Module – 2

Data Center Environment

Module 2: Data Center Environment

Upon completion of this module, you should be able to:
- Describe the core elements of a data center
- Describe virtualization at application and host layer
- Describe disk drive components and performance
- Describe host access to storage through DAS
- Describe working and benefits of flash drives

This module focuses on the key components of a data center. It cilnudes virtualization at compute, memory, desktop, and application.

This module also focuses on storage subsystems and provides details on components, geometry, and performance parameters of a disk drive. The connectivity between the host and storage facilitated by various technologies is also explained.

Module 2: Data Center Environment

Lesson 1: Application, DBMS, and Host (Compute)

During this lesson the following topics are covered:
- Application and application virtualization
- DBMS
- Components of host system
- Compute and memory virtualization

This lesson covers three key components of a data center – application, DBMS and compute. Hardware and software components of a compute system including OS, logical volume manager, file system, and device driver are also explained. Virtualization at application and compute is also discussed in the lesson.

Application

- A software program that provides logic for computing operations
- Commonly deployed applications in a data center
  - Business applications – email, enterprise resource planning (ERP), decision support system (DSS)
  - Management applications – resource management, performance tuning, virtualization
  - Data protection applications – backup, replication
  - Security applications – authentication, antivirus
- Key I/O characteristics of an application
  - Read intensive vs. write intensive
  - Sequential vs. random
  - I/O size

An *application* is a computer program that provides the logic for computing operations. The application sends requests to the underlying operating system to perform read/write (R/W) operations on the storage devices. Applications can be layered on the database, which in turn uses the OS services to perform R/W operations on the storage devices.

Applications deployed in a data center environment are commonly categorized as business applications, infrastructure management applications, data protection applications, and security applications. Some examples of these applications are e-mail, enterprise resource planning (ERP), decision support system (DSS), resource management, backup, authentication and antivirus applications, and so on.

The characteristics of I/Os (Input/Output) generated by the application influence the overall performance of storage system and storage solution designs. Common I/O characteristics of an application are:

- Read verses Write intensive
- Sequential verse Random
- I/O size

Application Virtualization

Application Virtualization

It is the technique of presenting an application to an end user without any installation, integration, or dependencies on the underlying computing platform.

- Allows application to be delivered in an isolated environment
  - Aggregates Operating System (OS) resources and the application into a virtualized container
  - Ensures integrity of Operating System (OS) and applications
  - Avoids conflicts between different applications or different versions of the same application

Application virtualization breaks the dependency between the application and the underlying platform (OS and hardware). Application virtualization encapsulates the application and the required OS resources within a virtualized container. This technology provides the ability to deploy applications without making any change to the underlying OS, file system, or registry of the computing platform on which they are deployed. Because virtualized applications run in an isolated environment, the underlying OS and other applications are protected from potential corruptions. There are many scenarios in which conflicts might arise if multiple applications or multiple versions of the same application are installed on the same computing platform. Application virtualization eliminates this conflict by isolating different versions of an application and the associated O/S resources.

Database Management System (DBMS)

- Database is a structured way to store data in logically organized tables that are interrelated
  - Helps to optimize the storage and retrieval of data
- DBMS controls the creation, maintenance, and use of databases
  - Processes an application's request for data
  - Instructs the OS to retrieve the appropriate data from storage
- Popular DBMS examples are MySQL, Oracle RDBMS, SQL Server, etc.

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

Host (Compute)

- Resource that runs applications with the help of underlying computing components
  - Example: Servers, mainframes, laptop, desktops, tablets, server clusters, etc.
- Consists of hardware and software components
- Hardware components
  - Include CPU, memory, and input/output (I/O) devices
- Software components
  - Include OS, device driver, file system, volume manager, and so on
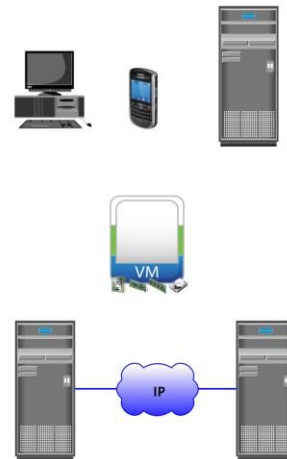
Users store and retrieve data through applications. The computers on which these applications run are referred to as *hosts* or *compute systems*. Hosts can be physical or virtual machines. A compute virtualization software enables creating virtual machines on top of physical compute infrastructure. Compute virtualization and virtual machines are discussed later in this module.

Examples of physical hosts include desktop computers, servers or a cluster of servers, virtual servers, laptops, and mobile devices. A host consists of CPU, memory, I/O devices, and a collection of software to perform computing operations. This software includes the operating system, file system, logical volume manager, device drivers, and so on. These software can be installed individually or may be part of the operating system.

Operating Systems and Device Driver

- In a traditional environment OS resides between the applications and the hardware
  - ▶ Responsible for controlling the environment
- In a virtualized environment virtualization layer works between OS and hardware
  - ▶ Virtualization layer controls the environment
  - ▶ OS works as a guest and only controls the application environment
  - ▶ In some implementation OS is modified to communicate with virtualization layer
- Device driver is a software that enables the OS to recognize the specific device

In a traditional computing environment, an *operating system* controls all the aspects of computing. It works between the application and physical components of a compute system. One of the services it provides to the application is data access. The operating system also monitors and responds to user actions and the environment. It organizes and controls hardware components and manages the allocation of hardware resources. It provides basic security for the access and usage of all managed resources. An operating system also performs basic storage management tasks while managing other underlying components, such as the file system, volume manager, and device drivers.

In a virtualized compute environment, the virtualization layer works between the operating system and the hardware resources. Here the OS might work differently based on the type of the compute virtualization implemented. In a typical implementation, the OS works as a guest and performs only the activities related to application interaction. In this case, hardware management functions are handled by the virtualization layer.

A *device driver* is special software that permits the operating system to interact with a specific device, such as a printer, a mouse, or a disk drive. A device driver enables the operating system to recognize the device and to access and control devices. Device drivers are hardware-dependent and operating-system-specific.

Memory Virtualization

- An OS feature that presents larger memory to the application than physically available
  - Additional memory space comes from disk storage
  - Space used on the disk for virtual memory is called 'swap space/swap file or page file'
  - Inactive memory pages are moved from physical memory to the swap file
  - Provides efficient use of available physical memory
  - Data access from swap file is slower – use of flash drives for swap space gives best performance
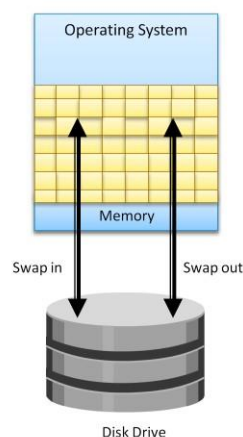
Operating System

Memory

Swap in   Swap out

Disk Drive

Memory has been, and continues to be, an expensive component of a host. It determines both the size and number of applications that can run on a host. *Memory virtualization* enables multiple applications and processes, whose aggregate memory requirement is greater than the available physical memory, to run on a host without impacting each other.

Memory virtualization is an operating system feature that virtualizes the physical memory (RAM) of a host. It creates a virtual memory with an address space larger than the physical memory space present in the compute system. The virtual memory encompasses the address space of the physical memory and part of the disk storage. The operating system utility that manages the virtual memory is known as the *virtual memory manager* (VMM). The VMM manages the virtual-to-physical memory mapping and fetches data from the disk storage when a process references a virtual address that points to data at the disk storage. The space used by the VMM on the disk is known as a swap space. A *swap space* (also known as *page file* or *swap file*) is a portion of the disk drive that appears like physical memory to the operating system.

In a virtual memory implementation, the memory of a system is divided into contiguous blocks of fixed-size pages. A process known as *paging* moves inactive physical memory pages onto the swap file and brings them back to the physical memory when required. This enables efficient use of the available physical memory among different applications. The operating system typically moves the least used pages into the swap file so that enough RAM is available for processes that are more active. Access to swap file pages is slower than physical memory pages because swap file pages are allocated on the disk drive which is slower than physical memory.

Logical Volume Manager (LVM)

- Responsible for creating and controlling host level logical storage
  - Physical view of storage is converted to a logical view
  - Logical data blocks are mapped to physical data blocks
- One or more Physical Volumes form a Volume Group
  - LVM manages Volume Groups as a single entity
- Logical volumes are created from the volume group

Logical Volume

Logical Volume

Logical Disk Block

Physical Vol. 1   Physical Vol. 2   Physical Vol. 3

Volume Group
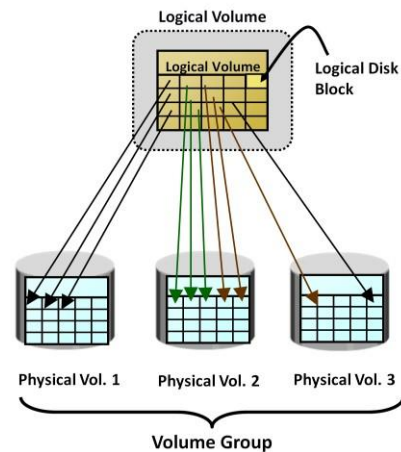
In the early days, entire disk drive would be allocated to the file system or other data entity used by the operating system or application. The disadvantage was lack of flexibility. When a disk drive ran out of space, there was no easy way to extend the file system's size. Also, as the storage capacity of the disk drive increased, allocating the entire disk drive for the file system often resulted in underutilization of storage capacity.

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

The LVM provides optimized storage access and simplifies storage resource management. It hides details about the physical disk and the location of data on the disk. It enables administrators to change the storage allocation even when the application is running.
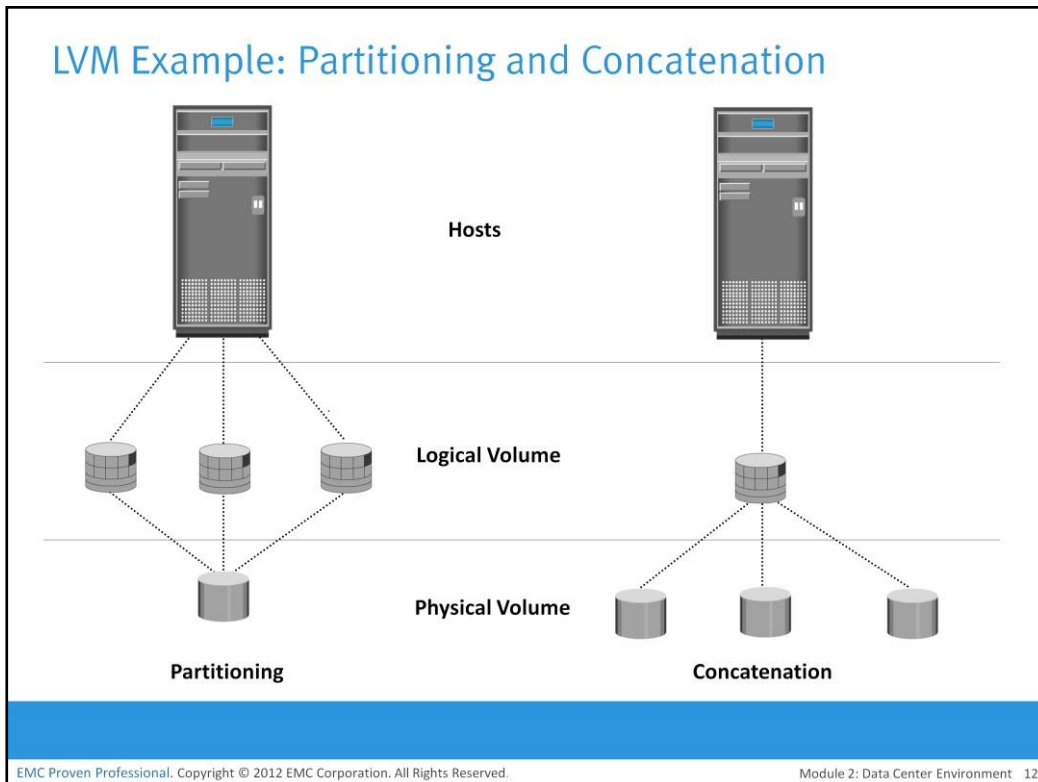
Cont…

The basic LVM components are physical volumes, volume groups, and logical volumes. In LVM terminology, each physical disk connected to the host system is a *physical volume* (PV). A *volume group* is created by grouping together one or more physical volumes. A unique *physical volume identifier* (PVID) is assigned to each physical volume when it is initialized for use by the LVM. Physical volumes can be added or removed from a volume group dynamically. They cannot be shared between different volume groups; which means, the entire physical volume becomes part of a volume group. Each physical volume is divided into equal-sized data blocks called *physical extents* when the volume group is created.

Logical volumes (LV) are created within a given volume group. A LV can be thought of as a disk partition, whereas the volume group itself can be thought of as a disk. The size of a LV is based on a multiple of the physical extents. The LV appears as a physical device to the operating system. A LV is made up of noncontiguous physical extents and may span over multiple physical volumes. A file system is created on a logical volume. These LVs are then assigned to the application. A logical volume can also be mirrored to provide enhanced data availability.

Today, logical volume managers are mostly offered as part of the operating system.

LVM Example: Partitioning and Concatenation

Hosts

Logical Volume

Physical Volume

Partitioning

Concatenation

Disk partitioning was introduced to improve the flexibility and utilization of disk drives. In partitioning, a disk drive is divided into logical containers called *logical volumes* (LVs). For example, a large physical drive can be partitioned into multiple LVs to maintain data according to the file system and application requirements. The partitions are created from groups of contiguous cylinders when the hard disk is initially set up on the host. The host's file system accesses the logical volumes without any knowledge of partitioning and physical structure of the disk.

Concatenation is the process of grouping several physical drives and presenting them to the host as one big logical volume.
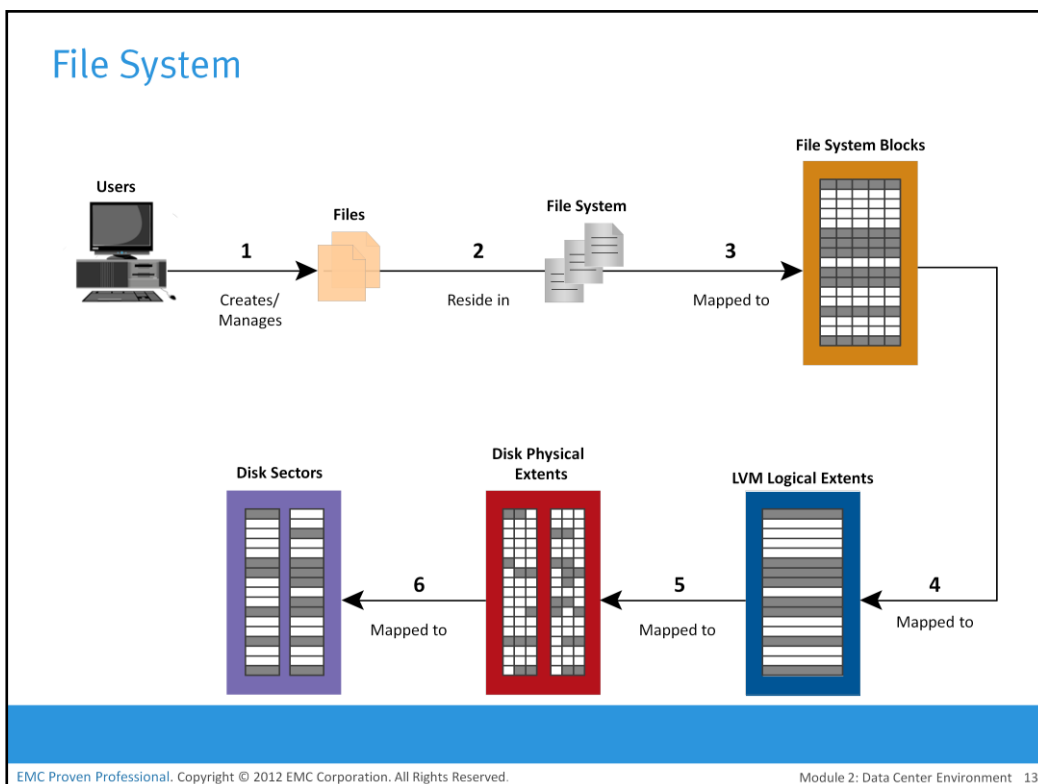
File System

A *file* is a collection of related records or data stored as a unit with a name. A *file system* is a hierarchical structure of files. A file system enables easy access to data files residing within a disk drive, a disk partition, or a logical volume. A file system consists of logical structures and software routines that control access to files. It provides users with the functionality to create, modify, delete, and access files. Access to files on the disks is controlled by the permissions assigned to the file by the owner, which are also maintained by the file system.

A file system organizes data in a structured hierarchical manner via the use of directories, which are containers for storing pointers to multiple files. All file systems maintain a pointer map to the directories, subdirectories, and files that are part of the file system. The following list shows the process of mapping user files to the disk storage that uses an LVM:

1.   Files are created and managed by users and applications.

2.   These files reside in the file systems.

3.   The file systems are mapped to file system blocks.

4.   The file system blocks are mapped to logical extents of a logical volume.

5.   These logical extents in turn are mapped to the disk physical extents either by the operating system or by the LVM.

6.   These physical extents are mapped to the disk sectors in a storage subsystem.

If there is no LVM, then there are no logical extents. Without LVM, file system blocks are directly mapped to disk sectors.

A file system *block* is the smallest unit allocated for storing data. Each file system block is a contiguous area on the physical disk. The block size of a file system is fixed at the time of its creation. The file system size depends on the block size and the total number of files system blocks. A file can span multiple file system blocks because most files are larger than the predefined block size of the file system. File system blocks cease to be contiguous and become fragmented when new blocks are added or deleted. Over time, as files grow larger, the file system becomes increasingly fragmented.

Apart from the files and directories, the file system also includes a number of other related records, which are collectively called the *metadata*. The metadata of a file system must be consistent for the file system to be considered healthy.
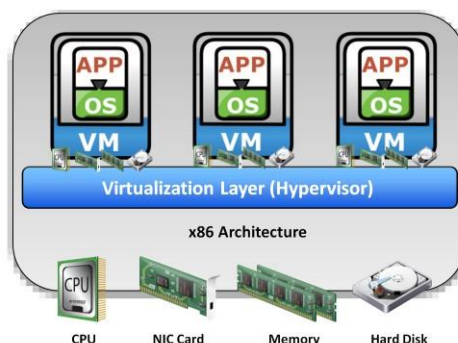
Examples of some common file systems are, FAT 32 (File Allocation Table) and NT File System (NTFS) for Microsoft Windows, UNIX File System (UFS) and Extended File System (EXT2/3) for Linux.

Compute Virtualization

Compute Virtualization

It is a technique of masking or abstracting the physical compute hardware and enabling multiple operating systems (OSs) to run concurrently on a single or clustered physical machine(s).

- Enables creation of multiple virtual machines (VMs), each running an OS and application
  - VM is a logical entity that looks and behaves like physical machine
- Virtualization layer resides between hardware and VMs
  - Also known as hypervisor
- VMs are provided with standardized hardware resources

APP / OS / VM   APP / OS / VM   APP / OS / VM

Virtualization Layer (Hypervisor)

x86 Architecture

CPU   NIC Card   Memory   Hard Disk
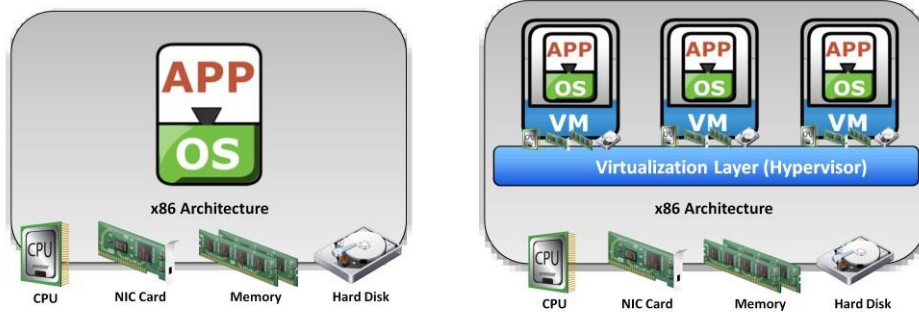
Compute virtualization is a technique of masking or abstracting the physical hardware from the operating system. It enables multiple operating systems to run concurrently on a single or clustered physical machine(s). This technique enables creating portable virtual compute systems called *virtual machines (VMs).* Each VM runs an operating system and application instance in an isolated manner.

Compute virtualization is achieved by a virtualization layer that resides between the hardware and virtual machines. This layer is also called the *hypervisor*. The hypervisor provides hardware resources, such as CPU, memory, and network to all the virtual machines. Within a physical server, a large number of virtual machines can be created depending on the hardware capabilities of the physical server.

A virtual machine is a logical entity but appears like a physical host to the operating system, with its own CPU, memory, network controller, and disks. However, all VMs share the same underlying physical hardware in an isolated manner. From a hypervisor perspective, virtual machines are discrete sets of files that include VM configuration file, data files, and so on.

Need for Compute Virtualization

x86 Architecture

Virtualization Layer (Hypervisor)

x86 Architecture

Before Virtualization	After Virtualization
• Runs single operating system (OS) per machine at a time • Couples s/w and h/w tightly • May create conflicts when multiple applications run on the same machine • Underutilizes resources • Is inflexible and expensive	• Runs multiple operating systems (OSs) per physical machine concurrently • Makes OS and applications h/w independent • Isolates VM from each other, hence, no conflict • Improves resource utilization • Offers flexible infrastructure at low cost

Typically, a physical server often faces resource-conflict issues when two or more applications running on the server have conflicting requirements. For example, applications might need different values in the same registry entry, different versions of the same DLL, and so on. These issues are further compounded with an application's high-availability requirements. As a result, the servers are limited to serve only one application at a time. This causes organizations to purchase new physical machines for every application they deploy, resulting in expensive and inflexible infrastructure. On the other hand, many applications do not take full advantage of the hardware capabilities available to them. Consequently, resources such as processors, memory, and storage remain underutilized.

Compute virtualization enables to overcome these challenges by allowing multiple operating systems and applications to run on a single physical machine. This technique significantly improves server utilization and provides server consolidation. Server consolidation enables organizations to run their data center with fewer servers. This, in turn, cuts down the cost of new server acquisition, reduces operational cost, and saves data center floor and rack space. Creation of VMs takes less time compared to a physical server setup; organizations can provision servers faster and with ease. Individual VMs can be restarted, upgraded, or even crashed, without affecting the other VMs on the same physical machine. Moreover, VMs can be copied or moved from one physical machine to another without causing application downtime.

Desktop Virtualization

Desktop Virtualization

It is a technology which enables detachment of the user state, the Operating System (OS), and the applications from endpoint devices.

- Enables organizations to host and centrally manage desktops
  - Desktops run as virtual machines within the data center and accessed over a network
- Desktop virtualization benefits
  - Flexibility of access due to enablement of thin clients
  - Improved data security
  - Simplified data backup and PC maintenance

Pcs and thin clients

LAN/WAN
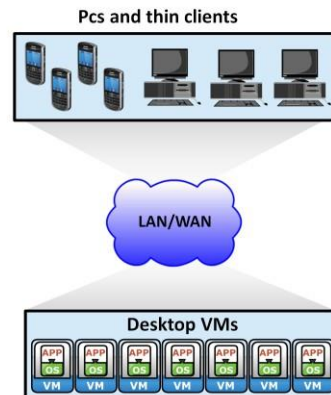
Desktop VMs

With the traditional desktop, the OS, applications, and user profiles are all tied to a specific piece of hardware. With legacy desktops, business productivity is impacted greatly when a client device is broken or lost. *Desktop virtualization* breaks the dependency between the hardware and its OS, applications, user profiles, and settings. This enables the IT staff to change, update, and deploy these elements independently. Desktops hosted at the data center and runs on virtual machines, whereas users remotely access these desktops from a variety of client devices, such as laptop, desktop, and mobile devices (also called Thin devices). Application execution and data storage are performed centrally at the data center instead of at the client devices. Because desktops run as virtual machines within an organization's data center, it mitigates the risk of data leakage and theft. It also helps to perform centralized backup and simplifies compliance procedures. Virtual desktops are easy to maintain because it is simple to apply patches, deploy new applications and OS, and provision or remove users centrally.

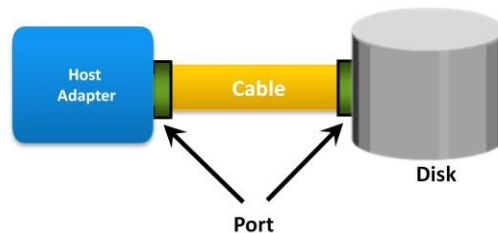## Module 2: Data Center Environment

Lesson 2: Connectivity

During this lesson the following topics are covered:
- Physical components of connectivity
- Storage connectivity protocols

This lesson covers physical components of connectivity and storage connectivity protocols. These protocols include IDE/ATA, SCSI, Fibre Channel and IP.

Connectivity

- Interconnection between hosts or between a host and peripheral devices, such as storage
- Physical Components of Connectivity are:
  - Host interface card, port, and cable
- Protocol = a defined format for communication between sending and receiving devices
  - Popular storage interface protocols: IDE/ATA and SCSI

Host Adapter — **Cable** — **Disk**

Port

Connectivity refers to the interconnection between hosts or between a host and peripheral devices, such as printers or storage devices. The discussion here focuses only on the connectivity between the host and the storage device. Connectivity and communication between host and storage are enabled using physical components and interface protocols.

The *physical components* of connectivity are the hardware elements that connect the host to storage. Three physical components of connectivity between the host and storage are host interface device, port, and cable

A *host interface device* or *host adapter* connects a host to other hosts and storage devices. Examples of host interface devices are host bus adapter (HBA) and network interface card (NIC). *Host bus adaptor (HBA)* is an application-specific integrated circuit (ASIC) board that performs I/O interface functions between the host and storage, relieving the CPU from additional I/O processing workload. A host typically contains multiple HBAs.

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

A *protocol* enables communication between the host and storage. Protocols are implemented using interface devices (or controllers) at both source and destination. The popular interface protocols used for host to storage communications are *Integrated Device Electronics/Advanced Technology Attachment* (IDE/ATA), *Small Computer System Interface* (SCSI), *Fibre Channel* (FC) and *Internet Protocol* (IP).

IDE/ATA and Serial ATA

- Integrated Device Electronics (IDE)/Advanced Technology Attachment (ATA)
  - Popular interface used to connect hard disks or CD-ROM drives
  - Available with varity of standards and names
- Serial Advanced Technology Attachment (SATA)
  - Serial version of the IDE/ATA specification that has replaced the parallel ATA
  - Inexpensive storage interconnect, typically used for internal connectivity
  - Provides data transfer rate up to 6 Gb/s (standard 3.0)

IDE/ATA is a popular interface protocol standard used for connecting storage devices, such as disk drives and CD-ROM drives. This protocol supports parallel transmission and therefore is also known as Parallel ATA (PATA) or simply ATA. IDE/ATA has a variety of standards and names. The Ultra DMA/133 version of ATA supports a throughput of 133 MB per second. In a master-slave configuration, an ATA interface supports two storage devices per connector.
However, if the performance of the drive is important, sharing a port between two devices is not recommended.

The serial version of this protocol supports single bit serial transmission and is known as Serial ATA (SATA). High performance and low cost SATA has largely replaced PATA in the newer systems. SATA revision 3.0 provides a data transfer rate up to 6 Gb/s.

SCSI has emerged as a preferred connectivity protocol in high-end computers. This protocol supports parallel transmission and offers improved performance, scalability, and compatibility compared to ATA. However, the high cost associated with SCSI limits its popularity among home or personal desktop users. Over the years, SCSI has been enhanced and now includes a wide variety of related technologies and standards. SCSI supports up to 16 devices on a single bus and provides data transfer rates up to 640 MB/s (for the Ultra-640 version).

Serial attached SCSI (SAS) is a point-to-point serial protocol that provides an alternative to parallel SCSI. A newer version (SAS 2.0) of serial SCSI supports a data transfer rate up to 6 Gb/s.

Fibre Channel and IP

- Fibre Channel (FC)
  - Widely used protocol for high speed communication to the storage device
  - Provides a serial data transmission that operates over copper wire and/or optical fiber
  - Latest version of the FC interface '16FC' allows transmission of data up to 16 Gb/s
- Internet Protocol (IP)
  - Traditionally used to transfer host-to-host traffic
  - Provide opportunity to leverage existing IP based network for storage communication
    - Examples: iSCSI and FCIP protocols

Fibre Channel is a widely used protocol for high-speed communication to the storage device. The Fibre Channel interface provides gigabit network speed. It provides a serial data transmission that operates over copper wire and optical fiber. The latest version of the FC interface '16FC' allows transmission of data up to 16 Gb/s. The FC protocol and its features are covered in more detail in Module 5.

IP is a network protocol that has been traditionally used for host-to-host traffic. With the emergence of new technologies, an IP network has become a viable option for host-to- storage communication. IP offers several advantages in terms of cost and maturity and enables organizations to leverage their existing IP-based network. iSCSI and FCIP protocols are common examples that leverage IP for host-to-storage communication. These protocols are detailed in module 6.

This lesson covers the most important element of a data center – Storage. Various storage medias and options are discussed with focus on disk drives. Components, structure, addressing and factors that impacts disk drives performance are detailed in the lesson.

Further it covers new generation flash drives and their benefits. Finally it introduces various methods of accessing storage from the host with details of direct-attached storage options.

Storage Options

- Magnetic Tape
  - Low cost solution for long term data storage
    - Preferred option for backup destination in the past
  - Limitations
    - Sequential data access
    - Single application access at a time
    - Physical wear and tear
    - Storage/retrieval overheads

The storage is a core component in a data center. A storage device uses magnetic, optic, or solid state media. Disks, tapes, and diskettes use magnetic media, whereas CD/DVD uses optical media for storage. Removable Flash memory or Flash drives are examples of solid state media.

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

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

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

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

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

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

Storage Options (contd.)

- Optical discs
  - Popularly used as distribution medium in small, single-user computing environments
  - Limited in capacity and speed
  - Write once and read many (WORM): CD-ROM, DVD-ROM
  - Other variations: CD-RW, Blu-ray discs
- Disk drive
  - Most popular storage medium
  - Large storage capacity
  - Random read/write access
- Flash drives
  - Uses semiconductor media
 - Provide high performance and low power consumption

Optical disc storage is popular in small, single-user computing environments. It is frequently used by individuals to store photos or as a backup medium on personal or laptop computers. It is also used as a distribution medium for small applications, such as games, or as a means to transfer small amounts of data from one computer to another.

Optical discs have limited capacity and speed, which limit the use of optical media as a business data storage solution. The capability to write once and read many (WORM) is one advantage of optical disc storage. A CD-ROM is an example of a WORM device. Optical discs, to some degree, guarantee that the content has not been altered. Therefore, it can be used as a low-cost alternative for long-term storage of relatively small amounts of fixed content that do not change after it is created. Collections of optical discs in an array, called a *jukebox*, are still used as a fixed-content storage solution. Other forms of optical discs include CD-RW, Blu-ray disc, and other variations of DVD.

Disk drives are the most popular storage medium used in modern computers for storing and accessing data for performance-intensive, online applications. Disks support rapid access to random data locations. This means that data can be written or retrieved quickly for a large number of simultaneous users or applications. In addition, disks have a large capacity. Disk storage arrays are configured with multiple disks to provide increased capacity and enhanced performance.

Flash drives (or solid stated drives - SSDs) uses semiconductor media and provides high performance and low power consumption. Flash drives are discussed in detail later in this module.

Disk Drive Components
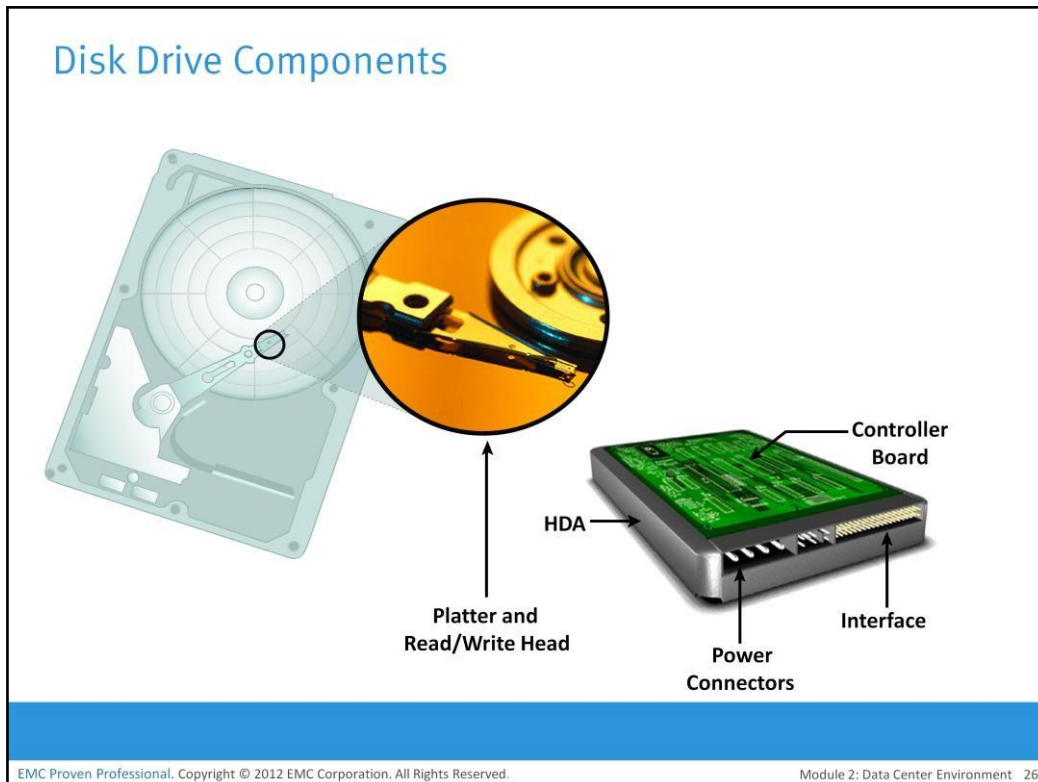
Controller Board

HDA

Platter and Read/Write Head

Power Connectors

Interface

The key components of a hard disk drive are platter, spindle, read-write head, actuator arm assembly, and controller board. I/O operations in a HDD is performed by rapidly moving the arm across the rotating flat platters coated with magnetic particles. Data is transferred between the disk controller and magnetic platters through the read-write (R/W) head which is attached to the arm. Data can be recorded and erased on magnetic platters any number of times.

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

Spindle: A spindle connects all the platters and is connected to a motor. The motor of the spindle rotates with a constant speed. The disk platter spins at a speed of several thousands of revolutions per minute (rpm). Common spindle speeds are 5,400 rpm, 7,200 rpm, 10,000 rpm, and 15,000 rpm. The speed of the platter is increasing with improvements in technology; although, the extent to which it can be improved is limited.

Cont…

Read/Write Head: *Read/Write (R/W) heads,* read and write data from or to platters. Drives have two R/W heads per platter, one for each surface of the platter. The R/W head changes the magnetic polarization on the surface of the platter when writing data. While reading data, the head detects the magnetic polarization on the surface of the platter. During reads and writes, the R/W head senses the magnetic polarization and never touches the surface of the platter.

When the spindle is rotating, there is a microscopic air gap maintained between the R/W heads and the platters, known as the *head flying height*. This air gap is removed when the spindle stops rotating and the R/W head rests on a special area on the platter near the spindle. This area is called the *landing zone*. The landing zone is coated with a lubricant to reduce friction between the head and the platter. The logic on the disk drive ensures that heads are moved to the landing zone before they touch the surface. If the drive malfunctions and the R/W head accidentally touches the surface of the platter outside the landing zone, a *head crash* occurs. In a head crash, the magnetic coating on the platter is scratched and may cause damage to the R/W head. A head crash generally results in data loss.

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

Drive Controller Board: The controller is a printed circuit board, mounted at the bottom of a disk drive. It consists of a microprocessor, internal memory, circuitry, and firmware. The firmware controls the power to the spindle motor and the speed of the motor. It also manages the communication between the drive and the host. In addition, it controls the R/W operations by moving the actuator arm and switching between different R/W heads, and performs the optimization of data access.
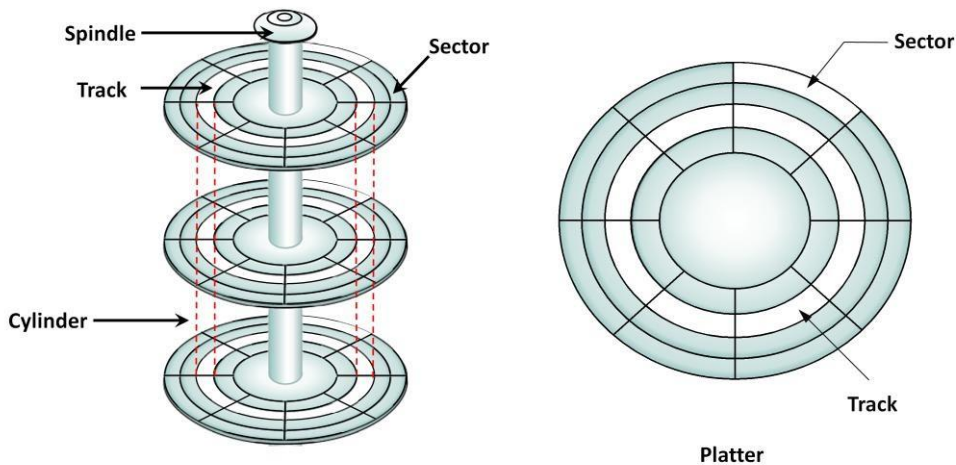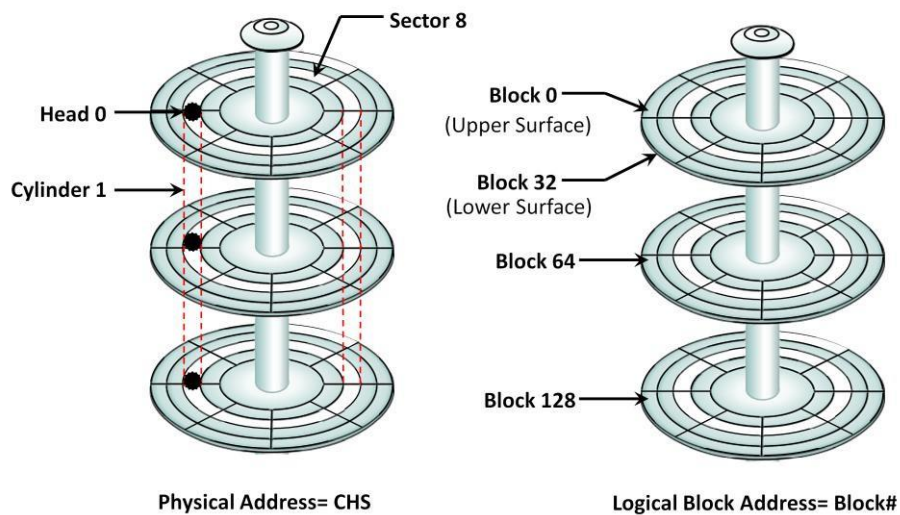
Data on the disk is recorded on *tracks*, which are concentric rings on the platter around the spindle. The tracks are numbered, starting from zero, from the outer edge of the platter. The number of *tracks per inch* (TPI) on the platter (or the *track density*) measures how tightly the tracks are packed on a platter.

Each track is divided into smaller units called *sectors*. A sector is the smallest, individually addressable unit of storage. The track and sector structure is written on the platter by the drive manufacturer using a low-level formatting operation. The number of sectors per track varies according to the drive type. The first personal computer disks had 17 sectors per track. Recent disks have a much larger number of sectors on a single track. There can be thousands of tracks on a platter, depending on the physical dimensions and recording density of the platter.

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

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

Logical Block Addressing

Sector 8

Head 0

Cylinder 1

Block 0
(Upper Surface)

Block 32
(Lower Surface)

Block 64

Block 128

Physical Address= CHS

Logical Block Address= Block#

Module 2: Data Center Environment 29

Earlier drives used physical addresses consisting of the *cylinder, head, & sector (CHS)* number to refer to specific locations on the disk, and the host operating system had to be aware of the geometry of each disk used. *Logical block addressing* (LBA) has simplified the addressing by using a linear address to access physical blocks of data. The disk controller translates LBA to a CHS address, and the host needs to know only the size of the disk drive in terms of the number of blocks. The logical blocks are mapped to physical sectors on a 1:1 basis.

In the slide, the drive shows eight sectors per track, six heads, and four cylinders. This means a total of 8 × 6 × 4 = 192 blocks, so the block number ranges from 0 to 191. Each block has its own unique address.

Assuming that the sector holds 512 bytes, a 500-GB drive with a formatted capacity of 465.7 GB has in excess of 976,000,000 blocks.

Disk Drive Performance

- Electromechanical device
  - Impacts the overall performance of the storage system
- Disk service time
  - Time taken by a disk to complete an I/O request, depends on:
    - Seek time
   - Rotational latency
    - Data transfer rate

Disk service time = seek time + rotational latency + data transfer time

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

- Seek time
- Rotational latency
- Data transfer rate

Seek Time

- Time taken to position the read/write head
- The lower the seek time, the faster the I/O operation
- Seek time specifications include
  - Full stroke
  - Average
  - Track-to-track
- The seek time of a disk is specified by the drive manufacturer
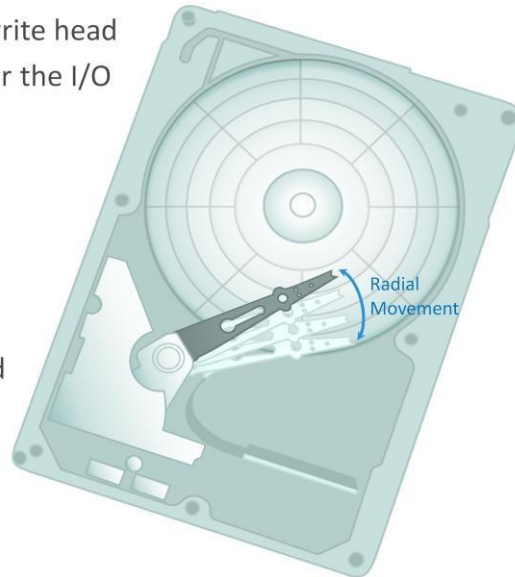
Radial Movement

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

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

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

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

Each of these specifications is measured in milliseconds. The seek time of a disk is typically specified by the drive manufacturer. The average seek time on a modern disk is typically in the range of 3 to 15 milliseconds. Seek time has more impact on the I/O operation of random tracks rather than the adjacent tracks. To minimize the seek time, data can be written to only a subset of the available cylinders. This results in lower usable capacity than the actual capacity of the drive. For example, a 500-GB disk drive is set up to use only the first 40 percent of the cylinders and is effectively treated as a 200-GB drive. This is known as *short-stroking* the drive.

Rotational Latency

- The time taken by the platter to rotate and position the data under the R/W head
- Depends on the rotation speed of the spindle
- Average rotational latency
  - One-half of the time taken for a full rotation
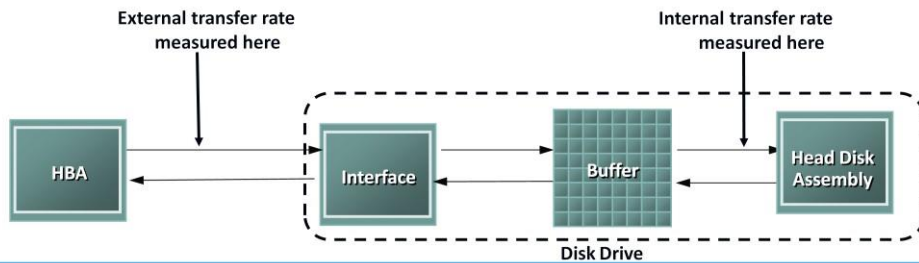 - For 'X' rpm, drive latency is calculated in milliseconds as:

$$= \frac{1/2}{(X/60)}$$
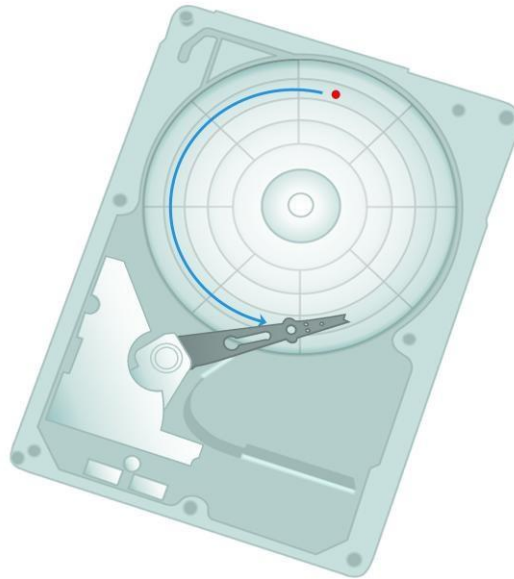
To access data, the actuator arm moves the R/W head over the platter to a particular track while the platter spins to position the requested sector under the R/W head. The time taken by the platter to rotate and position the data under the R/W head is called *rotational latency*. This latency depends on the rotation speed of the spindle and is measured in milliseconds. The average rotational latency is one-half of the time taken for a full rotation. Similar to the seek time, rotational latency has more impact on the reading/writing of random sectors on the disk than on the same operations on adjacent sectors.

Average rotational latency is approximately 5.5 ms for a 5,400-rpm drive, and around 2.0 ms for a 15,000-rpm (or 250-rps revolution per second) drive as shown here. Av. rotational latency for 15K rpm or 250 rps (15000/60) drive is = $(1/2)/250=2$ milliseconds

Data Transfer Rate

- Average amount of data per unit time that the drive can deliver to the HBA
  - Internal transfer rate : Speed at which data moves from a platter's surface to the internal buffer of the disk
  - External transfer rate: Rate at which data move through the interface to the HBA

External transfer rate measured here

Internal transfer rate measured here

HBA → Interface → Buffer → Head Disk Assembly

Disk Drive

The *data transfer rate* (also called *transfer rate*) refers to the average amount of data per unit time that the drive can deliver to the HBA. In a *read operation*, the data first moves from disk platters to R/W heads; then it moves to the drive's internal *buffer*. Finally, data moves from the buffer through the interface to the host HBA. In a *write operation*, the data moves from the HBA to the internal buffer of the disk drive through the drive's interface. The data then moves from the buffer to the R/W heads. Finally, it moves from the R/W heads to the platters. The data transfer rates during the R/W operations are measured in terms of internal and external transfer rates, as shown in the slide.

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

I/O Controller Utilization Vs. Response Time

- Based on fundamental laws of disk drive performance:

$$Av. Response\ Time = \frac{Service\ Time}{(1 - Utilization)}$$

  ▸ Service time is time taken by the controller to serve an I/O
- For performance-sensitive applications disks are commonly utilized below 70% of their I/O serving capability
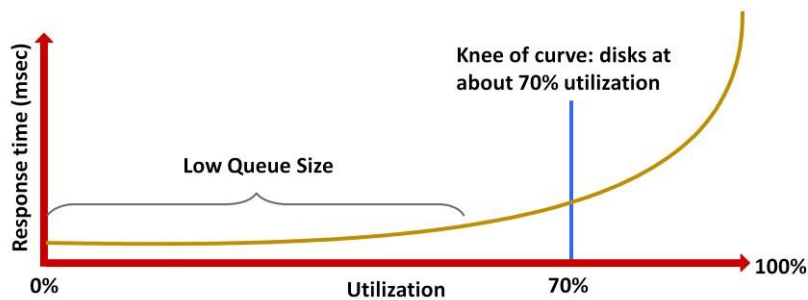
Utilization of a disk I/O controller has a significant impact on the I/O response time. Consider that a disk is viewed as a black box consisting of two elements queue and disk I/O controller. *Queue* is the location where an I/O request waits before it is processed by the I/O controller and *disk I/O controller* processes I/Os waiting in the queue one by one.

The I/O requests arrive at the controller at the rate generated by the application. The I/O arrival rate, the queue length, and the time taken by the I/O controller to process each request determines the I/O response time. If the controller is busy or heavily utilized, the queue size will be large and the response time will be high. Based on the fundamental laws of disk drive performance, the relationship between controller utilization and average response time is given as:

Average response time = Service time / (1 – Utilization)

where, service time is the time taken by the controller to serve an I/O.

As the utilization reaches 100 percent that is, as the I/O controller saturates, the response time is closer to infinity. In essence, the saturated component, or the bottleneck, forces the serialization of I/O requests; meaning, each I/O request must wait for the completion of the I/O requests that preceded it. Figure in the slide shows a graph plotted between utilization and response time. The graph indicates that the response time changes are nonlinear as the utilization increases. When the average queue sizes are low, the response time remains low. The response time increases slowly with added load on the queue and increases exponentially when the utilization exceeds 70 percent. Therefore, for performance-sensitive applications, it is common to utilize disks below their 70 percent of I/O serving capability.

Storage Design Based on Application Requirements and Disk Drive Performance

- Disks required to meet an application's capacity need (D_C):

$$D_C = \frac{Total\ capacity\ required}{Capacity\ of\ a\ single\ disk}$$

- Disks required to meet application's performance need (D_P):

$$D_P = \frac{IOPS\ generated\ by\ an\ application\ at\ peak\ workload}{IOPS\ serviced\ by\ single\ disk}$$

- IOPS serviced by a disk (S) depends upon disk service time (T_S):

$$T_S = Seek\ time + \frac{0.5}{(Disk\ rpm/60)} + \frac{Data\ block\ size}{Data\ transfer\ rate}$$

  - T_S is time taken for an I/O to complete, therefore IOPS serviced by a disk (S) is equal to ($1/T_S$)

    - For performance sensitive application $(S) = 0.7 \times \frac{1}{T_S}$

> Disk required for an application = max (D_C,D_P)

Determining storage requirements for an application begins with determining the required storage capacity and I/O performance. Capacity can be easily estimated by the size and number of file systems and database components used by applications. The I/O size, I/O characteristics, and the number of I/Os generated by the application at peak workload are other factors that affect performance, I/O response time and design of storage system.

The disk service time (T_S) for an I/O is a key measure of disk performance; T_S, along with disk utilization rate (U), determines the I/O response time for an application. As discussed earlier the total disk service time is the sum of the seek time, rotational latency, and transfer time.

Note that transfer time is calculated based on the block size of the I/O and given data transfer rate of a disk drive—for example, an I/O with a block size of 32 KB and given disk data transfer rate 40MB/s; the transfer time will be 32 KB / 40 MB.

T_S determines the time taken by the I/O controller to serve an I/O, therefore, the maximum number of I/Os serviced per second or IOPS is ($1/T_S$).

The IOPS calculated above represents the IOPS that can be achieved at potentially high levels of I/O controller utilization (close to 100 percent). If the application demands a faster response time, then the utilization for the disks should be maintained below 70 percent.

Based on this discussion, the total number of disks required for an application is computed as :

= Max (Disks required for meeting capacity, Disks required for meeting performance)

Cont..

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

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

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

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

The preceding example indicates that from a capacity-perspective, 10 disks are sufficient; however, the number of disks required to meet application performance is 72. To optimize disk requirements from a performance perspective, various solutions are deployed in a real-time environment. Examples of these solutions are disk native command queuing, use of flash drives, RAID, and the use of cache memory. RAID and cache are detailed in module 3 and 4 respectively.

Enterprise Flash Drives

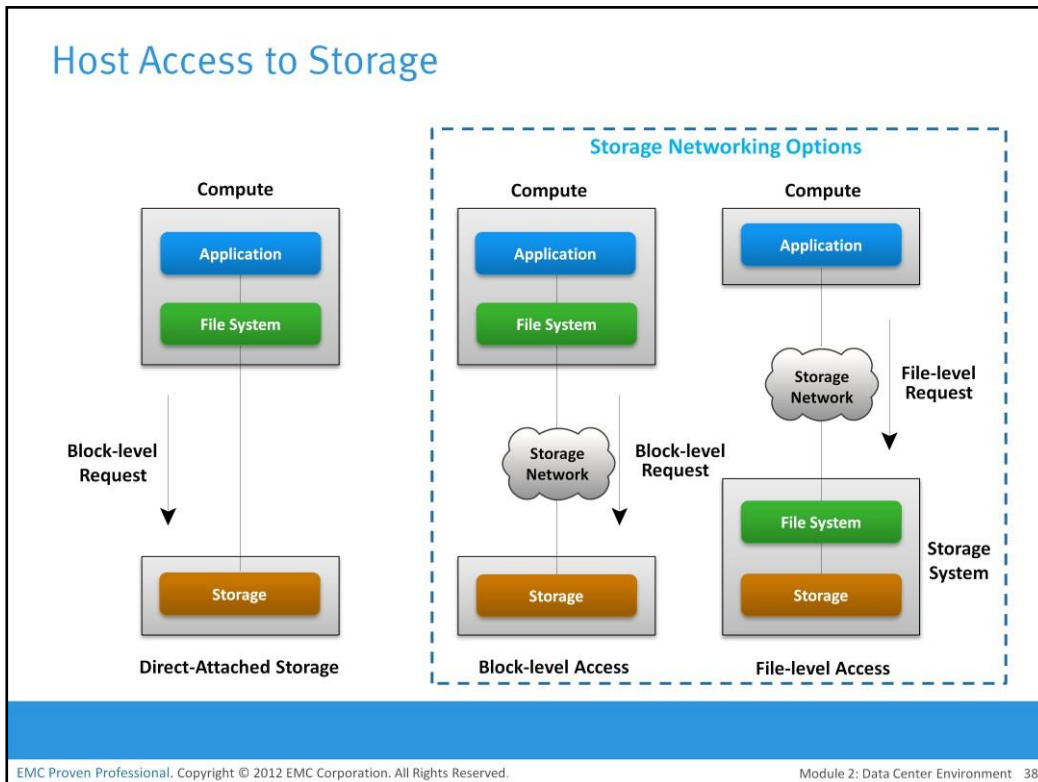| Conventional Hard Drives	Flash Drives
Mechanical delay due to seek time and rotational latency	Highest possible throughput per drive due to no mechanical movement
Limited performance and I/O serving capability	Very low latency per I/O and consistent I/O performance
More power consumption due to mechanical operations	High Energy efficiency • Lower power requirement per GB • Lower power requirement per IOPS
Low mean time between failure (MTBF)	High reliability due to no moving parts
Higher TCO due to more number of disks, power, cooling, and management cost	Overall less TCO

Traditionally, high I/O requirements of an application were met by simply using more disks. Availability of enterprise class flash drives (EFD) has changed the scenario.

Flash drives, also referred as solid state drives (SSDs), are new generation drives that deliver ultra-high performance required by performance-sensitive applications. Flash drives use semiconductor-based solid state memory (flash memory) to store and retrieve data. Unlike conventional mechanical disk drives, flash drives contain no moving parts; therefore, they do not have seek and rotational latencies. Flash drives deliver a high number of IOPS with very low response times. Also, being a semiconductor-based device, flash drives consume less power, compared to mechanical drives. Flash drives are especially suited for applications with small block size and random-read workloads that require consistently low (less than 1 ms) response times. Applications that need to process massive amounts of information quickly, such as currency exchange, electronic trading systems, and real-time data feed processing, benefit from flash drives.

Overall, flash drives provide better total cost of ownership (TCO) even though they cost more on $/GB basis. By implementing flash drives, businesses can meet application performance requirements with far fewer drives (approximately 20 to 30 times less number of drives compared to conventional mechanical drives). This reduction not only provides savings in terms of drive cost, but also translates to savings for power, cooling, and space consumption. Fewer numbers of drives in the environment also means less cost for managing the storage.

Host Access to Storage

Storage Networking Options

Compute | Compute | Compute

Application | File System

Block-level Request

Storage

Direct-Attached Storage

Block-level Request

Storage Network

Storage

Block-level Access

File-level Request

Storage Network

File System | Storage

Storage System

File-level Access

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

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

In a block-level access, the file system is created on a host, and data is accessed on a network at the block level. In this case, raw disks or logical volumes are assigned to the host for creating the file system.

In a file-level access, the file system is created on a separate file server or at the storage side, and the file-level request is sent over a network. Because data is accessed at the file level, this method has higher overhead, as compared to the data accessed at the block level.
Object-level access is an intelligent evolution, whereby data is accessed over a network in terms of self-contained objects with a unique object identifier. Details of storage networking technologies and deployments are covered in later modules of this course.

Direct-Attached Storage (DAS)
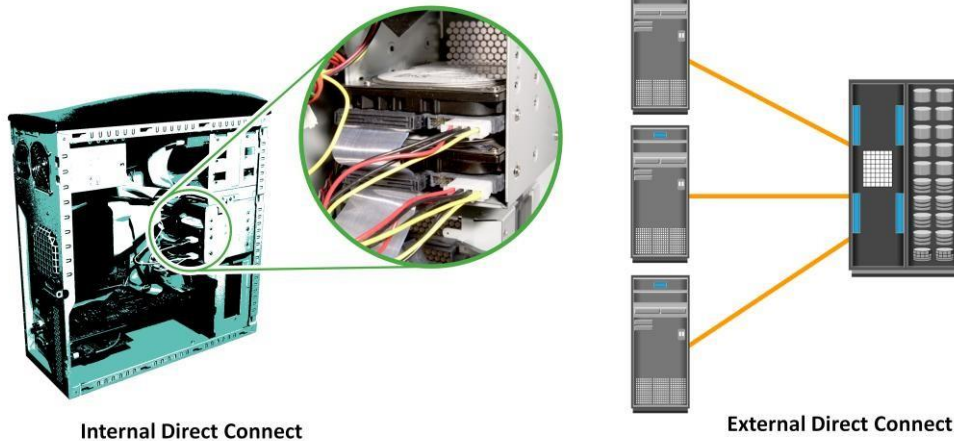
Internal Direct Connect

External Direct Connect

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

In *internal DAS* architectures, the storage device is internally connected to the host by a serial or parallel bus. The physical bus has distance limitations and can be sustained only over a shorter distance for high-speed connectivity. In addition, most internal buses can support only a limited number of devices, and they occupy a large amount of space inside the host, making maintenance of other components difficult. In *external DAS* architectures, the host connects directly to the external storage device, and data is accessed at the block level. In most cases, communication between the host and the storage device takes place over a SCSI or FC protocol. Compared to internal DAS, an external DAS overcomes the distance limitations and provides centralized management of storage devices.

DAS Benefits and limitations: DAS requires a relatively lower initial investment than storage networking architectures. The DAS configuration is simple and can be deployed easily and rapidly. It requires fewer management tasks and less hardware and software elements to set up and operate. However, DAS does not scale well. A storage array has a limited number of ports, which restricts the number of hosts that can directly connect to the storage.
Therefore, DAS does not make optimal use of resources and, moreover unused resources cannot be easily re-allocated, resulting in islands of over-utilized and under-utilized storage pools.

Module 2: Data Center Environment

Concept in Practice

- VMware ESXi

The concept in practice covers the product example of compute virtualization. It covers industry's leading hypervisor software VMware ESXi.

VMware ESXi

- Industry's leading hypervisor
  - Enable virtualization of x86 hardware platforms
- Physical machine that houses ESXi is called ESXi host
  - ESXi host abstracts physical compute resources to run multiple VMs concurrently on same physical server
- Two Components
  - VMKernel
    - Work similar to OS – responsible for process creation, resource scheduling, and so on
  - Virtual machine monitor
    - Performs binary translation for privileged OS instructions that can not be virtualized

VMware is the leading provider for server virtualization solution. VMware ESXi provides a platform called hypervisor. The hypervisor abstracts CPU, memory, and storage resources to run multiple virtual machines concurrently on the same physical server.

VMware ESXi is a hypervisor that installs on x86 hardware to enable server virtualization. It enables creating multiple virtual machines (VMs) that can run simultaneously on the same physical machine. A VM is a discrete set of files that can be moved, copied, and used as a template. All the files that make up a VM are typically stored in a single directory on a cluster file system called Virtual Machine File System (VMFS). The physical machine that houses ESXi is called ESXi host. The ESXi hosts provide physical resources used to run virtual machines.
ESXi has two key components: VMkernel and Virtual Machine Monitor.

VMkernel provides functionality similar to that found in other operating systems, such as process creation, file system management, and process scheduling. It is designed to specifically support running multiple VMs and provide core functionality such as resource scheduling, I/O stacks, and so on.

The virtual machine monitor is responsible for executing commands on the CPUs and performing Binary Translation (BT). A virtual machine monitor performs hardware abstraction to appear as a physical machine with its own CPU, memory, and I/O devices. Each VM is assigned a virtual machine monitor that has a share of the CPU, memory, and I/O devices to successfully run the VM.

Module 2: Summary

Key points covered in this module:
- Key data center elements
- Application and compute virtualization
- Disk drive components and performance
- Enterprise flash drives
- Host access to storage

This module covered the key elements of a data center – application, DBMS, compute, network, and storage.

It also covered virtualization at application and compute that enable better utilization of resources and ease of management. This module also elaborated on disk drive components and factors governing disk drive performance. It also covered enterprise flash drives that are superior to mechanical disk drives in many ways. This module also covered various options of host to storage access with focus on DAS.

Check Your Knowledge – 1

- Which is a benefit of compute virtualization?
    - A. Enables compute memory swapping
    - B. Improves compute utilization
    - C. Isolates compute memory from the applications
    - D. Isolates compute OS from the applications

- What best describes virtual machines (VMs)?
    - A. All VMs on a physical server must run same OS
    - B. VM files are deleted when VM is powered off
    - C. VMs are discrete sets of files
    - D. All VMs share available resources equally

Check Your Knowledge – 2

- What is concatenation?
    - A. Grouping multiple physical drives into a logical drive
    - B. Dividing a physical drive into multiple logical drives
    - C. Process of writing disk metadata on a logical drive
    - D. Adding more capacity to a physical drive through de-fragmentation

- Which factors contribute to the overall service time of a mechanical disk?
    - A. Disk buffer time, full stroke, and rotation latency
    - B. Internal transfer rate, external transfer rate, and buffer time
    - C. Full stroke, average seek time, and track-to-track seek time
    - D. Average seek time, rotational latency, and data transfer rate

Check Your Knowledge – 3

- Which is a challenge of DAS environment?
    - A. Low performance
    - B. Limited scalability
    - C. Deployment complexity
    - D. High initial investment

Exercise: Design Storage Solution for New Application

- Scenario
  - Characteristics of new application:
    - Require 1TB of storage capacity
    - Peak I/O workload 4900 IOPS
    - Typical I/O size is 4KB
  - Specifications of the available disk drives:
    - 15K rpm drive with storage capacity = 100 GB
    - Average seek time = 5ms
    - Data transfer rate = 40 MB/sec
  - As it is business critical application, response time must be within acceptable range
- Task
  - Calculate the number of disks required for the application

An organization is deploying a new business application in their environment. The new application requires 1TB of storage space for business and application data. During peak workload, application is expected to generate 4900 IOPS (I/O per second) with typical I/O size of 4KB.

The available disk drive option is 15,000 rpm drive with 100 GB capacity. Other specification of the drives are:

Av. Seek time = 5 millisecond and data transfer rate = 40MB/sec.

You are asked to calculate the required number of disk drives that can meet both capacity and performance requirements of an application.