Introduction to
Cloud COMPUTING

Giuliano Taffoni - I.N.A.F.

"Virtualization"

2024 @ Università di Trieste

Outline

Intro to Virtualization
and
Virtual Machines

Virtual Machine
Tutorial

What is virtualization

"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). 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."
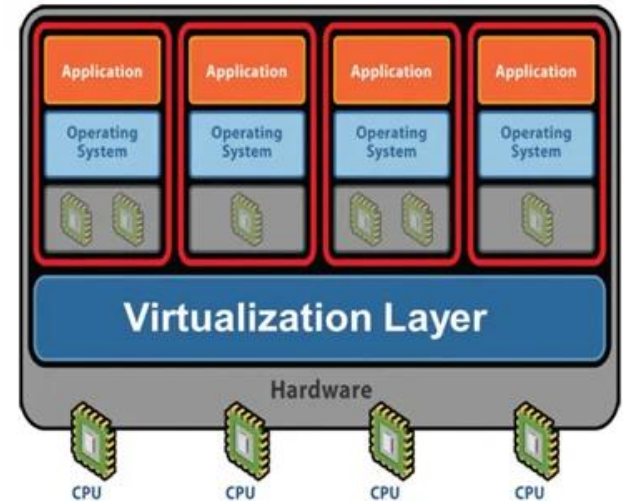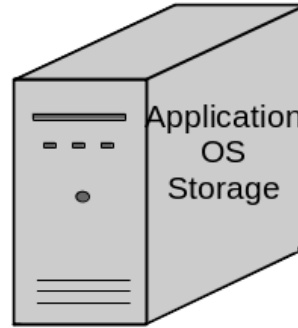
Traditional service delivery

Hardware Platform

Hardware Platform
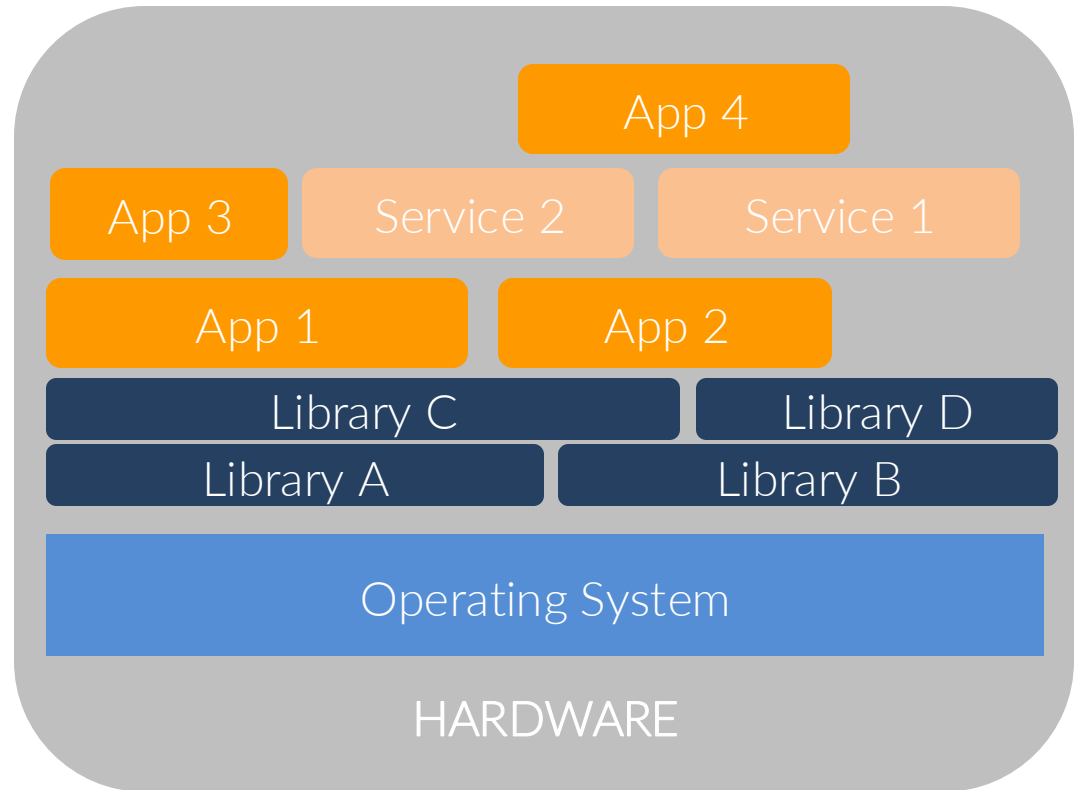
Hardware Platform

Hardware Platform

Application
OS
Storage

Application
OS
Storage

Application
OS
Storage

Application
OS
Storage

Web server
Windows
IIS

App server
Linux
Glassfish

E-mail
Windows
Exchange

Database
Linux
MariaDB

Traditional service delivery

Dependecy Hell

App 4

App 3 | Service 2 | Service 1

App 1 | App 2

Library C | Library D

Library A | Library B

Operating System

HARDWARE

Service delivery problem

Multiple servers are deployed to serve different operational use cases

Increase the operational costs

Server capacity is not fully exploited

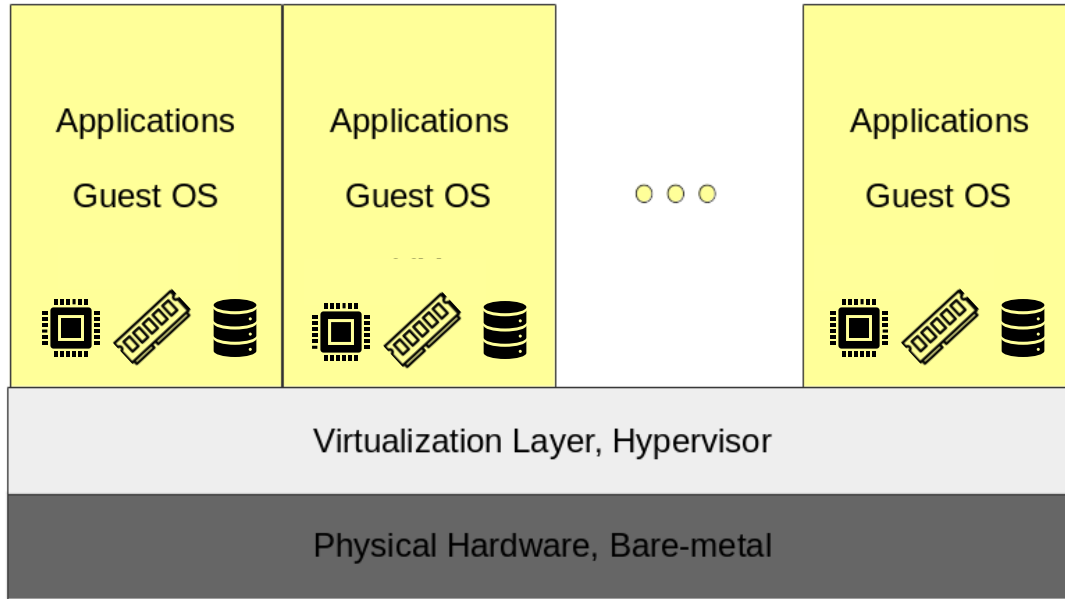Inefficiency and too high operating costs

Virtualization

Virtualization, in computing, is the ability to "simulate" a hardware platform, such as a server, storage device or network resource, in software. All the functionality is separated (**abstracted**) from the hardware and "simulated" as a "virtual instance" with the ability to operate just like the hardware solution. A single hardware platform can be used to support multiple virtual devices or machines, which are easy to spin up or down as needed.

Virtual Machine is the software simulation of a computer. It can run an Operating Systems and applications interacting with the virtualized abstracted resources, **not with the physical resources**, of the actual host computer.

Hypervisor (or Virtual Machine Monitor) is a software tool installed on the physical host system to provide the thin software layer of abstraction that decouples the OS from the physical bare-metal. It allows to split a computer in different separate environment, the Virtual Machines, distributing them the computer resources

|Virtualization componentes

Applications

Guest OS

Applications

Guest OS

○ ○ ○

Applications

Guest OS

Virtualization Layer, Hypervisor

Physical Hardware, Bare-metal

Host machine is the actual machine on which the virtualization takes place

Guest machine is the virtual machine.

Hypervisor or VM Manager is the software or firmware that creates a virtualization layer on the host hardware

| Virtual Machines

Virtual Machine is a virtual **computing system**

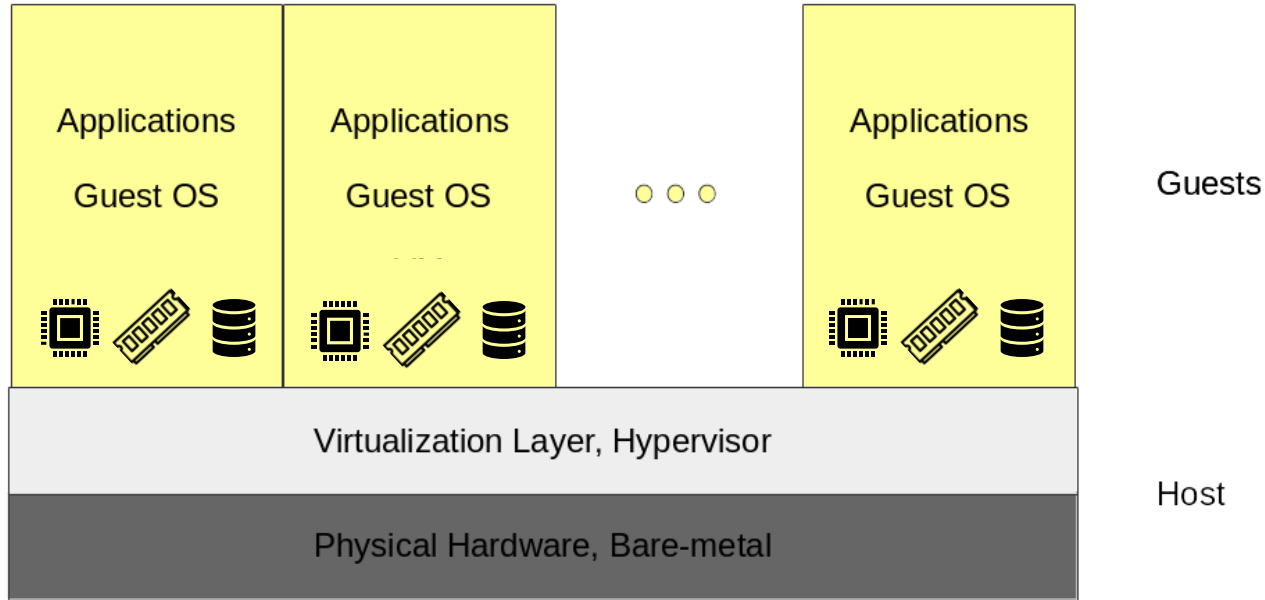It has tightly **isolated** software with an operating system and applications inside

Each Virtual Machine in a host is **independent**

In a single physical server can be put multiple VMs enabling the run of multiple OSes and Applications (**partition/multi-tenancy**)

Native HW performance (as much as possible...)
Can we measure how much is the degrade in performances?
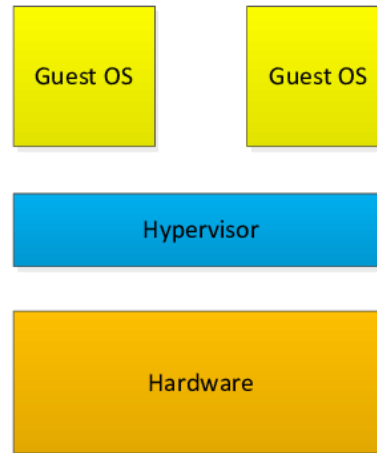
| VMs and Virtual Appliance

Applications

Guest OS

Applications

Guest OS

○ ○ ○

Applications

Guest OS

Guests

Virtualization Layer, Hypervisor

Physical Hardware, Bare-metal

Host

An application and its operating system packaged together for a virtualized environment

Virtual Machines Features

- **Consolidation** - Run different OSes on the same physical host Partition physical resources between VMs

- **Isolation** - Fault and security isolation at the hardware level Preserve performance with advanced resource control

- **Encapsulation** - Save the VM state to files VMs can be moved and copied moving and copying files (**migration**)

- **Hardware Independence -** VMs can be copied, moved or migrated to different physical servers

- **Other** – Debugging, profiling, reproducibility, software preservation

Virtualization models

1. Bare-metal Hypervisor or Native Hypervisors (Type 1)
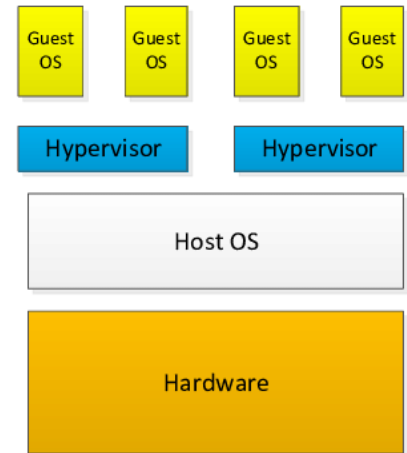
2. Hosted Hypervisor (Type 2)

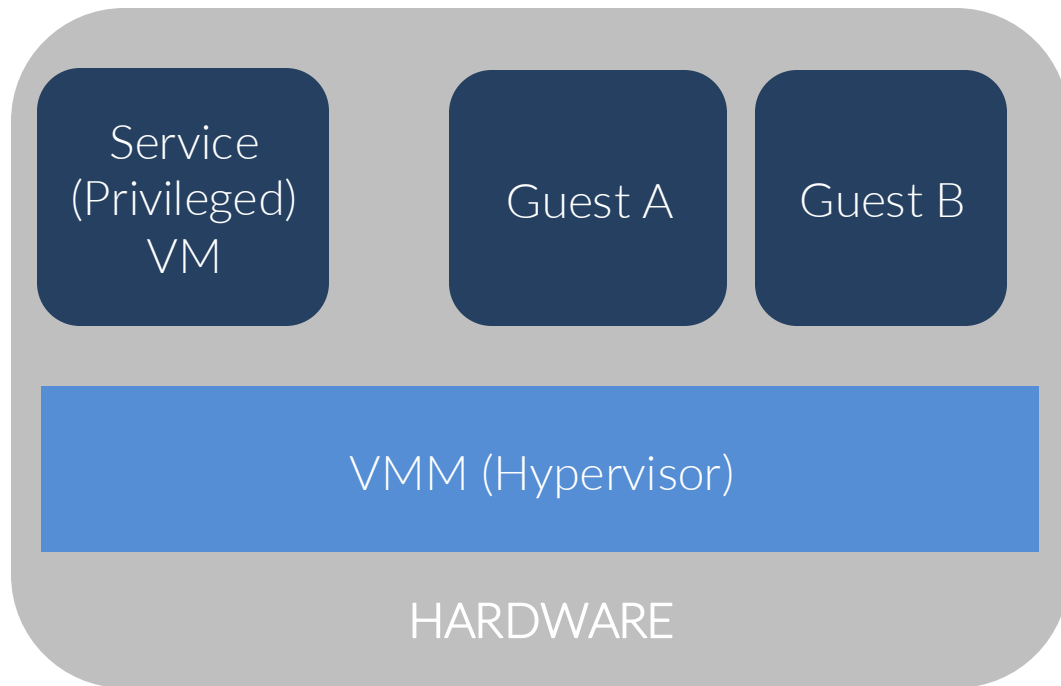

Type 1 (bare-metal)   **Type 2 (hosted)**

Type 1 Hypervisor

VMM manages all hardware resources and supports the execution of VMs.

Service (privileged) VM
Based on standard OS and has full privileged to access HW resources.

Service (Privileged) VM	Guest A	Guest B

VMM (Hypervisor)

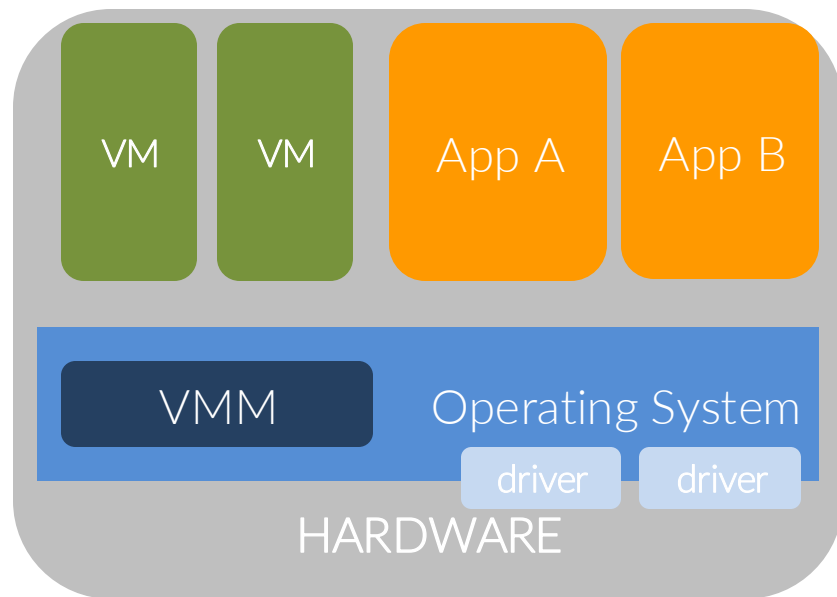HARDWARE

Type 1 Hypervisor examples

This model is adapted by the Xen virtualization solution (open source or Citrix Xen Server) and by the VMware hypervisor (the ESX hypervisor).

1. **Xen (Open source or Citrix Xen Server)** - The VMs that are run in the virtualized environment are referred to as domains. The privileged domain is called dom0, and the guest VMs are referred to as domU. Xen is the actual hypervisor, and all the drivers are running in the privileged domain, in dom0.

2. **ESX (VMware)** API based VM-VMM interaction. VMM enforce directly drivers for different HWs. Originally there was a linux monitor admin VM now integrated with APIs.

Hosted Virtualization model

Hosted hypervisors are designed to run within a traditional operating system. A hosted hypervisor owns all HW. VMM module provides HW interfaces to VMs and manage VM context switch.

A well-known example of a hosted hypervisor is Oracle VM VirtualBox. Others include KVM, QEMU and Parallels.

Type 2 Hypervisor Example

Kernel Based VM (KVM) – Linux based approach to Type 2

KVM	QEMU
Kernel module	Hypervisor
Low-level interface with processors using virtualization primitives	Hardware virtualization (No Emulation)
Resource allocation	Virtual Hard Drive
Kernel extention Libvirt (APIs+monitoring)	
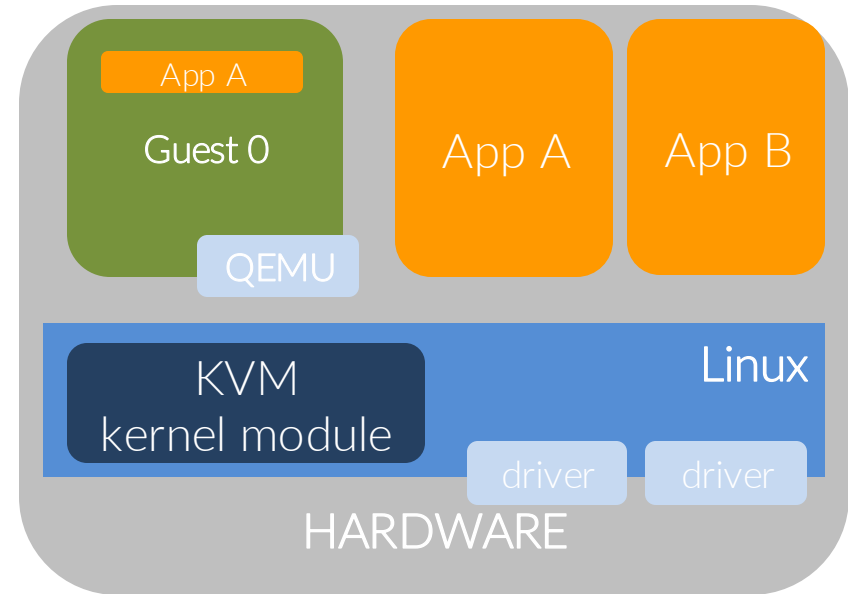
Virtualization Types

Full Virtualization: the hypervisor provides complete hardware abstraction creating simulated hardware devices. The guest OS don't know (or care) about the presence of a hypervisor and issue commands to what it thinks is actual hardware.

Paravirtualization: para means partial. The guest OS is aware that it is a guest, it recognizes the presence of a hypervisor, and it has drivers to issue some commands, mainly I/O operations, directly to the host OS, more efficiently than inside a virtual environment. The guest OS must be modified

Hardware assisted virtualization: is a type of full virtualization where the microprocessor architecture has special instructions to aid the virtualization of the hardware. These hardware extensions help the hypervisor tackle complex tasks at the processor level rather than through software emulation

What do we need to virtualize?
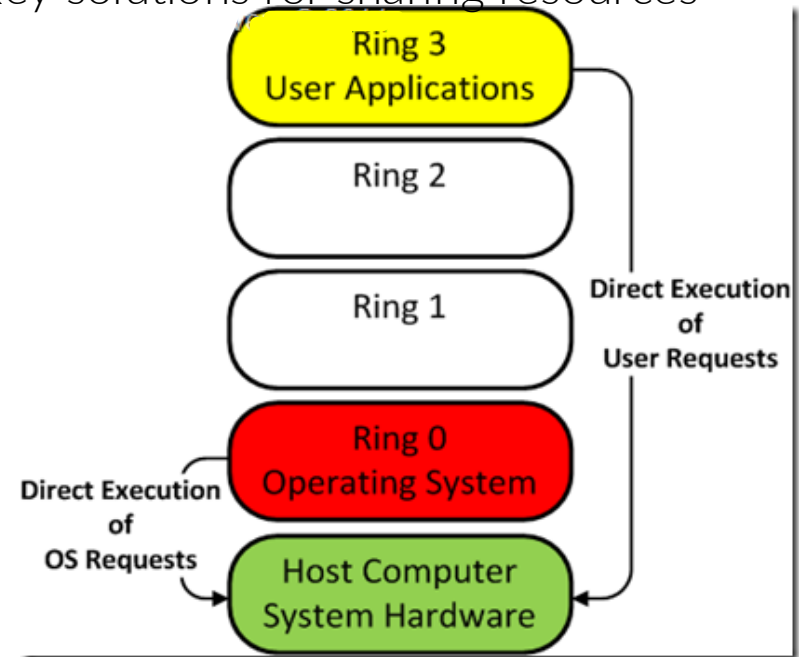
| Hardware Protection levels

Operating systems manage computer resources, like processing time on the CPU and accessing the memory. Since computers run more than one software process, this will bring some issues. Protection rings are one of the key solutions for sharing resources and hardware.

Commodity hardware has more than two protection levels. (×86 architecture has four protection levels called rings).

Ring 3: has the least level of privilege, so this is where the applications would reside.
.
.
.
Ring 0 has the highest privilege and can access all the resources and execute all hardware-supported instructions (OS level).
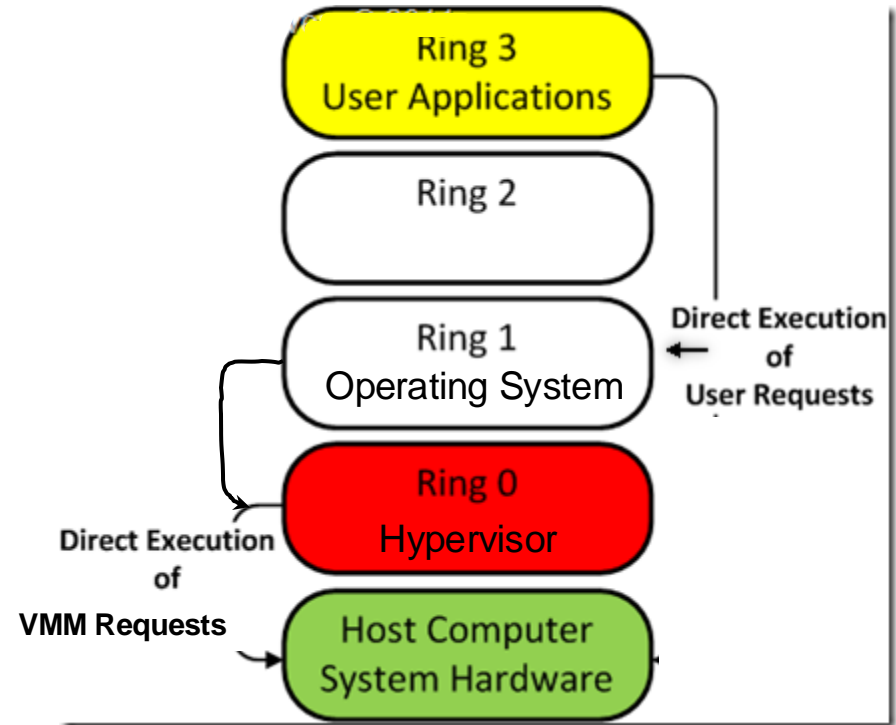
Hardware Protection levels VM

VM protection level on ×86 architecture.

Ring 3: has the least level of privilege, so this is where the applications would reside.
.
.
.
Ring 1: hosts the OS of VMs and a trap is caused to VMM
Ring 0 has the highest privilege and hosts the VMM that can access all the resources and execute all hardware-supported instructions.

Processor Virtualization

Guest instructions are executed directly by the hardware as much as possible: virtual machine monitor does not interfere with every single instruction that is issued by the guest operating system, or its applications. The non-privileged instructions will operate at hardware speeds.

Privileged instructions: Whenever a privileged instruction gets accessed, then the processor causes a trap, and control is automatically switched to the most privileged level, that is the hypervisor. At this point, the hypervisor can determine whether the operation is to be an allowed or not. Illegal operations will cause actions on the VM (as KILL), legal operations the hypervisor should perform the necessary emulation so that the guest operating system is under the impression that it does have control over the hardware.
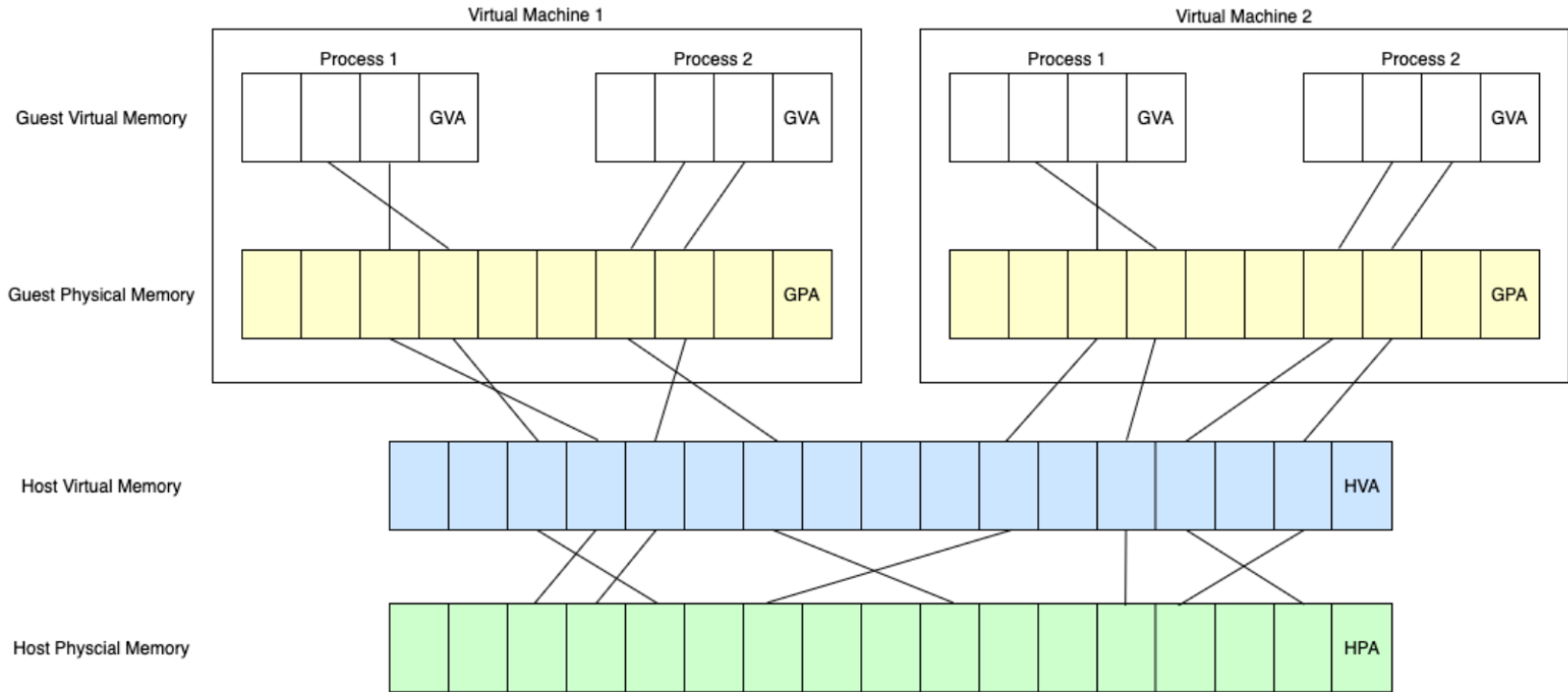
Memory Virtualization

Full virtualization: the guest operating system continues to observe a contiguous linear physical address space that starts from physical address 0.

Three types of addresses:
- **Virtual addresses**, these are the ones that are used by the applications in the guest.
- **Physical addresses**, these are the ones that the guest thinks are the addresses of the physical resource
- **The machine addresses**, these are the actual machine addresses with the actual physical addresses on the underlying platform.

Similar distinction of virtual verses physical verses machine will also apply to the page numbers and the page frame numbers.

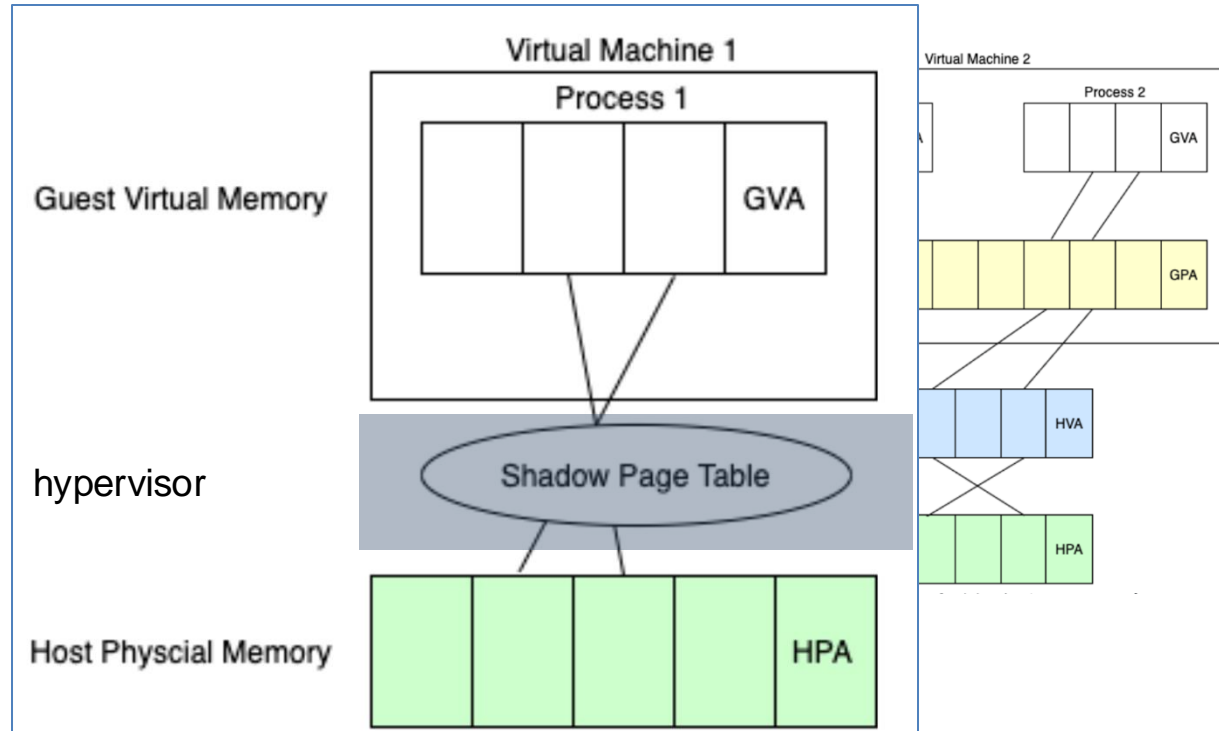# Memory Virtualization

expensive

SW

HW
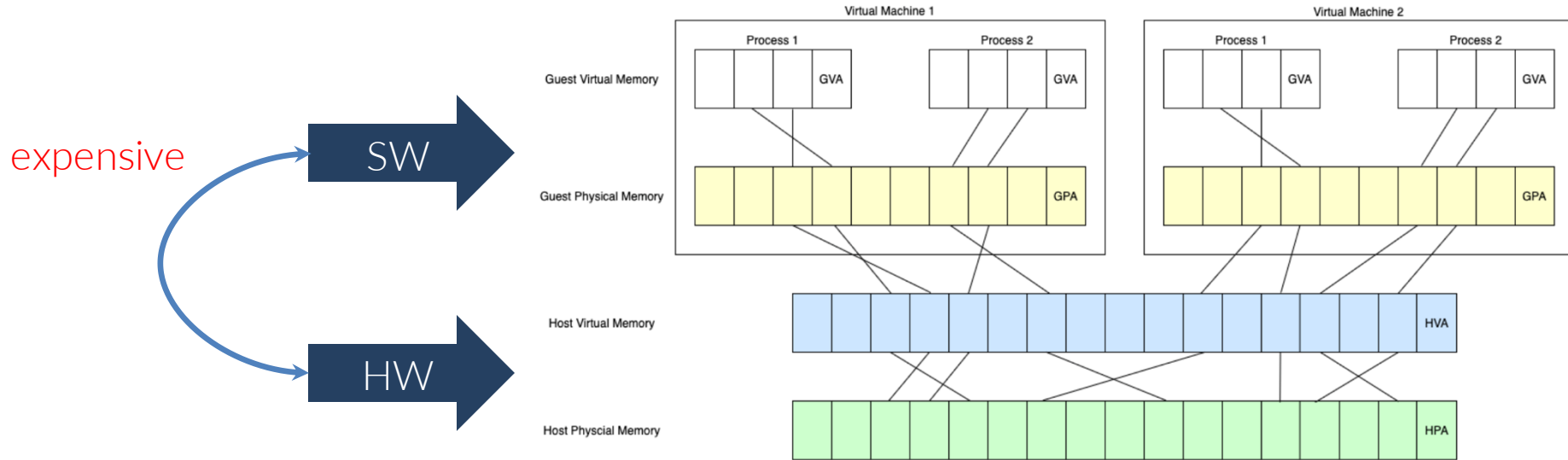
Every single memory access goes through two separate translation, the first one which will be done in software, and then the second one potentially can take advantage of hardware.

Shadow Page Table establishes a shortcut to directly manage the mapping from GVA to HPA.

It works the same way virtual address is mapped to physical address

VMM must maintain consistence between the page tables.

Memory paravirtualization

- Guests are aware of the virtualization
- No longer strict requirement on continuous physical memory starting at 0
- Guests are modified to explicitely registers page tables with VMM
- Operational costs are amortized by introducing some optimizations:
  - Batch page updates (bulk updates)
  - Cooperative inter VMs memory managment

Hardware virtualization on modern x86 platforms is optimizing the memory maintanance

Device Virtualization

- Large number of different devices

- No standardization of interfaces and of the semantic of the interfaces

- Different models for device virtualization

- VT* hardware extentions of the CPU architecture make some aspects of device virtualization simpler

Device virtualization models

Passthrough	Hypervisor direct mode	Split device driver model
VMM configure the access permission to a device so that the GuestVM has **direct access** to the device (VMM-bypass)	VMM intercept all the device access requests and **emulate** device operations.	Device operations are split between front end driver in guest VM and backend driver in **a service VM** (dom0) that hosts the actual regular driver for the device. (only on paravirtualized guests)
🟢 VM provided with exclusive access to the device; 🟢 VM can directly access the device bypassing the hypervisor;	🟢 VM decoupled with physical device; 🟢 VMM operates transaltiona and emulation; 🟢 Sharing and migration;	🟢 Eliminate emulation overhead 🟢 Backend reduce the VMM exposure to bugs and improve security
⊖ Device sharing (overhad or not duable); ⊖ VMs and HW must have the **exact type of device;** ⊖ Migration: no more decoupling with HW	⊖ Latency in device access due to emulation ⊖ VMM is extremely complex has it must support all devices ⊖ VMM exposed to bugs in drivers and drivers	⊖ Only for paravirtualization guests ⊖ Requires a modification in GuestVM front end device

| HW virtualization support

Hardware specifications to Hypervisors that reduced the overhead of VMM operations and greatly improve the speed and abilities of the VMM.

E.g. Intel VT tecnologies, AMD-V

- Improve x86 Instruction Set Architecture
- New modes: root/non-root
- VM control structure: describe the state of the virtual processor (vCPU)
    - VMM can interpret that structure and reduce the virtualization overheads
- Optimize memory:
    - extended page tables (HW mapping)
    - tagging the translation lookaside buffer with VM IDs (content switch efficency)
- Multiqueue devices (devices with multiple logical interfaces)
- Security (protect VM from one another)

HW Emulation Support

Creating an environment that imitates the properties of one system onto another.
mimics the qualities and logic of one processor to run in another platform.
Run an OS or software in any other system. Guest Operators need a translation.

Emulation brings higher overhead but has its perks too. It is highly inexpensive, easy to access, and helps us run the programs that have become obsolete in the available system.

An emulator converts the needed architecture CPU instructions and successfully runs it on another architecture.

Anyone can access the emulation platforms remotely and is easier to use. It is an excellent ability to have for embedded/OS development, without affecting the underlying OS.

Emulation can generally handle the size of the design under test (DUT), without considering the host's capabilities.

Emulation vs Virtualization

VIRTUALIZATION	EMULATION
In virtualization, hardware can be accessed directly.	In case of Emulation, you'd need a software connector to access hardware.
Virtual machine can run the code directly, which is available in different languages.	Emulator requires an interpreter to translate the source code.
Virtual machines are relatively faster in its operations.	Emulators are relatively slower.
VM solutions are costlier than Emulation.	Emulation is comparatively cheaper.
Virtualization provides better backup solutions.	Emulation falls short of virtualization as far as backup and recovery is considered.

Network virtualization

Abstraction of the physical network
- Support of multiple logical networks running on a common physical substrate

- Container of network services (switches, routers, ports, etc.)

Applications run on a virtual network as they where running on a physical network.
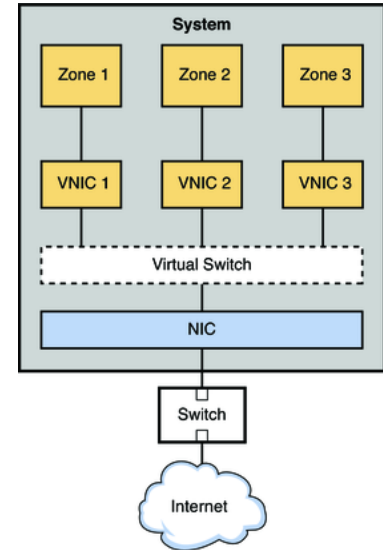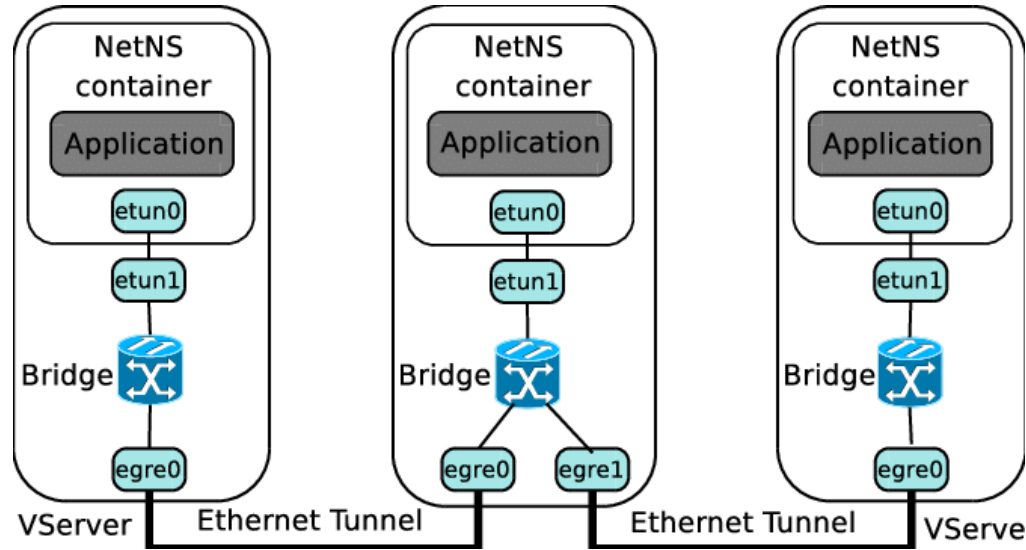
Network virtualization

✓ **Flexibility**: topologies, routing and forwarding architecture; independent configuration

✓ **Manageability**: separate policy and mechanism

✓ **Scalability**: maximize number of co-existing virtual networks

✓ **Security and Isolation**: isolate both the logical networks and the resources

✓ **Programmability**: programmable routers, etc.

✓ **Heterogeneity**: support for different technologies
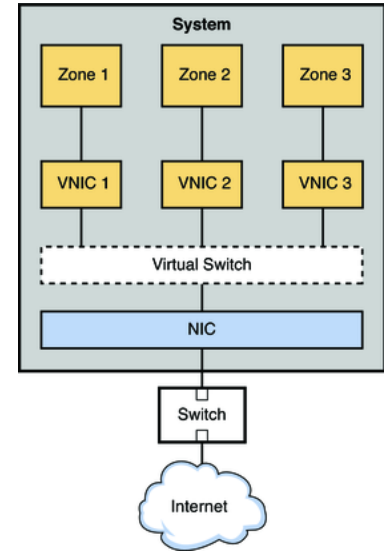
Network virtualization

Ethernet GRE Tunnelling (Ethernet frames are incapsulated in IP packets)

Virtual Disks are where guest operating systems are installed, making them the equivalent of traditional hard disks.

Fixed hard disk image. This type has the same size as the virtual disk and is characterized by a raw disk image followed by a VHD footer.

Expandable (or dynamic) hard disk image. This type is as large as the actual data it contains and includes a header and footer.

Differencing hard disk image. This image type saves all changes within the VHD to a child image, with the option to undo the changes or to merge the changes into the VHD. This type of image allows cloning of VHDs.

Pass-through disk image. This kind is linked to a physical hard drive or to one of its partitions.

Disk virtualization

Virtual Disk Image (VirtualBox): portable and supported by other hypervisors. Fixed-size and dynamically allocated storage. Expansible. No Incremental backups.

Virtual Hard Disk (VHDX – Microsoft): default used in the Microsoft Hyper-V hypervisor that was first introduced in Windows 2012. It features a storage capacity of 64 TB,

Virtual Machine Disk (**VMDK**, VMWARE): cloning of a physical hard disk and Backup, dynamic (sparse) or fixed (flat)

Key Concepts…

What is Virtualization?

What is a Virtual Machine?

What is a Hypervisor?

that's all, have fun