challenges demand a smarter approach to manage unstructured data based on its content rather than metadata about its name, location, and so on. *Object-based storage* is a way to store file data in the form of objects based on its content and other attributes rather than the name and location.

Due to varied application requirements, organizations have been deploying storage area networks (SANs), NAS, and object-based storage devices (OSDs) in their data centers. Deploying these disparate storage solutions adds management complexity, cost and environmental overhead. An ideal solution would be to have an integrated storage solution that supports block, file, and object access. Unified storage has emerged as a solution that consolidates block, file, and object-based access within one unified platform. It supports multiple protocols for data access and can be managed using a single management interface.

This chapter details object-based storage, its components, and operation. It also details *content addressed storage* (CAS), a special type of OSD. Further, this chapter covers the components and data access method in unified storage.

8.1 Object-Based Storage Devices

An OSD is a device that organizes and stores unstructured data, such as movies, office documents, and graphics, as objects. Object-based storage provides a scalable, self-managed, protected, and shared storage option. OSD stores data in the form of *objects*. OSD uses flat address space to store data. Therefore, there is no hierarchy of directories and files; as a result, a large number of objects can be stored in an OSD system (see Figure 8-1).

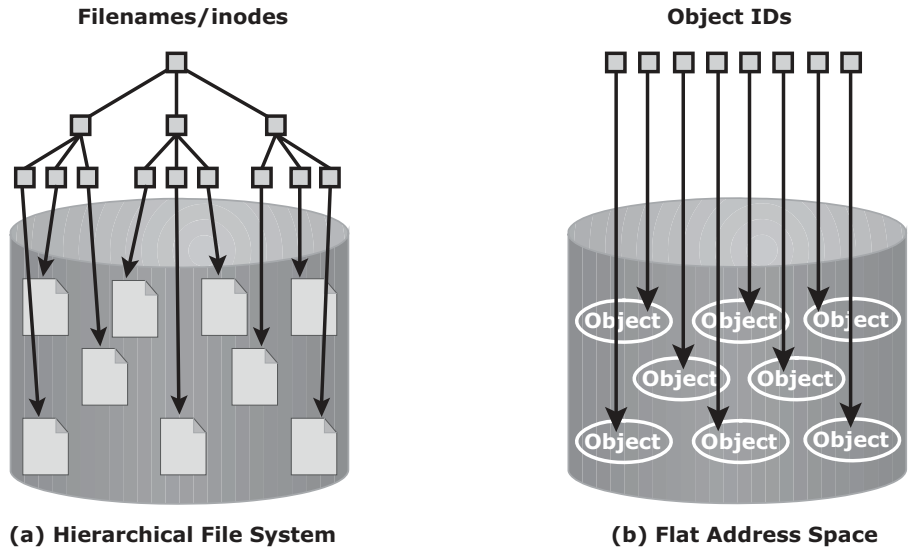


Figure 8-1: Hierarchical file system versus flat address space

An object might contain user data, related metadata (size, date, ownership, and so on), and other attributes of data (retention, access pattern, and so on); see Figure 8-2. Each object stored in the system is identified by a unique ID called the *object ID*. The object ID is generated using specialized algorithms such as hash function on the data and guarantees that every object is uniquely identified.

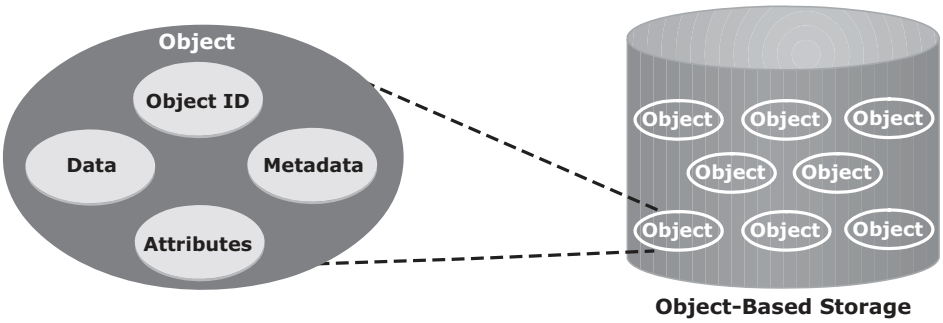


Figure 8-2: Object structure

8.1.1 Object-Based Storage Architecture

An I/O in the traditional block access method passes through various layers in the I/O path. The I/O generated by an application passes through the file system, the channel, or network and reaches the disk drive. When the file system receives the I/O from an application, the file system maps the incoming I/O to the disk blocks. The block interface is used for sending the I/O over the channel or network to the storage device. The I/O is then written to the block allocated on the disk drive. Figure 8-3 (a) illustrates the block-level access.

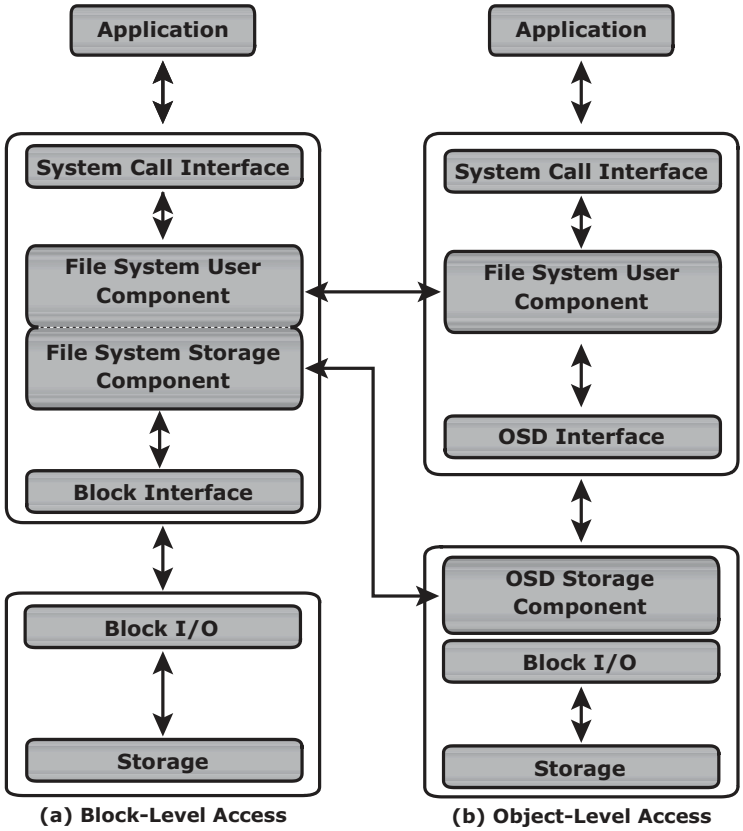


Figure 8-3: Block-level access versus object-level access

The file system has two components: user component and storage component. The user component of the file system performs functions such as hierarchy management, naming, and user access control. The storage component maps the files to the physical location on the disk drive.

When an application accesses data stored in OSD, the request is sent to the file system user component. The file system user component communicates to the OSD interface, which in turn sends the request to the storage device. The storage device has the OSD storage component responsible for managing the access to the object on a storage device. Figure 8-3 (b) illustrates the object-level access. After the object is stored, the OSD sends an acknowledgment to the application server. The OSD storage component manages all the required low-level storage and space management functions. It also manages security and access control functions for the objects.

8.1.2 Components of OSD

The OSD system is typically composed of three key components: nodes, private network, and storage. Figure 8-4 illustrates the components of OSD.

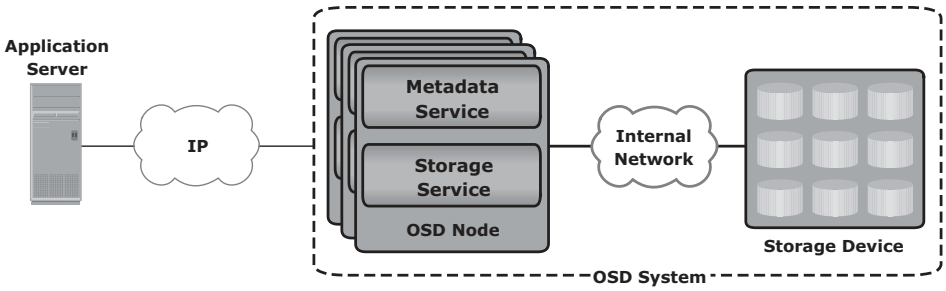


Figure 8-4: OSD components

The OSD system is composed of one or more *nodes*. A node is a server that runs the OSD operating environment and provides services to store, retrieve, and manage data in the system. The OSD node has two key services: metadata service and storage service. The metadata service is responsible for generating the object ID from the contents (and can also include other attributes of data) of a file. It also maintains the mapping of the object IDs and the file system namespace. The storage service manages a set of disks on which the user data is stored. The OSD nodes connect to the storage via an internal network. The internal network provides node-to-node connectivity and node-to-storage connectivity. The application server accesses the node to store and retrieve data over an external network. In some implementations, such as CAS, the metadata service might reside on the application server or on a separate server.

OSD typically uses low-cost and high-density disk drives to store the objects. As more capacity is required, more disk drives can be added to the system.

8.1.3 Object Storage and Retrieval in OSD

The process of storing objects in OSD is illustrated in Figure 8-5. The data storage process in an OSD system is as follows:

1. The application server presents the file to be stored to the OSD node.
2. The OSD node divides the file into two parts: user data and metadata.
3. The OSD node generates the object ID using a specialized algorithm. The algorithm is executed against the contents of the user data to derive an ID unique to this data.
4. For future access, the OSD node stores the metadata and object ID using the metadata service.
5. The OSD node stores the user data (objects) in the storage device using the storage service.
6. An acknowledgment is sent to the application server stating that the object is stored.

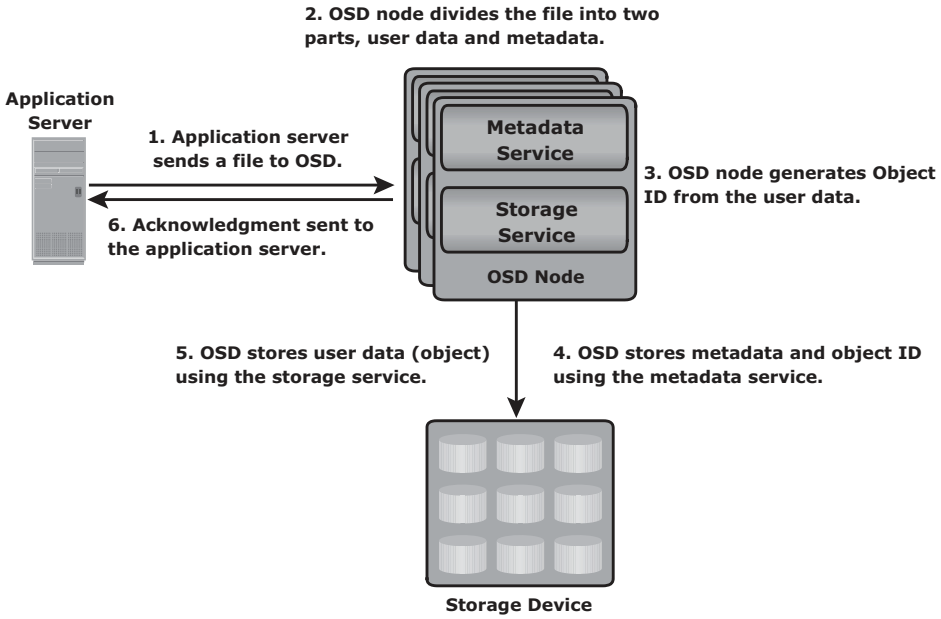


Figure 8-5: Storing objects on OSD

After an object is stored successfully, it is available for retrieval. A user accesses the data stored on OSD by the same filename. The application server retrieves the stored content using the object ID. This process is transparent to the user.

The process of retrieving objects in OSD is illustrated in Figures 8-6. The process of data retrieval from OSD is as follows:

1. The application server sends a read request to the OSD system.
2. The metadata service retrieves the object ID for the requested file.
3. The metadata service sends the object ID to the application server.
4. The application server sends the object ID to the OSD storage service for object retrieval.
5. The OSD storage service retrieves the object from the storage device.
6. The OSD storage service sends the file to the application server.

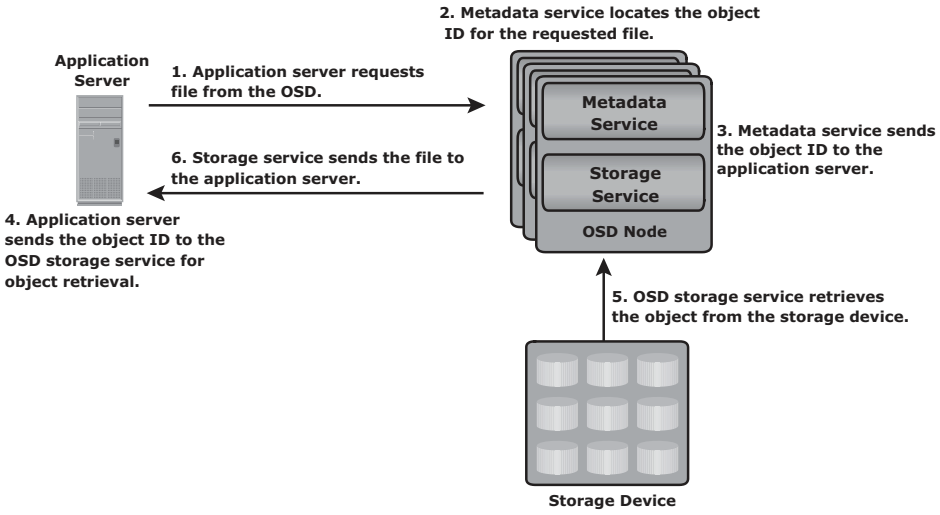


Figure 8-6: Object retrieval from an OSD system

8.1.4 Benefits of Object-Based Storage

For unstructured data, object-based storage devices provide numerous benefits over traditional storage solutions. An ideal storage architecture should provide performance, scalability, security, and data sharing across multiple platforms. Traditional storage solutions, such as SAN and NAS, do not offer all these benefits as a single solution. Object-based storage combines benefits of both the worlds. It provides platform and location independence, and at the same time, provides scalability, security, and data-sharing capabilities. The key benefits of object-based storage are as follows:

- **Security and reliability:** Data integrity and content authenticity are the key features of object-based storage devices. OSD uses specialized algorithms

to create objects that provide strong data encryption capability. In OSD, request authentication is performed at the storage device rather than with an external authentication mechanism.

- **Platform independence:** Objects are abstract containers of data, including metadata and attributes. This feature allows objects to be shared across heterogeneous platforms locally or remotely. This platform-independence capability makes object-based storage the best candidate for cloud computing environments.
- **Scalability:** Due to the use of flat address space, object-based storage can handle large amounts of data without impacting performance. Both storage and OSD nodes can be scaled independently in terms of performance and capacity.
- **Manageability:** Object-based storage has an inherent intelligence to manage and protect objects. It uses self-healing capability to protect and replicate objects. Policy-based management capability helps OSD to handle routine jobs automatically.

8.1.5 Common Use Cases for Object-Based Storage

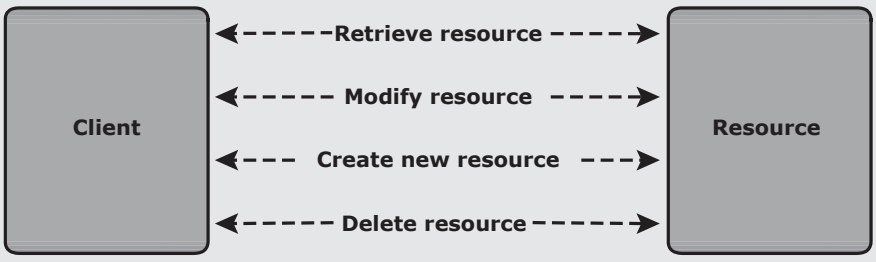
A data archival solution is a promising use case for OSD. Data integrity and protection is the primary requirement for any data archiving solution. Traditional archival solutions — CD and DVD-ROM — do not provide scalability and performance. OSD stores data in the form of objects, associates them with a unique object ID, and ensures high data integrity. Along with integrity, it provides scalability and data protection. These capabilities make OSD a viable option for long term data archiving for fixed content. Content addressed storage (CAS) is a special type of object-based storage device purposely built for storing fixed content. CAS is covered in the following section.

Another use case for OSD is cloud-based storage. OSD uses a web interface to access storage resources. OSD provides inherent security, scalability, and automated data management. It also enables data sharing across heterogeneous platforms or tenants while ensuring integrity of data. These capabilities make OSD a strong option for cloud-based storage. Cloud service providers can leverage OSD to offer storage-as-a-service.

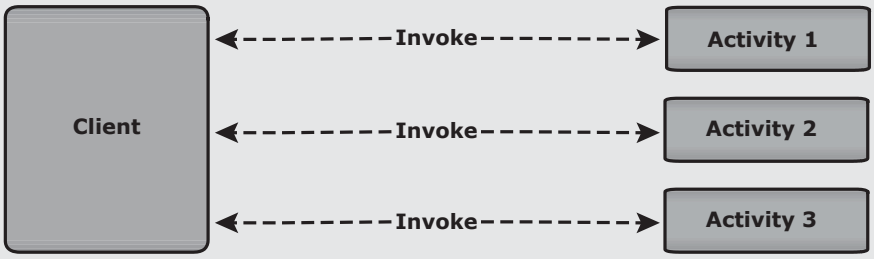
OSD supports web service access via *representational state transfer* (REST) and *simple object access protocol* (SOAP). REST and SOAP APIs can be easily integrated with business applications that access OSD over the web.

REST AND SOAP

REST is an architectural style developed for modern web applications. REST provides lightweight web services to access resources (for example, documents, blogs, and so on) on which a few basic operations can be performed, such as retrieving, modifying, creating, and deleting resources. REST-style web services are resource-oriented services. Resources can be uniquely located and identified by a Universal Resource Identifier (URI), and operations can be performed on those resources using an HTTP specification. For example, if a user accesses a blog using REST via a unique identifier, the request returns the representation of the blog in a particular format (XML or HTML).



(a) REST



(b) SOAP

SOAP is an XML-based protocol that enables communication between the web applications running on different OSes and based on different programming languages. SOAP provides processes to encode HTTP headers and XML files to enable and pass information between different computers.

8.2 Content-Addressed Storage

CAS is an object-based storage device designed for secure online storage and retrieval of fixed content. CAS stores user data and its attributes as an object. 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 storage solution. Data access in CAS differs from other OSD devices. In CAS, the application server access the CAS device only via the CAS API running on the application server. However, the way CAS stores data is similar to the other OSD systems.

CAS provides all the features required for storing fixed content. The key features of CAS are as follows:

- **Content authenticity:** It assures the genuineness of stored content. This is achieved by generating a unique content address for each object and validating the content address for stored objects at regular intervals. Content authenticity is assured because the address assigned to each object is as unique as a fingerprint. Every time an object is read, CAS uses a hashing algorithm to recalculate the object's content address as a validation step and compares the result to its original content address. If the object fails validation, CAS rebuilds the object using a mirror or parity protection scheme.
- **Content integrity:** It provides assurance that the stored content has not been altered. CAS uses a hashing algorithm for content authenticity and integrity. If the fixed content is altered, CAS generates a new address for the altered content, rather than overwrite the original fixed content.
- **Location independence:** CAS uses a unique content address, rather than directory path names or URLs, to retrieve data. This makes the physical location of the stored data irrelevant to the application that requests the data.
- **Single-instance storage (SIS):** CAS uses a unique content address to guarantee the storage of only a single instance of an object. When a new object is written, the CAS system is polled to see whether an object is already available with the same content address. If the object is available in the system, it is not stored; instead, only a pointer to that object is created.
- **Retention enforcement:** Protecting and retaining objects is a core requirement of an archive storage system. After an object is stored in the CAS system and the retention policy is defined, CAS does not make the object available for deletion until the policy expires.
- **Data protection:** CAS ensures that the content stored on the CAS system is available even if a disk or a node fails. CAS provides both local and remote

protection to the data objects stored on it. In the local protection option, data objects are either mirrored or parity protected. In mirror protection, two copies of the data object are stored on two different nodes in the same cluster. This decreases the total available capacity by 50 percent. In parity protection, the data object is split in multiple parts and parity is generated from them. Each part of the data and its parity are stored on a different node. This method consumes less capacity to protect the stored data, but takes slightly longer to regenerate the data if corruption of data occurs.

In the remote replication option, data objects are copied to a secondary CAS at the remote location. In this case, the objects remain accessible from the secondary CAS if the primary CAS system fails.

- **Fast record retrieval:** CAS stores all objects on disks, which provides faster access to the objects compared to tapes and optical discs.
- **Load balancing:** CAS distributes objects across multiple nodes to provide maximum throughput and availability.
- **Scalability:** CAS allows the addition of more nodes to the cluster without any interruption to data access and with minimum administrative overhead.
- **Event notification:** CAS continuously monitors the state of the system and raises an alert for any event that requires the administrator's attention. The event notification is communicated to the administrator through SNMP, SMTP, or e-mail.
- **Self diagnosis and repair:** CAS automatically detects and repairs corrupted objects and alerts the administrator about the potential problem. CAS systems can be configured to alert remote support teams who can diagnose and repair the system remotely.
- **Audit trails:** CAS keeps track of management activities and any access or disposition of data. Audit trails are mandated by compliance requirements.

8.3 CAS Use Cases

Organizations have deployed CAS solutions to solve several business challenges. Two solutions are described in detail in the following sections.

8.3.1 Healthcare Solution: Storing Patient Studies

Large healthcare centers examine hundreds of patients every day and generate large volumes of medical records. Each record might be composed of one