ASSIGNMENT - 2

01. Explain the different Levels at which virtualization can be implemented.

→ • Instruction Set Architecture Level :

- Defines the way in which a microprocessor is programmed at the machine level.
- Is performed by emulating a given ISA by the ISA of the host machine.
- Basic emulation method is
    - Code interpretation
    - Dynamic binary translation

• Hardware Abstraction level :

- Generates a virtual hardware envioronment for a VM.
- The idea is to virtualized the computer's resources, such as its processors, memory and I/O devices.

• Operating System Level:

- Commonly used in creating virtual hosting envioronments to allocate hardware resources among a large no of mutually distributing users.

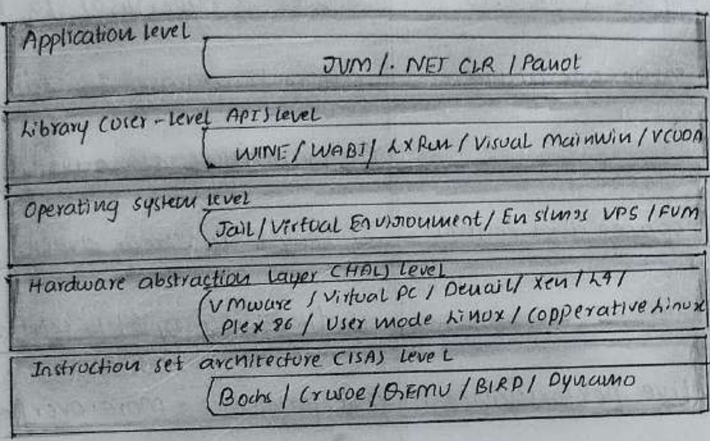
• Library Support level :

- Interfaces is possible by controlling the communication link between application and rest of the system through API hooks.

· User - Application Level:

- Also known as Process level virtualization

- The Microsoft .NET CLR and JVM are two good examples



Q2. What is a Virtual Machine Monitor (VMM)? List its key design requirements.

→ · A Virtualization system that partitions a single physical machine into multi-ple virtual machines.

· In order to create and deploy virtual machines and services to physical servers, vmm is a solution for virtualized environments.

· The host software that provides virtualization is often referred to as a virtual machine monitor (vmm) or hypervisor.

<u>Design requirements:</u>

There are three requirements for a vmm

→ First, a vmm should provide an environment for programs which is essenti-ally identical to the orginal machine.

→ Second, programs run in this environment should show, at worst, only minor decreases in speed.

→ Third, a VMM should be in complete control of the system resources.

Q3. How does OS-level virtualization differ from hardware-level virtualization?

→

OS-Level Virtualization	Hardware-Level virtualization
1) Multiple isolated user-space instances (containers) share the same operating system kernel.	1) Uses a hypervisor to emulate entire hardware for each VM; each VM runs its own full OS.
2) Isolation: Uses namespaces and control groups (cgroups) for isolation.	2) Isolation: Very strong - each VM is like a completely separate computer.
3) Performance: Near-native performance because there's no need to emulate hard-ware.	3) Performance: More overhead than containers because of full OS per VM and hardware emulation.
4) Use Case: Running microservices, lightweight app deployment, DevOps.	4) Use case: Running different OSes on the same machine, legacy software support
5) Limitations: All containers must use the same OS kernel. (e.g., can't run Windows containers on Linux host without additional layers).	5) Use-Examples: VMware, Virtual-Box, KVM, Hyper-V.
6) Example: Docker, LXC, Podman.	

Q4. Explain the architecture of Xen with a neat diagram.

→ Xen is an open-source hypervisor developed by Cambridge University. Xen is an open-source micro-kernel hypervisor that separates policy from mechanism, that means it separates the OS from hardware, enabling virtualization efficiently.

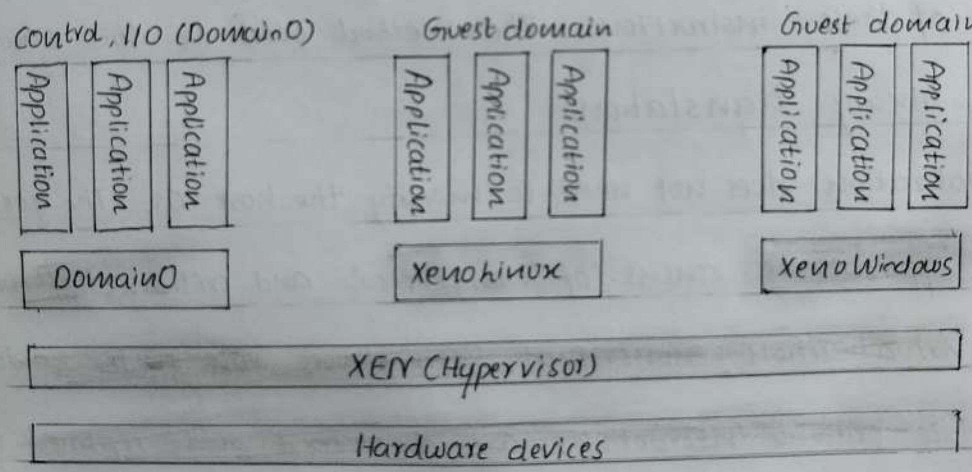
The Xen hypervisor implements all the mechanisms, leaving the policy to be handled by Domain 0, as shown in fig. It does not include native device drivers. It just provides a mechanism by which a guest OS can have direct access to the physical devices.

The Xen system consists of three core components:

① Hypervisor - Manages virtual machines.

② Kernel - Provides OS-level functions.

③ Applications - Run on top of the hypervisor.



Xen follows a domain-based architecture:

Domain 0 (Dom0): Domain 0 is a privileged guest OS of Xen. It is first loaded when Xen boots without any file system drivers being available.

Guest Domains (DomU): Regular VMs running on Xen without direct hard-ware access.

Security risks: If Domain 0 is compromised, an attacker can control all guest domains.

VM Management : Allows dynamic changes, including rollback, cloning, and state preservation, enabling fault tolerance and live system updates.

Tree-Like execution : multiple VM instances can branch into different execut-ion states, making Xen useful for distributed and dynamic computing environments.

05. What is binary translation ? How is it used in full virtualization ?

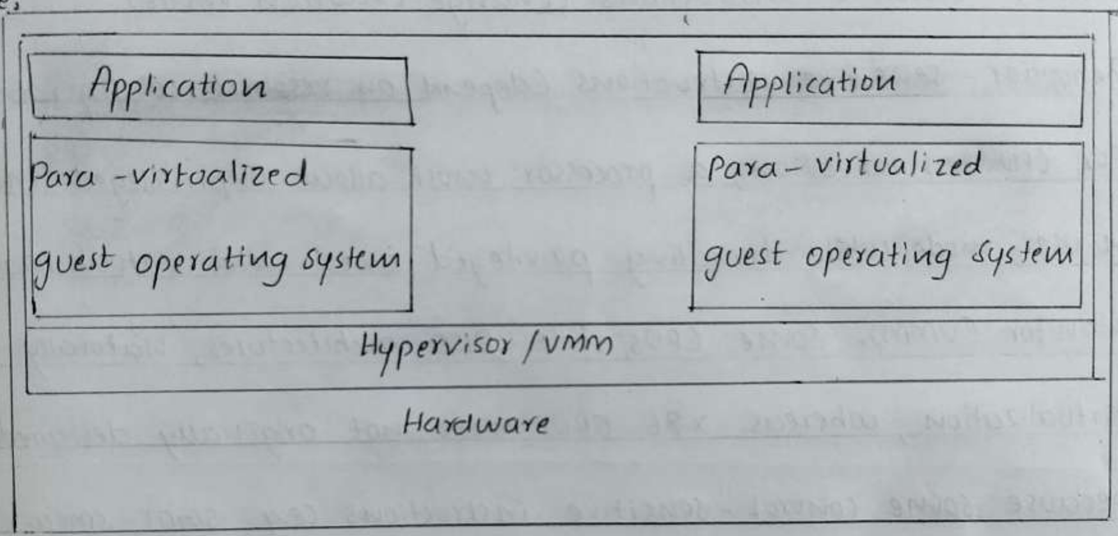
→ The VMM scans the instruction steam and identifies the privileged, control - and behavior - sensitive instructions. When these instructions are identified, they are trapped into the VMM, which emulates the behaviour of these instructions. The method used in this emulation is called binary translation.

Full virtualization does not need to modify the host OS. The guest OSes and their applications consist of noncritical and critical instructions. With full virtualization, noncritical instructions run on the hardware directly while critical instructions are discovered and replaced with binary translation to trap. Non critical instructions do not control hardware or threaten the security of the system, but critical instructi-ons do. Therefore, running noncritical instructions on hardware not only can promote efficiency, but also can ensure system security.

Q6. Explain para-virtualization. How does it differ from full virtualization.

→ Para-virtualization needs to modify the guest operating systems. A para-virtualized VM provides special APIs requiring substantial OS modifications in user applications.

Performance degradation is a critial issue of a virtualized system. No one wants to use a VM if it is much slower than using a physical machine.



Diff b/w para & full virtualization:

para virtualization	full virtualization
* Para-virtualization requires OS modification	* full virtualization does not.
* Para-virtualization has better performance due to lower overhead	* Low performance compare to para virtualization.
* Para-virtualization may not need hardware support	* Full virtualization often needs hardware support.
* Does not support OSes.	* Full virtualization supports unmodified OSes.

07. Explain how CPU virtualization is achieved in modern systems.

→ A Virtual Machine (VM) is a duplicate of a physical system where most unprivileged VM instructions run directly on the host CPU for efficiency. However, critical instructions must be handled carefully for correctne -ss and stability. These are categorized as:

- Privileged instructions (executed supervisor mode)
- Control - sensitive instructions (change execution mode)
- Behavior - sensitive instructions (depend on resource configuration)

For CPU virtualization, a processor must allow unprivileged instructions in user mode while handling privileged ones in a Virtual Machine Monitor (VMM). Some CPUs, like RISC architectures, naturally support virtualization, whereas x86 CPUs were not originally designed for it because some control - sensitive instructions (e.g., SGOT, SMSW) are not privileged and thus cannot be trapped by the VMM.

Hardware - Assisted CPU Virtualization:

Intel and AMD introduced hardware -assited virtualization to simplify VM execution. This technique adds CPU instructions that allow the hypervisor to handle privileged and sensitive operations automatically. It enables the OS to run in a VM without modification, improving efficiency.

Ex:- Intel Hardware - Assisted CPU virtualization.

08. What is memory virtualization? How does it enhance system performance?

→ In a traditional execution environment, the operating system maintains mappings of virtual memory to machine memory using page tables, which is a one-stage mapping from virtual memory to machine memory. All modern x 86 CPUs include a memory management unit (MMU) and a translation look aside buffer (TLB) to optimize virtual memory performance.

However, in a virtual execution environment, virtual memory virtualization involves sharing the physical system memory in RAM and dynamically allocating it to the physical memory of the VMs. That means a two-stage mapping process should be maintained by the guest OS and the VMM, respectively: virtual memory to physical memory and physical memory to machine memory.

Memory virtualization allows VMs to efficiently use system memory by implementing a two-stage memory mapping:

① Virtual memory -> Physical memory (Guest OS)

② Physical memory -> Machine memory (VMM / Hypervisor).

09. List the benefits of I/O virtualization in cloud environments.

→ 1. Improved Resource Utilization - Enables sharing of I/O devices among multiple VMs, reducing hardware waste.

2. Enhanced Scalability - Easily scales I/O resources as workloads increase.

3. Better Performance - Reduces I/O bottlenecks and latency through

Optimized data paths.

4. Cost Efficiency - Lowers hardware and maintance costs by reducing the number of physical I/O devices needed.

5. Simplified Management - Centralized control and management of I/O resources.

6. Increased Flexibility - Allows dynamic allocation and reallocation of I/O resources based on demand.

7. High Availability - Supports failover and redundancy, improving system reliability.

8. Faster Provisioning - Speeds up deployment of new virtual machines and services.

9. Reduced Downtime - Enables live migration of VMs with minimal disruption to I/O processes.

10. Isolation and Security - Ensures secure and isolated I/O channels for different VMs.

10. Explain the architecture of a virtual cluster. 11.

→ Architecture of a Virtual cluster typically includes the following components and structure:

1. Virtual Machines (VMs):

These are the individual computing nodes of the cluster. Each vm acts like a physical node and runs its own operating system and applications.

2. Hypervisor:

The hypervisor (like VMware, KVM, or Hyper-V) manages multiple VMs on a single physical server.

3. Physical Host Machines:

These are the actual servers that host the VMs.

4. Virtual Network:

A software-defined network connects the VMs within the cluster, enabli-ng communication among them as if they were on a physical network.

5. Shared Storage System: All VMs typically access a centralized storage system (like NAS or SAN) for consistent data access and high availabili-ty.

6. Cluster Management Software:

Tools like OpenStack, VMware vSphere, or Kubernetes manage the Lifecycle of the cluster - provisioning, scaling, monitoring, and orchestrating workloads.

11. What is live VM migration? List and explain its steps.

→ In clusters with both host and guest nodes, live VM migration enables failure recovery by moving VMs to different nodes. If a VM fails, anoth-er VM with the same OS can take over. This offers flexibility and reduces the need for physical failover.

There are four main ways to manage a virtual cluster:

1. Guest-based manager: Cluster management runs inside the guest system (e.g., OpenMosix, Solaris zones).
2. Host-based manger: Cluster manager runs on the host system and manages VMs directly (e.g., VMware HA).
3. Independent cluster manager: Runs both on host and guest but without interaction; more complex.
4. Integrated cluster manager: Shares information between host and guest for enhanced resource usage and performance.

The goal is to design a migration scheme that:

* Minimizes service disruption.
* Maintains reasonable migration time.
* Uses the least possible network bandwidth.