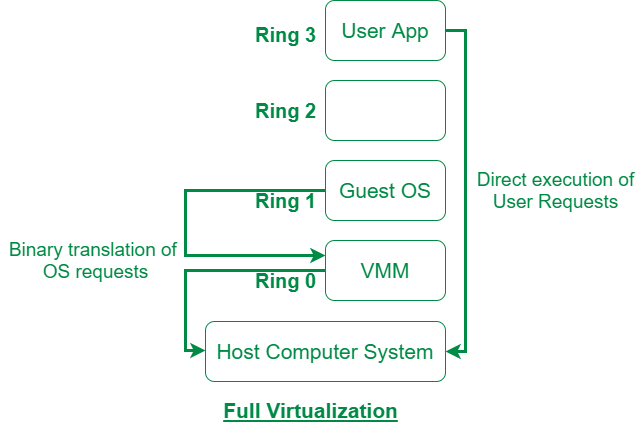
**Full virtualization :**

* Full Virtualization was introduced by IBM in 1966. It is the first software solution for server virtualization and uses binary translation and direct approach techniques.
* VMware's full virtualization enables guest operating systems to run without modification by replicating primary hardware.
* It uses type-II hypervisors for desktops and type-I hypervisors for servers.
* It is achieved by abstracting the underlying hardware completely, so the guest OS operates unaware of the virtualization environment.

**Diagram :**



**Working of Full Virtualization :**

**Hypervisor Setup:**

* The hypervisor installs directly on the physical hardware (in Type-I hypervisor) or on top of the host OS (in Type-II hypervisor).
* It creates virtual machines (VMs) that mimics the real hardware (CPU, memory, disk, network, etc.) for each guest OS.

**Guest OS Installation:**

* The guest OS is installed inside the VM just like it would be on a physical machine. ( The OS thinks it’s running on a physical machine.)
* It is unaware that it's not running directly on real hardware.

**Execution of Tasks:**

When the guest OS needs to execute tasks (e.g., accessing the CPU, memory, or disk), it sends the requests as if it were interacting with the physical hardware.

**Hypervisor Handling**:

* Directly executes non-sensitive instructions (simple operations).
* Simulates hardware for more complex, sensitive instructions (like accessing hardware directly).
* It ensures that each VM remains isolated from others, and each gets the resources it needs.

**Key Characteristics:**

1. **Hardware Abstraction**: The hypervisor replicates the hardware environment, including CPU, memory, and I/O devices.
2. **Guest OS Independence**: No modification is required in the guest OS to run on the virtualized hardware.
3. **Execution Techniques**:
   * **Direct Execution**: Non-sensitive instructions are executed directly on the CPU for high performance.
   * **Binary Translation**: Sensitive instructions are dynamically translated and executed safely to maintain isolation and control.
4. **Hypervisor Types**:
   * **Type-I Hypervisor**: Runs directly on hardware (e.g., VMware ESXi, Microsoft Hyper-V).
   * **Type-II Hypervisor**: Runs on a host operating system (e.g., VMware Workstation, Oracle VirtualBox).
5. **Performance Overhead**: Slightly higher overhead due to binary translation, but modern hardware-assisted virtualization (e.g., Intel VT-x, AMD-V) reduces this significantly.

**Advantages:**

* Works with any operating system.
* Provides strong isolation between virtual machines.

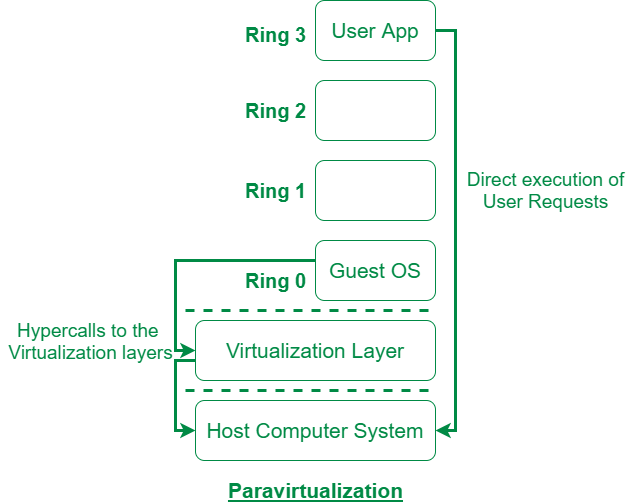
**Disadvantages:**

* Can be slower due to emulation.
* Requires more resources (CPU, memory).

**Paravirtualization:**

* Paravirtualization is a virtualization technique that involves modifying the guest operating system (OS) to make it "aware" of the virtualization environment.
* In paravirtualization, guest OS is not completely isolated but it is partially isolated by the virtual machine from the virtualization layer and hardware. [VMware](https://www.geeksforgeeks.org/difference-between-vmware-and-virtualbox/) and Xen are some examples of paravirtualization.
* This approach eliminates the need for full hardware emulation, enabling the guest OS to communicate directly with the hypervisor for optimized performance.

**Diagram :**



**How Paravirtualization Works**

**Hypervisor Setup**:

* The hypervisor creates a virtual environment for the guest OS.
* It provides an **API** (a set of rules or commands) that the guest OS uses to request resources like CPU, memory, and I/O.

**Guest OS Modification**:

The guest OS is modified to include support for **hypercalls** instead of making direct hardware access attempts.

**Execution of Tasks**:

* When the guest OS needs to perform privileged operations (e.g., accessing memory or managing hardware), it doesn’t directly execute instructions.
* Instead, it sends a **hypercalls** request to the hypervisor.

**Hypervisor Response**:

The hypervisor handles the request, performs the operation (like accessing hardware), and sends the result back to the guest OS.

**Key Characteristics of Paravirtualization :**

1. **Guest OS Needs to be Changed :** The guest OS has to be modified to know it's running on a virtual machine and use special commands to talk to the hypervisor (the virtualization layer).
2. **Talks Directly to Hypervisor :** Instead of accessing hardware directly, the guest OS calls the hypervisor for tasks like managing memory or input/output (I/O).
3. **Better Performance :** Since the guest OS doesn’t need to simulate hardware, it can work faster and more efficiently, especially when dealing with tasks like reading or writing data.
4. **No Full Hardware Emulation :** Paravirtualization doesn’t try to recreate every piece of hardware. Instead, it gives the guest OS virtual versions of hardware, making everything run more smoothly.
5. **Works Best for Performance :** It’s designed to run faster than full virtualization, so it’s good for situations like in cloud computing or large data centers.

 **Advantages:**

* Faster performance.
* Uses resources more efficiently.

 **Disadvantages:**

* Needs modified operating systems.
* Not compatible with all operating systems.