Virtualization

1. Virtualization is the Software method (including hypervisors) used to allow multiple virtual computing resources to run on a single hardware platform//
2. Virtualization uses software to create an abstraction layer over computer hardware that allows the hardware elements of a single computer—processors, memory, storage and more—to be divided into multiple virtual computers, commonly called **virtual machines** (VMs).//
3. Each VM runs its own operating system (OS) and behaves like an independent computer, even though it is running on just a portion of the actual underlying computer hardware.//
4. Virtualization enables more efficient utilization of physical computer hardware and allows a greater return on an organization's hardware investment.//

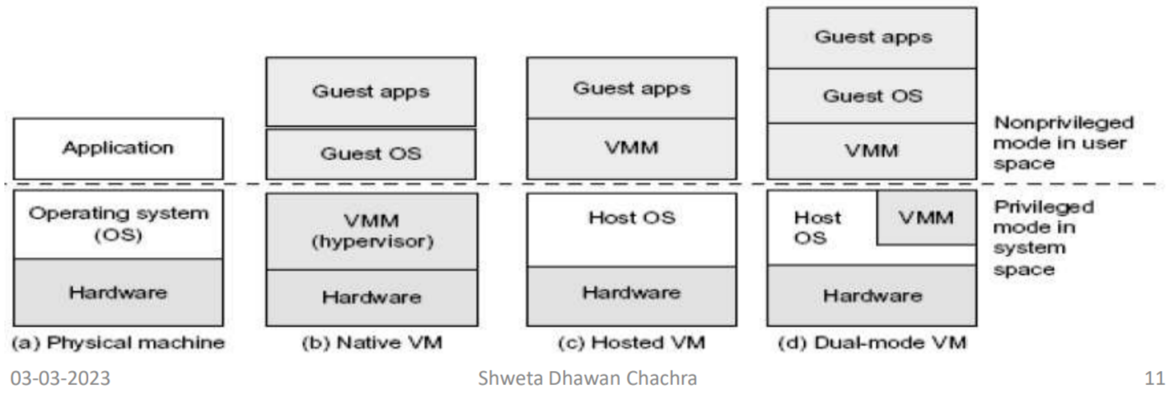
Benefits of Virtualization

- ❖ **Resource efficiency:** Before virtualization, each application server required its own dedicated physical CPU—IT staff would purchase and configure a separate server for each application they wanted to run. Invariably, each physical server would be underused. In contrast, server virtualization lets you run several applications—each on its own VM with its own OS—on a single physical computer (typically an x86 server) without sacrificing reliability. This enables maximum utilization of the physical hardware's computing capacity.
- ❖ **Easier management:** Replacing physical computers with software-defined VMs makes it easier to use and manage policies written in software.Automated deployment and configuration tools enable administrators to define collections of virtual machines and applications as services, in software templates. This means that they can install those services repeatedly and consistently without cumbersome, time-consuming. and error-prone manual setup.
- ❖ **Minimal downtime:** OS and application crashes can cause downtime and disrupt user productivity. Admins can run multiple redundant virtual machines alongside each other and failover between them when problems arise. Running multiple redundant physical servers is more expensive.
- ❖ **Faster provisioning:** Buying, installing, and configuring hardware for each application is time-consuming. Provided that the hardware is already in place, provisioning virtual machines to run all your applications is significantly faster.

—--

- Main Files: • Configuration file • Virtual disk file • NVRAM settings file • Log file

- Virtual machines (VMs) are virtual environments that simulate a physical computer in software form.

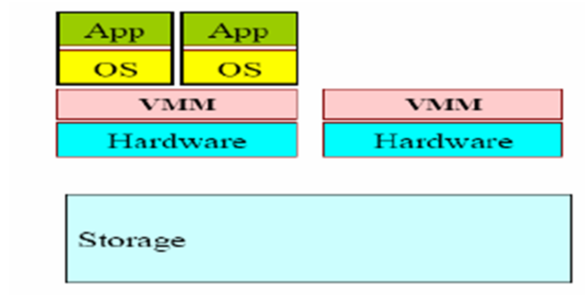


(b)- This Hypervisor handles the bare hardware(CPU, memory, I/O) directly, hence called as Bare Metal Hypervisor or Native VM

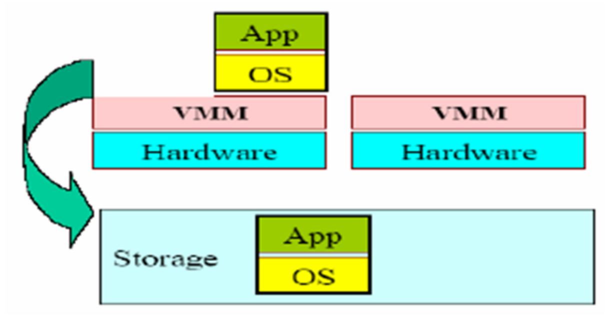
(c)-VMM runs in Non-priviledged mode. The Host OS needs not to be modified.

(d)-VM implemented in Dual Mode. Part of the VMM runs at the user level and another part runs at the supervisor level. Host OS has to be modified to some extent.

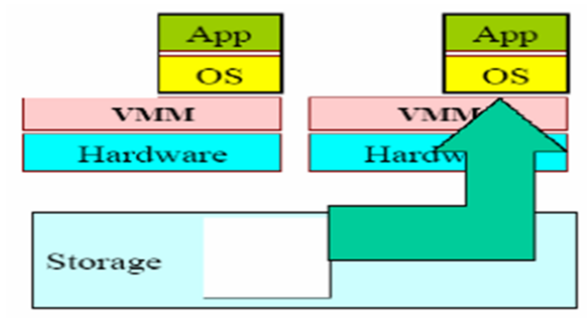
Primitive Operations in Virtual Machines:



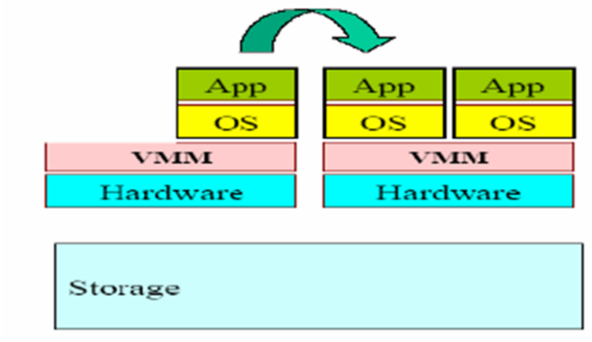
(a) Multiplexing



(b) Suspension (Storage)

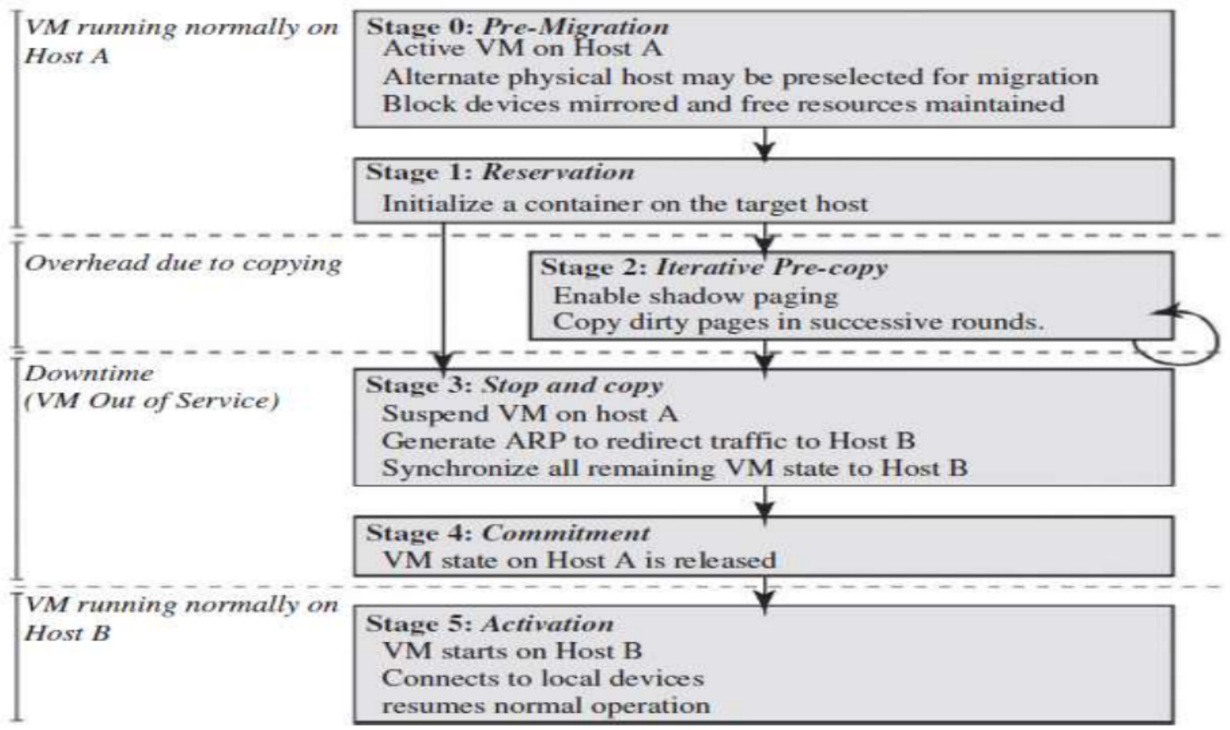


(c) Provision (Resume)



(d) Life migration

Live Migration


VM running normally on Host A
Stage 0: Pre-Migration
Active VM on Host A
Alternate physical host may be preselected for migration
Block devices mirrored and free resources maintained
Stage 1: Reservation
Initialize a container on the target host
Overhead due to copying
Stage 2: Iterative Pre-copy
Enable shadow paging
Copy dirty pages in successive rounds.
Downtime (VM Out of Service)
Stage 3: Stop and copy
Suspend VM on host A
Generate ARP to redirect traffic to Host B
Synchronize all remaining VM state to Host B
Stage 4: Commitment
VM state on Host A is released
VM running normally on Host B
Stage 5: Activation
VM starts on Host B
Connects to local devices
resumes normal operation

Cold Migration(PPT)

Levels Of Virtualization:

The main function of the software layer for virtualization is to virtualize the physical hardware of a host machine into virtual resources to be used by the VMs, exclusively. This can be implemented at various operational levels,

Instruction Set Architecture level

- Transforming the physical architecture of the system's instruction set completely into software.//
- The guest systems issue instructions for the emulator to process and execute. //
- The instructions are received by the emulator, which transforms them into a native instruction set. //
- The native instructions are run on the host m/c's hardware. //
- The instructions include both the processor-oriented instructions and the I/O specific ones//
- For example, MIPS binary code can run on an x86-based host machine with the help of ISA emulation.
- The basic emulation method is through code interpretation.
- An interpreter program interprets the source instructions to target instructions one by one.One source instruction may require tens or hundreds of native target instructions to perform its function. Obviously, this process is relatively slow.
- QEMU is a generic and open source machine emulator and virtualizer. //

Advantage:

With this approach, it is possible to run a large amount of legacy binary code written for various processors on any given new hardware host machine.

Disadvantage:

An interpreter program interprets the source instructions to target instructions one by one. Therefore the system with the virtualization at ISA level shows a poor performance.

Hardware Abstraction Level

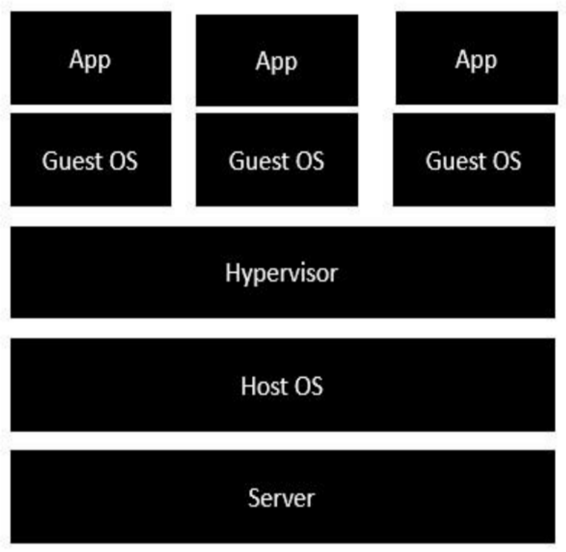
- It generates virtual hardware environments for VMs, and manages the underlying hardware through virtualization. //
- Typical systems: VMware, Virtual PC, Denali, Xen//
- The idea is to virtualize a computer's resources, such as its: – Processors – memory, and – I/O devices. //
- The intention is to upgrade the hardware utilization rate by multiple users concurrently//
- It gives the appearance of multiple separate m/cs each of which can be used as a normal system.
- The Xen hypervisor has been applied to virtualize x86-based machines to run Linux or other guest OS applications
- Hardware Level virtualization needs trapping the execution of privileged instructions by the virtual m/c, which must pass these instructions to the VMM for being handled properly. //

  Advantage: has higher performance and good application isolation //

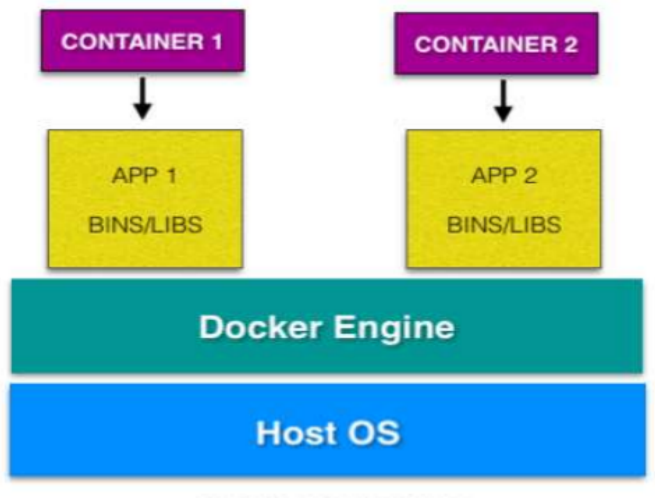
  Shortcoming & limitation: very expensive to implement.//

Operating system Level

- To overcome the issues of redundancy (In case the physical and virtual Oss are the same)and time consumption we implement virtualization at a higher level, i.e. at the level of the OS//
- Here the kernel of an OS allows more than one isolated user-space instance to exist. Such instances are called containers/software containers or virtualization engines. In other words, the OS kernel will run a single operating system & provide that operating system's functionality to replicate on each of the isolated partitions.
- OS virtualization usually imposes little or no overhead.//
- OS Virtualization is capable of live migration//
- It can also use dynamic load balancing of containers between nodes and a cluster.//



- Dockers containers are analogous to physical containers that you can use to store, package, and transport goods.//
- But instead of tangible goods, they're containers for software applications//
- A container is an environment that runs an application that is not dependent on the operating system. //
- A docker container is a portable unit of software that has the application along with the associated dependency and configuration//
- Each container runs isolated tasks. //
- It cannot harm the host machine nor come in conflict with other apps running in separate containers.



Lighter weight: Unlike VMs, containers don't carry the payload of an entire OS instance and hypervisor. They include only the OS processes and dependencies necessary to execute the code. Container sizes are measured in megabytes (vs. gigabytes for some VMs), make better use of hardware capacity, and have faster startup times. //

Improved developer productivity: Containerized applications can be written once and run anywhere. And compared to VMs, containers are faster and easier to deploy, provision and restart.//

Greater resource efficiency: With containers, developers can run several times as many copies of an application on the same hardware as they can using VMs. This can reduce cloud spending.//

Library Support Level

Library Level

OS system calls are lengthy and cumbersome. Which is why applications opt for APIs from user-level libraries.//

Create execution environments for running alien programs on a platform • rather than creating a VM to run the entire operating system.

• It is done by API call interception and remapping.

Also called

User-level Application Binary Interface (ABI) or API emulation

Library interfacing virtualization is made possible by API hooks. These API hooks control the communication link from the system to the applications.//

Some tools available today, such as **vCUDA and WINE**, have successfully demonstrated this technique.

Advantage: It has very low implementation effort • Shortcoming & limitation: poor application flexibility and isolation.//

Application Level

Application-level virtualization comes handy when you wish to virtualize only an application. It does not virtualize an entire platform or environment.//

On an operating system, applications work as one process. Hence it is also known as process-level virtualization.//

It is generally useful when running virtual machines with high-level languages. Here, the application sits on top of the virtualization layer, which is above the application program.//

The application program is, in turn, residing in the operating system.

Programs written in high-level languages and compiled for an application-level virtual machine can run fluently here.

Advantage: has the best application isolation

Shortcoming & limitation: low performance, low application flexibility and high implementation complexity.

Virtualization Design Requirements

- **Equivalence Requirement**
    - A m/c that is developed through virtualization must have a logical equivalence with the real m/cs //
    - Needs to match the capabilities of the physical system in its computational performance.//
    - execute all the applications and programs that are designed to execute on the real m/cs.
- **Efficiency Requirement**
    - the virtual m/c must be as efficient in its performance as a real system.//
    - Virtualization is primarily done with a purpose of getting efficient software without the physical hardware.
- **Resource Control requirement**
    - A typical computer system is a combination of various resources, including processors, memory, and I/O devices.//
    - All these resources must be managed and controlled effectively by the VMM. //

- **HyperVisor:**

    The hypervisor supports hardware-level virtualization on bare metal devices like CPU, memory, disk and network interfaces.

- **Type 1 hypervisor**: –
    1. Type 1 hypervisor or the bare metal hypervisor //
    2. Sits on the bare metal computer hardware like the CPU, memory, etc. //
    3. All the guest operating systems are a layer above the hypervisor. //
    4. So the hypervisor is the first layer over the hardware. //
    5. Examples are Microsoft Hyper-V
- **Type 2 hypervisor: –**
    1. Type 2 or the hosted hypervisor //
    2. Does not run over the bare metal hardware but they run over a host operating system. //
    3. The hypervisor is the second layer over the hardware.//
    4. The guest operating systems run a layer over the hypervisor and so they form the third layer.
    5. Examples are FreeBSD.

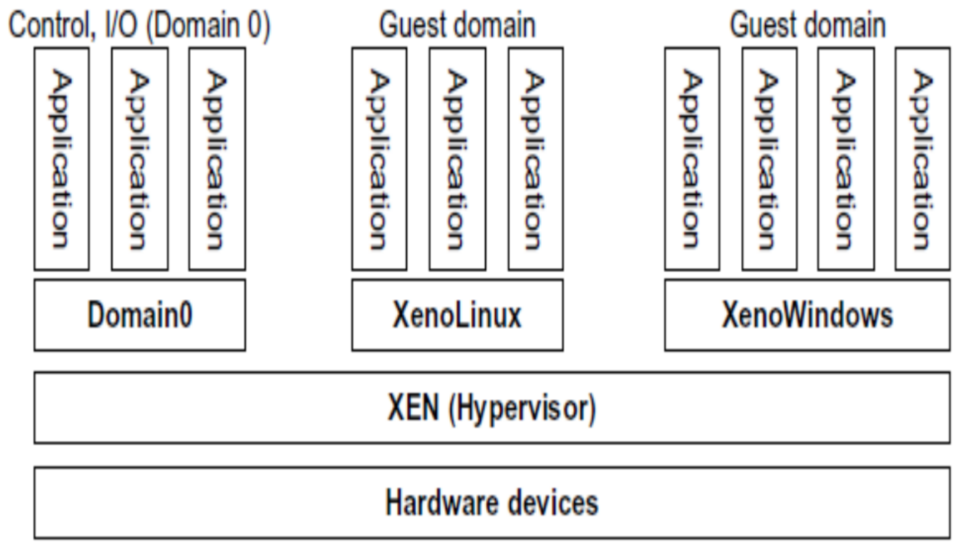
Architecture Types:

- ❖ **microkernel hypervisor** –
    1. includes only the basic and unchanging functions (such as physical memory management and processor scheduling). //
    2. The device drivers and other changeable components are outside the hypervisor.//
    3. Micro-kernel architecture like the Microsoft Hyper-V.
- ❖ **monolithic hypervisor** –
    1. implements all the aforementioned functions, including those of the device drivers. //
    2. Size of the hypervisor code of a micro-kernel hypervisor is smaller than that of a monolithic hypervisor.//

—-------

Xen Hypervisor

1. Xen is a Type 1 Hypervisor//
2. Xen is a microkernel hypervisor as it does not include any device drivers natively. //
3. It just provides a mechanism by which a guest OS can have direct access to the physical devices.
4. The Xen Hypervisor is primarily based on para virtualization. It is achieved by modifying portions of the guest operating system by Xen.//
5. Recently Xen has been advanced to support full virtualization using hardware assisted virtualization.//



6. The core components of a Xen system are //
    - the hypervisor
    - Kernel
    - applications.
7. Many guest OSes can run on top of the hypervisor. Not all guest OSes are created equal, and one in particular controls the others.
8. Guest operating systems are executed within Domains which represent VMs instances.
9. The guest OS, which has control ability, is called Domain 0 while the others are called Domain U.//
10. Domain 0 is the privileged guest OS of Xen and has privileged access to the host //
11. It is also called the Management VM as it has the privilege to manage other VMs implemented on the same host.//
12. It is first loaded when Xen boots without any file system drivers being available. .//
13.
14. Domain 0 is designed to access hardware directly and manage devices.
15. Therefore, one of the responsibilities of Domain 0 is to allocate and map hardware resources for the other guest domains(Domain U).//
16. If Domain 0 is compromised, the hacker can control the entire system. //
17. Security policies are needed to improve the security of Domain 0.//

Hyper Calls://

"The hypervisor provides hypercalls for the guest OSes and applications. "//

Sensitive system calls need to be re-implemented with hypercalls.//

The Xen hypervisor is able to catch the execution of all the sensitive instructions manage them and return the control to the guest operating system by means of a handler.//

A hypercall is a software trap from a domain to the hypervisor, just as a syscall is a software trap from an application to the kernel.//

All are the same thing: a system call to the hypervisor below. //

Microsoft Hyper-V hypervisor

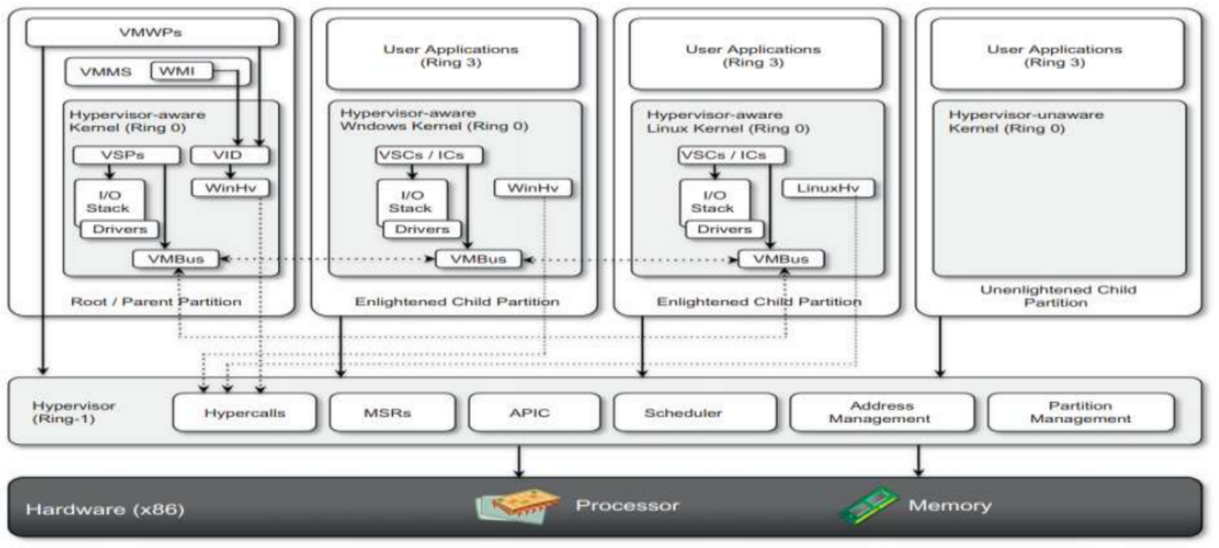

VMWPs
VMMS
WMI
Hypervisor-aware Kernel (Ring 0)
VSPs
VID
I/O Stack
WinHv
Drivers
VMBus
Root / Parent Partition
User Applications (Ring 3)
Hypervisor-aware Wndows Kernel (Ring 0)
VSCs / ICs
I/O Stack
Drivers
WinHv
VMBus
Enlightened Child Partition
User Applications (Ring 3)
Hypervisor-aware Linux Kernel (Ring 0)
VSCs / ICs
I/O Stack
Drivers
LinuxHv
VMBus
Enlightened Child Partition
User Applications (Ring 3)
Hypervisor-unaware Kernel (Ring 0)
Unenlightened Child Partition
Hypervisor (Ring-1)
Hypercalls
MSRs
APIC
Scheduler
Address Management
Partition Management
Hardware (x86)
Processor
Memory

FIGURE 3.17
Microsoft Hyper-V architecture.

1. Microsoft Hyper-V is a Type 1 Hypervisor//
2. Hyper-V supports multiple and concurrent execution of guest operating systems by means of partitions. //
3. A partition is a completely isolated environment in which an operating system is installed and run. //
4. The parent partition (also called the root partition) is the only one that has direct access to the hardware. //
5. It runs the virtualization stack, hosts all the drivers required to configure guest operating systems, and creates child partitions through the hypervisor.//
6. Child partitions are used to host guest operating systems and do not have access to the underlying hardware, but their interaction with it is controlled by either the parent partition or the hypervisor itself. //
7. The hypervisor is the component that directly manages the underlying hardware (processors and memory). //
8. It is logically defined by the following components:
   - ❖ **Hypercalls interface-**
     - This is the entry point for all the partitions for the execution of sensitive instructions.
     - This is an implementation of the paravirtualization approach
     - The parent partition also uses this interface to create child partitions
   - ❖ **Memory service routines (MSRs)-**
     - These are the set of functionalities that control the memory and its access from partitions.
   - ❖ **Advanced programmable interrupt controller (APIC).**

        - This component represents the interrupt controller, which manages the signals coming from the underlying hardware when some event occurs – timer expired, – I/O ready, – exceptions and – traps
    - ❖ **Scheduler-**
        - This component schedules the virtual processors to run on available physical processors.
        - The scheduling is controlled by policies that are set by the parent partition.
    - ❖ **Address manager-**
        - This component is used to manage the virtual network addresses that are allocated to each guest operating system
    - ❖ **Partition manager-**
        - This component is in charge of performing partition creation, finalization, destruction, enumeration, and configurations.

Popek and Goldberg Instructions:

- Privileged instructions
- Those that trap if the processor is in user mode and do not trap if it is in system mode (supervisor mode).
- Control sensitive instructions
- Those that attempt to change the configuration of resources in the system.
- Behavior sensitive instructions
- Those whose behavior or result depends on the configuration of resources (the content of the relocation register or the processor's mode).

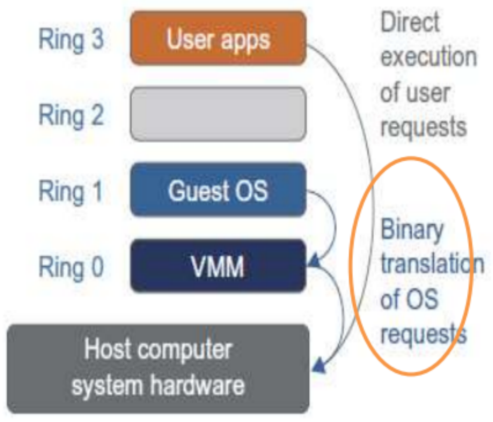
Full and para virtualization

1. With full virtualization, non-critical instructions run on the hardware directly while critical instructions are discovered and replaced with traps into the VMM to be emulated by software.//
2. Only critical instructions are trapped into the VMM because non-critical instructions do not control hardware or do not threaten the security of the system, but critical instructions do.//
3. Full virtualization does not need to modify guest OS, and critical instructions are emulated by software through the use of binary translation. //
4. However, this approach of binary translation slows down the performance a lot. Also binary translation can incur a large performance overhead.//
5. On the other hand, para virtualization needs to modify guest OS, and non-virtualizable instructions are replaced by hypercalls that communicate directly with the hypervisor or VMM. //

Full virtualization:
The operating system running in the virtual machine is unaware that it is running on virtualized hardware, and can interact with it as if it were running on physical hardware. Full virtualization is typically used for running legacy applications, testing software, or isolating applications from each other.

Para-virtualization:
Involves modifying the operating system of the virtual machine to make it aware that it is running on virtualized hardware. This allows the virtual machine to interact directly with the underlying hardware and can result in improved performance compared to full virtualization. However, para-virtualization requires more effort to set up and maintain because the operating system of the virtual machine needs to be modified.

Binary Translation of Guest OS Requests using VMM


Ring 3
User apps
Ring 2
Ring 1
Guest OS
Ring 0
VMM
Host computer system hardware
Direct execution of user requests
Binary translation of OS requests

1. VMware puts the VMM at Ring 0 and the guest OS at Ring 1.
2. The VMM scans the instruction stream and identifies the privileged, control- and behavior-sensitive instructions.
3. They are trapped into the VMM, which emulates the behavior of these instructions.
4. The method used in this emulation is called binary translation.
5. Thus, full virtualization combines binary translation and direct execution.//
6. The trap triggers the -
   translation of the offending instructions into
   an equivalent set of instructions that achieve the same goal without generating exceptions.

Advantage:
Guests can run unmodified in a virtualized environment.

Binary translation is only applied to a subset of the instruction set, while the others are managed through direct execution on the underlying hardware.

This reduces the impact on performance of binary translation

Disadvantage:
Translating instructions at runtime introduces additional overhead that is not present in other approaches like para-virtualization and hardware assisted virtualization.

Rather time-consuming, In particular for the full virtualization of I/O-intensive applications.

Increases the cost of memory usage.

Para-Virtualization with Compiler Support:

- Para-virtualization needs to modify the guest operating systems.//
- A para-virtualized VM provides special APIs requiring substantial OS modifications in user applications.//
- However, when the guest OS kernel is modified for virtualization, it can no longer run on the hardware directly.//
- Para-virtualization attempts to reduce the virtualization overhead, and thus improve performance by modifying only the guest OS kernel.
- The cost of maintaining paravirtualized OS is high.

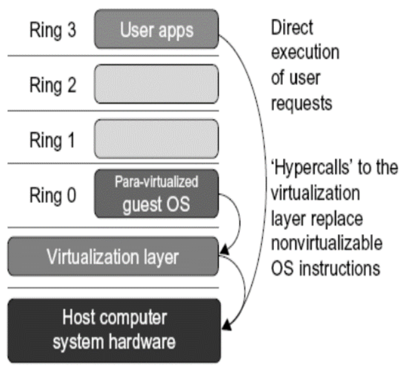

Ring 3
User apps
Ring 2
Ring 1
Ring 0
Para-virtualized guest OS
Virtualization layer
Host computer system hardware
Direct execution of user requests
'Hypercalls' to the virtualization layer replace nonvirtualizable OS instructions

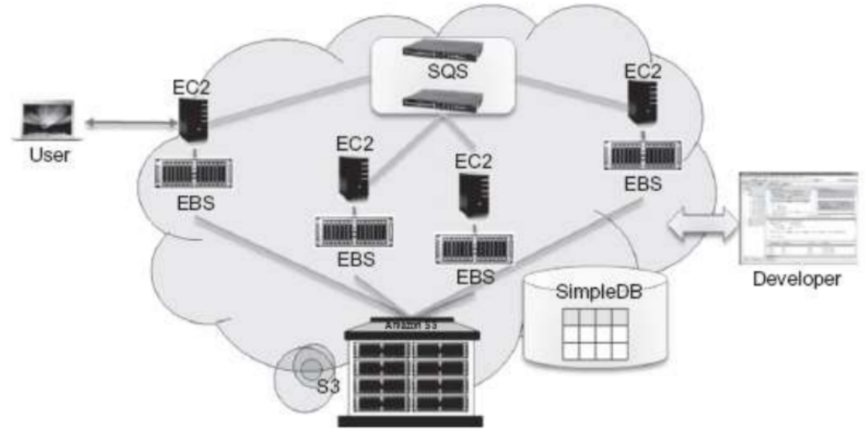
FIGURE 3.8

The Use of a para-virtualized guest OS assisted by an intelligent compiler to replace nonvirtualizable OS instructions by hypercalls.

Features	Full Virtualization	ParaVirtualization
Definition	It is the first generation of software solutions for server virtualization.	The interaction of the guest operating system with the hypervisor to improve performance and productivity is known as paravirtualization.
Security	It is less secure than paravirtualization.	It is more secure than full virtualization.
Performance	Its performance is slower than paravirtualization.	Its performance is higher than full virtualization.
Guest OS Modification	It supports all the Guest OS without any change.	The Guest OS has to be modified in paravirtualization.
Guest OS hypervisor independent	It enables the Guest OS to run independently.	It enables the Guest OS to interact with the hypervisor.
Efficient	It is less efficient than paravirtualization.	It is more simplified than full virtualization.
Examples	It is used in Microsoft, VMware, and Parallels systems.	It is mainly used in VMware and Xen systems.

Page 225 to end(mod 2) from ppt:

Amazon Web Architecture



1. EC2 (Elastic Compute Cloud) provides the virtualized platforms to the host VMs where the cloud application can run.
2. S3 (Simple Storage Service) provides the object-oriented storage service for users.
3. EBS (Elastic Block Service) provides the block storage interface which can be used to support traditional applications.
4. SQS stands for Simple Queue Service, and its job is to ensure a reliable message service between two processes. The message can be kept reliably even when the receiver processes are not running.
5. Users can access their objects through SOAP with either browsers or other client programs which support the SOAP standard.

Amazon Web Services

- ❖ **Compute Services**
    1. Compute services constitute the fundamental element of cloud computing systems.
    2. The fundamental service in this space is Amazon EC2, which delivers an IaaS solution.
    3. We can identify six major categories:
        - Standard instances - This class offers a set of configurations that are suitable for most applications.
        - Micro instances - This class is suitable for those applications that consume a limited amount of computing power and memory and occasionally need bursts in CPU cycles to process surges in the workload.
        - High-memory instances - This class targets applications that need to process huge workloads and require large amounts of memory.
        - High-CPU instances - This class targets compute-intensive applications.
        - Cluster Compute instances - This class is used to provide virtual cluster services.

        - Cluster GPU instances - This class is particularly suited for cluster applications that perform heavy graphic computations, such as rendering clusters.
4. EC2 instances are executed within a virtual environment, which provides them with the services they require to host applications.
5. The EC2 environment is in charge of
    - allocating addresses,
    - attaching storage volumes, and
    - configuring security in terms of access control and network connectivity

❖ **Storage Services**

AWS provides a collection of services for data storage

Amazon Simple Storage Service (S3)

The core service in this area is represented by Amazon Simple Storage Service (S3).

1. S3 is a distributed object store that allows users to store information in different formats.
2. S3 has been designed to provide a simple storage service that's accessible through a Representational State Transfer (REST) interface
3. The important points that allow the infrastructure to be highly efficient:
    - **The storage is organized in a two-level hierarchy - Buckets & Objects**
        1. Buckets represent virtual containers in which to store objects
        2. Objects represent the content that is actually stored.
        3. Buckets are top level elements of the S3 storage architecture and do not support nesting i.e., they cannot be further partitioned.
        4. This means that it is not possible to create directories or other kinds of physical groupings for objects stored in a bucket.
        5. Buckets, objects, and attached metadata are made accessible through a REST interface.
        6. Users create a bucket by sending a PUT request to http://s3.amazonaws.com/ with
            - the name of the bucket and,
            - if they want to specify the availability zone,
            - additional information about the preferred location.
        7. The content of a bucket can be listed by sending a GET request specifying the name of the bucket.
        8. Once created, the bucket cannot be renamed or relocated.
        9. If it is necessary to do so, the bucket needs to be deleted and recreated.

10. The deletion of a bucket is performed by a DELETE request, which can be successful if and only if the bucket is empty.
11. An object is identified by a name that needs to be unique within the bucket in which the content is stored.
12. Users create an object via a PUT request that specifies
    - the name of the object together
    - with the bucket name,
    - its contents, and
    - additional properties.
13. The maximum size of an object is 5 GB.
14. Once an object is created, it cannot be modified, renamed, or moved into another bucket
15. It is possible to retrieve an object via a GET request;
16. Deleting an object is performed via a DELETE request

- **Stored objects cannot be manipulated like standard files.**
    1. S3 has been designed to essentially provide storage for objects that will not change over time.
    2. Therefore, it does not allow renaming, modifying, or relocating an object.
    3. Once an object has been added to a bucket, its content and position is immutable, and the only way to change it is to remove the object from the store and add it again
- **Content is not immediately available to users**
    1. S3 uses replication to provide redundancy and efficiently serve objects across the globe; this practice introduces latencies when adding objects to the store— especially large ones—which are not available instantly across the entire globe
- **Requests will occasionally fail**
    1. Due to the large distributed infrastructure being managed, requests for object may occasionally fail

Elastic Block Storage

1. Elastic block storage is block storage for Amazon EC2 compute instances
2. It is just similar to hard disks attached to your computers or laptops, but the only difference is that it is used for virtualized instances.
3. With EBS, a volume can be mounted on an EC2 instance and it would appear just like a hard disk partition.

4. It can be formatted with any file system and files can be written or read by the EC2 instance just like it would to a hard drive
5. EBS has a standard limit of 20 volumes and each volume can hold data up to 1TB.
6. A limitation of EBS is its inability to be used by multiple instances at once. Once it is mounted by an instance, no other instance can use it

❖ **Communication Services**

1. Amazon provides facilities to structure and facilitate the communication among existing applications and services residing within the AWS infrastructure.
2. These facilities can be organized into two major categories: –

❖ **Virtual networking**

- It comprises a collection of services that allow AWS users to control the connectivity to and between compute and storage services.
- Amazon Virtual Private Cloud (VPC) and Amazon Direct Connect provide connectivity solutions in terms of infrastructure.
- it is possible to control connectivity between different services (EC2 instances and S3 buckets) by using the Identity Access Management (IAM) service

❖ **Messaging**.

The three different types of messaging services offered are

1) Amazon Simple Queue Service (SQS)

- SQS constitutes a disconnected model for exchanging messages between applications by means of message queues, hosted within the AWS infrastructure.
- Applications can send messages to any queue they have access to.
- These messages are securely and redundantly stored within the AWS infrastructure for a limited period of time, and they can be accessed by other (authorized) applications.
- While a message is being read, it is kept locked to avoid spurious processing from other applications. Such a lock will expire after a given period

2) Amazon Simple Notification Service (SNS)

- Amazon SNS allows applications to be notified when new content of interest is available.
- This feature is accessible through a Web service whereby AWS users can create a topic, which other applications can subscribe to.
- At any time, applications can publish content on a given topic and subscribers can be automatically notified.

3) Amazon Simple Email Service (SES)

        - Amazon SES provides AWS users with a scalable email service that leverages the AWS infrastructure.
        - Once users are signed up for the service, they have to provide an email that SES will use to send emails on their behalf.

- ❖ **Additional Services**
    1. Besides compute, storage, and communication services, AWS provides a collection of services that allow users to utilize services in aggregation.
    2. The two relevant services are
        - Amazon CloudWatch
        - Amazon Flexible Payment Service (FPS).