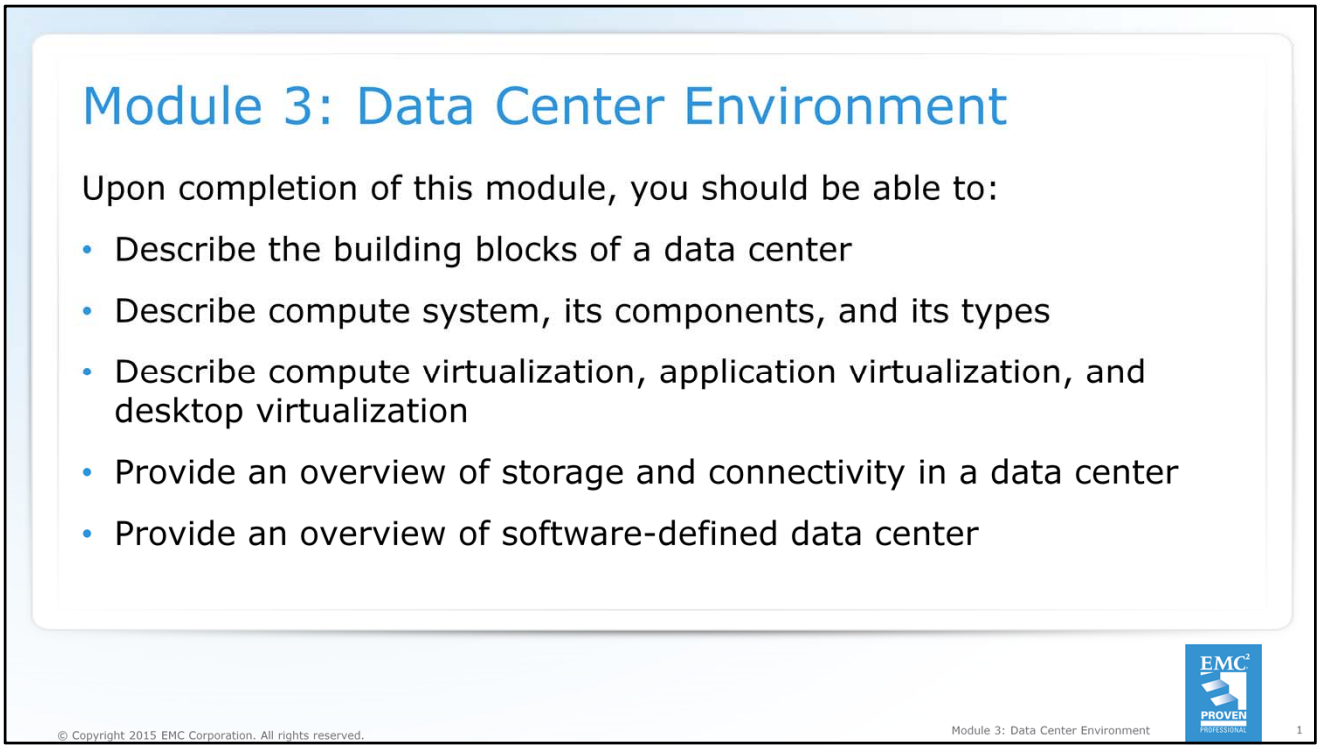
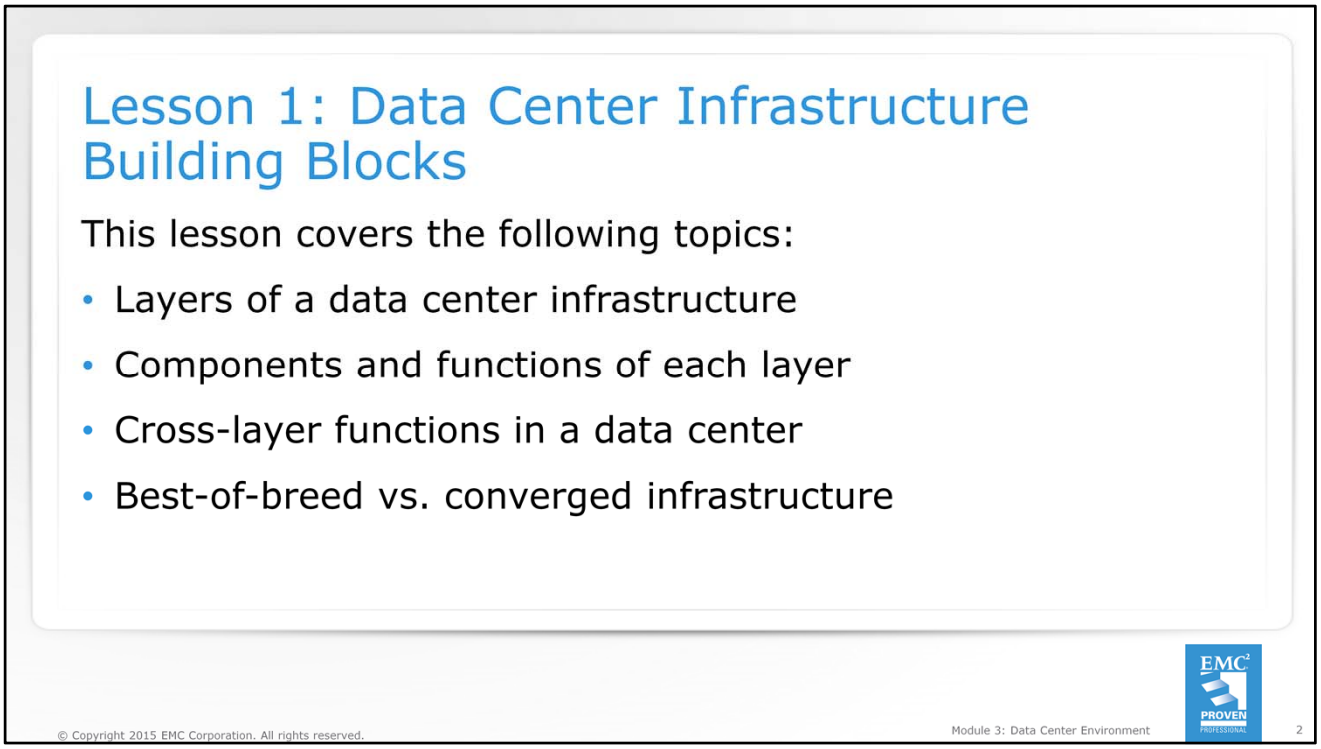
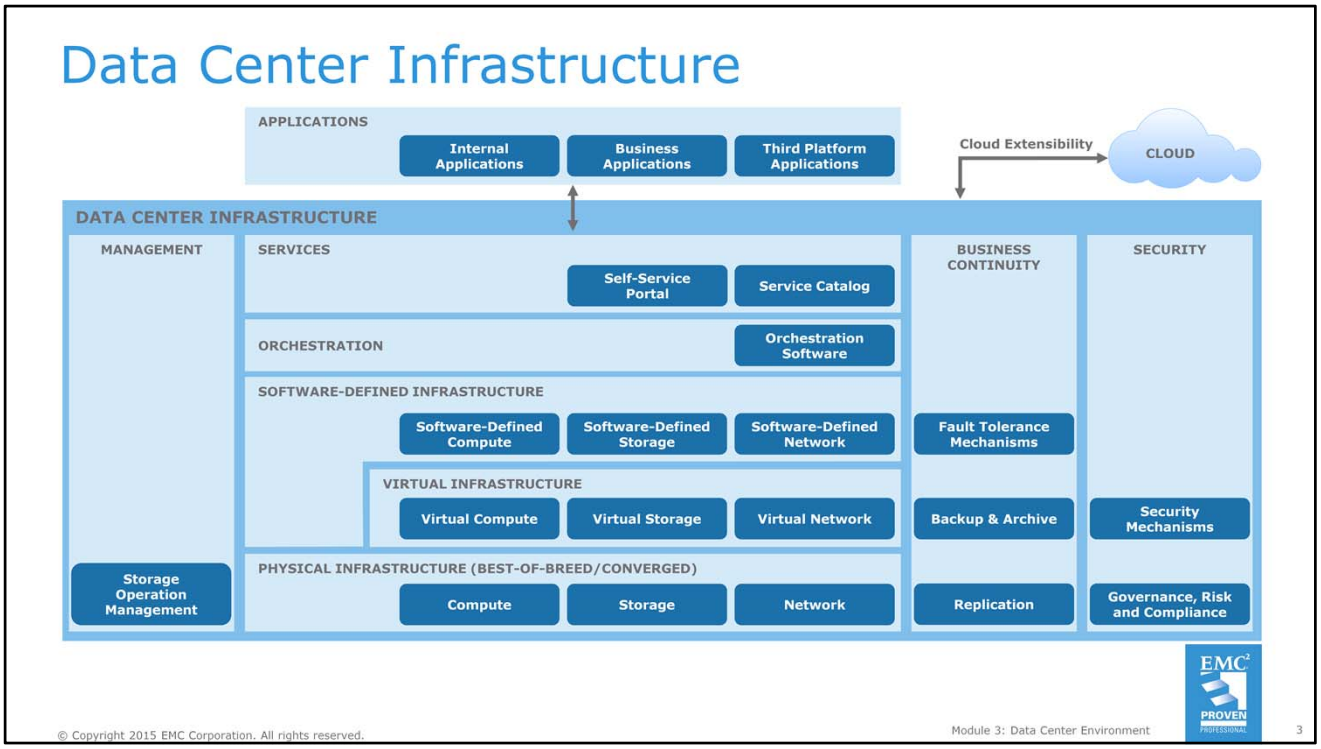


This module focuses on the building blocks of a data center environment. This module also focuses on compute system, its components, and its types. Additionally, this module focuses on compute virtualization, application virtualization, and desktop virtualization. Further, this module focuses on an overview of storage and connectivity in a data center. Finally, this module focuses on an overview of software-defined data center.



This lesson covers the building blocks of a data center infrastructure. It covers the components and functions of the five layers of a data center. It also covers the three cross-layer functions in a data center. Further, this lesson covers best-of-breed versus converged infrastructure.

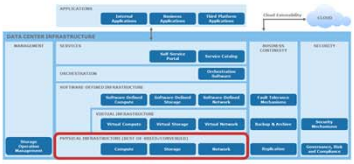


Module 1, 'Introduction to Information Storage', defined data center and specified the IT infrastructure and support infrastructure that comprise a data center. The figure on the slide is a block diagram depicting the core IT infrastructure building blocks that make up a data center. The IT infrastructure is arranged in five logical layers and three cross-layer functions. The five layers are physical infrastructure, virtual infrastructure, software-defined infrastructure, orchestration, and services. Each of these layers has various types of hardware and/or software components as shown in the figure. The three cross-layer functions are business continuity, security, and management. Business continuity and security functions include mechanisms and processes that are required to provide reliable and secure access to applications, information, and services. The management function includes various processes that enable the efficient administration of the data center and the services for meeting business requirements.

The building blocks depicted in the figure may be implemented in part or in whole to create either a second platform, platform 2.5, or a third platform data center. Applications that are deployed in the data center may be a combination of internal applications, business applications, and third platform applications that are either custom-built or off-the-shelf. By ensuring the fulfillment of the five essential cloud characteristics, the infrastructure can be transformed into a cloud infrastructure that could be either private or public. Further, by integrating cloud extensibility, the infrastructure can be connected to an external cloud to leverage the hybrid cloud model.

Data Center Infrastructure

Physical Infrastructure

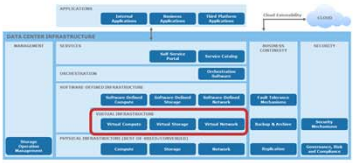


- Foundation layer of the data center infrastructure
- Physical components:
  - Compute systems, storage, and network devices
    - Require operating systems, system software, and protocols for their functions
- Executes the requests generated by the virtual and software-defined layers

The physical infrastructure forms the foundation layer of a data center. It includes equipment such as compute systems, storage systems, and networking devices along with the operating systems, system software, protocols, and tools that enable the physical equipment to perform their functions. A key function of physical infrastructure is to execute the requests generated by the virtual and software-defined infrastructure, such as storing data on the storage devices, performing compute-to-compute communication, executing programs on compute systems, and creating backup copies of data. Compute systems are covered later in this module. Different storage systems are covered in Modules 4, 'Intelligent Storage Systems (ISS)', 5, 'Block-based Storage System', 6, 'File-based Storage System (NAS)', and 7, 'Object-based and Unified Storage'. Networking is covered in Modules 9, 'Fibre Channel (FC) SAN', 10, 'Internet Protocol (IP) SAN', and 11, 'FC over Ethernet (FCoE) SAN'.

Data Center Infrastructure

Virtual Infrastructure



- Virtualization abstracts physical resources and creates virtual resources
- Virtual components:
  - Virtual compute, virtual storage, and virtual network
    - Created from physical resource pools using virtualization software
- Benefits of virtualization:
  - Resource consolidation and multitenant environment
  - Improved resource utilization and increased ROI
  - Flexible resource provisioning and rapid elasticity

Virtualization is the process of abstracting physical resources, such as compute, storage, and network, and creating virtual resources from them. Virtualization is achieved through the use of virtualization software that is deployed on compute systems, storage systems, and network devices. Virtualization software aggregates physical resources into resource pools from which it creates virtual resources. A resource pool is an aggregation of computing resources, such as processing power, memory, storage, and network bandwidth. For example, storage virtualization software pools the capacity of multiple storage devices to create a single large storage capacity. Similarly, compute virtualization software pools the processing power and memory capacity of a physical compute system to create an aggregation of the power of all processors (in megahertz) and all memory (in megabytes). Examples of virtual resources include virtual compute (virtual machines), virtual storage (LUNs), and virtual networks.

Virtualization enables a single hardware resource to support multiple concurrent instances of systems, or multiple hardware resources to support a single instance of system. For example, a single disk drive can be partitioned and presented as multiple disk drives to a compute system. Similarly, multiple disk drives can be concatenated and presented as a single disk drive to a compute system. With virtualization, it is also possible to make a resource appear larger or smaller than it actually is.

Virtualization offers several benefits in a data center. It enables the consolidation of physical IT resources, and supports a multitenant environment. This optimizes the utilization of physical resources that, in turn, results in an increased return-on-investment (ROI) and enables reducing the costs of purchasing of new hardware. Virtualization also reduces space and energy requirements and simplifies infrastructure management. It also increases the flexibility of resource provisioning through the dynamic creation and reclamation of virtual resources. Virtualization is a key enabling technology to meet the resource pooling and rapid elasticity characteristics of cloud computing.

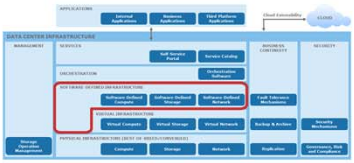
(Cont'd)

Compute virtualization is covered later in this module, while different storage virtualization and network virtualization techniques are covered later in the course in the storage modules and network modules respectively.

Note: While deploying a data center, an organization may choose not to deploy virtualization. In such an environment, the software-defined layer is deployed directly over the physical infrastructure. Further, it is also possible that part of the infrastructure is virtualized and rest is not virtualized.

Data Center Infrastructure

Software-defined Infrastructure

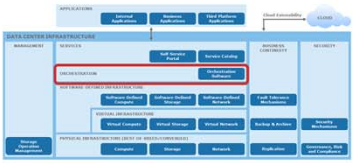


- Deployed either on virtual layer or on physical layer
- All infrastructure components are virtualized and aggregated into pools
  - Underlying resources are abstracted from applications
  - Enables ITaaS
- Centralized, automated, and policy-driven management and delivery of heterogeneous resources
- Components:
  - Software-defined compute
  - Software-defined storage
  - Software-defined network

The software-defined infrastructure layer is deployed either on the virtual layer or on the physical layer. In the software-defined approach, all infrastructure components are virtualized and aggregated into pools. This abstracts all underlying resources from applications. The software-defined approach enables ITaaS, in which consumers provision all infrastructure components as services. It centralizes and automates the management and delivery of heterogeneous resources based on policies. The key architectural components in the software-defined approach include software-defined compute (equivalent to compute virtualization), software-defined storage (SDS), and software-defined network (SDN). Software-defined data center is covered later in this module. Software-defined storage is covered in Module 8, whereas software-defined network is covered in the network modules.

Data Center Infrastructure

Orchestration



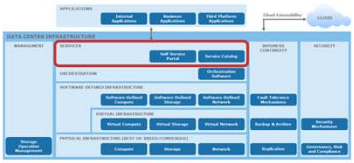
- Component:
  - Orchestration software
- Provides workflows for executing automated tasks
- Interacts with various components across layers and functions to invoke provisioning tasks

The orchestration layer includes the orchestration software. The key function of this layer is to provide workflows for executing automated tasks to accomplish a desired outcome. Workflow refers to a series of inter-related tasks that perform a business operation. The orchestration software enables this automated arrangement, coordination, and management of the tasks. This helps to group and sequence tasks with dependencies among them into a single, automated workflow.

Associated with each service listed in the service catalog, there is an orchestration workflow defined. When a service is selected from the service catalog, an associated workflow in the orchestration layer is triggered. Based on this workflow, the orchestration software interacts with the components across the software-defined layer and the BC, security, and management functions to invoke the provisioning tasks to be executed by the entities.

Data Center Infrastructure

Services

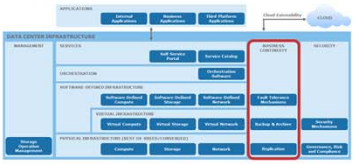


- Delivers IT resources as services to users
  - Enables users to achieve desired business results
  - Users have no liabilities associated with owning the resources
- Components:
  - Service catalog
  - Self-service portal
- Functions of service layer:
  - Stores service information in service catalog and presents them to the users
  - Enables users to access services via a self-service portal

Similar to a cloud service, an IT *service* is a means of delivering IT resources to the end users to enable them to achieve the desired business results and outcomes without having any liabilities such as risks and costs associated with owning the resources. Examples of services are application hosting, storage capacity, file services, and email. The service layer is accessible to applications and end users. This layer includes a service catalog that presents the information about all the IT resources being offered as services. The service catalog is a database of information about the services and includes a variety of information about the services, including the description of the services, the types of services, cost, supported SLAs, and security mechanisms. The provisioning and management requests are passed on to the orchestration layer, where the orchestration workflows—to fulfill the requests—are defined.

Data Center Infrastructure

Business Continuity



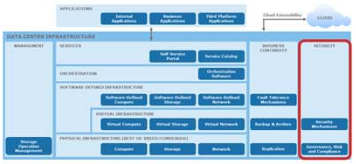
- Enables ensuring the availability of services in line with SLA
- Supports all the layers to provide uninterrupted services
- Includes adoption of measures to mitigate the impact of downtime

Measure	Description
Proactive	• Business impact analysis • Risk assessment • Technology solutions deployment (backup and replication)
Reactive	• Disaster recovery • Disaster restart

The business continuity (BC) cross-layer function specifies the adoption of proactive and reactive measures that enable an organization to mitigate the impact of downtime due to planned and unplanned outages. The proactive measures include activities and processes such as business impact analysis, risk assessment, and technology solutions such as backup, archiving, and replication. The reactive measures include activities and processes such as disaster recovery and disaster restart to be invoked in the event of a service failure. This function supports all the layers—physical, virtual, software-defined, orchestration, and services—to provide uninterrupted services to the consumers. The BC cross-layer function of a cloud infrastructure enables a business to ensure the availability of services in line with the service level agreement (SLA). BC and BC solutions are covered in Modules 12, 'Introduction to Business Continuity', 13, 'Backup and Archive', and 14, 'Replication'.

Data Center Infrastructure

Security



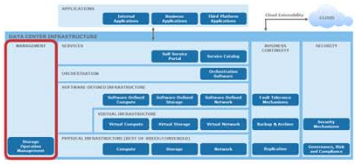
- Supports all the layers to provide secure services
- Specifies the adoption of:
  - Administrative mechanisms
    - Security and personnel policies
    - Standard procedures to direct safe execution of operations
  - Technical mechanisms
    - Firewall
    - Intrusion detection and prevention systems
    - Antivirus
- Security mechanisms enable to meet governance, risk, and compliance (GRC) requirements

The security cross-layer function supports all the infrastructure layers—physical, virtual, software-defined, orchestration, and service—to provide secure services to the consumers. Security specifies the adoption of administrative and technical mechanisms that mitigate or minimize the security threats and provide a secure data center environment. Administrative mechanisms include security and personnel policies or standard procedures to direct the safe execution of various operations. Technical mechanisms are usually implemented through tools or devices deployed on the IT infrastructure. Examples of technical mechanisms include firewall, intrusion detection and prevention systems, and antivirus software.

Governance, risk, and compliance (GRC) specifies processes that help an organization in ensuring that their acts are ethically correct and in accordance with their risk appetite (the risk level an organization chooses to accept), internal policies, and external regulations. Security mechanisms should be deployed to meet the GRC requirements. Security and GRC are covered in Module 15, ‘Securing the Storage Infrastructure’.

Data Center Infrastructure

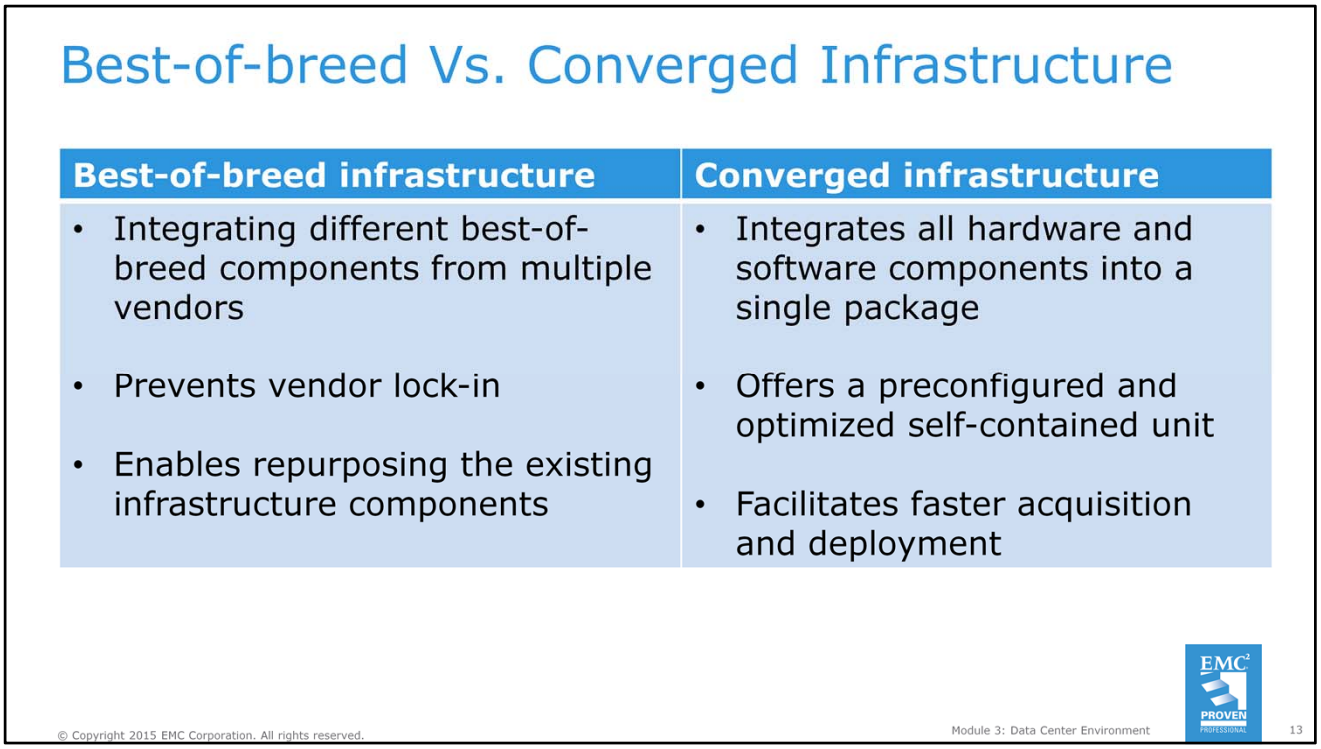
Management



- Enables storage infrastructure configuration and capacity provisioning
- Enables problem resolution
- Enables capacity and availability management
- Enables compliance conformance
- Enables monitoring services

The management cross-layer function specifies the adoption of activities related to data center operations management. Adoption of these activities enables an organization to align the creation and delivery of IT services to meet their business objectives. This course focuses on the aspect of storage infrastructure management.

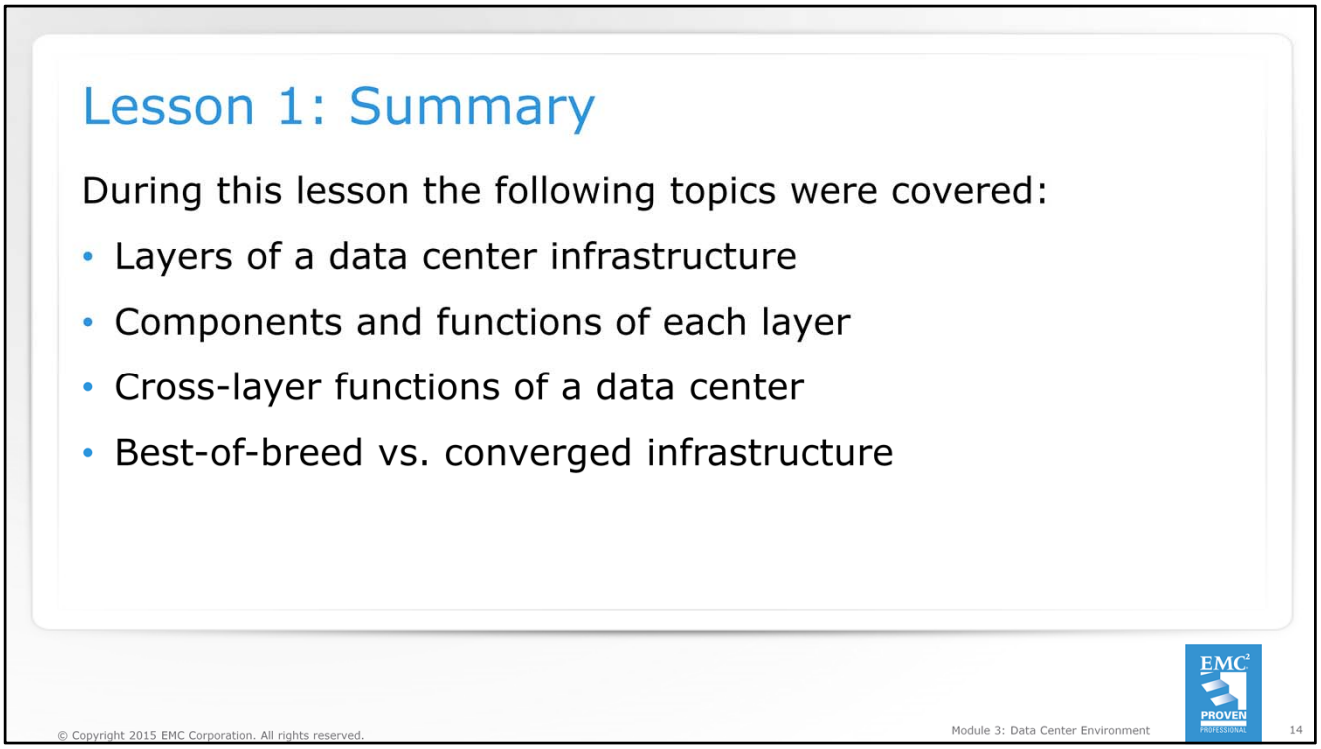
Storage operation management enables IT administrators to manage the data center infrastructure and services. Storage operation management tasks include handling of infrastructure configuration, resource provisioning, problem resolution, capacity, availability, and compliance conformance. This function supports all the layers to perform monitoring, management, and reporting for the entities of the infrastructure. Storage infrastructure management is covered in Module 16, 'Managing the Storage Infrastructure'.



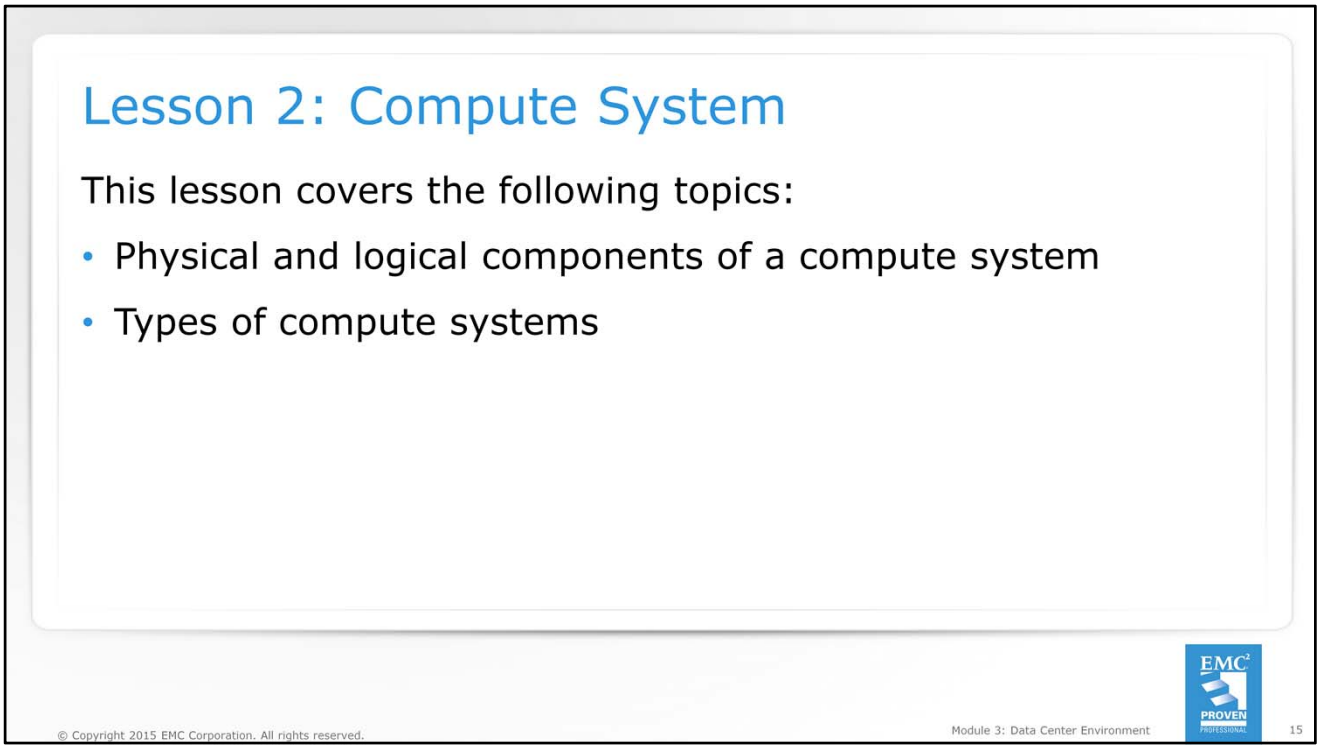
There are two options for building the data center infrastructure – by integrating best-of-breed infrastructure components, or by acquiring and deploying a converged infrastructure.

Best-of-breed infrastructure: In this approach, organizations integrate the best-of-breed infrastructure components (hardware and software) purchased from multiple different vendors. This enables the organizations to leverage the advantages of high quality products and services from the respective leading vendors in the segment. It provides the flexibility to change the individual vendors in case the committed support is not provided and the SLAs are not met. Additionally, this approach allows organizations to repurpose the existing infrastructure components, providing a cost benefit. However, this approach requires significant CAPEX, OPEX, and time as it involves evaluation, purchase, testing, deployment, configuration, and integration of multiple disparate hardware and software components. Further, scaling of such an infrastructure takes longer because each new component goes through the process from evaluation to integration.

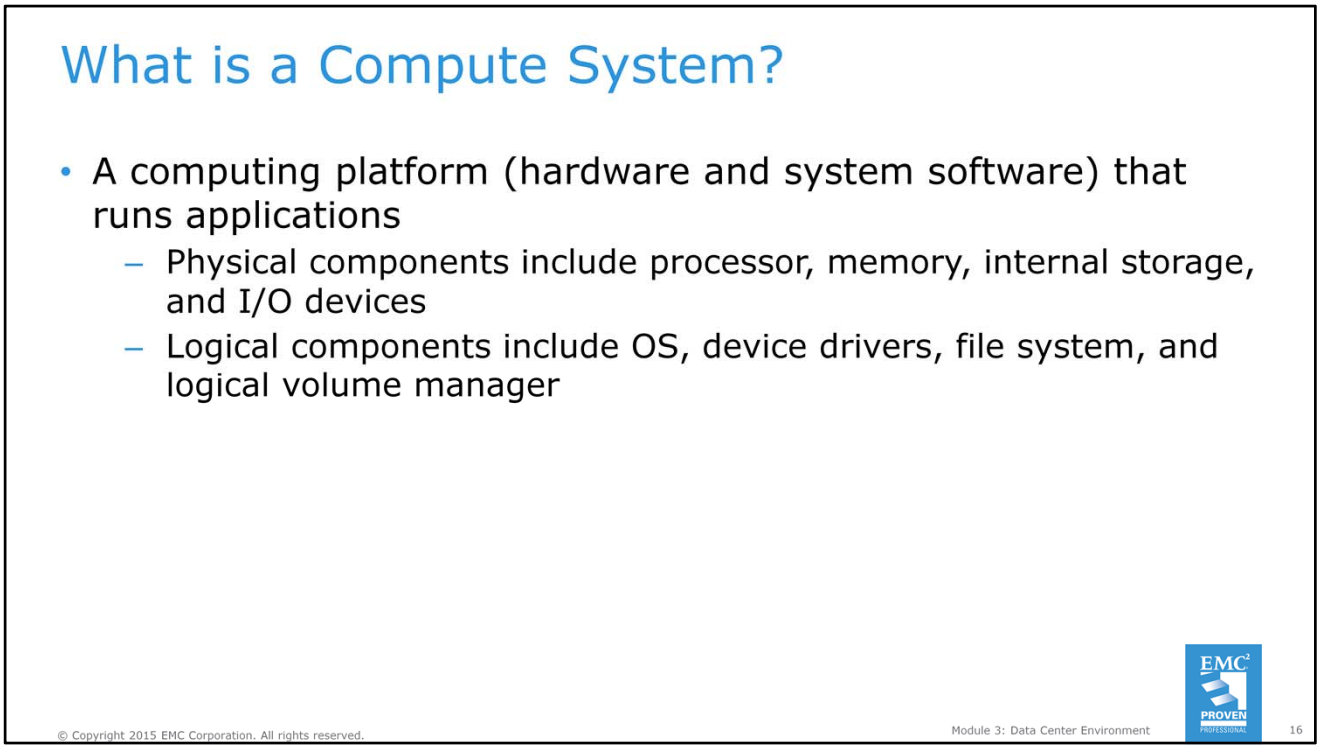
Converged infrastructure: A converged infrastructure integrates hardware and software components that make up a data center into a single packaged solution. This package is a self-contained unit that can be deployed independently, or aggregated with other packages to meet the additional capacity and performance requirements. The package is pre-configured and optimized, which reduces the time to acquire and deploy the infrastructure. It also lowers power and space requirements. Vendors also provide cloud-ready converged infrastructure with built-in capabilities for secure multi-tenancy. Converged infrastructure has a single management software capable of managing all hardware and software within the package. A potential area of concern regarding the converged infrastructure solutions is the lack of flexibility to use infrastructure components from different vendors. Some vendors may provide the flexibility to choose multi-vendor infrastructure components such as network devices, compute systems, and hypervisors for the solution.



This lesson covered the building blocks of a data center infrastructure. It covered the components and functions of the five layers of a data center. It also covered the three cross-layer functions of a data center. Further, this lesson covered best-of-breed versus converged infrastructure.



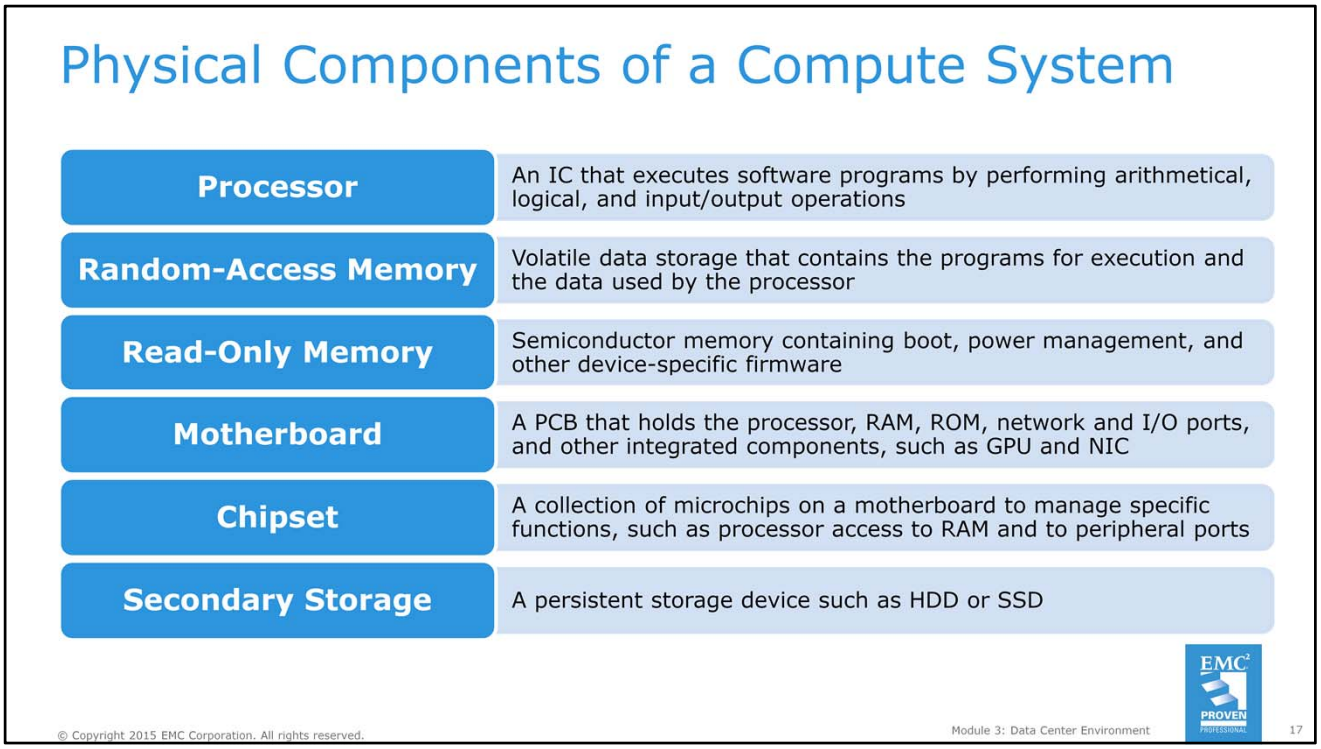
This lesson covers compute system, and its key physical and logical components. This lesson also covers the types of compute systems.



A compute system is a computing device (combination of hardware, firmware, and system software) that runs business applications. Examples of compute systems include physical servers, desktops, laptops, and mobile devices. As mentioned previously in Module 1, 'Introduction to Information Storage' in this course, the term compute system refers to physical servers and hosts on which platform software, management software, and business applications of an organization are deployed.

A compute system's hardware consists of processor(s), memory, internal storage, and I/O devices. The logical components of a compute system include the operating system (OS), file system, logical volume manager, and device drivers. The OS may include the other software or they can be installed individually.

In an enterprise data center, applications are typically deployed on compute clusters for high availability and for balancing computing workloads. A compute cluster is a group of two or more compute systems that function together, sharing certain network and storage resources, and logically viewed as a single system. Compute clustering is covered in detail in Module 12, 'Introduction to Business Continuity'.



A compute system comprises multiple physical hardware components assembled inside a metal enclosure. Some key components are described below.

Processor: A processor, also known as a Central Processing Unit (CPU), is an integrated circuit (IC) that executes the instructions of a software program by performing fundamental arithmetical, logical, and input/output operations. A common processor/instruction set architecture is the x86 architecture with 32-bit and 64-bit processing capabilities. Modern processors have multiple cores (independent processing units), each capable of functioning as an individual processor.

Random-Access Memory (RAM): The RAM or main memory is an IC that serves as a volatile data storage internal to a compute system. The RAM is directly accessible by the processor, and holds the software programs for the execution and the data used by the processor.

Read-Only Memory (ROM): A ROM is a type of non-volatile semiconductor memory from which data can only be read but not written to. It contains the boot firmware (that enables a compute system to start), power management firmware, and other device-specific firmware.

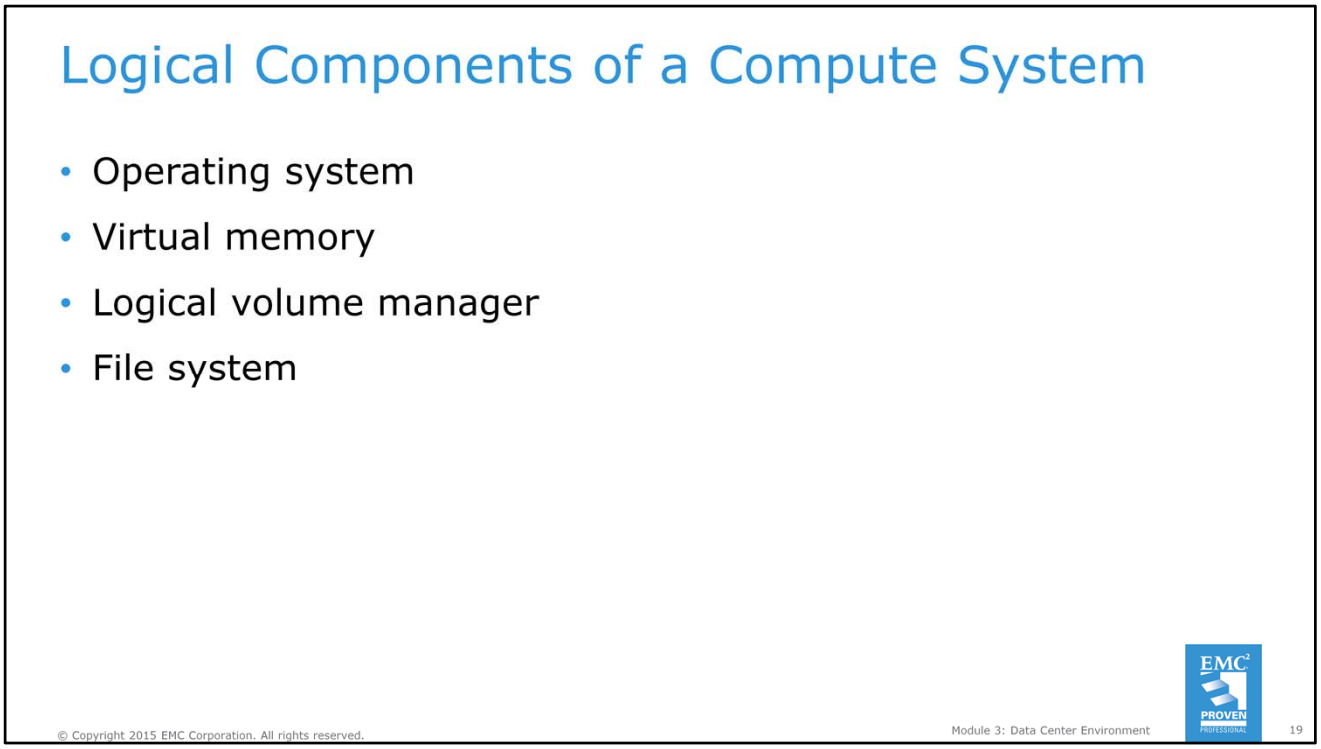
Motherboard: A motherboard is a printed circuit board (PCB) to which all compute system components connect. It has sockets to hold components such as the microprocessor chip, RAM, and ROM. It also has network ports, I/O ports to connect devices such as keyboard, mouse, and printers, and essential circuitry to carry out computing operations. A motherboard may additionally have integrated components, such as a graphics processing unit (GPU), a network interface card (NIC), and adapters to connect to external storage devices. Motherboards (and other internal components) receive power from a power supply unit.

(Cont'd)

Chipset: A chipset is a collection of microchips on a motherboard and it is designed to perform specific functions. The two key chipset types are Northbridge and Southbridge. Northbridge manages processor access to the RAM and the GPU, while Southbridge connects the processor to different peripheral ports, such as USB ports.

Secondary storage: Secondary storage is a persistent storage device, such as a hard disk drive or a solid state drive, on which the OS and the application software are installed. The processor cannot directly access secondary storage. The desired applications and data are loaded from the secondary storage on to the RAM to enable the processor to access them.

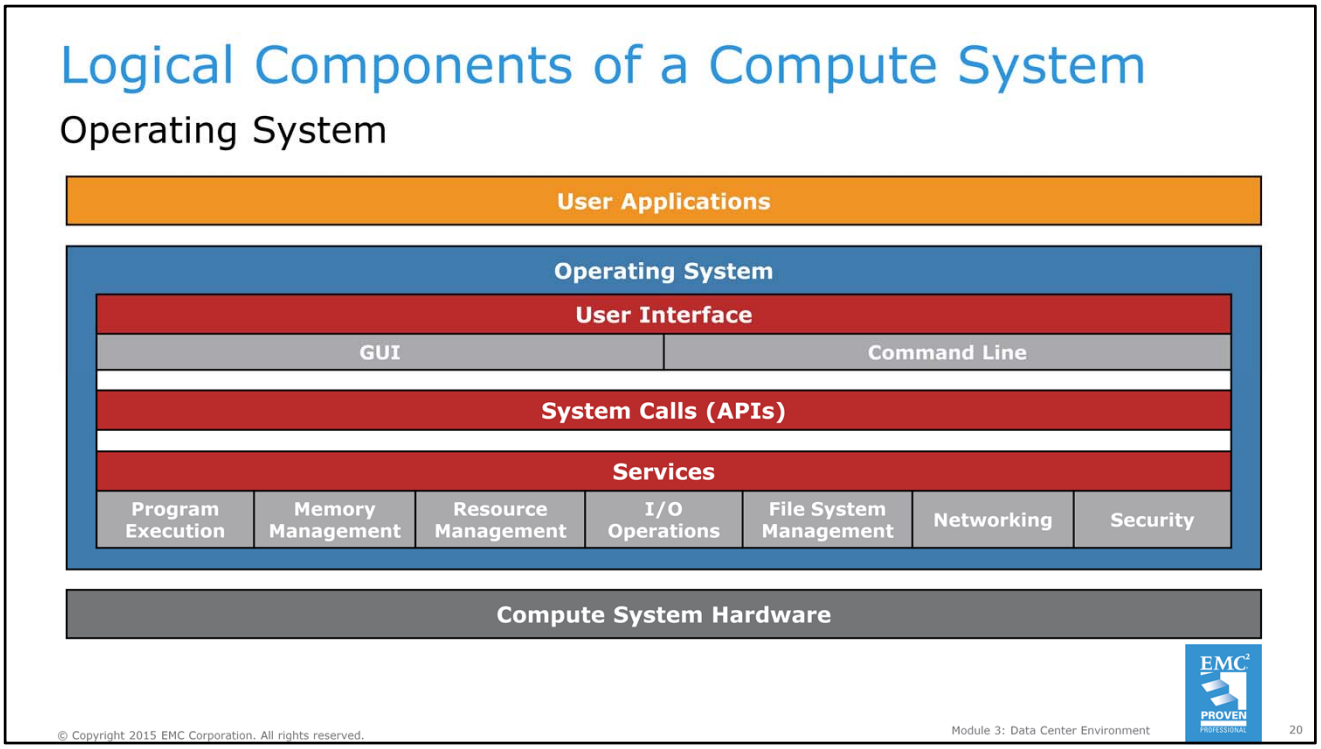
Based on business and performance requirements, cost, and expected rate of growth, an organization has to make multiple important decisions about the choice of compute system hardware to be deployed in a data center. These decisions include the number of compute systems to deploy, the number, the type, and the speed of processors, the amount of RAM required, the motherboard's RAM capacity, the number and type of expansion slots on a motherboard, the number and type of I/O cards, and installation and configuration effort.



The key logical components of a compute system are:

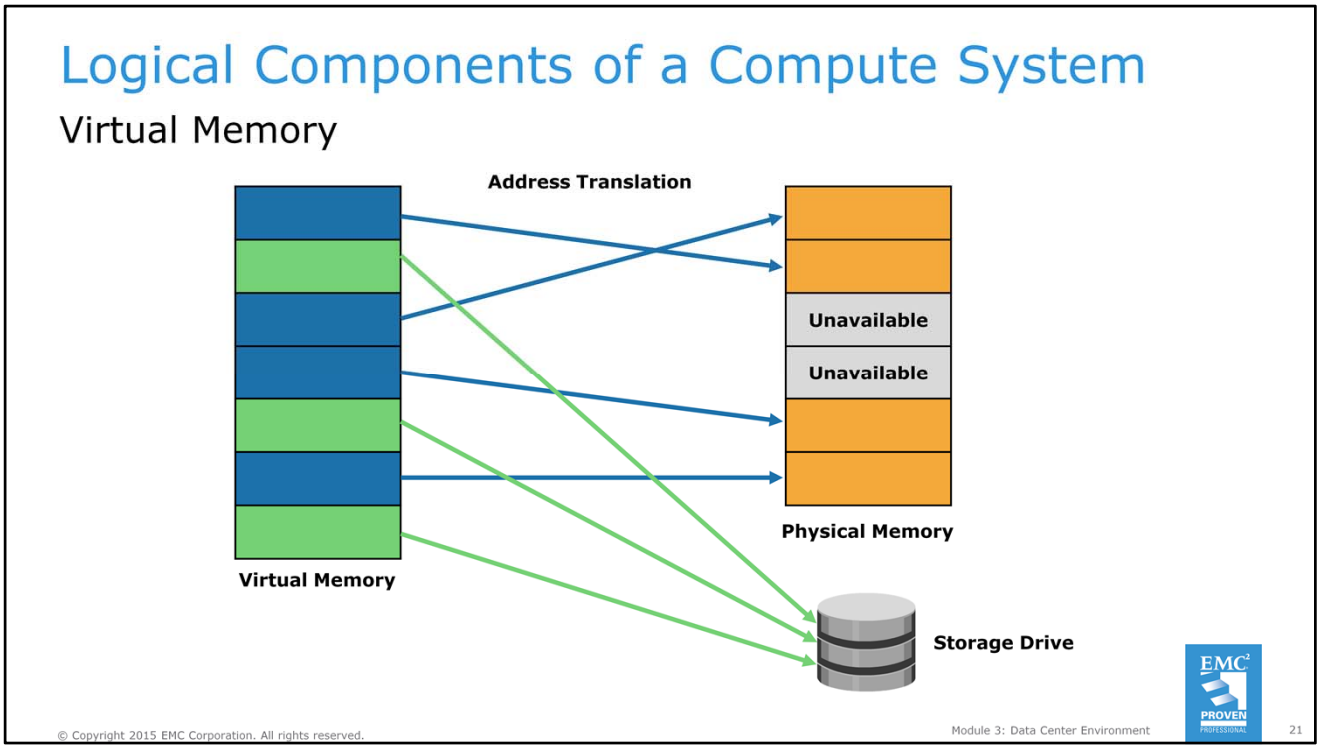
- Operating system
- Virtual memory
- Logical volume manager
- File system

A detailed description of the logical components is beyond the scope of this course. However, the components are covered in brief next.



The *operating system* (OS) is a software that acts as an intermediary between a user of a compute system and the compute system hardware. It controls and manages the hardware and software on a compute system. The OS manages hardware functions, applications execution, and provides a user interface (UI) for users to operate and use the compute system. The figure on the slide depicts a generic architecture of an OS. Some functions (or services) of an OS include program execution, memory management, resources management and allocation, and input/output management. An OS also provides networking and basic security for the access and usage of all managed resources. It also performs basic storage management tasks while managing other underlying components, such as the device drivers, logical volume manager, and file system. An OS also contains high-level Application Programming Interfaces (APIs) to enable programs to request services.

To interact with a particular hardware resource, an OS requires a *device driver*, which is a special system software that permits the OS to interact with the specific device. For example, hardware such as printer, mouse, disk drive, network adapters, and graphics cards require device drivers. A device driver enables the OS to recognize the device, and to access and control it. Device drivers are hardware-dependent and OS-specific.



The amount of physical memory (RAM) in a compute system determines both the size and the number of applications that can run on the compute system. Memory virtualization presents physical memory to applications as a single logical collection of contiguous memory locations called *virtual memory*. While executing applications, the processor generates logical addresses (virtual addresses) that map into the virtual memory. The memory management unit of the processor then maps the virtual address to the physical address. The OS utility, known as the *virtual memory manager* (VMM), manages the virtual memory and also the allocation of physical memory to virtual memory.

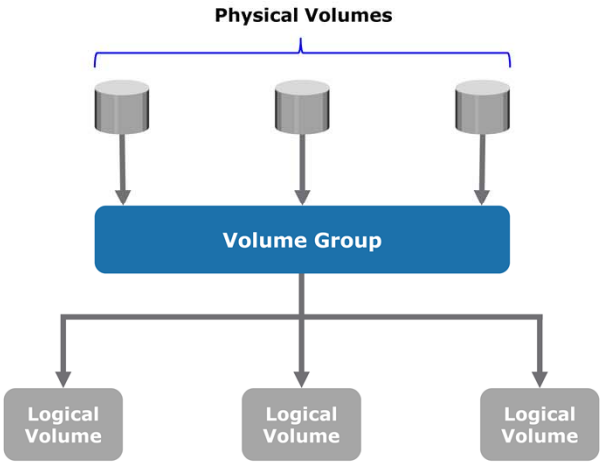
An additional memory virtualization feature of an OS enables the capacity of secondary storage devices to be allocated to the virtual memory. This creates a virtual memory with an address space that is much larger than the actual physical memory space present in the compute system. This enables multiple applications and processes, whose aggregate memory requirement is greater than the available physical memory, to run on a compute system without impacting each other. The VMM manages the virtual-to-physical memory mapping and fetches data from the secondary storage when a process references a virtual address that points to data at the secondary storage. The space used by the VMM on the secondary storage is known as a swap space. A *swap space* (also known as *page file* or *swap file*) is a portion of the storage drive that is used as physical memory.

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 OS typically moves the least-used pages into the swap file so that enough RAM is available for processes that are more active. The access to swap file pages is slower than physical memory pages because swap file pages are allocated on the storage drive, which is slower than the physical memory.

Logical Components of a Compute System

Logical Volume Manager (LVM)

- Creates and controls compute level logical storage
  - Provides a logical view of physical storage
  - Logical data blocks are mapped to physical data blocks
- Physical volumes form a volume group
  - LVM manages volume groups as a single entity
- Logical volumes are created from a volume group



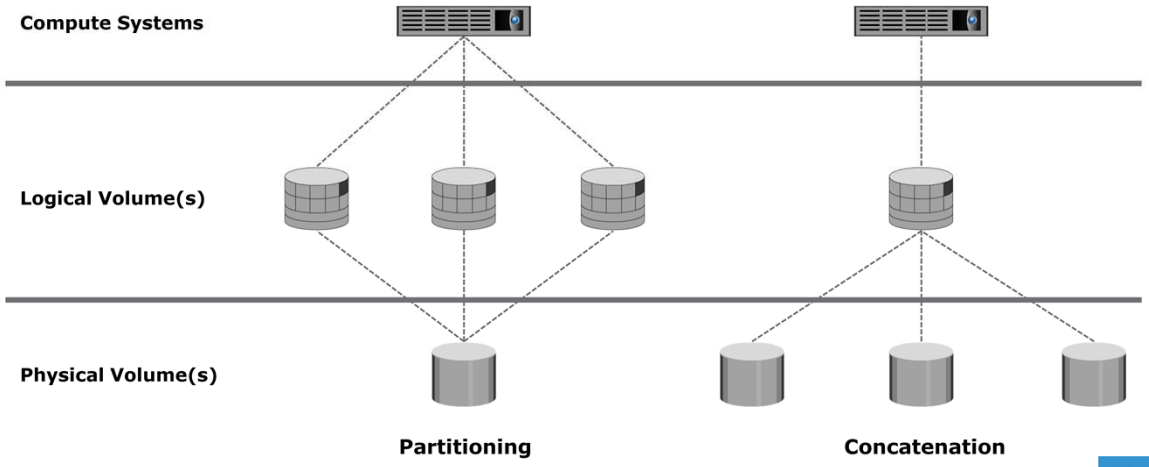
Logical Volume Manager (LVM) is software that runs on a compute system and manages logical and physical storage. LVM is an intermediate layer between the file system and the physical drives. It can partition a larger-capacity disk into virtual, smaller-capacity volumes (*partitioning*) or aggregate several smaller disks to form a larger virtual volume (*concatenation*). LVMs are mostly offered as part of the OS. Earlier, an entire storage drive would be allocated to the file system or the other data entity used by the OS or application. The disadvantage of this was a lack of flexibility. When a storage drive ran out of space, there was no easy way to extend the file system's size. As the storage capacity of the disk drive increased, allocating the entire disk drive for the file system often resulted in underutilization of the storage capacity. The evolution of LVMs enabled dynamic extension of file system capacity and efficient storage management. 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.

The basic LVM components are physical volumes, logical volume groups, and logical volumes. In LVM terminology, each physical disk connected to the compute system is a *physical volume* (PV). A *volume group* is created by grouping together one or more PVs. A unique *physical volume identifier* (PVID) is assigned to each PV 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 PV becomes part of a volume group. Each PV 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 number of physical extents. The LV appears as a physical device to the OS. 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.

Logical Components of a Compute System

LVM Example: Partitioning and 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. 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.

Logical Components of a Compute System

File System

- File is a collection of related records stored as a single named unit in contiguous logical address space
- A file system controls and manages the storage and retrieval of files
  - Enables users to perform various operations on files
  - Groups and organizes files in a hierarchical structure
- File system may be broadly classified as:
  - Disk-based file system
  - Network-based file system
  - Virtual file system

A *file* is a collection of related records or data stored as a single named unit in contiguous logical address space. Files are of different types, such as text, executable, image, audio/video, binary, library, and archive. Files have a number of attributes, such as name, unique identifier, type, size, location, owner, and protection.

A *file system* is an OS component that controls and manages the storage and retrieval of files in a compute system. A file system enables easy access to the files residing on a storage drive, a partition, or a logical volume. It consists of logical structures and software routines that control access to files. It enables users to perform various operations on files, such as create, access (sequential/random), write, search, edit, and delete.

A file system typically groups and organizes files in a tree-like hierarchical structure. It enables users to group files within a logical collection called a *directory*, which are containers for storing pointers to multiple files. A file system maintains a pointer map to the directories, subdirectories (if any), and files that are part of the file system. It also stores all the metadata (file attributes) associated with the files.

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 file 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 the course of time, as files grow larger, the file system may become fragmented.

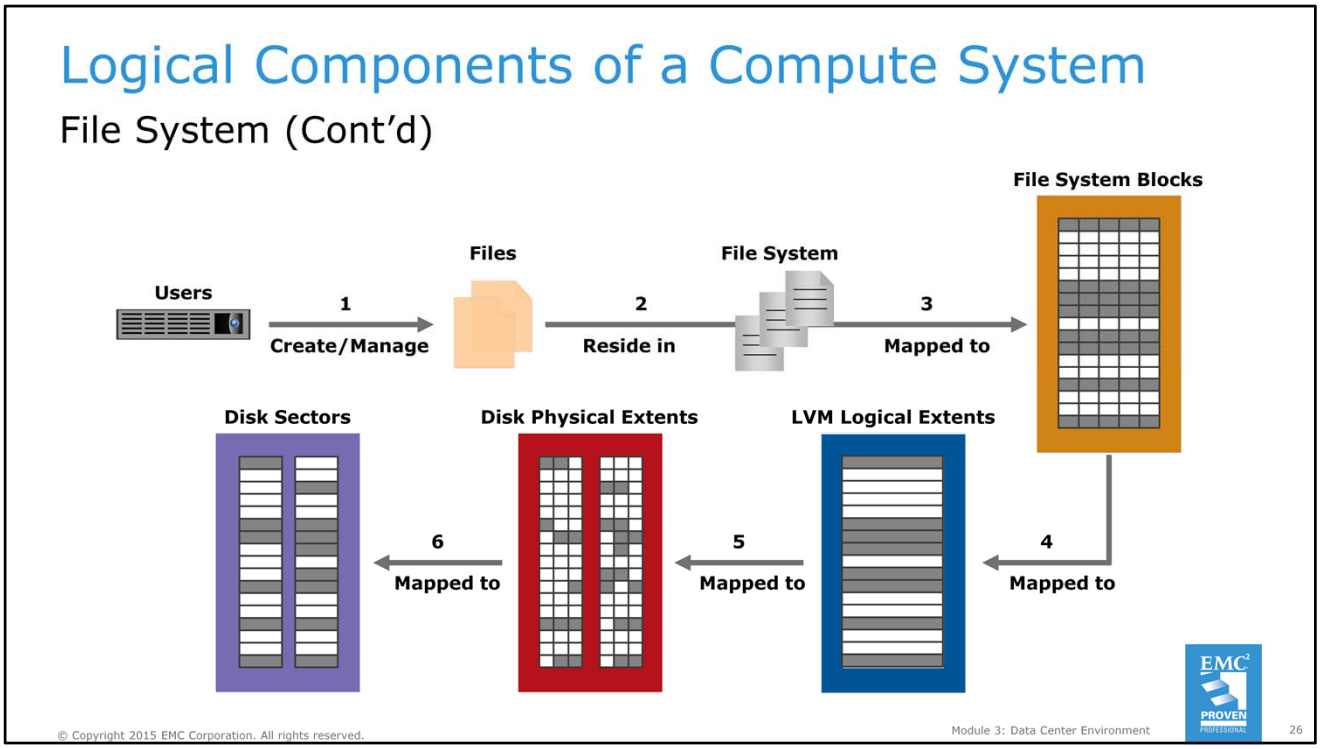
(Cont'd)

File system may be broadly classified as follows disk-based, network-based, and virtual file systems. These are described below.

Disk-based file system: A disk-based file system manages the files stored on storage devices such as solid-state drives, disk drives, and optical drives. Examples of disk-based file systems are Microsoft NT File System (NTFS), Apple Hierarchical File System (HFS) Plus, Extended File System family for Linux, Oracle ZFS, and Universal Disk Format (UDF).

Network-based file system: A network-based file system uses networking to allow file system access between compute systems. Network-based file systems may use either the client-server model, or may be distributed/clustered. In the client-server model, the file system resides on a server, and is accessed by clients over the network. The client-server model allows clients to mount the remote file systems from the server. NFS for UNIX environment and CIFS for Windows environment (both covered in Module 6, 'File-based Storage System (NAS)') are two standard client-server file sharing protocols. A *clustered file system* is a file system that is simultaneously mounted on multiple compute systems (or nodes) in a cluster. It allows the nodes in the cluster to share and concurrently access the same storage device. Clustered file systems provide features like location-independent addressing and redundancy. A clustered file system may also spread data across multiple storage nodes, for redundancy and/or performance. Examples of network-based file systems are Microsoft Distributed File System (DFS), Hadoop Distributed File System (HDFS), VMware Virtual Machine File System (VMFS), Red Hat GlusterFS, and Red Hat CephFS.

Virtual file system: A virtual file system is a memory-based file system that enables compute systems to transparently access different types of file systems on local and network storage devices. It provides an abstraction layer that allows applications to access different types of file systems in a uniform way. It bridges the differences between the file systems for different operating systems, without the application's knowledge of the type of file system they are accessing. The examples of virtual file systems are Linux Virtual File System (VFS) and Oracle CacheFS.

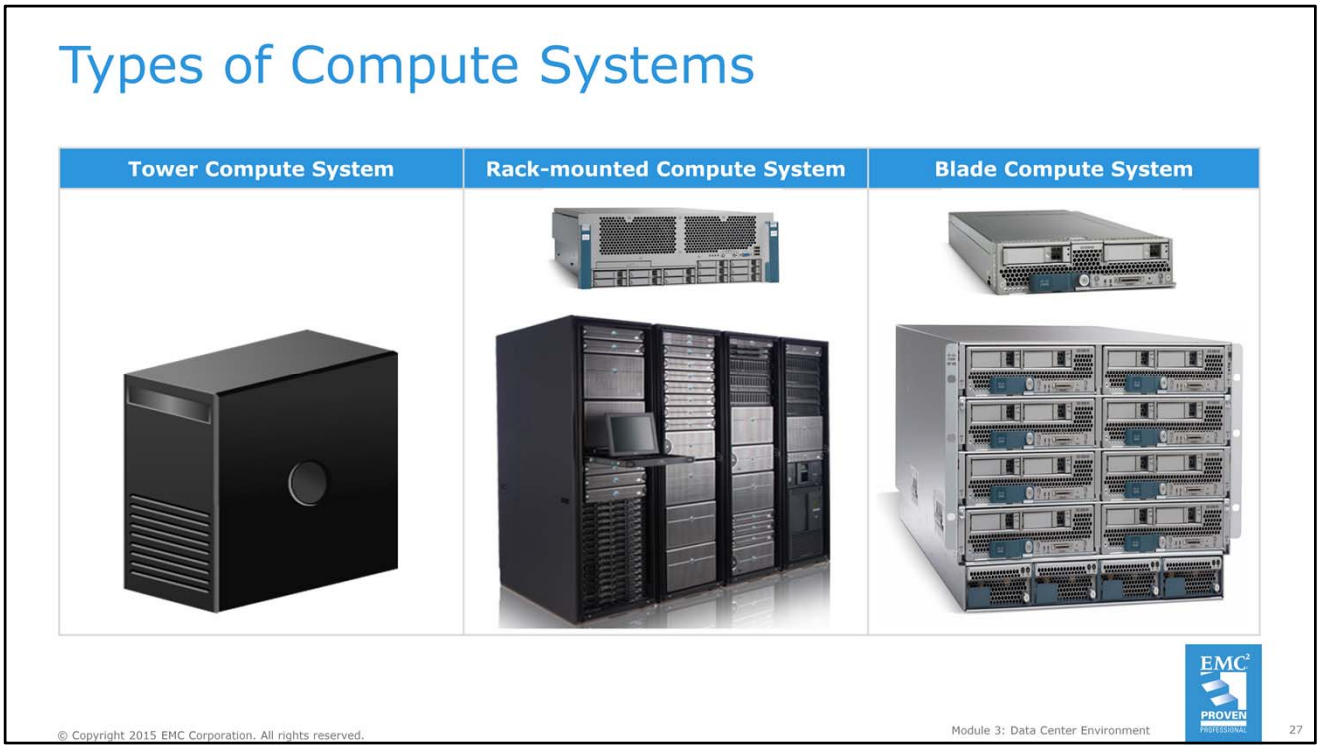


The following is the process of mapping user files to the storage that uses an LVM:

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 physical extents either by the OS or by the LVM.
6. These physical extents are mapped to the 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 sectors.

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.



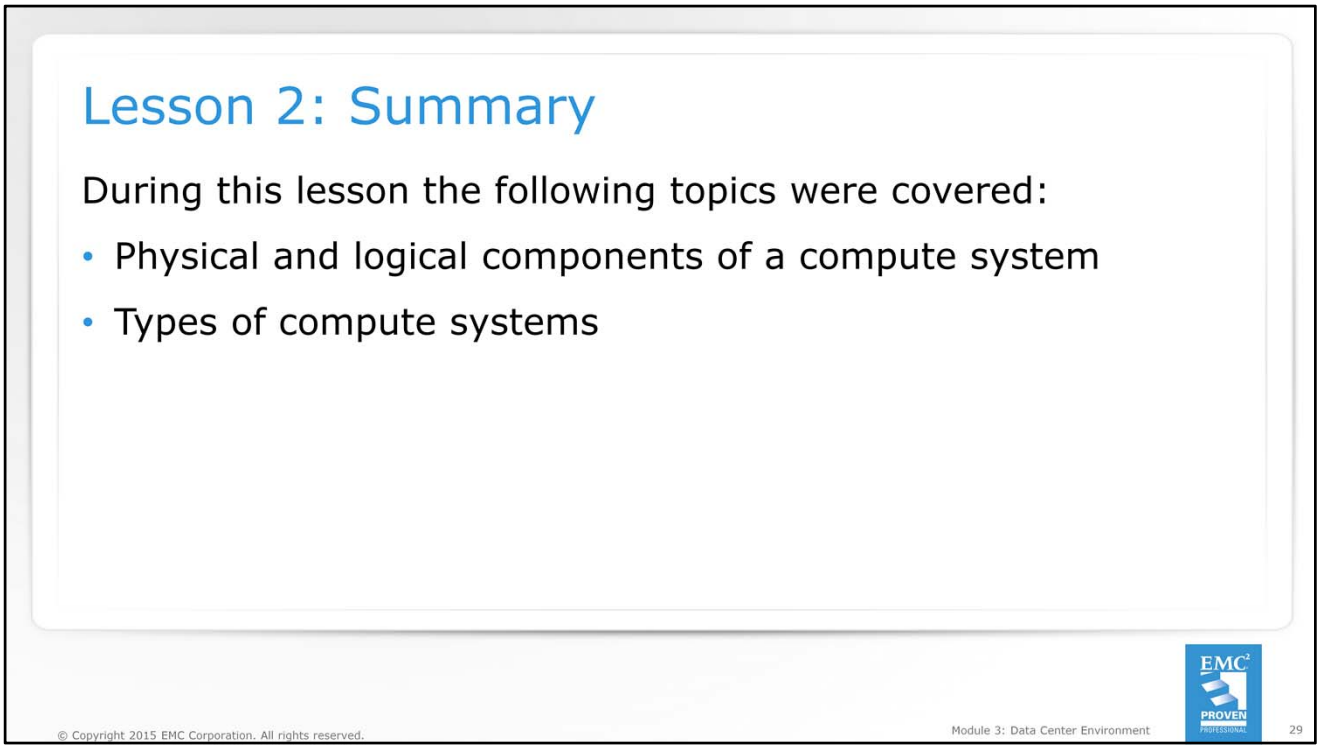
The compute systems used in building data centers are typically classified into three categories: tower compute system, rack-mounted compute system, and blade compute system

A tower compute system, also known as a tower server, is a compute system built in an upright standalone enclosure called a “tower”, which looks similar to a desktop cabinet. Tower servers have a robust build, and have integrated power supply and cooling. They typically have individual monitors, keyboards, and mice. Tower servers occupy significant floor space and require complex cabling when deployed in a data center. They are also bulky and a group of tower servers generate considerable noise from their cooling units. Tower servers are typically used in smaller environments. Deploying a large number of tower servers in large environments may involve substantial expenditure.

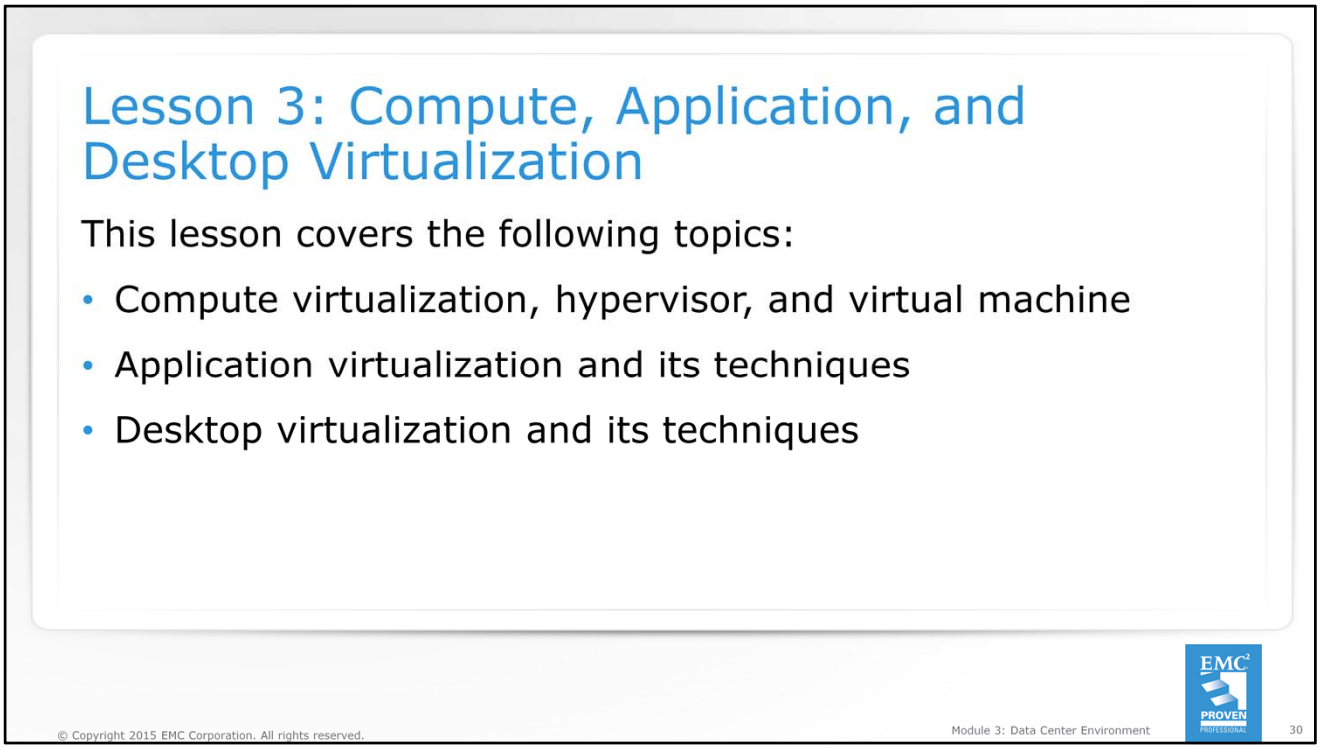
A rack-mounted compute system, also known as a *rack server*, is a compute system designed to be fixed inside a frame called a “rack”. A *rack* is a standardized enclosure containing multiple mounting slots called “bays”, each of which holds a server in place with the help of screws. A single rack contains multiple servers stacked vertically in bays, thereby simplifying network cabling, consolidating network equipment, and reducing the floor space use. Each rack server has its own power supply and cooling unit. Typically, a console is mounted on a rack to enable administrators to manage all the servers in the rack. Some concerns with rack servers are that they are cumbersome to work with, and they generate a lot of heat because of which more cooling is required, which in turn increases power costs. A “rack unit” (denoted by U or RU) is a unit of measure of the height of a server designed to be mounted on a rack. One rack unit is 1.75 inches (44.45 mm). A 1 U rack server is typically 19 inches (482.6 mm) wide. The standard rack cabinets are 19 inches wide and the common rack cabinet sizes are 42U, 37U, and 27U. The rack cabinets are also used to house network, storage, telecommunication, and other equipment modules. A rack cabinet may also contain a combination of different types of equipment modules.

(Cont’d)

A blade compute system, also known as a *blade server*, is an electronic circuit board containing only core processing components, such as processor(s), memory, integrated network controllers, storage drive, and essential I/O cards and ports. Each blade server is a self-contained compute system and is typically dedicated to a single application. A blade server is housed in a slot inside a blade enclosure (or chassis), which holds multiple blades and provides integrated power supply, cooling, networking, and management functions. The blade enclosure enables interconnection of the blades through a high-speed bus and also provides connectivity to external storage systems. The modular design of the blade servers makes them smaller, which minimizes the floor space requirements, increases the compute system density and scalability, and provides better energy efficiency as compared to the tower and the rack servers. It also reduces the complexity of the compute infrastructure and simplifies compute infrastructure management. It provides these benefits without compromising on any capability that a non-blade compute system provides. Some concerns with blade servers include the high cost of a blade system (blade servers and chassis), and the proprietary architecture of most blade systems due to which a blade server can typically be plugged only into a chassis from the same vendor.



This lesson covered compute system, and its key physical and logical components. This lesson also covered the types of compute systems.

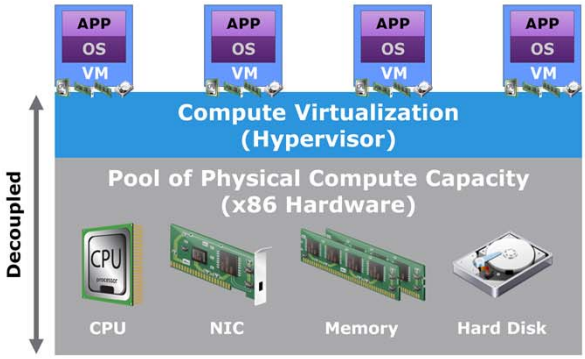


This lesson covers compute virtualization, hypervisor, and virtual machine. This lesson also covers application virtualization and its techniques. Further, this lesson covers desktop virtualization and its techniques.

What is Compute Virtualization?

Compute Virtualization

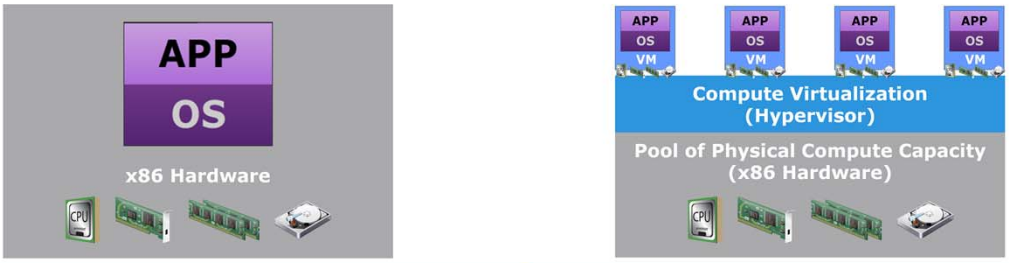
The technique of abstracting the physical compute hardware from the operating system and applications enabling multiple operating systems to run concurrently on a single or clustered physical compute system(s).



Compute virtualization is a technique of abstracting the physical hardware of a compute system from the operating system (OS) and applications. The decoupling of the physical hardware from the OS and applications enables multiple operating systems to run concurrently on a single or clustered physical compute system(s). Compute virtualization enables the creation of virtual compute systems called virtual machines (VMs). Each VM runs an OS and applications, and is isolated from the other VMs on the same compute system. Compute virtualization is achieved by a hypervisor, which is virtualization software that is installed on a physical compute system. The hypervisor provides virtual hardware resources, such as CPU, memory, storage, and network resources to all the VMs. Depending on the hardware capabilities, a large number of VMs can be created on a single physical compute system.

A VM is a logical entity; but to the OS running on the VM, it appears as a physical compute system, with its own processor, memory, network controller, and disks. However, all VMs share the same underlying physical hardware of the compute system and the hypervisor allocates the compute system's hardware resources dynamically to each VM. From a hypervisor's perspective, each VM is a discrete set of files. This is covered later in this lesson.

Need for Compute Virtualization



Before Virtualization	After Virtualization
• IT silos and underutilized resources • Inflexible and expensive • Management inefficiencies • Risk of downtime	• Server consolidation and improved resource utilization • Flexible infrastructure at lower costs • Increased management efficiency • Increased availability and improved business continuity

In an x86-based physical compute system, the software and hardware are tightly coupled and it can run only one OS at a time. A physical compute system often faces resource conflicts when multiple applications running on the compute have conflicting requirements. For example, conflicts may arise when applications need different values in the same registry entry, and different versions of the same DLL. These issues are further compounded when an application has high-availability requirements. As a result, a compute system is typically configured to serve only one application at a time. Therefore organizations purchase and configure new compute systems for every application they deploy, which is expensive, inflexible, and results in server sprawl and creation of IT silos. Moreover, many applications do not take full advantage of the hardware capabilities available to them. Consequently, resources such as processors, memory, and storage frequently remain underutilized. A large number of compute systems also requires complex network cabling and considerable floor space and power requirements. Hardware configuration, provisioning, and management become complex and require more time. A physical compute is a single point of failure because its failure leads to application unavailability.

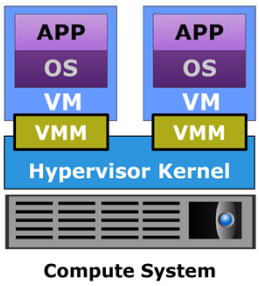
Compute virtualization enables to overcome these challenges by allowing multiple operating systems and applications to run on a single compute system. It converts physical machines to virtual machines and consolidates the converted machines onto a single compute system. Server consolidation significantly improves resource utilization and enables organizations to run their data center with a fewer machines. This, in turn, reduces the hardware acquisition costs and operational costs, and saves the data center space and energy requirements. Compute virtualization increases the management efficiency and reduces the maintenance time. The creation of VMs takes less time compared to a physical compute setup and organizations can provision compute resources faster, and with greater ease to meet the growing resource requirements. Individual VMs can be restarted, upgraded, or even crashed, without affecting the other VMs on the same physical compute. Moreover, VMs are portable and can be copied or moved from one physical compute to another without causing application unavailability.

What is Hypervisor?

Hypervisor

Software that provides a virtualization layer for abstracting compute system hardware, and enables the creation of multiple virtual machines.

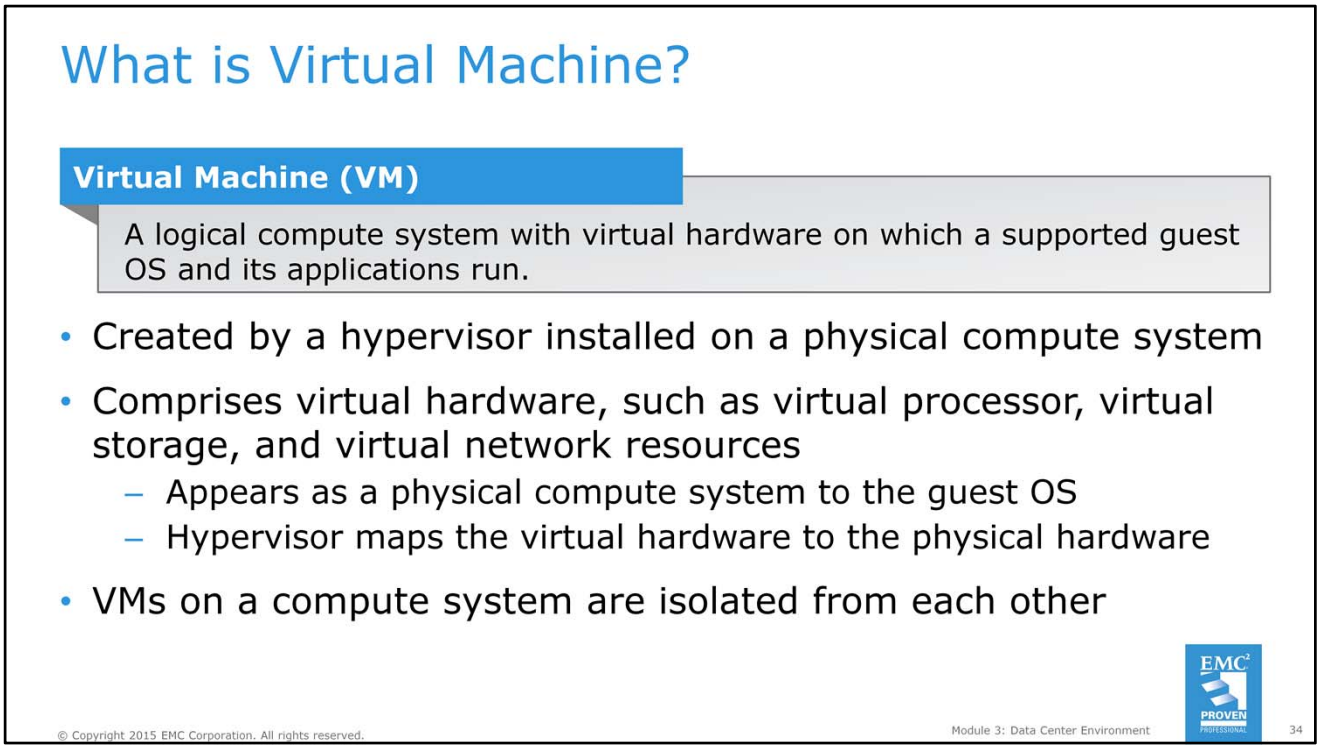
- Two key components:
  - Hypervisor kernel
    - Provides functionality similar to an OS kernel
    - Presents resource requests to physical hardware
  - Virtual machine manager (VMM)
    - Each VM is assigned a VMM
    - Abstracts physical hardware and presents to VM
- Two types of hypervisor: bare-metal and hosted



Hypervisor is compute virtualization software that is installed on a compute system. It provides a virtualization layer that abstracts the processor, memory, network, and storage of the compute system and enables the creation of multiple virtual machines. Each VM runs its own OS, which essentially enables multiple operating systems to run concurrently on the same physical compute system. The hypervisor provides standardized hardware resources to all the VMs.

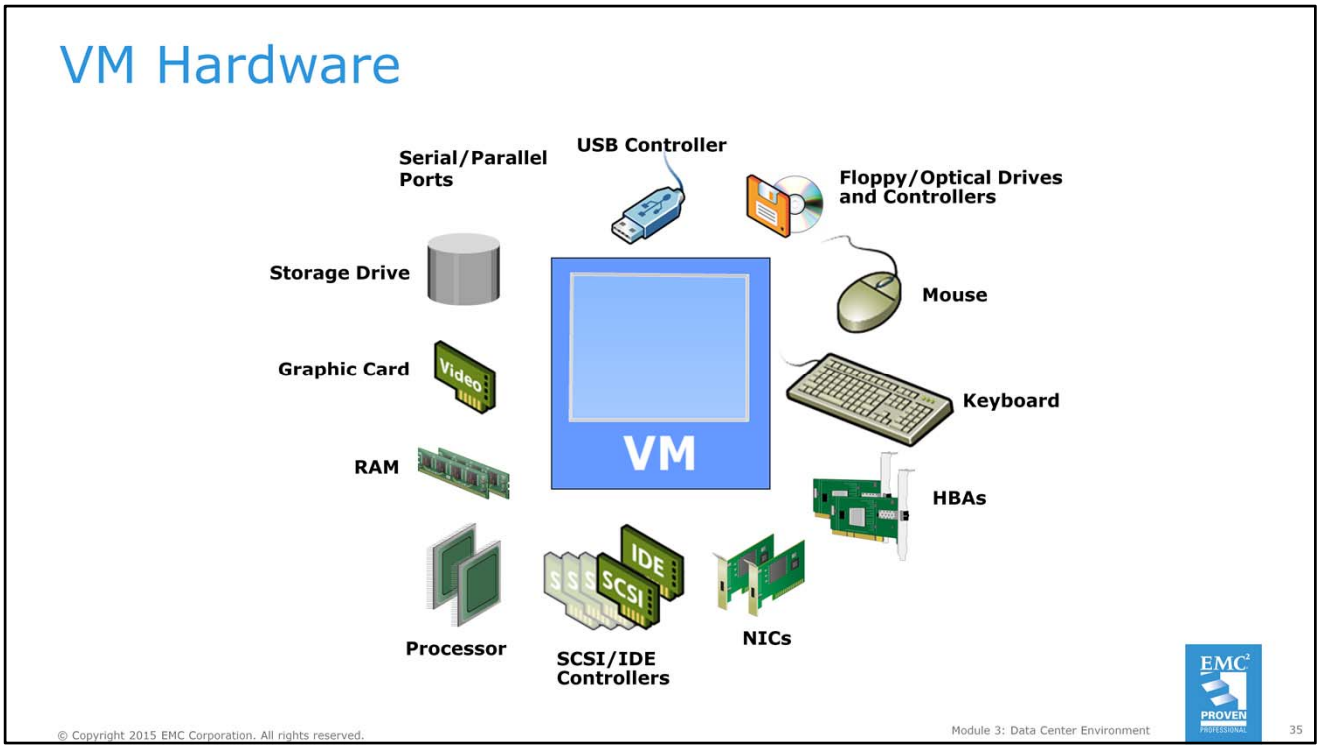
A hypervisor has two key components: kernel and virtual machine manager (VMM). A hypervisor kernel provides the same functionality like the kernel of any OS, including process management, file system management, and memory management. It is designed and optimized to run multiple VMs concurrently. It receives requests for resources through the VMM, and presents the requests to the physical hardware. Each virtual machine is assigned a VMM that gets a share of the processor, memory, I/O devices, and storage from the physical compute system to successfully run the VM. The VMM abstracts the physical hardware, and appears as a physical compute system with processor, memory, I/O devices, and other components that are essential for an OS and applications to run. The VMM receives resource requests from the VM, which it passes to the kernel, and presents the virtual hardware to the VM.

Hypervisors are categorized into two types: bare-metal and hosted. A *bare-metal hypervisor* is directly installed on the physical compute hardware in the same way as an OS. It has direct access to the hardware resources of the compute system and is therefore more efficient than a hosted hypervisor. A bare-metal hypervisor is designed for enterprise data centers and third platform infrastructure. It also supports the advanced capabilities such as resource management, high availability, and security. The figure on the slide represents a bare-metal hypervisor. A *hosted hypervisor* is installed as an application on an operating system. The hosted hypervisor does not have direct access to the hardware, and all requests pass through the OS running on the physical compute system. A hosted hypervisor adds an overhead compared to a bare-metal hypervisor. This is because there are other processes being executed by the OS that consume compute resources. Therefore, a hosted hypervisor is more suitable for development, testing, and training purposes.



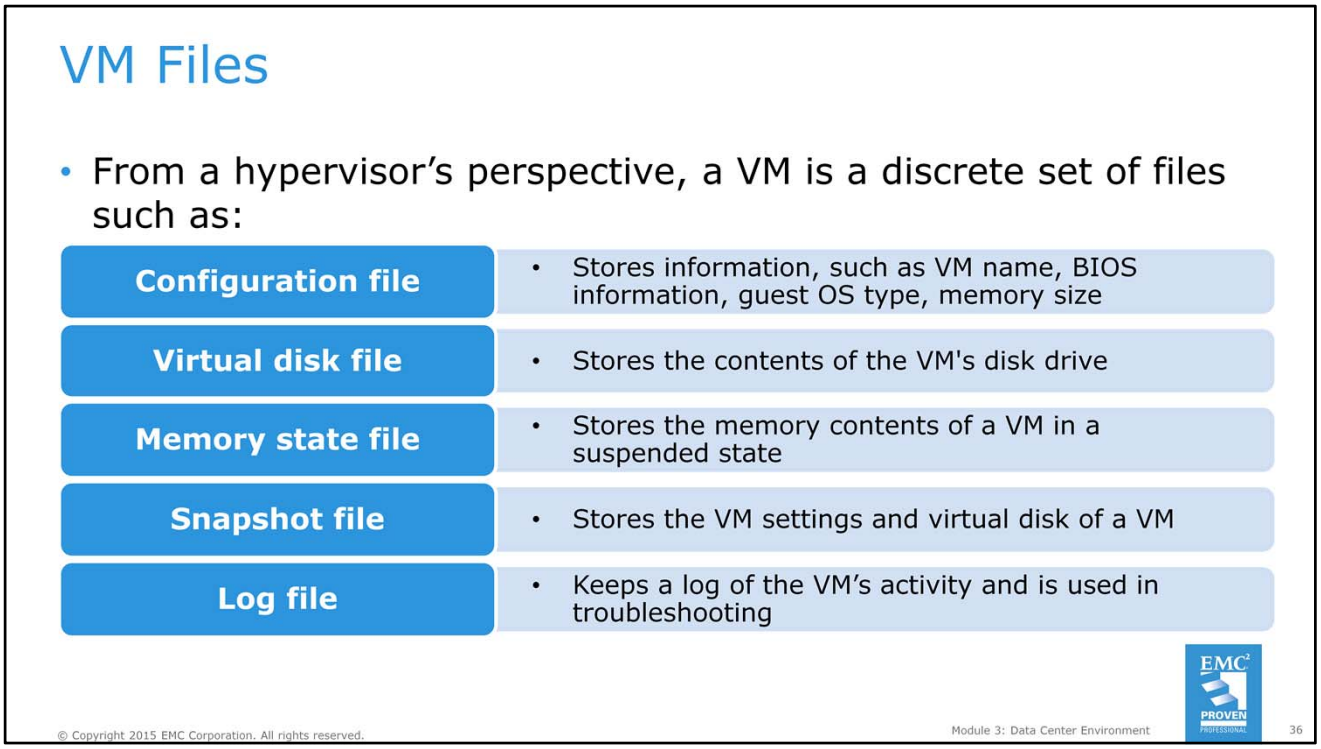
A *virtual machine* (VM) is a logical compute system with virtual hardware on which a supported guest OS and its applications run. A VM is created by a hosted or a bare-metal hypervisor installed on a physical compute system. An OS, called a "guest OS", is installed on the VM in the same way it is installed on a physical compute system. From the perspective of the guest OS, the VM appears as a physical compute system. A VM has a self-contained operating environment, comprising OS, applications, and virtual hardware, such as a virtual processor, virtual memory, virtual storage, and virtual network resources. As discussed previously, a dedicated virtual machine manager (VMM) is responsible for the execution of a VM. Each VM has its own configuration for hardware, software, network, and security. The VM behaves like a physical compute system, but does not have direct access either to the underlying host OS (when a hosted hypervisor is used) or to the hardware of the physical compute system on which it is created. The hypervisor translates the VM's resource requests and maps the virtual hardware of the VM to the hardware of the physical compute system. For example, a VM's I/O requests to a virtual disk drive are translated by the hypervisor and mapped to a file on the physical compute system's disk drive.

Compute virtualization software enables creating and managing several VMs—each with a different OS of its own—on a physical compute system or on a compute cluster. VMs are created on a compute system, and provisioned to different users to deploy their applications. The VM hardware and software are configured to meet the application's requirements. The different VMs are isolated from each other, so that the applications and the services running on one VM do not interfere with those running on other VMs. The isolation also provides fault tolerance so that if one VM crashes, the other VMs remain unaffected.



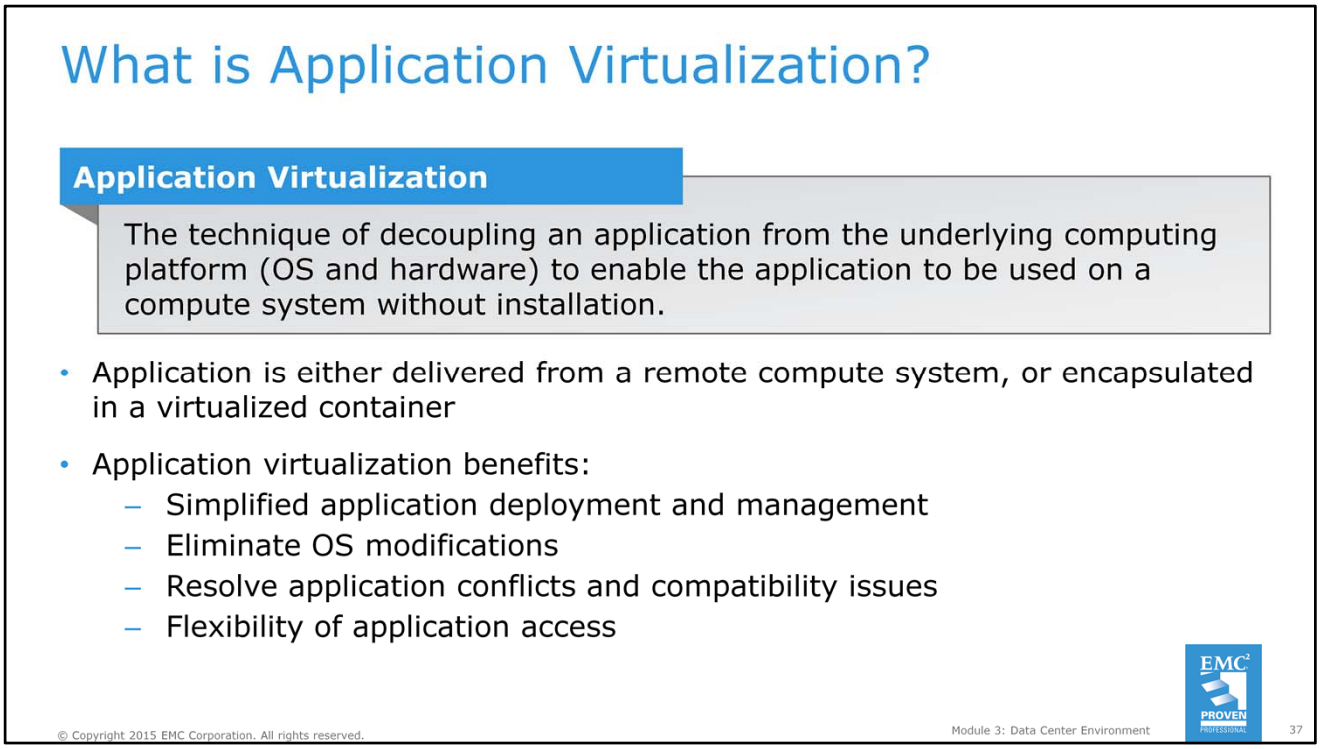
When a VM is created, it is presented with virtual hardware components that appear as physical hardware components to the guest OS. Within a given vendor's environment, each VM has standardized hardware components that make them portable across physical compute systems. Based on the requirements, the virtual components can be added or removed from a VM. However, not all components are available for addition and configuration. Some hardware devices are part of the virtual motherboard and cannot be modified or removed. For example, the video card and the PCI controllers are available by default and cannot be removed. The figure on the slide shows the typical hardware components of a VM. This includes virtual processor(s), virtual motherboard, virtual RAM, virtual disk, virtual network adapter, optical drives, serial and parallel ports, and peripheral devices.

A VM can be configured with one or more virtual processors. The number of virtual processors in a VM can be increased or reduced based on the requirements. When a VM starts, its virtual processors are scheduled by the hypervisor kernel to run on the physical processors. Each VM is assigned a virtual motherboard with the standardized devices essential for a compute system to function. Virtual RAM is the amount of physical memory allocated to a VM and it can be configured based on the requirements. The virtual disk is a large physical file, or a set of files that stores the VM's OS, program files, application data, and other data associated with the VM. A virtual network adapter functions like a physical network adapter. It provides connectivity between VMs running on the same or different compute systems, and between a VM and physical compute systems. Virtual optical drives and floppy drives can be configured to connect to either physical devices or to image files, such as ISO and floppy images (.flp), on the storage. SCSI/IDE virtual controllers provide a way for the VMs to connect to the storage devices. The virtual USB controller is used to connect to a physical USB controller and to access the connected USB devices. Serial and parallel ports provide an interface for connecting peripherals to the VM.



From a hypervisor's perspective, a VM is a discrete set of files on a storage device. Some of the key files that make up a VM are the configuration file, the virtual disk file, the memory file, and the log file. The configuration file stores the VM's configuration information, including VM name, location, BIOS information, guest OS type, virtual disk parameters, number of processors, memory size, number of adapters and associated MAC addresses, SCSI controller type, and disk drive type. The virtual disk file stores the contents of a VM's disk drive. A VM can have multiple virtual disk files, each of which appears as a separate disk drive to the VM. The memory state file stores the memory contents of a VM and is used to resume a VM that is in a suspended state. The snapshot file stores the running state of the VM including its settings and the virtual disk, and may optionally include the memory state of the VM. It is typically used to revert the VM to a previous state. Log files are used to keep a record about the VM's activity and are often used for troubleshooting purposes.

For managing VM files, a hypervisor may use a native clustered file system, or the Network File System (NFS). A hypervisor's native clustered file system is optimized to store VM files. It may be deployed on Fibre Channel and iSCSI storage (covered later in the course), apart from the local storage. The virtual disks are stored as files on the native clustered file system. Network File System enables storing VM files on remote file servers (NAS device) accessed over an IP network. The NFS client built into the hypervisor uses the NFS protocol to communicate with the NAS device. NAS devices and NFS are covered in Module 6, 'File-based Storage System (NAS)'.



Application virtualization is the technique of decoupling an application from the underlying computing platform (OS and hardware) in order to enable the application to be used on a compute system without installation. In application virtualization, the application is either deployed on a remote compute system and delivered to a client system over a network, or encapsulated in a virtualized container along with the assets it requires for execution. In both the cases, the application can be used without the need to install it on the application user's compute system. Some key benefits of application virtualization are described below.

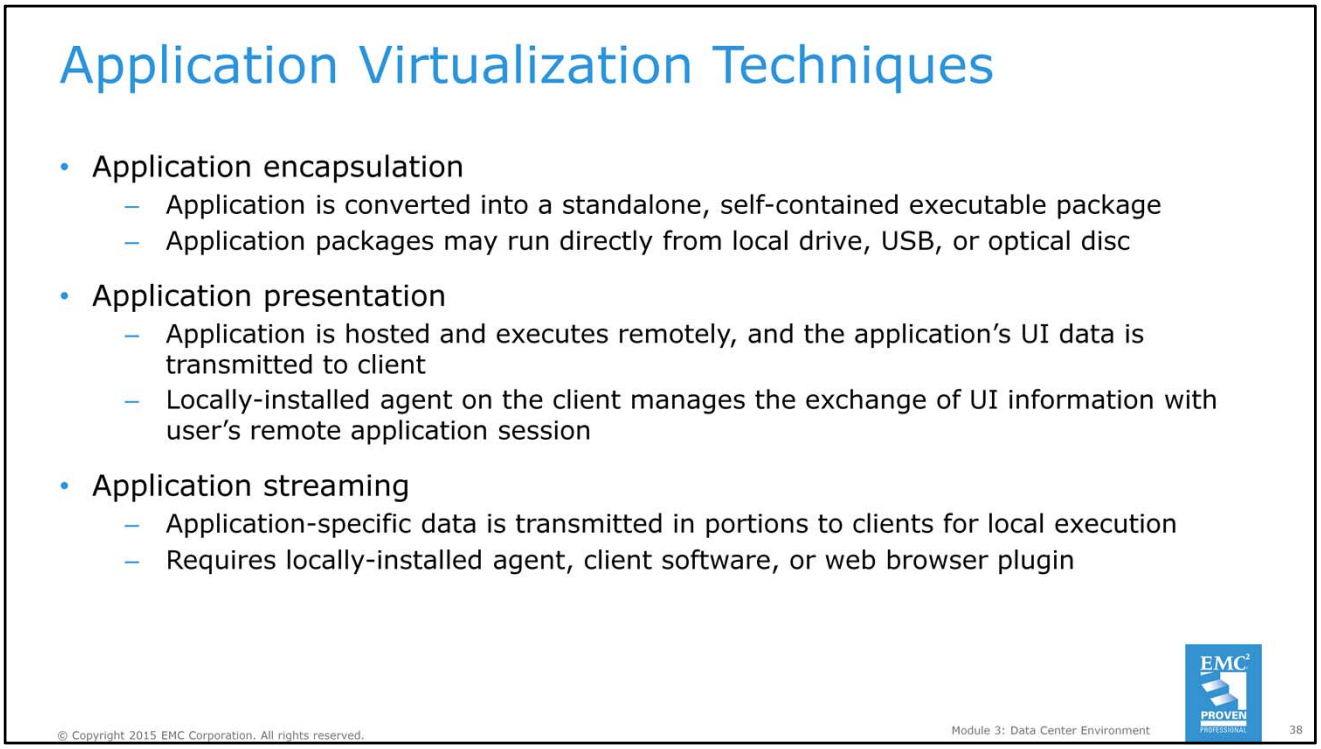
Simplified application management: Application virtualization provides a solution to meet an organization's need for simplified and improved application deployment, delivery and manageability. It reduces system integration and maintenance costs by providing a common software baseline across multiple diverse end-point devices.

Eliminate OS modifications: Since application virtualization decouples an application from the OS, it leaves the underlying OS unaltered. This provides additional security, and protects the OS from potential corruptions and problems that may arise due to changes to the file system and registry.

Resolve application conflicts and compatibility issues: Application virtualization allows the use of conflicting applications on the same end-point device. It also enables the use of applications that otherwise do not execute on an end-point device due to incompatibility with the underlying computing platform.

Simplified OS image management: Application virtualization simplifies OS image management. Since application delivery is separated from the OS, there is no need to include "standard" applications in end-point images. As a result, managing images is simpler, especially in the context of OS patches and upgrades.

Flexibility of access: Application virtualization enables an organization's workforce and customers to access applications hosted on a remote compute system from any location, and through diverse end-point devices types.



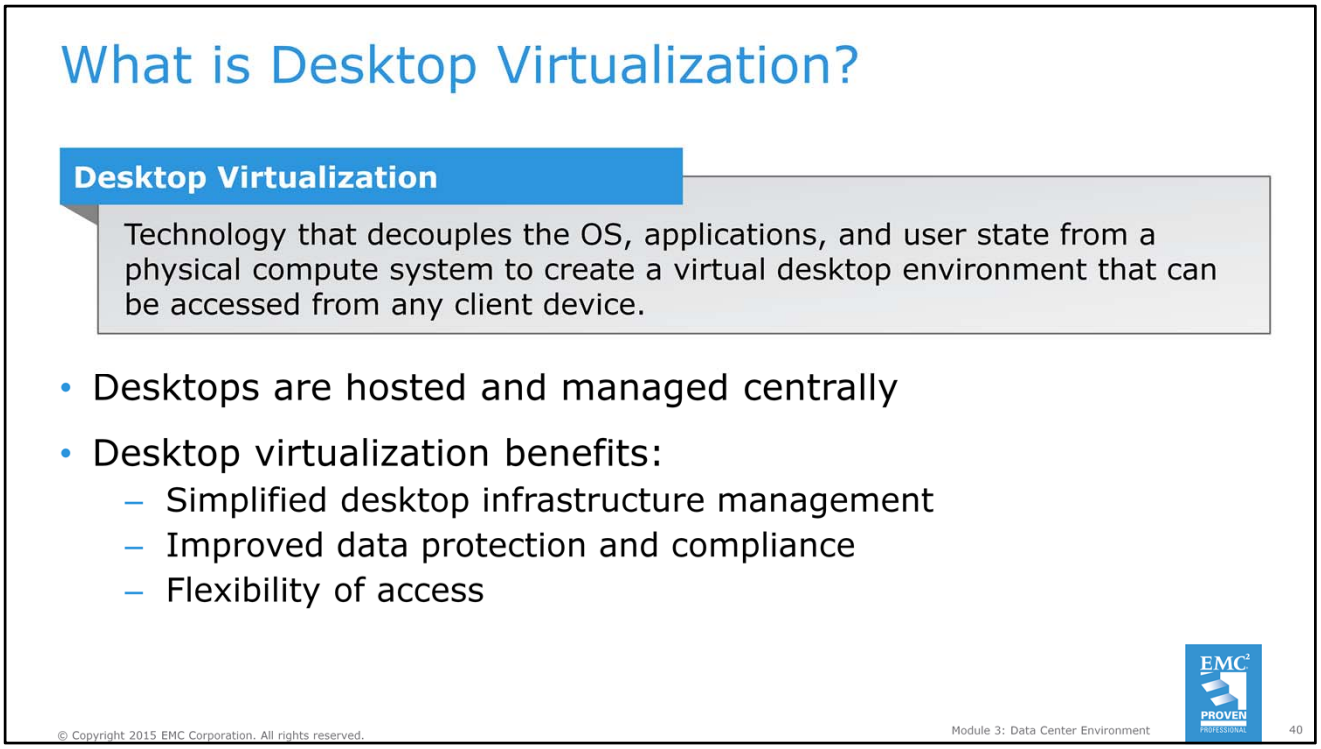
There are three techniques for application virtualization: application encapsulation, application presentation, and application streaming.

In *application encapsulation*, an application is aggregated within a virtualized container, along with the assets, such as files, virtual registry, and class libraries that it requires for execution. This process, known as *packaging* or *sequencing*, converts an application into a standalone, self-contained executable package that can directly run on a compute system. The assets required for execution are included within the virtual container. Therefore, the application does not have any dependency on the underlying OS, and does not require a traditional installation on the compute system. The application's virtual container isolates it from the underlying OS and other applications, thereby minimizing application conflicts. During application execution, all function calls made by the application to the OS for assets get redirected to the assets within the virtual container. The application is thus restricted from writing to the OS file system or registry, or modifying the OS in any other way.

In *application presentation*, an application's user interface (UI) is separated from its execution. The application executes on a remote compute system, while its UI is presented to an end-point client device over a network. When a user accesses the application, the screen pixel information and the optional sound for the application are transmitted to the client. A software agent installed on the client receives this information and updates the client's display. The agent also transmits the keystrokes and graphical input information back from the client, allowing the user to control the application. This process makes it appear as if the application is running on the client when, in fact, it is running on the remote compute system. Application presentation enables the delivery of an application on devices that have less computing power than what is normally required to execute the application. In application presentation, application sessions are created in the remote compute system and a user connects to an individual session from a client by means of the software agent. Individual sessions are isolated from each other, which secures the data of each user and also protects the application crashes.

(Cont'd)

In *application streaming*, an application is deployed on a remote compute system, and is downloaded in portions to an end-point client device for local execution. A user typically launches the application from a shortcut, which causes the client to connect to the remote compute system to start the streaming process. Initially, only a limited portion of the application is downloaded into memory. This portion is sufficient to start the execution of the application on the client. Since a limited portion of the application is delivered to the client before the application starts, the user experiences rapid application launch. The streaming approach also reduces network traffic. As the user accesses different application functions, more of the application is downloaded to the client. The additional portions of the application may also be downloaded in the background without user intervention. Application streaming requires an agent or client software on clients. Alternatively, the application may be streamed to a web browser by using a plug-in installed on the client. In some cases, application streaming enables offline access to the application by caching them locally on the client.



With the traditional desktop machine, 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. Managing a vast desktop environment is also a challenging task.

Desktop virtualization decouples the OS, applications, and user state (profiles, data, and settings) from a physical compute system. These components, collectively called a *virtual desktop*, are hosted on a remote compute system, and can be accessed by a user from any client device, such as laptops, desktops, thin clients, or mobile devices. A user accesses the virtual desktop environment over a network on a client through a web browser or a client application.

The OS and applications of the virtual desktop execute on the remote compute system, while a view of the virtual desktop's user interface (UI) is presented to the end-point device. The view of the virtual desktop enables the user to interact with it by using peripheral devices, such as keyboard and mouse, on the end-point device. Desktop virtualization uses a *remote display protocol* to transmit the virtual desktop's UI to the end-point devices. The remote display protocol also sends back key strokes and graphical input information from the end-point device, enabling the user to interact with the virtual desktop. Although the virtual desktop runs remotely, the user experience is similar to using an OS, and applications installed locally on an end-point device.

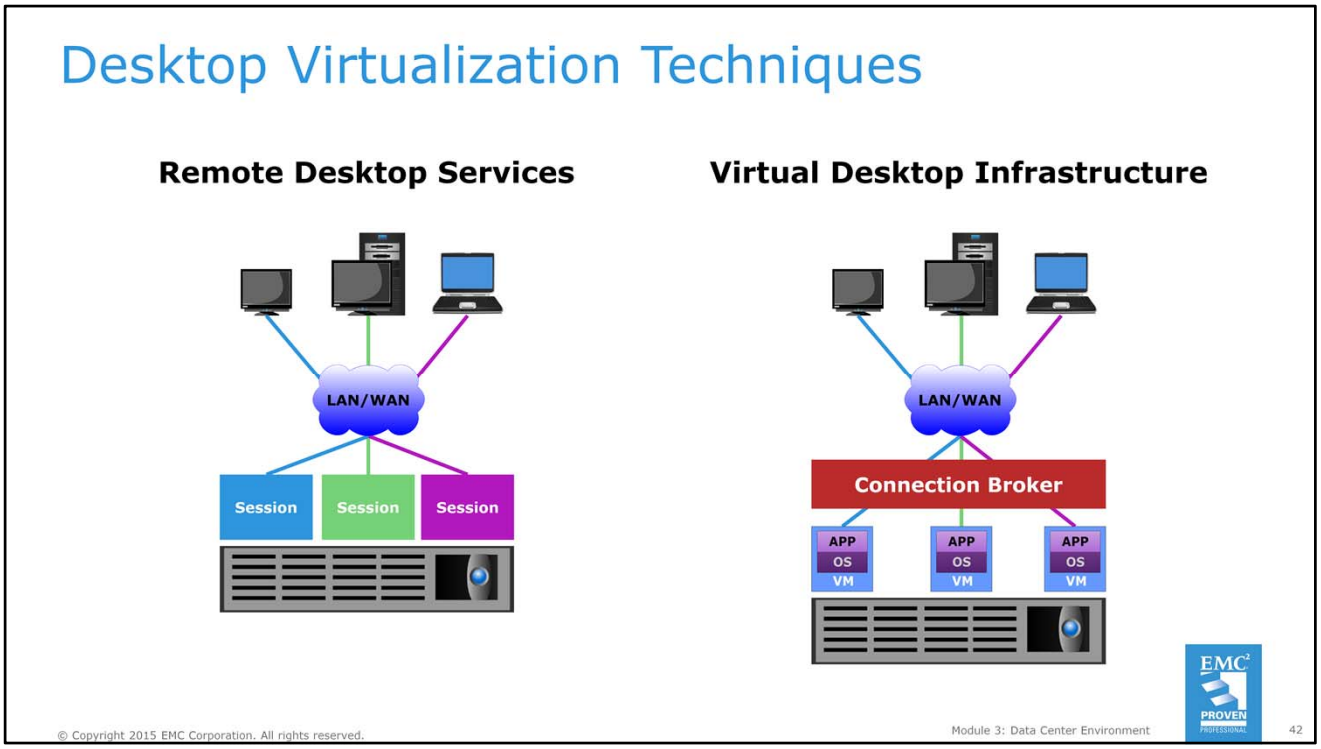
(Cont'd)

Some key benefits of desktop virtualization are describe below.

Simplified desktop infrastructure management: Desktop virtualization simplifies desktop infrastructure management, and creates an opportunity to reduce the maintenance costs. New virtual desktops can be configured and deployed faster than physical machines. The patches, updates, and upgrades can be centrally applied to the OS and applications. This simplifies or eliminates many redundant, manual, and time-consuming tasks. Virtual desktops are also based on standardized images, which make the environment simpler to manage. It is also easier to diagnose and troubleshoot problems.

Improved data protection and compliance: Applications and data are located centrally, which ensures that business-critical data is not at risk in case of loss or theft of the device. Virtual desktops are also easier to back up compared to deploying backup solutions on end-point devices.

Flexibility of access: Desktop virtualization enables users to access their desktops and applications without being bound to a specific end-point device. The virtual desktops can be accessed remotely from different end-point devices. This creates a flexible work scenario and enables user productivity from remote locations. Desktop virtualization also enables Bring Your Own Device (BYOD), which creates an opportunity to reduce acquisition and operational costs.



There are two techniques for desktop virtualization: remote desktop services (RDS) and virtual desktop infrastructure (VDI).

In *remote desktop services*, the OS and applications are hosted on a remote compute system and are shared by multiple users. RDS is similar to application presentation, but provides the capability to deliver virtual desktops rather than individual applications. Each user has an individual virtual desktop session within which applications execute. A user typically connects to a remote desktop session by means of client software. To connect to a remote desktop session, a user specifies the URL of the remote desktop service. When the client connects to the remote desktop service, a unique session, in which the user can execute applications, is created. This session provides a complete desktop experience to the user on the end-point device. The desktop's UI is transmitted to the end-point device through a remote display protocol enabling the user to interact with the desktop and applications. Each user session is isolated from the sessions of other users, which protects the application and data. In RDS, users are typically restricted from installing or modifying applications on the remote OS.

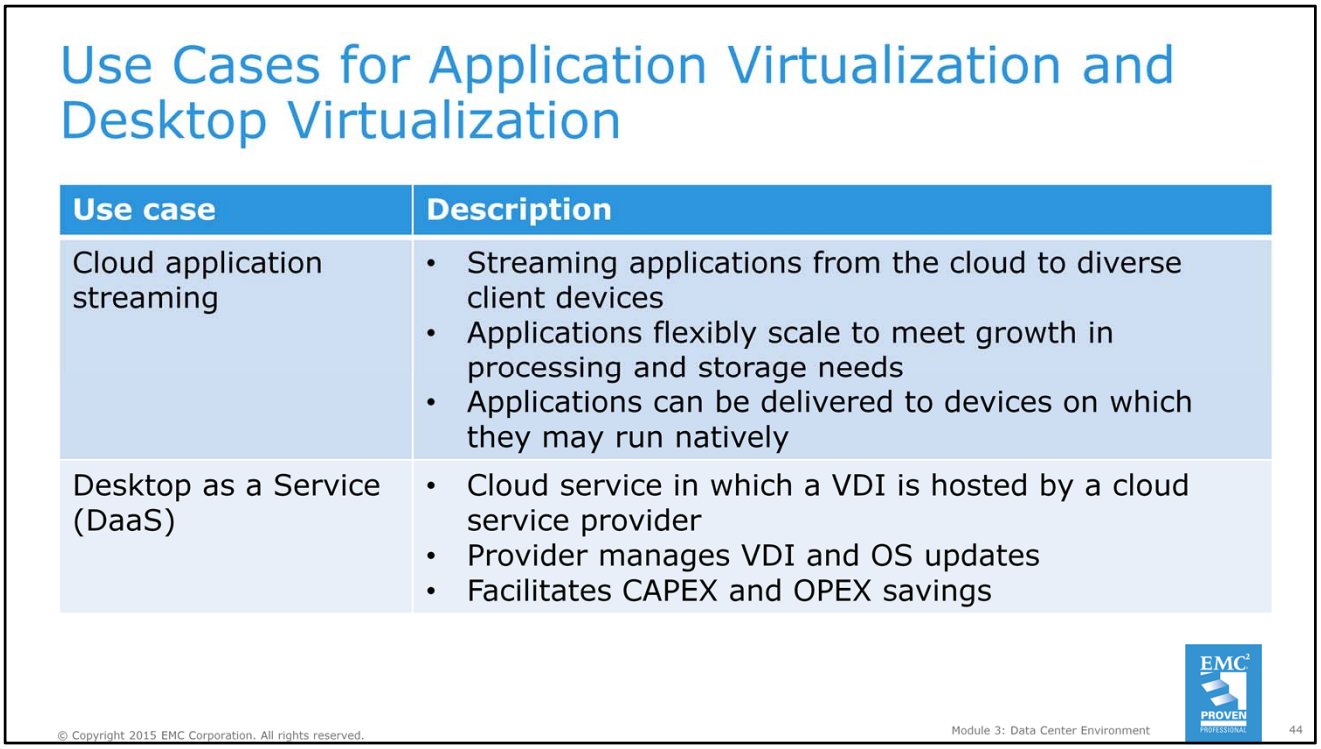
RDS supports a very high user density, as several desktop sessions can be served per processor core. It is typically used in a scenario where a core set of applications are accessed on-demand by a group of users. It provides a streamlined and standardized environment for those applications to multiple users. However, unpredictable application loads may lead to inconsistent performance. The multi-user environment used in RDS may not also be suitable for all applications. Some applications may have to be re-written for use in RDS environments.

(Cont'd)

In *virtual desktop infrastructure*, the OS and applications are hosted on virtual machines running on a remote compute system. Virtual desktop VMs are typically hosted on a bare-metal hypervisor. Every user is provided a separate desktop VM with its own OS and applications. The UI of the OS and applications on the desktop VM is transmitted to an end-point device via a remote display protocol, which enables the user to interact with the virtual desktop. In VDI, a desktop VM may be persistent or non-persistent. A persistent desktop VM retains a user's data and settings between logins and is typically dedicated to a specific user. A non-persistent desktop VM does not retain user data and customizations. It is allocated to a user from a pool of desktop VMs. When the user logs out, the desktop VM is returned to the desktop VM pool and is available for reallocation to other users. Desktop VMs can be provisioned from templates. The changes made to the templates are then automatically propagated to the corresponding desktop VMs without affecting the user data and the applications.

In VDI, desktop VMs are typically accessed from client devices by means of connection broker software. A *connection broker* establishes and manages the connection between an end-point device and the desktop VM. If desktop VMs are provisioned from a pool, the connection broker connects the user to an available desktop VM in the pool. The connection broker may authenticate a desktop request before establishing a connection to the desktop VM. The connection broker may have capabilities to suspend and resume desktop VMs, based on policies, to enable efficient usage of resources, such as processor and memory. The connection broker may also support encryption mechanisms to secure the connection between the end-point device and the desktop VM.

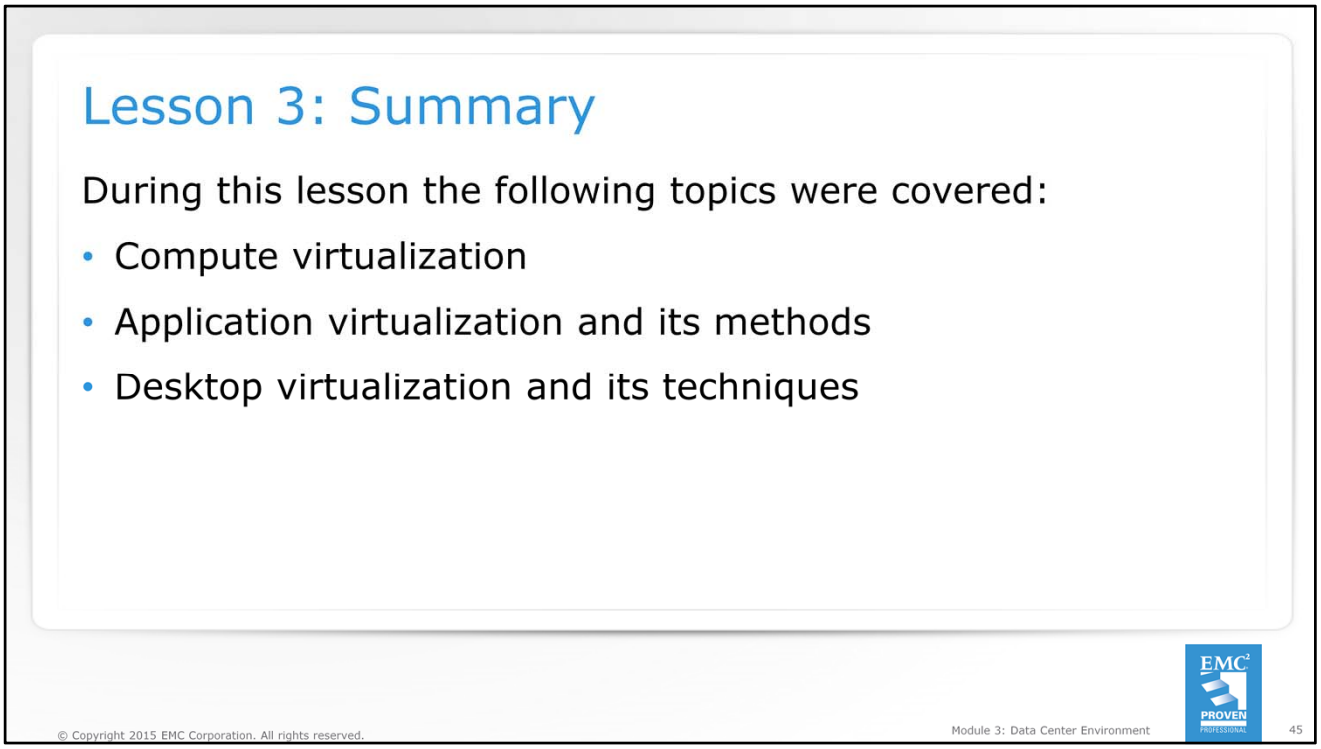
In VDI, desktop VMs are fully isolated from one another. This provides reliability and security. For persistent desktop VMs, users may be given full administrative and local application installation privileges in their personal desktop VM. Desktop VMs can be migrated within a cluster without disruption, thereby enabling efficient manageability and maintenance operations. At the OS level, a desktop VM is indistinguishable from a physical compute system, which provides greater software compatibility with VDI as compared to remote desktop technology.



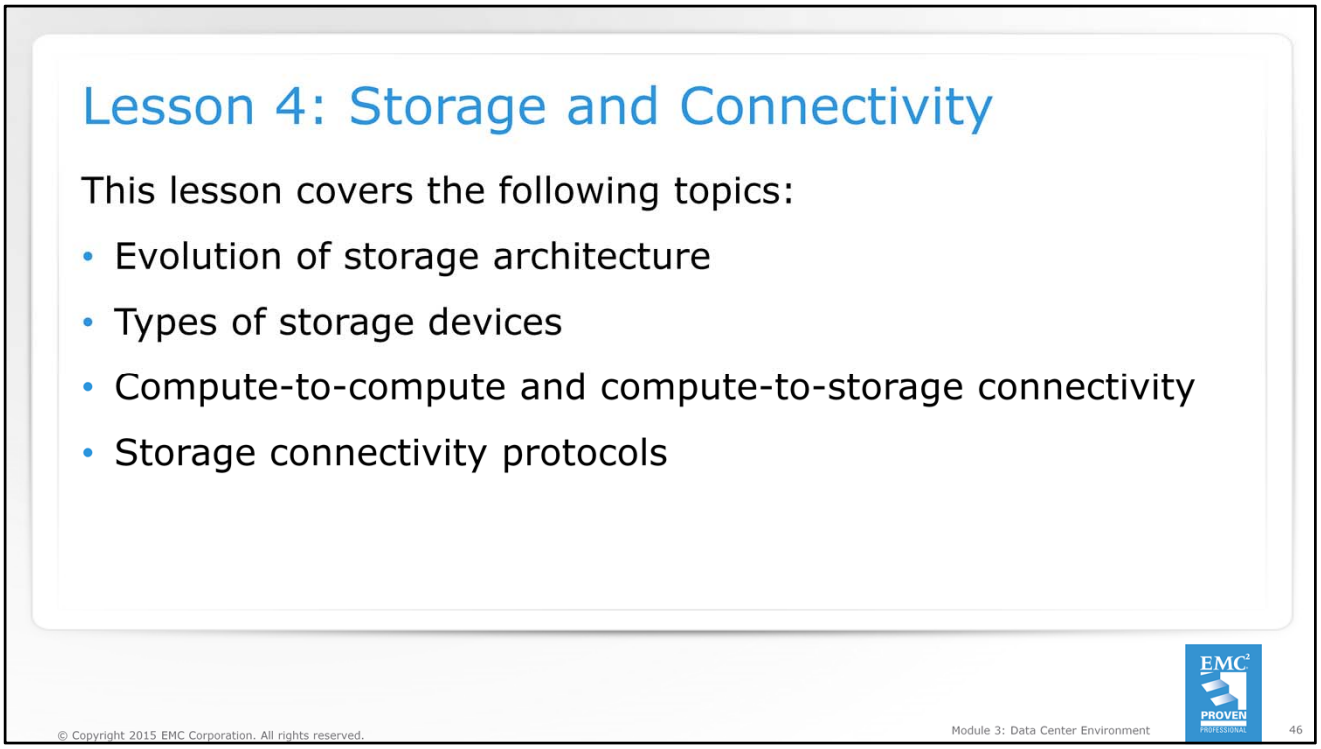
Application virtualization and desktop virtualization provide several benefits to organizations and facilitate the transformation to the third platform. The two use cases of application virtualization and desktop virtualization are described below.

Cloud application streaming: Cloud application streaming employs application virtualization to stream applications from the cloud to client devices. Streaming applications from the cloud enable organizations to reach more users on multiple devices, without modifying the application code significantly. The application is deployed on a cloud infrastructure, and the output is streamed to client devices, such as desktops, tablets, and mobile phones. Because the application runs in the cloud, it can flexibly scale to meet the massive growth in processing and storage needs, regardless of the client devices the end users are using. The cloud service can stream either all or portions of the application from the cloud. Cloud application streaming enables an application to be delivered to client devices on which it may not be possible to run the application natively.

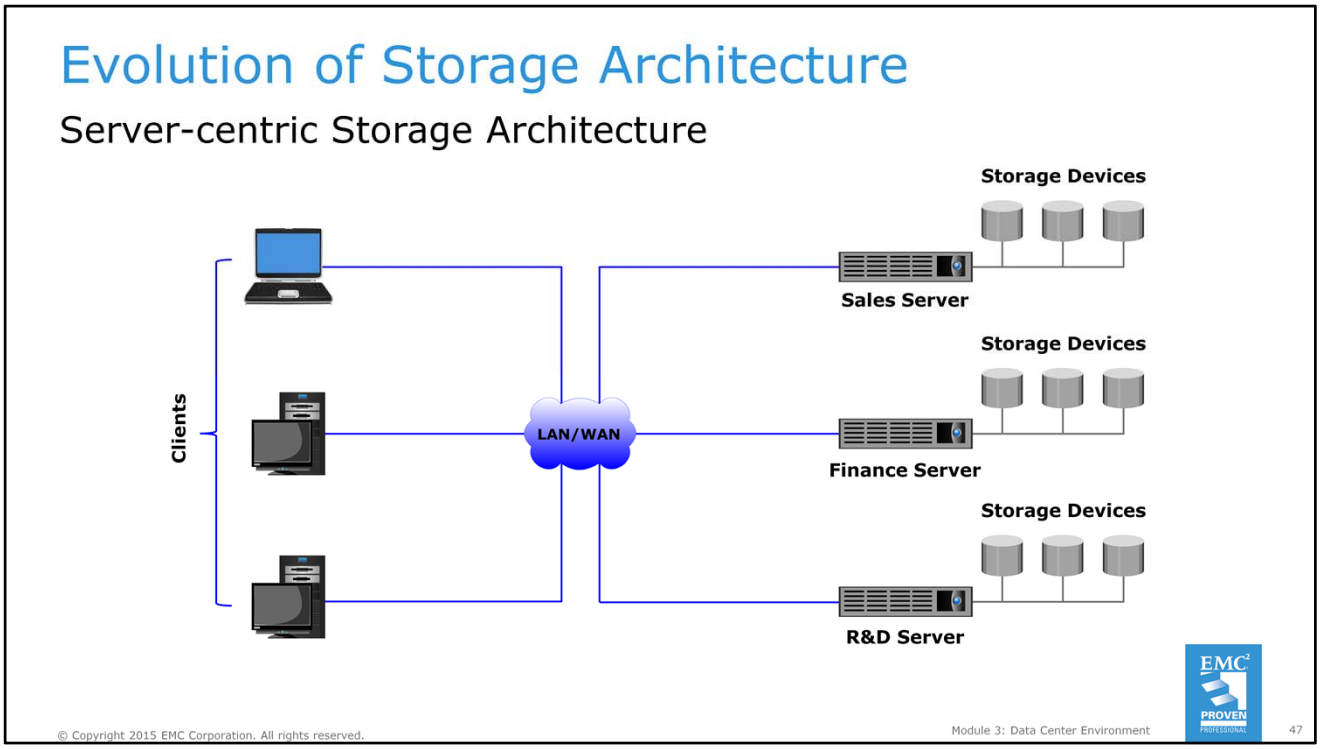
Desktop as a Service: Desktop as a Service (DaaS) is a cloud service in which a virtual desktop infrastructure (VDI) is hosted by a cloud service provider. The provider offers a complete, business-ready VDI solution, delivered as a cloud service with either subscription-based or pay-as-you-go billing. The service provider (internal IT or public) manages the deployment of the virtual desktops, data storage, backup, security, and OS updates/upgrades. The virtual desktops are securely hosted in the cloud and managed by the provider. DaaS has a multi-tenant architecture, wherein virtual desktops of multiple users share the same underlying infrastructure. However, individual virtual desktops are isolated from each other and protected against unauthorized access and crashes on other virtual desktops. The virtual desktops can be easily provisioned by consumers and they are delivered over the Internet to any client device. DaaS provides organizations with a simple, flexible, and efficient approach to IT. It enables to lower CAPEX and OPEX for acquiring and managing end-user computing infrastructure.



This lesson covered compute virtualization, hypervisor, and virtual machine. This lesson also covered application virtualization and its techniques. Further, this lesson covered desktop virtualization and its techniques.

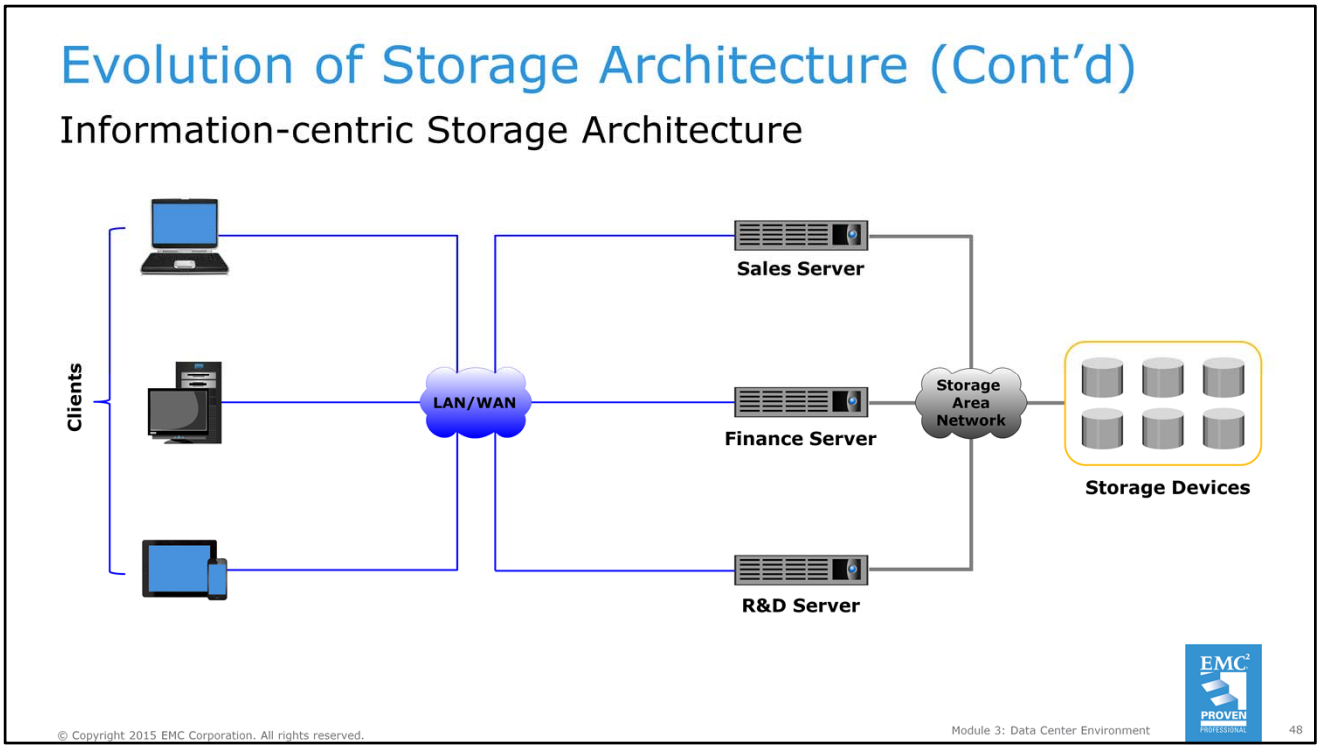


This lesson covers evolution of storage architecture and the types of storage devices. This lesson also covers compute-to-compute and compute-to-storage connectivity. Further, this lesson covers different storage connectivity protocols.



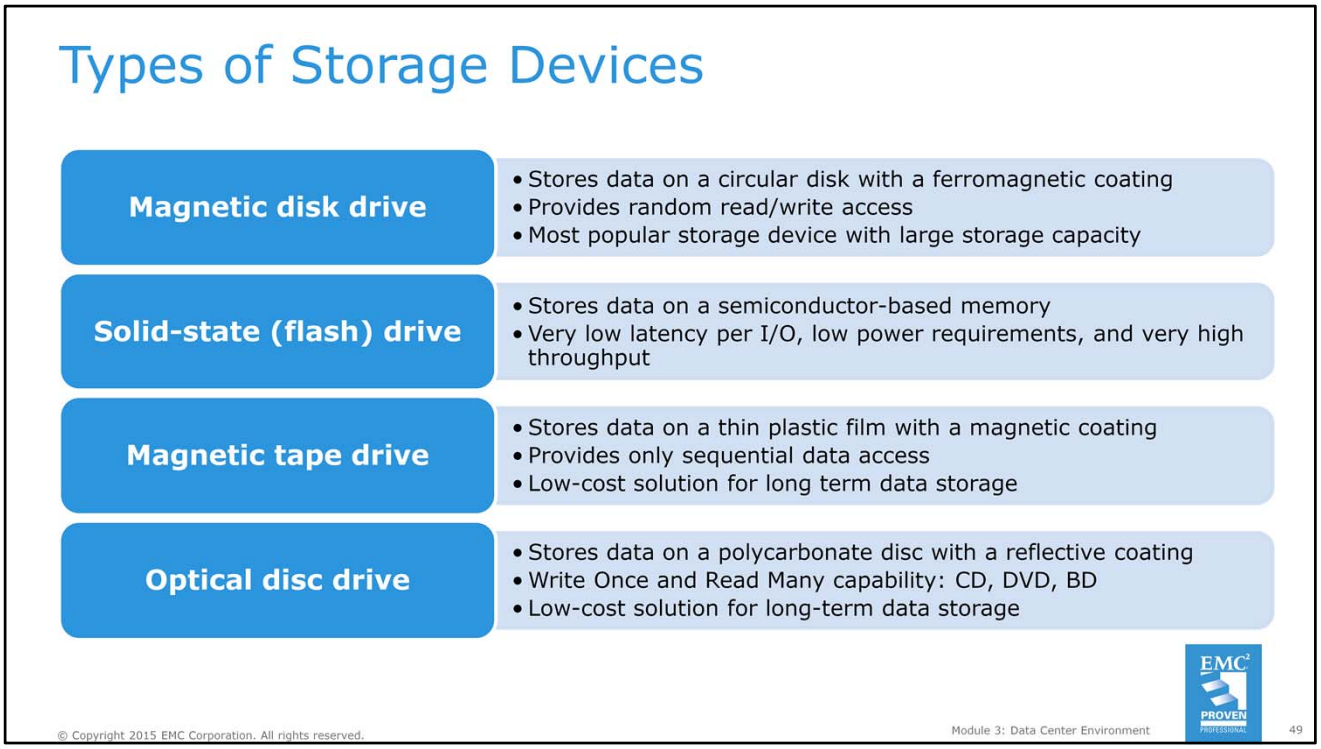
In a traditional environment, business units/departments in an organization have their own servers running the business applications of the respective business unit/department. Storage devices are connected directly to the servers and are typically internal to the server. These storage devices cannot be shared with any other server. This is called *server-centric storage architecture*. In this architecture, each server has a limited number of storage devices, and each storage device exists only in relation to the server to which it is connected. The figure on the slide depicts an example of server-centric architecture. In the figure, the servers of different departments in an organization have directly-connected storage and clients connect to the servers over a local area network (LAN) or a wide area network (WAN).

Traditional server-centric architecture has several limitations, and is therefore inadequate to satisfy the growing demand for storage capacity in third platform environments. The number of storage devices that can be connected to one server is limited, and it is not possible to scale the storage capacity. Moreover, a server cannot directly access the unused storage space available on other servers. A server failure or any administrative tasks, such as maintenance of the server or increasing its storage capacity, also results in unavailability of information. Furthermore, the proliferation of departmental servers in an organization results in silos of information, that are difficult to manage and lead to an increase in capital expenditure (CAPEX) and operating expenditure (OPEX).



To overcome the challenges of the server-centric architecture, storage evolved to the information-centric architecture. In information-centric architecture, storage devices exist completely independently of servers, and are managed centrally and shared between multiple compute systems. Storage devices assembled within storage systems form a storage pool, and several compute systems access the same storage pool over a specialized, high-speed storage area network (SAN). A SAN is used for information exchange between compute systems and storage systems, and for connecting storage systems. It enables compute systems to share storage resources, improve the utilization of storage systems, and facilitate centralized storage management. SANs are classified based on protocols they support. Common SAN deployment types are Fibre Channel SAN (FC SAN), Internet Protocol SAN (IP SAN), and Fibre Channel over Ethernet SAN (FCoE SAN). These are covered later in the course.

The figure on the slide depicts an example of information-centric architecture. In the figure, the servers of different departments in an organization are connected to the shared storage over a SAN, while clients connect to the servers over a LAN or a WAN. When a new server is deployed in the environment, storage is assigned to the server from the same shared pool of storage devices. The storage capacity can be increased dynamically and without impacting information availability by adding storage devices to the pool. This architecture improves the overall storage capacity utilization, while making management of information and storage more flexible and cost-effective.



A *magnetic disk* is a circular storage medium made of non-magnetic material (typically an alloy) and coated with a ferromagnetic material. Data is stored on both surfaces (top and bottom) of a magnetic disk by polarizing a portion of the disk surface. A disk drive is a device that comprises multiple rotating magnetic disks, called platters, stacked vertically inside a metal or plastic casing. Each platter has a rapidly moving arm to read from and write data to the disk. Disk drives are currently the most popular storage medium for storing and accessing data for performance-intensive applications. Disks support rapid access to random data locations and data can be written or retrieved quickly for a number of simultaneous users or applications. Disk drives use pre-defined protocols, such as Advanced Technology Attachment (ATA), Serial ATA (SATA), Small Computer System Interface (SCSI), Serial Attached SCSI (SAS), and Fibre Channel (FC). These protocols reside on the disk interface controllers that are typically integrated with the disk drives. Each protocol has its unique performance, cost, and capacity characteristics.

A *solid-state drive* (SSD) uses semiconductor-based memory, such as NAND and NOR chips, to store data. SSDs, also known as “flash drives”, deliver the ultra-high performance required by performance-sensitive applications. These devices, unlike conventional mechanical disk drives, contain no moving parts and therefore do not exhibit the latencies associated with read/write head movement and disk rotation. Compared to other available storage devices, SSDs deliver a relatively higher number of input/output operations per second (IOPS) with very low response times. They also consume less power and typically have a longer lifetime as compared to mechanical drives. However, flash drives do have the highest cost per gigabyte ($/GB) ratio.

(Cont’d)

A *magnetic tape* is a thin, long strip of plastic film that is coated with a magnetizable material, such as barium ferrite. The tape is packed in plastic cassettes and cartridges. A tape drive is the device to record and retrieve data on a magnetic tape. Tape drives provide linear sequential read/write data access. A tape drive may be standalone or part of a tape library. A tape library contains one or more tape drives and a storage area where a number of tape cartridges are held in slots. Tape is a popular medium for long-term storage due to its relative low cost and portability. Tape drives are typically used by organizations to store large amounts of data, typically for backup, offsite archiving, and disaster recovery. The low access speed due to the sequential access mechanism, the lack of simultaneous access by multiple applications, and the degradation of the tape surface due to the continuous contact with the read/write head are some of the key limitations of tape.

An *optical disc* is a flat, circular storage medium made of polycarbonate with one surface having a special, reflective coating (such as aluminum). An optical disc drive uses a writing laser to record data on the disc in the form of microscopic light and dark dots. A reading laser reads the dots, and generates electrical signals representing the data. The common optical disc types are compact disc (CD), digital versatile disc (DVD), and Blu-ray disc (BD). These discs may be recordable or re-writable. Recordable or read-only memory (ROM) discs have Write Once and Read Many (WORM) capability and are typically used as a distribution medium for applications or as a means to transfer small amounts of data from one system to another. The limited capacity and speed of optical discs constrain their use as a general-purpose enterprise data storage solution. However, high-capacity optical discs are sometimes used as a storage solution for fixed-content and archival data. Some cloud providers of Storage as a Service offer a facility wherein they copy backup files on encrypted optical discs, if required, and ship them to the consumers.

Overview of Storage Virtualization

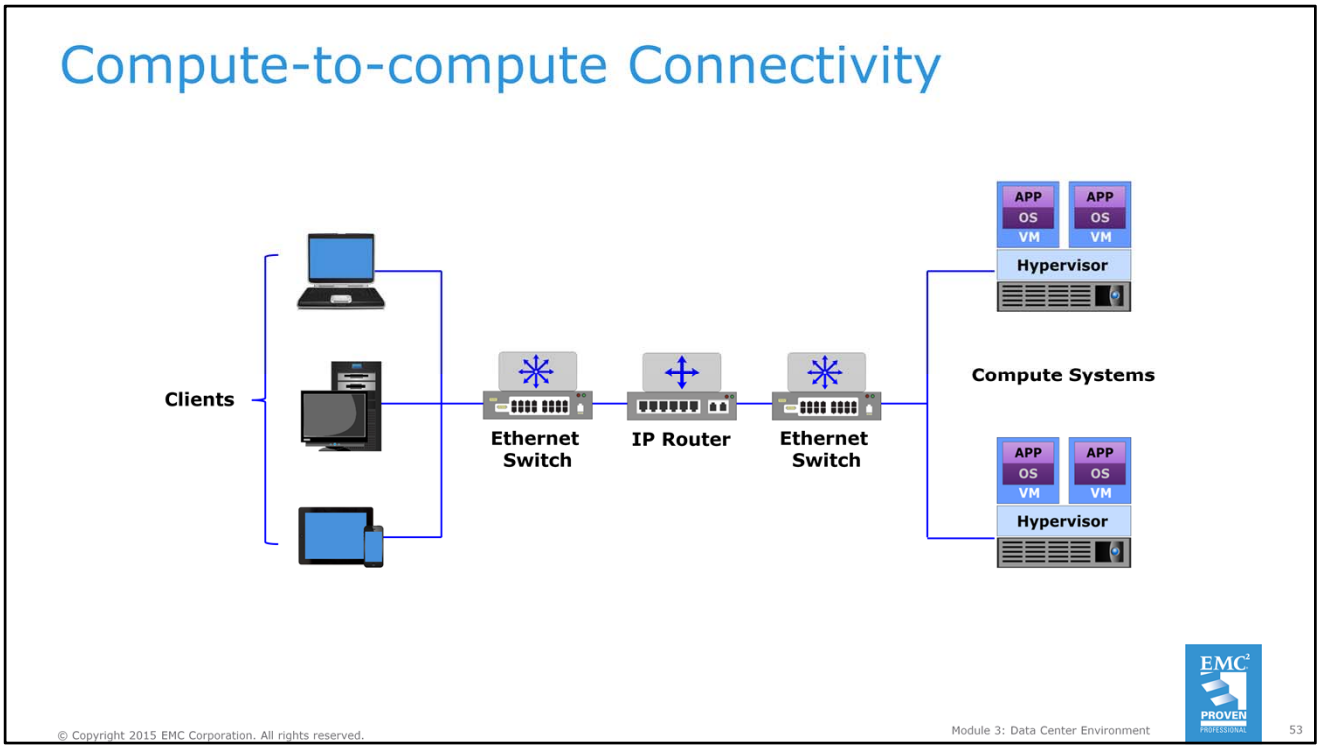
- Abstracts physical storage resources to create virtual storage resources:
  - Virtual volumes
  - Virtual disk files
  - Virtual storage systems
- Storage virtualization software can be:
  - Built into the operating environment of a storage system
  - Installed on an independent compute system
  - Built into a hypervisor

Storage virtualization is the technique of abstracting physical storage resources to create virtual storage resources. Storage virtualization software has the ability to pool and abstract physical storage resources, and present them as a logical storage resources, such as virtual volumes, virtual disk files, and virtual storage systems. Storage virtualization software is either built into the operating environment of a storage system, installed on an independent compute system, or available as hypervisor's capability. Storage virtualization will be covered in detail in the storage modules of this course.

Introduction to Connectivity

- Communication paths between IT infrastructure components for information exchange and resource sharing
- Types of connectivity:
  - Compute-to-compute connectivity
  - Compute-to-storage connectivity

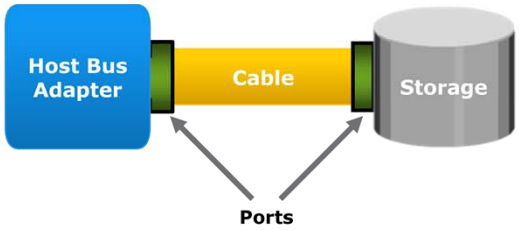
Connectivity refers to the communication paths between IT infrastructure components for information exchange and resource sharing. The two primary types of connectivity include the interconnection between compute systems, and between a compute system and storage.



Compute-to-compute connectivity typically uses protocols based on the Internet Protocol (IP). Each physical compute system is connected to a network through one or more host interface devices, called a *network interface controller* (NIC). Physical switches and routers are the commonly-used interconnecting devices. A switch enables different compute systems in the network to communicate with each other. A router is an OSI Layer-3 device that enables different networks to communicate with each other. The commonly-used network cables are copper cables and optical fiber cables. The figure on the slide shows a network (LAN or WAN) that provides interconnections among the physical compute systems. It is necessary to ensure that appropriate switches and routers, with adequate bandwidth and ports, are available to provide the required network performance.

Compute-to-storage Connectivity

- Enabled through physical components and interface protocols
- Physical connectivity components:
  - Host bus adapter, port, and cable
- Protocols define formats for communication between devices
  - Popular storage interface protocols are IDE/ATA, SCSI, and FC
- Storage may be connected directly or over a SAN



The discussion in this lesson focuses on the connectivity between compute systems and storage. Storage may be connected directly to a compute system or over a SAN as discussed previously in this lesson. Connectivity and communication between compute and storage are enabled through physical components and interface protocols. The physical components that connect compute to storage are host interface device, port, and cable.

Host bus adapter: A *host bus adapter* (HBA) is a host interface device that connects a compute system to storage or to a SAN. It is an application-specific integrated circuit (ASIC) board that performs I/O interface functions between a compute system and storage, relieving the processor from additional I/O processing workload. A compute system typically contains multiple HBAs.

Port: A port is a specialized outlet that enables connectivity between the compute system and storage. An HBA may contain one or more ports to connect the compute system to the storage. Cables connect compute systems to internal or external devices using copper or fiber optic media.

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

Storage Connectivity Protocols

Protocol	Description
IDE/ATA	• Popular interface used to connect hard disks and optical drives • The Ultra DMA/133 version of ATA supports a throughput of 133 MB/s
Serial ATA	• Serial version of the IDE/ATA specification typically used for internal connectivity • Provides data transfer rate up to 16 Gb/s (standard 3.2)
SCSI	• Popular standard for compute-to-storage connectivity • Supports up to 16 devices on a single bus • Ultra-640 version provides data transfer speed up to 640 MB/s
SAS	• Point-to-point serial protocol replacing parallel SCSI • Supports data transfer rate up to 12 Gb/s (SAS 3.0)
FC	• Widely-used protocol for high speed compute-to-storage communication • 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
IP	• Existing IP-based network leveraged for storage communication • Examples: iSCSI and FCIP protocols

Integrated Device Electronics (IDE)/Advanced Technology Attachment (ATA) is a popular interface protocol standard used for connecting storage devices, such as disk drives and optical 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/s. 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 3.2 provides a data transfer rate up to 16 Gb/s.

SCSI has emerged as a preferred connectivity protocol in high-end compute systems. 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 3.0) of serial SCSI supports a data transfer rate up to 12 Gb/s.

(Cont'd)

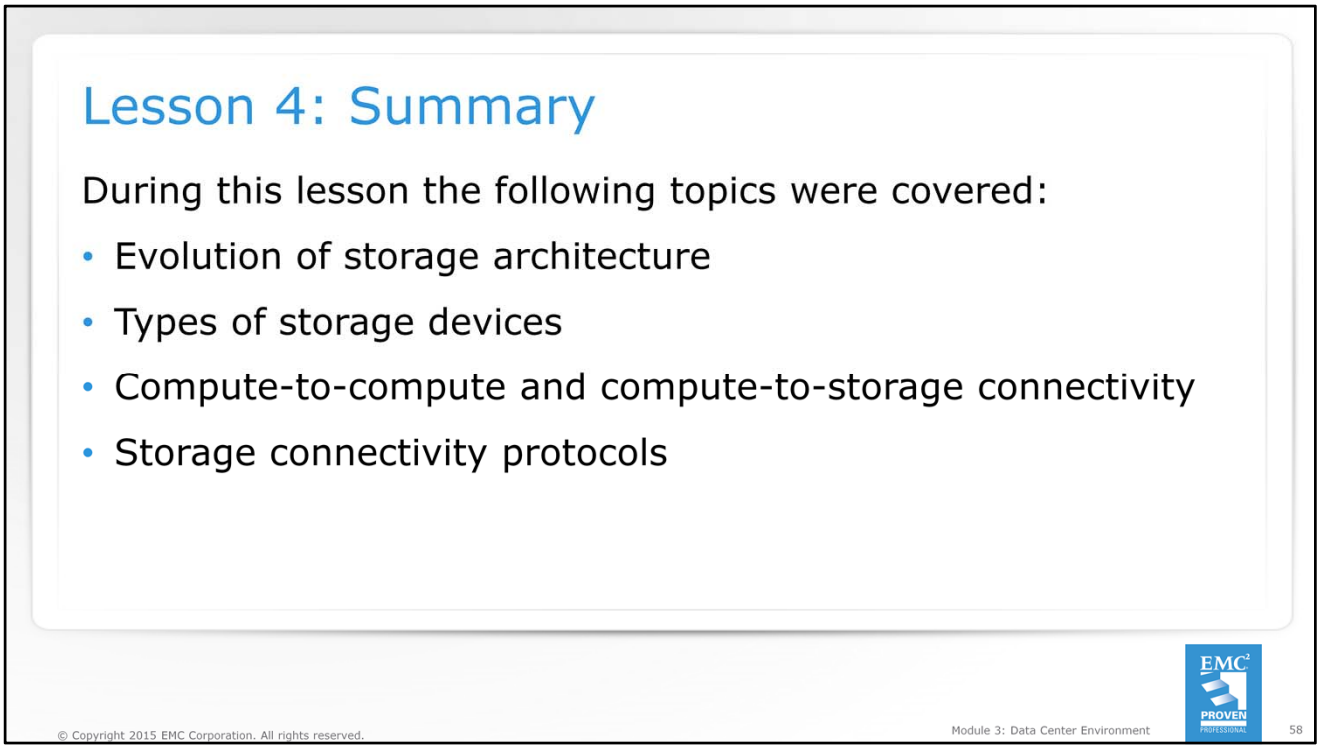
Fibre Channel is a widely-used protocol for high-speed communication to the storage device. The Fibre Channel interface provides gigabit network speed. It provides a serial data transmission that operates over copper wire and optical fiber. The latest version of the FC interface '16FC' allows transmission of data up to 16 Gb/s. The FC protocol and its features are covered in more detail in Module 9, 'Fibre Channel (FC) SAN'.

IP is a network protocol that has been traditionally used for compute-to-compute traffic. With the emergence of new technologies, an IP network has become a viable option for compute-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 compute-to-storage communication. These protocols are detailed in Module 10, 'Internet Protocol (IP) SAN'.

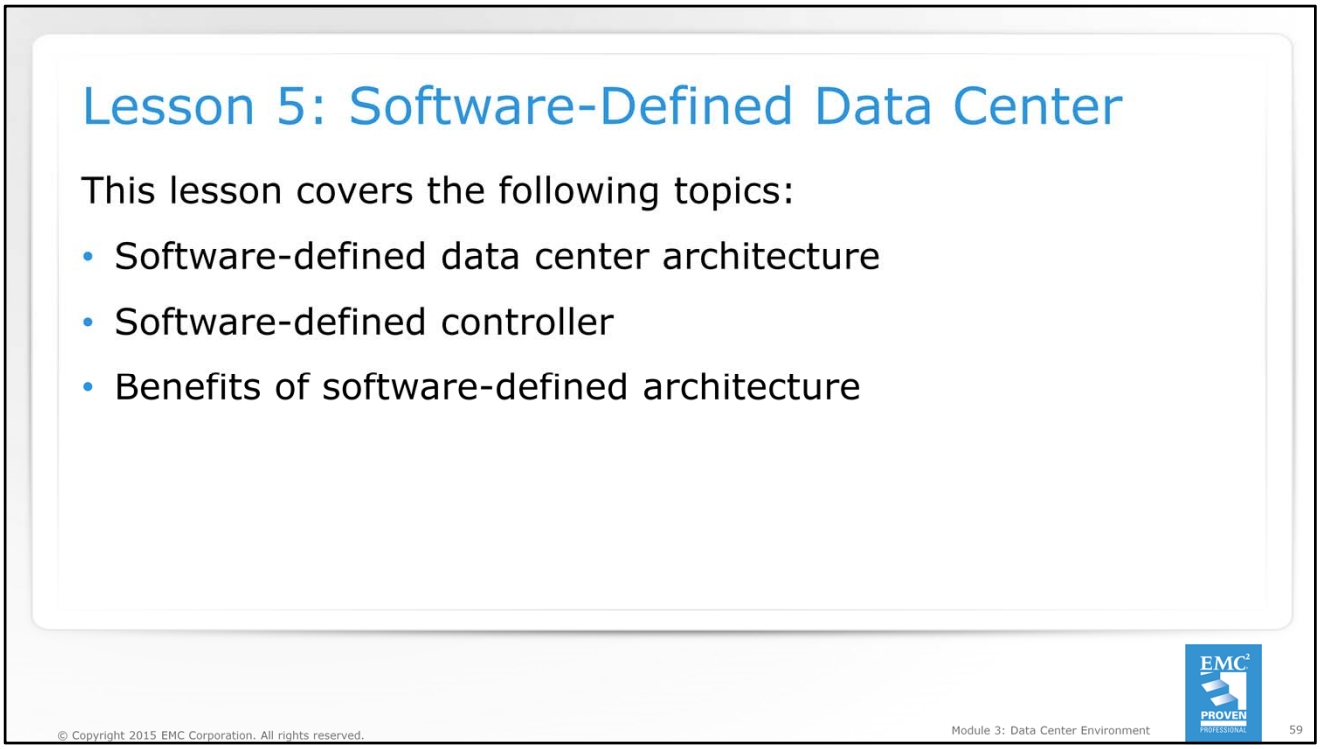
Overview of Network Virtualization

- Abstracts physical network resources to create virtual network resources:
  - Virtual switch
  - Virtual LAN
  - Virtual SAN
- Network virtualization software can be:
  - Built into the operating environment of a network device
  - Installed on an independent compute system
  - Built into a hypervisor

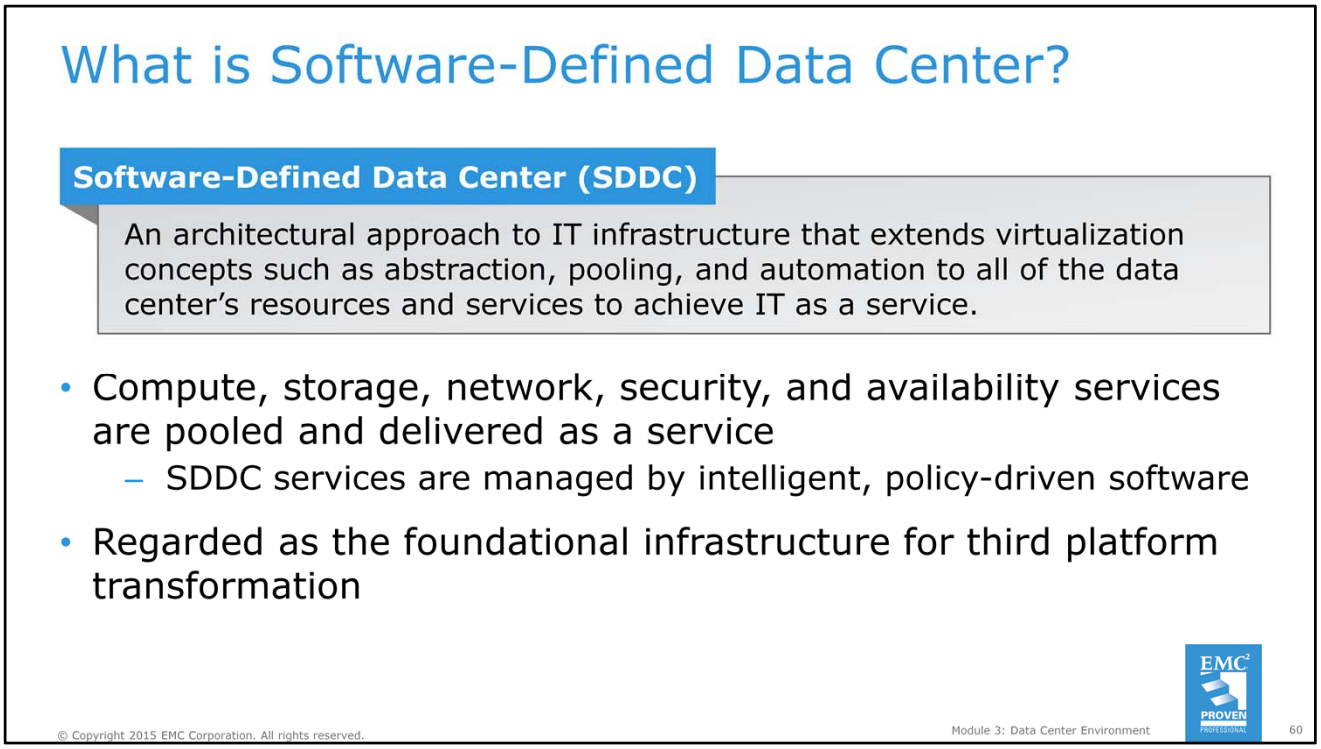
Network virtualization is the technique of abstracting physical network resources to create virtual network resources. Network virtualization software is either built into the operating environment of a network device, installed on an independent compute system or available as hypervisor's capability. Network virtualization software has the ability to abstract the physical network resources such as switches and routers to create virtual resources such as virtual switches. It also has the ability to divide a physical network into multiple virtual networks, such as virtual LANs and virtual SANs. Network virtualization available as a hypervisor's capability can emulate the network connectivity between virtual machines (VMs) on a physical compute system. It also enables creating virtual switches that appear to the VMs as physical switches. Network virtualization will be covered later in Module 9, 'Fibre Channel (FC) SAN', 10, 'Internet Protocol (IP) SAN', and 11, 'FC over Ethernet (FCoE) SAN' of this course.



This lesson covered evolution of storage architecture and the types of storage devices. This lesson also covered compute-to-compute and compute-to-storage connectivity. Further, this lesson covered different storage connectivity protocols.

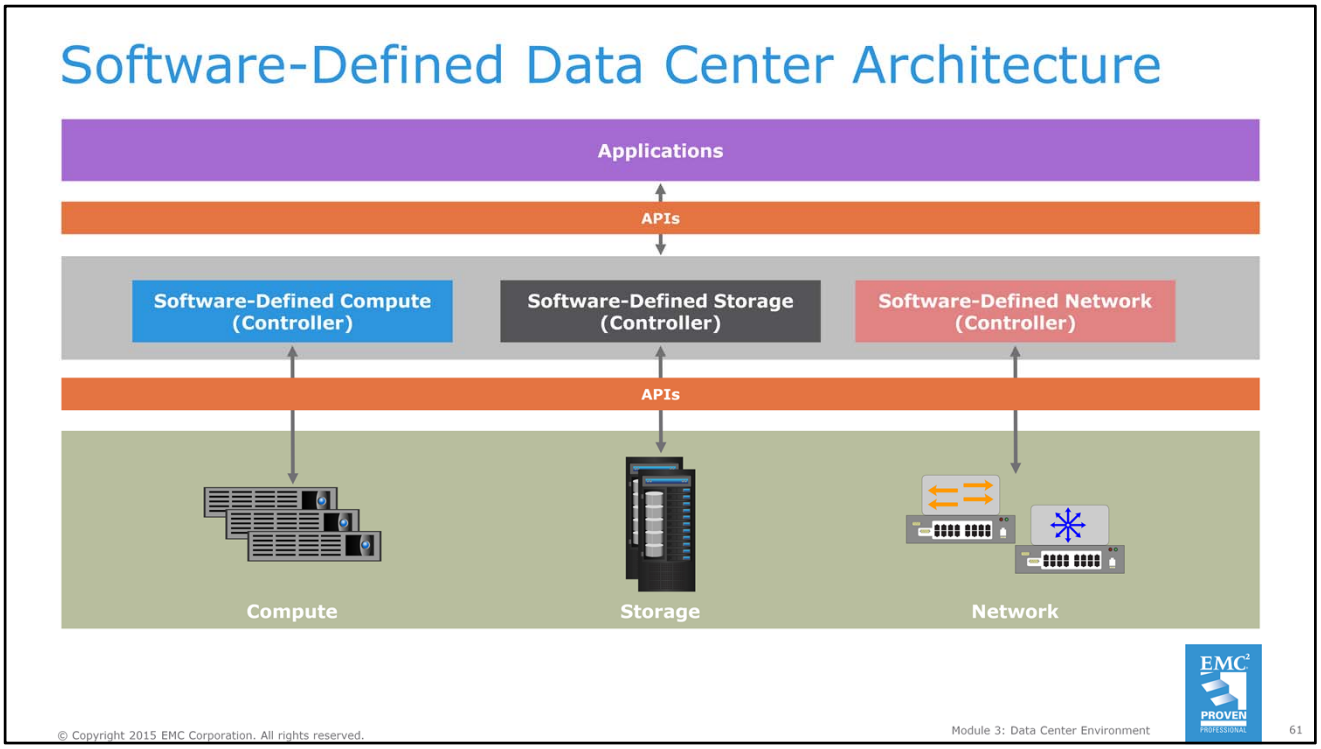


This lesson covers software-defined data center and its architecture. This lesson also covers software-defined controller and the benefits of software-defined architecture.



Software-defined data center (SDDC) is an architectural approach to IT infrastructure that extends virtualization concepts such as abstraction, pooling, and automation to all of the data center's resources and services to achieve IT as a service (ITaaS). In an SDDC, compute, storage, networking, security, and availability services are pooled, aggregated, and delivered as a service. SDDC services are managed by intelligent, policy-driven software.

SDDC is a vision that can be interpreted in many ways and can be implemented by numerous concrete architectures. Typically, an SDDC is viewed as a conglomeration of virtual infrastructure components, among which are software-defined compute (compute virtualization), software-defined network (SDN), and software-defined storage (SDS). SDDC is viewed as an important step in the progress towards a complete virtualized data center (VDC), and is regarded as the necessary foundational infrastructure for third platform transformation.



The software-defined approach separates the control or management functions from the underlying components and provides it to external software. The external software takes over the control operations and enables the management of multi-vendor infrastructure components centrally.

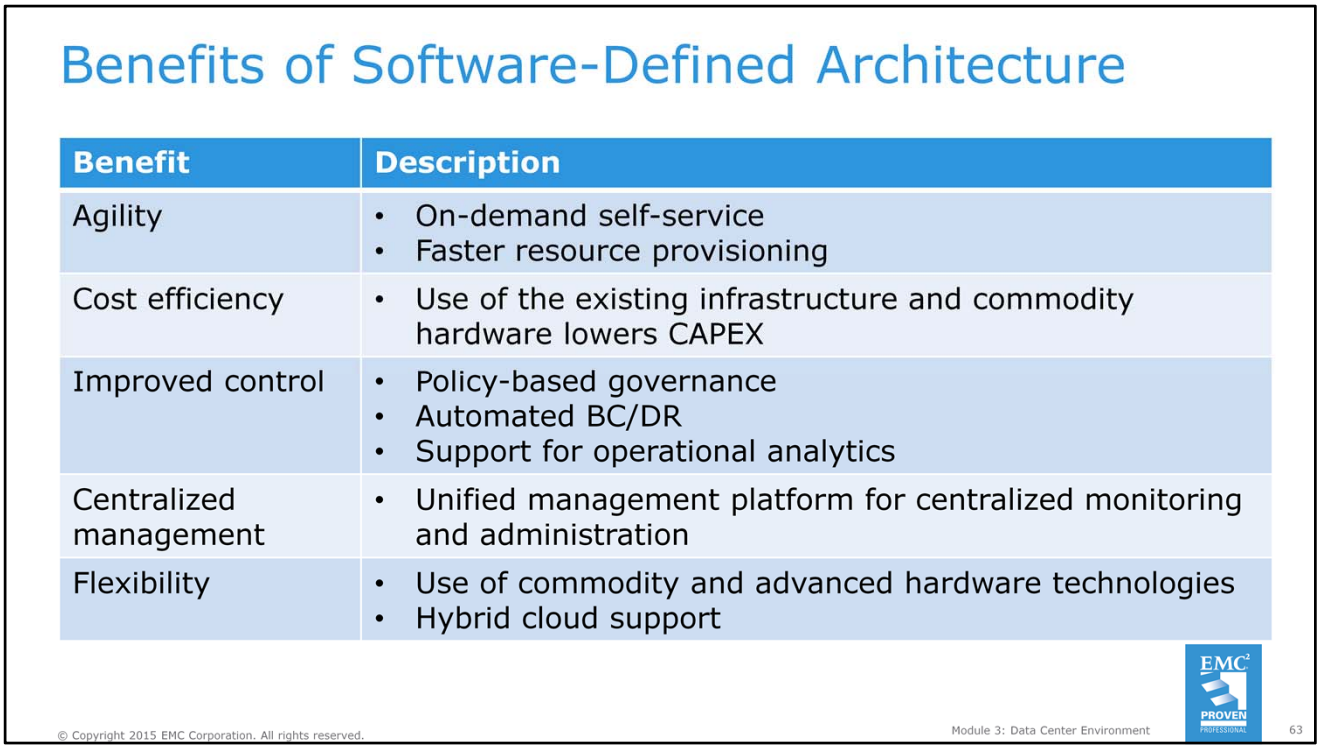
Principally, a physical infrastructure component (compute, network, and storage) has a control path and a data path. The *control path* sets and manages the policies for the resources, and the *data path* performs the actual transmission of data. The software-defined approach decouples the control path from the data path. By abstracting the control path, resource management function operates at the control layer. This gives the ability to partition the resource pools, and manage them uniquely by policy. This decoupling of the control path and data path enables the centralization of data provisioning and management tasks through software that is external to the infrastructure components. The software runs on a centralized compute system or a standalone device, called the *software-defined controller*. The figure on the slide illustrates the software-defined architecture, where the management function is abstracted from the underlying infrastructure components using controller.

Software-Defined Controller

- Discovers underlying resources and provides an aggregated view of resources
  - Abstracts the underlying hardware resources and pools them
- Enables the rapid provisioning of resources based on pre-defined policies
- Enables to apply policies uniformly across the infrastructure components, all from a software interface
- Provides interfaces that enable applications external to the controller to request resources and access them as services

A *software-defined controller* is software with built-in intelligence that automates provisioning and configuration based on the defined policies. It enables organizations to dynamically, uniformly, and easily modify and manage their infrastructure. The controller discovers the available underlying resources and provides an aggregated view of resources. It abstracts the underlying hardware resources (compute, storage, and network) and pools them. This enables the rapid provisioning of resources from the pool based on pre-defined policies that align to the service level agreements for different consumers.

The controller provides a single control point to the entire infrastructure enabling policy-based infrastructure management. The controller enables an administrator to use a software interface to manage the resources, node connectivity, and traffic flow; control behavior of underlying components; apply policies uniformly across the infrastructure components; and enforce security. The controller also provides interfaces that enable applications, external to the controller, to request resources and access these resources as services.



By extending virtualization throughout the data center, SDDC provides several benefits to the organizations. Some key benefits are described below.

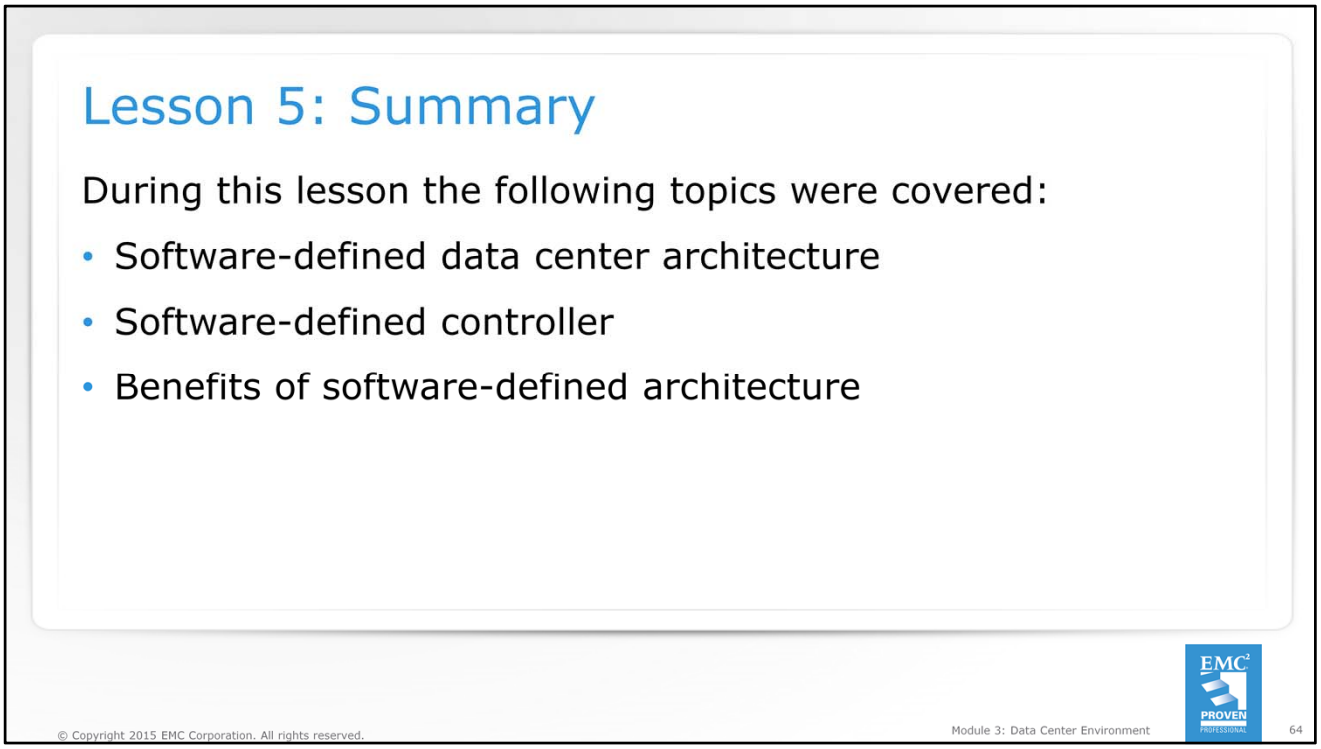
Agility: SDDC enables faster provisioning of resources based on workload policies. Consumers provision infrastructure resources via self-service portal. These significantly improve business agility.

Cost efficiency: SDDC enables organizations to use commodity hardware and existing infrastructure, which significantly lowers CAPEX.

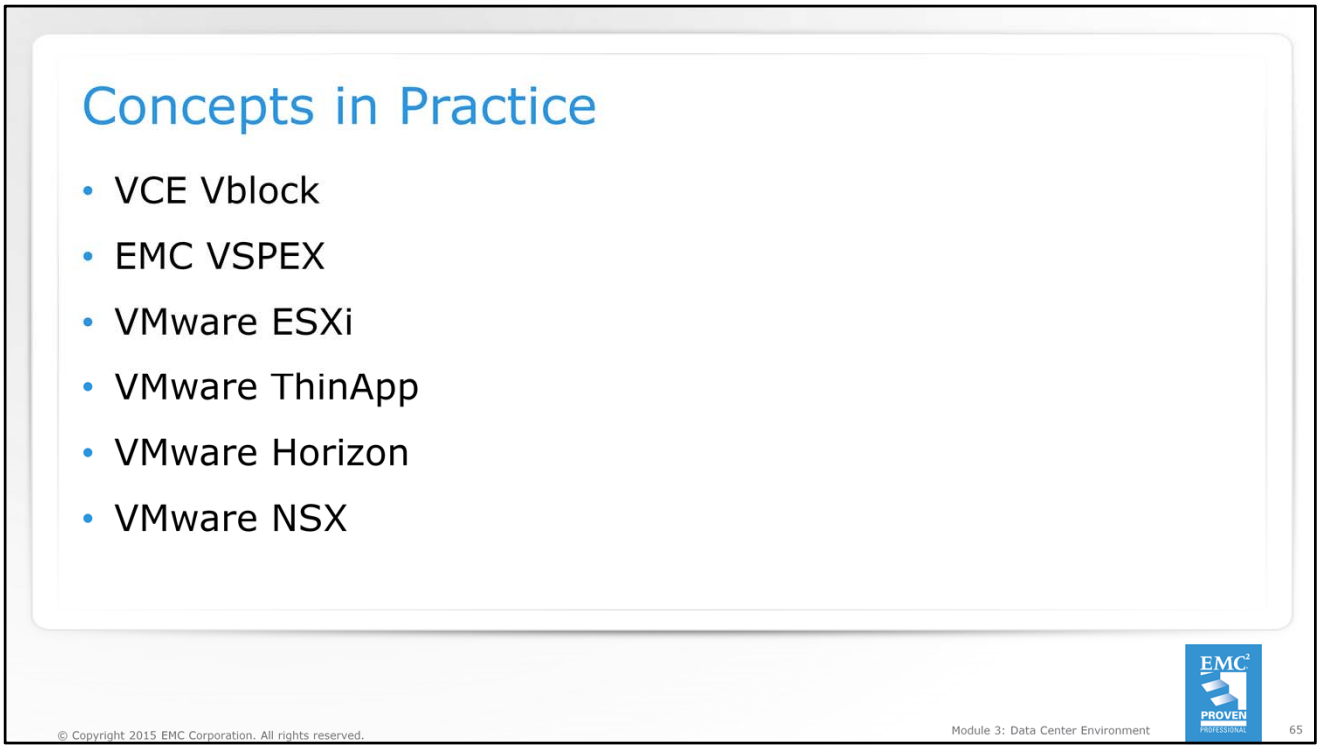
Improved control: SDDC provides improved control over application availability and security through policy-based governance. SDDC provides automated business continuity and disaster recovery features. It also has virtualization-aware security and compliance features, and provides support for performing operational analytics.

Centralized management: An SDDC is automated and managed by intelligent, policy-based data center management software, vastly simplifying governance and operations. A single, unified management platform allows central monitoring and administration of all heterogeneous physical and virtual resources across geographies and hybrid clouds.

Flexibility: SDDC enables organizations to use heterogeneous commodity hardware and the latest advanced hardware technologies as suitable. Lower-value workloads can run on commodity hardware, while software-based services and mission-critical applications can run on advanced, more-intelligent infrastructure. SDDC also supports adoption of the hybrid cloud model through the use of standard protocols and APIs.



This lesson covered software-defined data center and its architecture. This lesson also covered software-defined controller and the benefits of software-defined architecture.



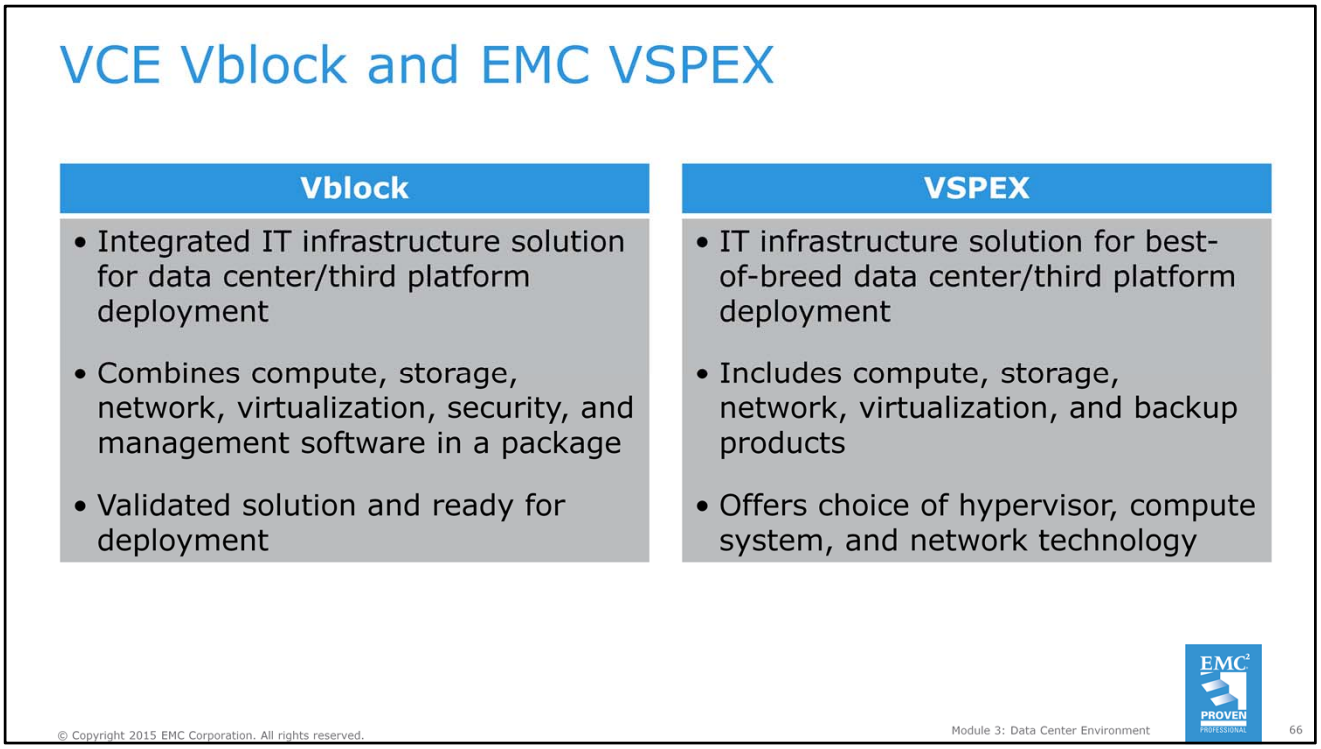
The Concepts in Practice section covers VCE Vblock, EMC VSPEX, VMware ESXi, VMware ThinApp, VMware Horizon, and VMware NSX.

Note:

For the latest information on VCE products, visit www.vce.com.

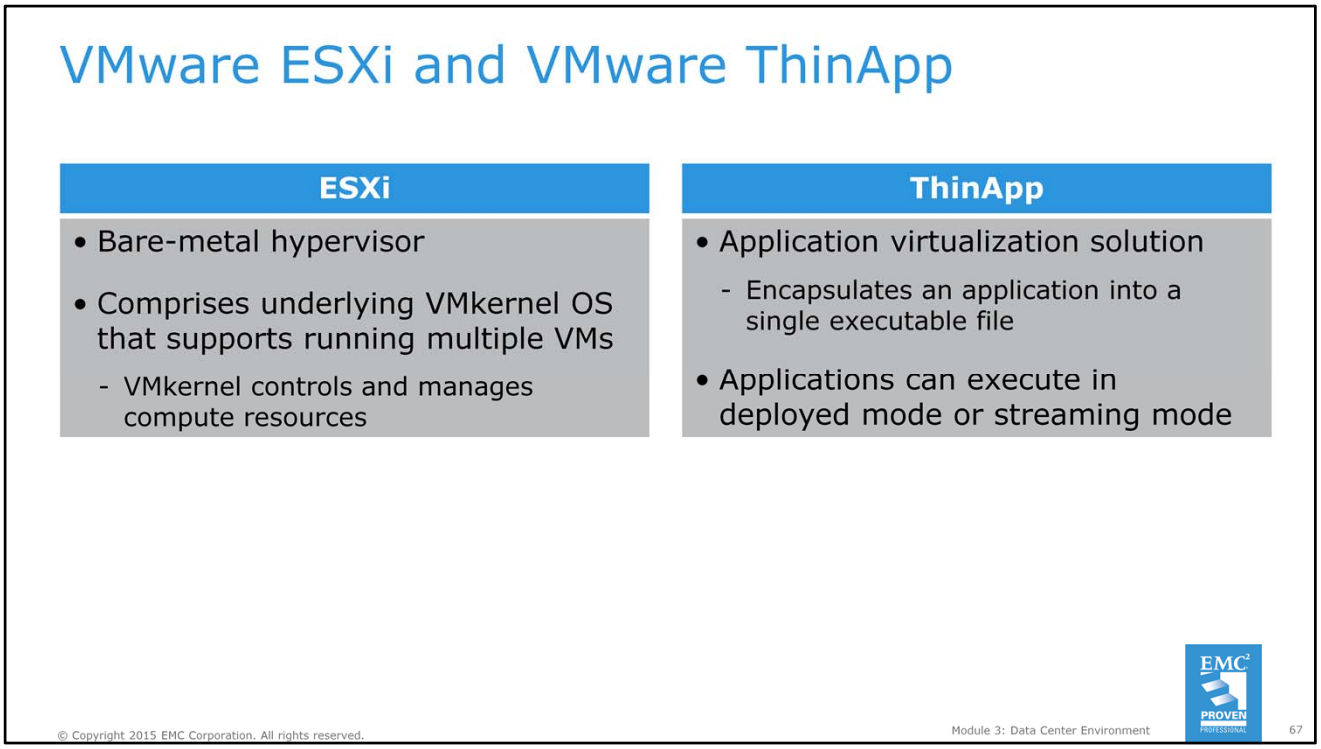
For the latest information on EMC products, visit www.emc.com.

For the latest information on VMware products, visit www.vmware.com.



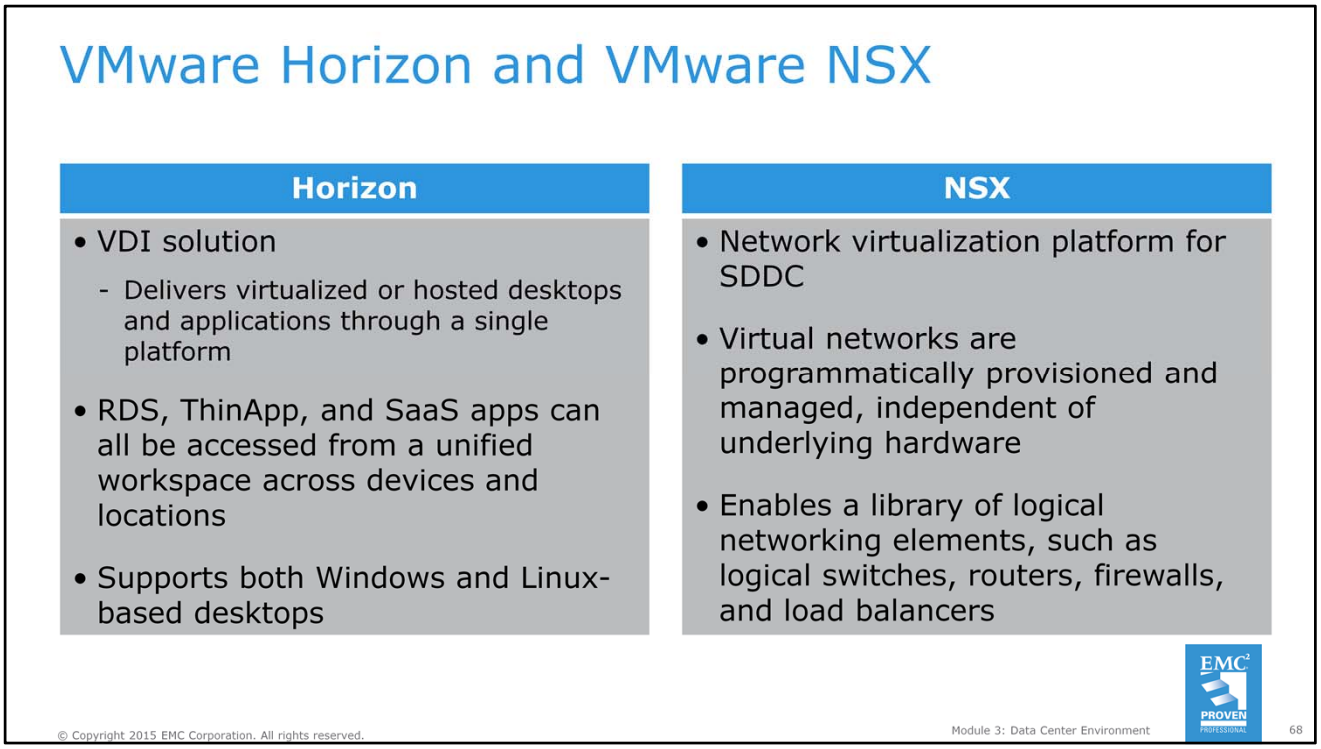
VCE Vblock is a completely integrated IT infrastructure offering from Virtual Computing Environment Company (VCE), and includes compute, storage, network, and virtualization products. These products are provided by EMC (storage solution provider), VMware (virtualization solution provider), and Cisco (networking and compute solution provider), who have formed a coalition to deliver Vblocks. Vblock is an integrated IT infrastructure solution that combines compute, storage, network, virtualization, security, and management software into a single package. This solution is a self-contained unit that accelerates deployment of a data center or a third platform infrastructure. Vblocks are pre-architected, preconfigured, pretested and have defined performance and availability attributes. Rather than the customers buying and assembling the individual IT infrastructure components, Vblock provides a validated solution and is factory-ready for deployment and production. This saves significant cost and deployment time associated with building a data center infrastructure.

EMC VSPEX is an end-to-end virtualized infrastructure solution, which includes compute, storage, network, virtualization, and backup products. The product vendors include EMC, Brocade, Cisco, Citrix, Intel, Microsoft, and VMware. VSPEX offers choice to the customers in terms of the hypervisor, compute systems, and networking components. Therefore, customers have the flexibility to choose the infrastructure components that fit their existing IT infrastructures. EMC VSPEX is a complete virtualization solution that accelerates the deployment of data center infrastructures. It provides customers the flexibility to choose the hypervisor, compute system, and network technology they prefer along with EMC's VNX and VNXe unified storage, and EMC's backup and recovery solutions. Regardless of customer's choice of hypervisor, compute system, and network technologies, validation of VSPEX ensures fast and low-risk deployment. VSPEX significantly reduces the planning, sizing, and configuration burdens that typically come with designing, integrating, and deploying a best-of-breed solution. VSPEX, unlike Vblock, does not offer unified management. It comes with element management tools such as Microsoft System Center, VMware vCenter Operations Management Suite, and EMC Unisphere. But, it offers customers the choice of service elements that make up the solution.



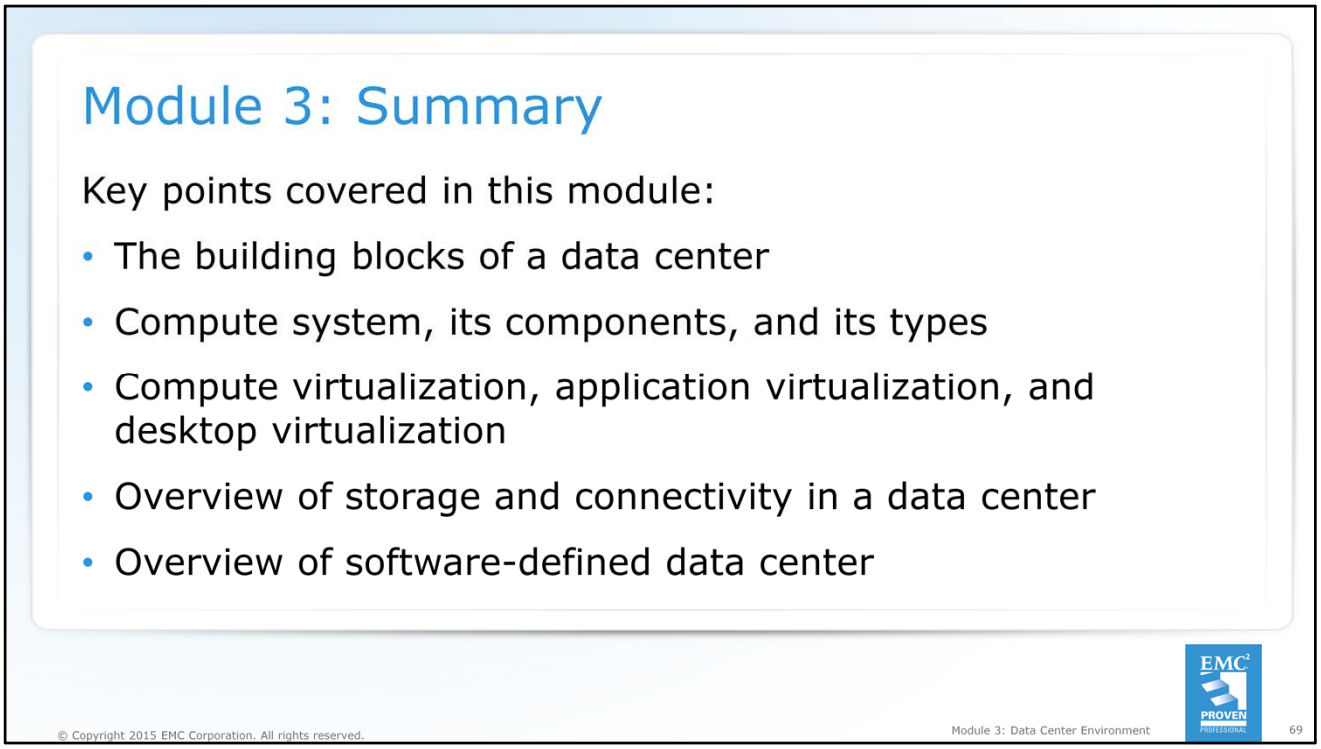
VMware ESXi is a bare-metal hypervisor. ESXi has a compact architecture that is designed for integration directly into virtualization-optimized compute system hardware, enabling rapid installation, configuration, and deployment. ESXi abstracts processor, memory, storage, and network resources into multiple VMs that run unmodified operating systems and applications. The ESXi architecture comprises underlying operating system called VMkernel, that provides a means to run management applications and VMs. VMkernel controls all hardware resources on the compute system and manages resources for the applications. It provides core OS functionality, such as process management, file system, resource scheduling, and device drivers.

VMware ThinApp is an application virtualization solution. ThinApp encapsulates an application, along with the assets it requires, into a single package that can be deployed, managed, and updated independently from the underlying OS. A ThinApp application is a single executable file that can be directly executed on a compute system. The application package can be distributed across an organization's environment from a centralized location, such as a cloud. ThinApp applications can be executed in two modes: deployed mode or streaming mode. In either case, agent software is not required on the end-point device. In deployed mode, ThinApp packages are copied to the compute system, which then executes the package locally. In deployed mode, an application can run regardless of the availability of the network connectivity. In streaming mode, the application is streamed from the centralized location, over a network, to the compute system.



VMware Horizon is a VDI solution for delivering virtualized or hosted desktops and applications through a single platform to the end users. These desktop and application services—including RDS, hosted apps, packaged apps with VMware ThinApp, and SaaS apps—can all be accessed from one unified workspace across devices and locations. Horizon provides IT with a streamlined approach to deliver, protect, and manage desktops and applications while containing costs and ensuring that end users can work anytime, anywhere, on any device. Horizon supports both Windows as well as Linux-based desktops.

VMware NSX is a network virtualization platform for the Software-defined Data Center (SDDC). Similar to virtual machines for compute, virtual networks are programmatically provisioned and managed independent of underlying hardware. NSX reproduces the entire network model in software, enabling any network topology—from simple to complex multi-tier networks—to be created and provisioned. It enables a library of logical networking elements and services, such as logical switches, routers, firewalls, load balancers, VPN, and workload security. Users can create isolated virtual networks through custom combinations of these capabilities. NSX is ideal for data centers with more than 500 virtual machines, for multi-tenant clouds, large enterprise private and R&D clouds, and multi-hypervisor cloud environments.



This module covered the building blocks of a data center environment. This module also covered compute system, its components, and its types. Additionally, this module covered compute virtualization, application virtualization, and desktop virtualization. Further, this module covered an overview of storage and connectivity in a data center. Finally, this module covered an overview of software-defined data center.