# Week 3 Virtualization and Containerization

Dr Zhi Zhang

## Overview

- Operating System Concepts
- Virtualization
- Containerization

## Operating System Concepts

- Process scheduling
- Virtual memory

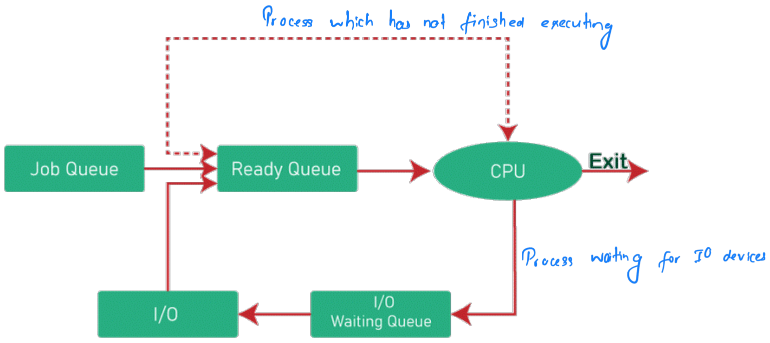
## What is a process?

- A process is a program that is executed
- It is a basic unit of execution in an OS.
- At a minimum, process execution requires CPU and memory without I/O devices.

## Process scheduling (time-sharing)

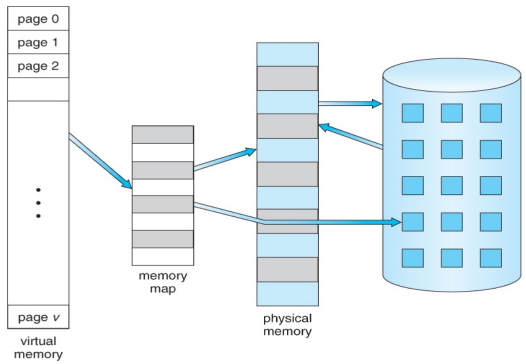
- Each process is scheduled and executed within their time share.
- The number of processes can be much larger than that of the CPUs.

## Process scheduling




https://www.tutorialandexample.com/process-scheduling-in-operating-system

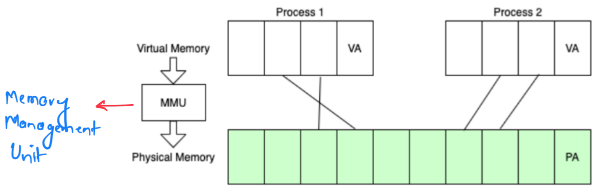
## Virtual memory




https://www.cs.uic.edu/~jbell/CourseNotes/OperatingSystems/9_VirtualMemory.html

## Virtual memory

- Page table:
  - In a modern OS (e.g., Linux, Windows or MacOS), the OS uses a set of page tables to map virtual memory within a process to their corresponding physical memory in main memory.

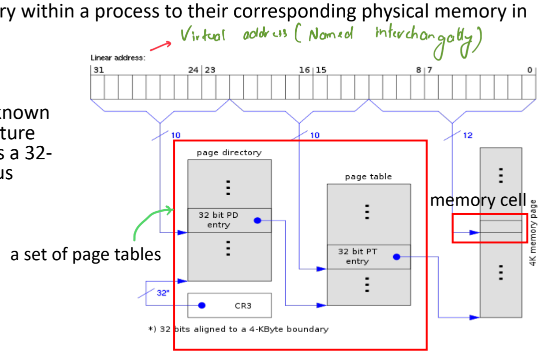



  - VA refers to virtual address, pointing to a memory cell in a virtual page.
  - PA refers to physical address, pointing to a memory cell in a physical page.

https://www.alibabacloud.com/blog/a-tribute-to-hackers-the-way-to-explore-memory-virtualization_599058

## Virtual memory

- Page table:
  - In a modern OS (e.g., Linux, Windows or OS X), the OS uses a set of page tables to map virtual memory within a process to their corresponding physical memory in main memory.
- 32-bit version of x86, known as IA-32 (Intel Architecture 32-bit), which supports a 32-bit memory address bus




https://www.cs.virginia.edu/~bjc8c/class/cs6456-s20/slides/01-intro.pptx

## What is virtualization?

- Virtualization is the ability to run multiple operating systems on a single physical system and share the underlying hardware resources [1]
- Allows one computer software (called Virtual Machine Monitors or Hypervisor) to provide the appearance of many computers (called virtual machines).
- Goals:
  - Provide flexibility for users
  - Amortize hardware costs
  - Isolate completely separate users

[1] VMWare white paper, Virtualization Overview

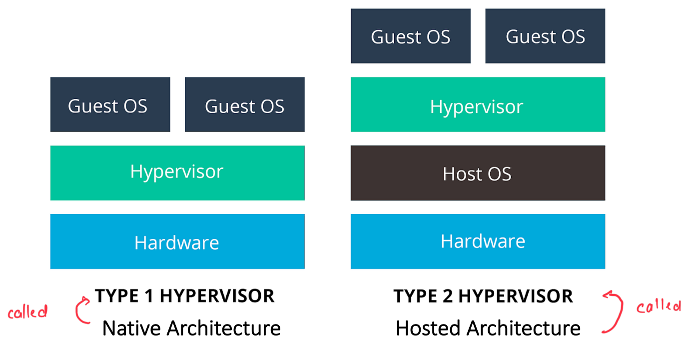
## Characteristics for Virtualizable Computer Architectures

- First, the VMM provides an environment for VM. The environment is essentially identical with the original machine.
- Second, VM running in this environment shows minor decreases in speed.
- Last, the VMM is in complete control of all system resources, such as CPUs, memory, disk, etc.

## VMM Types

- **Hosted Architecture**
  - Install as an application on existing x86 "host" OS, *e.g.* Windows, Linux, OS X
  - Leverage host I/O stack and resource management
  - Examples: Virtualbox, VMware
- **Bare-Metal/Native Architecture**
  - VMM or Hypervisor is installed directly on hardware
  - Acknowledged as preferred architecture for mainstream public clouds
  - Examples: KVM, Xen, Hyper-V

## VMM Types



## Virtualization: history

- 1960's: first track of virtualization
  - Time and resource sharing on expensive computers
  - IBM VM/370
- Late 1970s and early 1980s: became unpopular
  - Cheap hardware and multiprocessing OS
- **Late 1990s:** became popular again
  - Wide variety of OS and hardware configurations (compatibility issue)
  - VMWare
- **Since 2000:** hot and important
  - Cloud computing
  - Containers

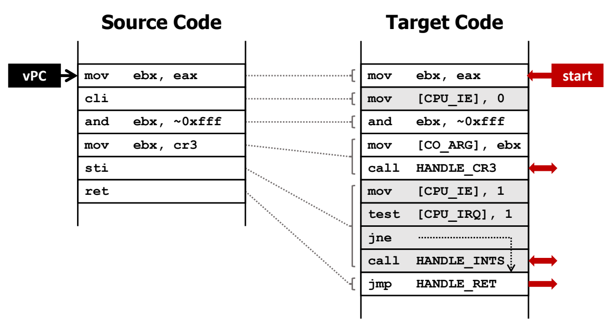
## Virtualization on x86 architecture

- Solutions: → Generations of virtualization solutions
  - Full virtualization implemented by Dynamic binary translation & Two-stage translation
  - Para-virtualization (Xen)
  - Full virtualization implemented by Hardware extension

## Dynamic Binary Translation

- The entire program (e.g., ARM format) is translated into a binary of another architecture (e.g., x86 format).
- The translation occurs on-the-fly during program execution.

## Dynamic Binary Translation

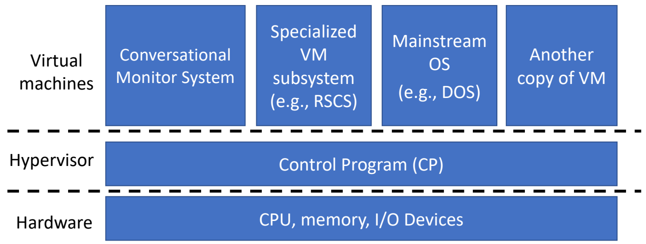



Courtesy Scott Define VMware Inc

## Dynamic Binary Translation

- For sensitive/privileged instructions, trap-and-emulate is needed.
  - e.g., IBM VM/370: the first virtualized system

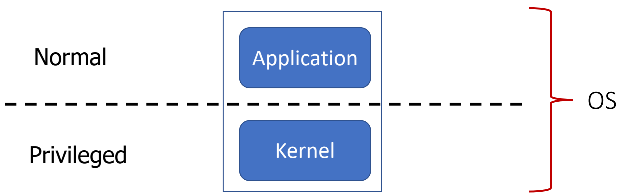
## IBM VM/370: the first virtualized system




https://www.cs.virginia.edu/~bjc8c/class/cs6456-s20/slides/01-intro.pptx

## IBM VM/370

- Virtualization technology: trap-and-emulate



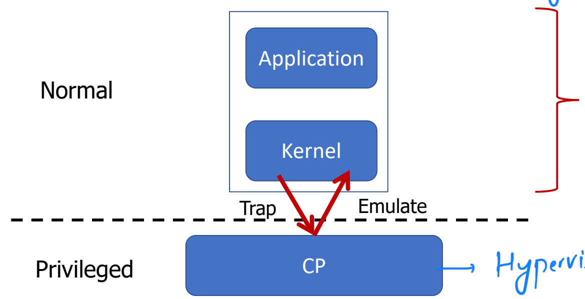

This is how normal OS would look. Forget about Guest OS here.

https://www.cs.virginia.edu/~bjc8c/class/cs6456-s20/slides/01-intro.pptx

## IBM VM/370

When guest OS wants to run some privileged instructions, those instructions will create a trap that goes into hypervisor and hypervisor will emulate the intended functionality of the guest OS

- Virtualization technology: trap-and-emulate




Hypervisor

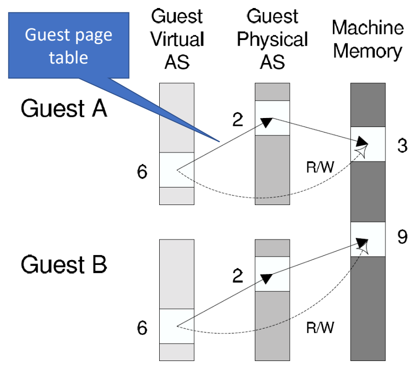
https://www.cs.virginia.edu/~bjc8c/class/cs6456-s20/slides/01-intro.pptx

## Two-stage translation

- Guest page table:
  - A virtual machine has its own guest page table (GPT), managed by the guest OS. The GPT translates guest virtual addresses to guest physical addresses.
- Shadow page table:
  - The hypervisor creates a shadow page table for each guest page table. The shadow page table translates guest virtual addresses to machine addresses.
- Two-stage translation:
  - When a virtual machine running on the hypervisor performs a memory access, the virtual address goes through two stages of translation.

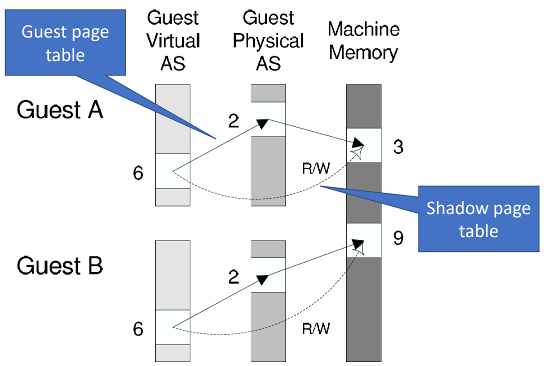
| Guest Virtual Address | $\xrightarrow[\text{Guest OS}]{v2p}$ | Guest Physical Address | $\xrightarrow[\text{VMM}]{p2m}$ | Machine Address |
|---|---|---|---|---|
| | guest page table | | shadow page table | |

## Two-stage translation




https://www.cs.virginia.edu/~bjc8c/class/cs6456-s20/slides/01-intro.pptx

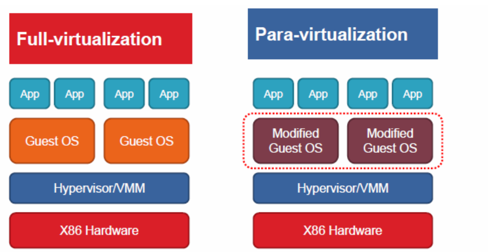
## Two-stage translation




https://www.cs.virginia.edu/~bjc8c/class/cs6456-s20/slides/01-intro.pptx

## Para-virtualization

- Full vs. Para virtualization

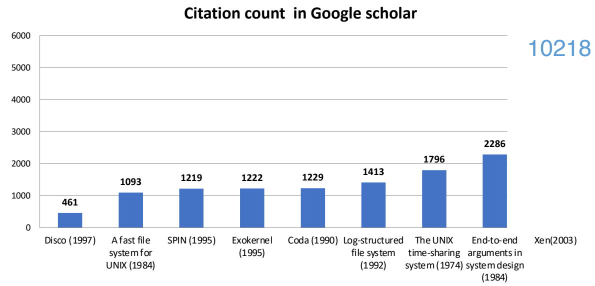



NO changes to OS but big overhead

Significant changes to OS but small overhead

## A typical example: Xen

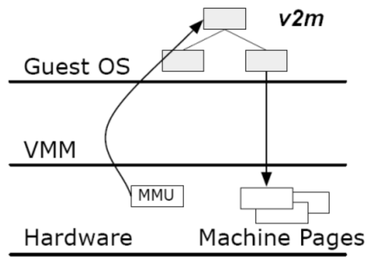
- SOSP (ACM Symposium on Operating Systems Principles ) in 2003.
- Very high impact (data collected in 2013 and 2023)




## Overview of Xen

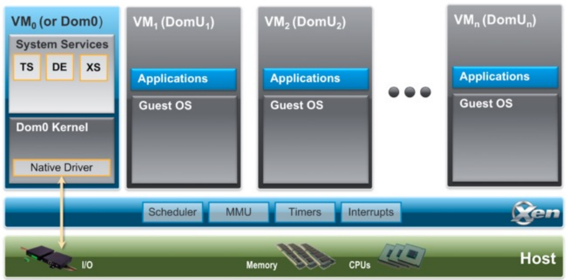
- Modify guest OS to be aware of virtualization
  - No emulation of privileged operations from guest OS
  - Better performance

**(a)** MMU Para-virtualiztion




← Single Translation stage but compatibility issue

## Xen architecture




- TS is Toolstack, providing a user interface to the Xen hypervisor administrator.
- DE is Device Emulation (DE), emulating devices when the devices are not available.
- XS is XenStore/XenBus, managing information that are shared between some domains.

https://wiki.xenproject.org/wiki/Xen_Project_Software_Overview

## A short conclusion

- Full virtualization
  - Unmodified guest OS
  - Performance issue
- Para virtualization:
  - Better performance
  - Modified guest OS

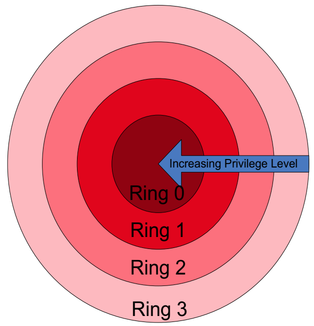
## Support from hardware extensions

- Full/Para virtualization: software-based virtualization
- Hardware support: Intel and AMD assist virtualization

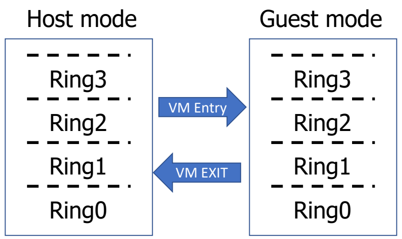
## x86 Protection Rings (CPL)

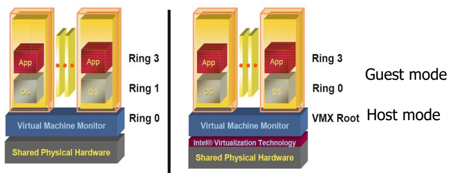
- x86 has four protection levels (rings) /CPU privilege levels (CPLs).
  - ring 0 – "Kernel mode" (most privileged)
  - ring 3 – "User mode"
  - ring 1 & 2 – Other
- Linux only uses ring 0 and ring 3.
  - "Kernel vs. user mode"
- For previous full virtualization, guest OS applications run in ring 3, guest OS kernel runs in ring 1, and the hypervisor runs in ring 0.

CPU Privilege Level (CPL)




## CPU virtualization: Intel VT-x & AMD SVM



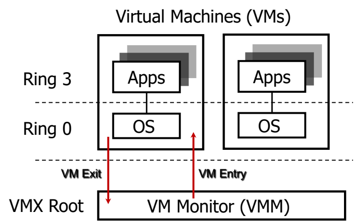




Software-based virtualization

Hardware-assisted virtualization

## **An example: Intel VT-x**

- Two VT-x operating modes
  - Less-privileged mode (guest or VMX non-root) for guest OSes
  - More-privileged mode (host or VMX root) for VMM
- Two transitions
  - VM entry to non-root mode
  - VM exit to root mode



## CPU virtualization

- New VM mode bit
  - Create two orthogonal modes
- If VM mode bit is clear → host mode
- If VM mode bit is set → guest mode

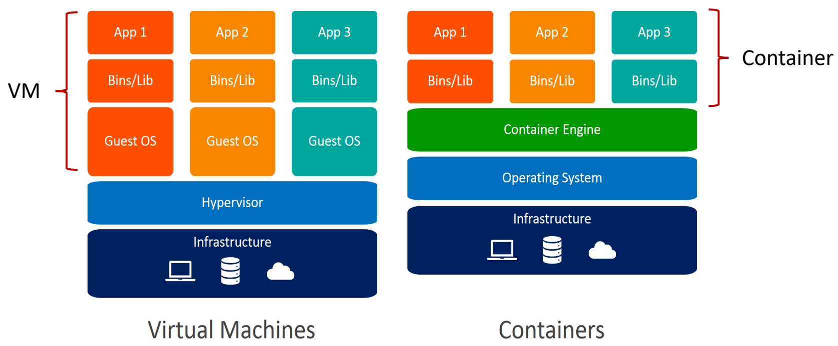
- Benefits:
  - Eliminating the need of binary translation: significant performance improvement.
  - Requiring no changes to privileged operations in guest OSes: compatible with existing modern OSes.

## Containers: concepts

- A container is a sandboxed process running on a host OS that is isolated from all other processes running on that host OS [2]
- A container image is a stand-alone and executable software package (e.g., dependencies, binaries, etc) that contains everything needed to run an application.
- A container is a runnable instance of the image.

[2] https://docs.docker.com/get-started/

## Containers




https://www.weave.works/blog/a-practical-guide-to-choosing-between-docker-containers-and-vms

## Containerization vs. Virtualization

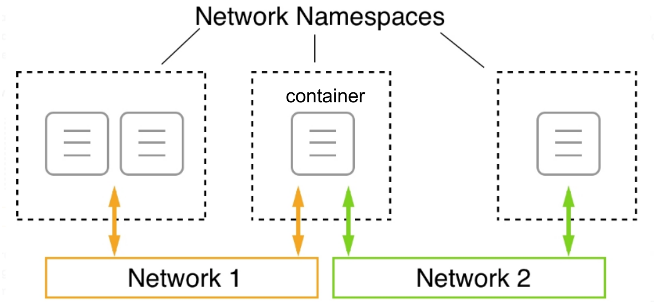
- Granularity
  - Containers are an abstraction of the process layer and VMs are a simulation of the hardware layer.
- Overhead
  - Required resources: containers are created to run one application and VMs support a whole OS.
  - Efficiency: containers are launched to run an application. VMs need to boot up an entire OS.
- Security/Isolation
  - Containers are isolated from each other at the process level. VMs are isolated at the OS level.

## Backbones of containers

- namespaces
- cgroups

## namespaces: limit what a container can see

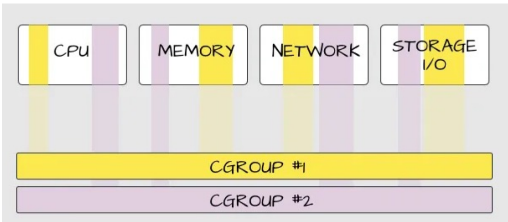
- Provide containers with their own view of system resources (e.g., network)




https://subscription.packtpub.com/book/cloud-and-networking/9781838827472/13/ch13lvl1sec97/running-in-an-existing-network-namespace

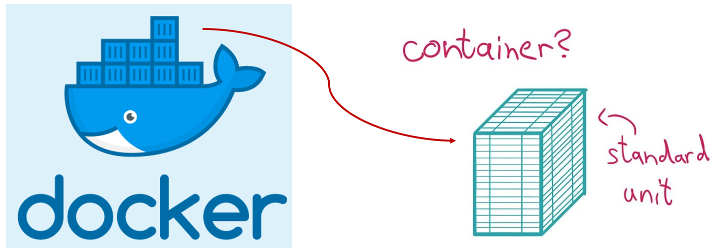
## cgroups: limits how much a container can use

- Control Groups: manage resource usage and limits for processes within a container.
  - Example: a container is limited to use only 1 Mbps of network bandwidth




## Docker container

- Docker provides an interface on top of the techniques
  - Popularized containers (a standard unit of software)




https://medium.com/@saschagrunert/demystifying-containers-part-i-kernel-space-2c53d6979504

## Docker container

Gabriela Georgieva <gabriela.georgieva@docker.com> Jul 14, 2023, 12:46 AM
to Security, me, G

Hi Zhi and Yansong,

We have reviewed your report and agree that this is a security issue. Thank you for bringing it to our attention. We are planning to address it by adding the relevant [redacted] be already covered as they require [redacted] We aim patch this for all of moby, containerd and runc.

We will keep you updated on the status of the issue. Please let us know if you have any questions or comments!

Best regards,
Security Team @ Docker, Inc.

## Docker cheatsheet

- docker run – Runs a command in a new container.
- docker start – Starts one or more stopped containers
- docker stop – Stops one or more running containers
- docker build – Builds an image form a Docker file
- docker pull – Pulls an image or a repository from a registry
- docker push – Pushes an image or a repository to a registry
- docker export – Exports a container's filesystem as a tar archive
- docker exec – Runs a command in a run-time container
- docker search – Searches the Docker Hub for images
- docker attach – Attaches to a running container
- docker commit – Creates a new image from a container's changes

## Practice Questions

- [7 marks] Q1: Describe what virtualization is and describe the characteristic attributes of the different types of virtualization (Language, Operating System and Hardware).

- [1 mark] Virtualization is a technology that allows multiple virtual instances of resources (e.g., operating systems) to run on a single physical machine.
- [2 marks] **Language virtualization**: provides a virtual runtime environment (e.g., JVM) that allows code written in a specific programming language (e.g., Java) to execute.
  - Cross-OS: code can be executed in a runtime environment across OSes without modification.
- [2 marks] **Operating system virtualization**: allows multiple user-space instances (e.g., docker containers) within a single operating-system kernel (e.g., Linux kernel).
  - Limited Isolation: instances share the host OS kernel, which means a vulnerability in the kernel could affect all instances.
- [2 marks] **Hardware virtualization**: creates multiple virtual machines sharing a single physical server.
  - VM Snapshots: allows saving the current state of a VM
  - Live VM Migration: enables moving runtime VMs between physical hosts seamlessly.