CAB432 Cloud Computing
Lecture 2: Container as a Service

Faculty of Science

When the machine isn't really a machine…

VIRTUALISATION

Motivation

- Virtualisation means abstracting hardware or software

- Hardware:
  - virtual machines (CPU, memory)
  - network interfaces, storage devices, GPUs, …

- Software:
  - operating systems,
  - middleware
  - (databases, application servers, messaging middleware, …)

Why?

- Operational cost: hardware time-sharing
  - Fewer physical machines for the same compute
  - Share over time or concurrently
- Normalisation
  - Cookie cutter resources
  - Define an instance type and machine image
  - So much easier to manage and maintain
  - (e.g., AWS instance types, “compute units”)
- Software lifecycle:
  - Deployment of identical machine image updates

Why?

- Isolation:
  - Multiple users/tenants on same hardware
  - But ‘hard’ separation of virtualised resources

- Instance Lifecycle:
  - Continuity: quick fail-over to another resource
  - Scale up: quick replacement with a more powerful resource
  - Scale out: quickly start up additional, identical instances

  - *Note the difference between scale up and scale out*

Types, Terminology, The Rings

HARDWARE VIRTUALISATION

Virtualisation by Resource Type

- Operating system:
  - (re-)implementation of another OS' ABI
  - WINE (Win16/Win32 ABI to POSIX calls)
- Virtual machine ("hardware virtualisation"):
  - full/para/partial virtualisation of hardware platform
  - (typically CPU, memory controller, etc.)
  - VMWare, Xen, Microsoft Hyper-V, KVM, …
  - Storage, network, …

Virtualisation by Use Case

- Desktop virtualisation:
  - Parallels, VMWare Player, Oracle VirtualBox, WSL
  - https://devblogs.microsoft.com/commandline/announcing-wsl-2

- Data centre virtualisation:
  - Xen, KVM, VMWare, …

Hardware Virtualization Terminology

- Host OS:
  - operating system of (non-virtualised) hardware resource
  - “bare metal” running the hypervisor/VMM

- Guest OS:
  - operating system running inside the virtualised hardware

- Hypervisor:
  - The software that runs virtual machines
  - The “virtual machine monitor”
  - Type 1: runs directly on the hardware (like an operating system)
  - Type 2: runs inside an operating system

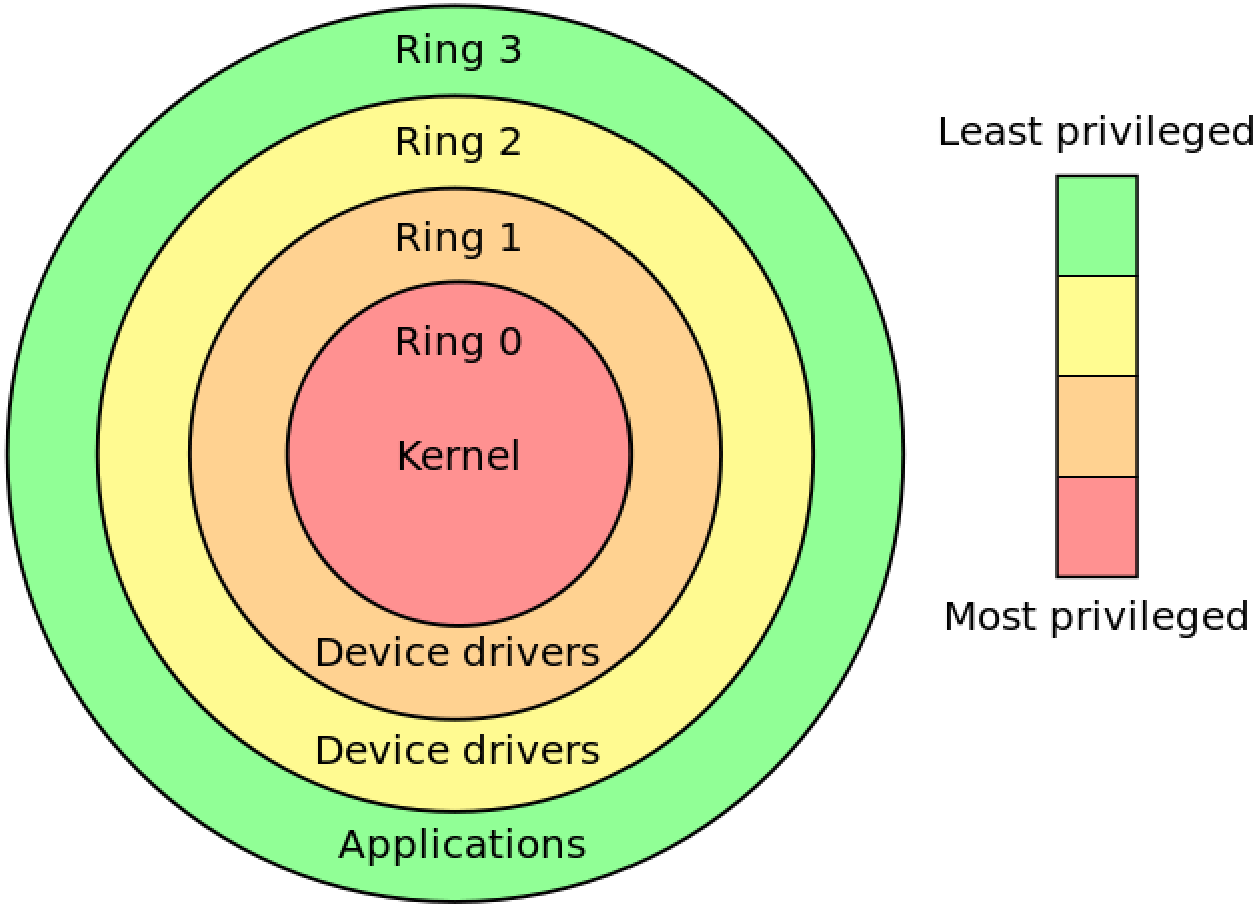
Hardware Virtualisation

- Modern CPUs support virtualisation directly
  - Distinction between Hardware Virtualisation and Hardware-Assisted Virtualisation

- Hardware Virtualisation
  - Partial of full emulation of the machine (usually slooooooooow)

- Hardware-Assisted Virtualisation
  - Use privileged CPU instructions to call OS level instructions

Protection Layers or Rings



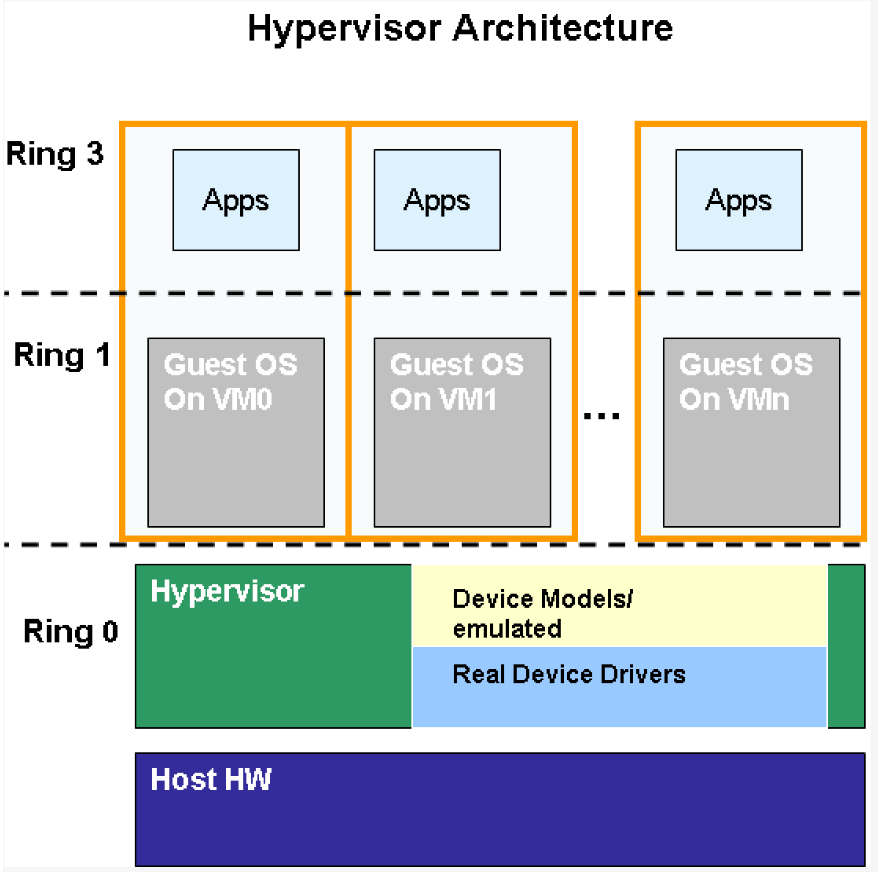
By Hertzsprung at English Wikipedia, CC BY-SA 3.0, https://commons.wikimedia.org/w/index.php?curid=8950144

The Rings

- Kernel mode:
  - privileged instructions (interrupts, memory management, etc.)
  - normally reserved for OS kernel (ring 0)
  - Applications can’t normally see this (see below)

- User mode:
  - non-privileged, “safe” instructions (ring 3)
  - application software asks the kernel nicely to perform privileged instructions on its behalf
  - Guest OS given higher privilege level via the hypervisor

Hardware (-assisted) virtualisation basics



Hardware (-assisted) virtualisation basics

- Multiple guests share the same host hardware
  - Includes OS and application software stack

- Hypervisor provides software layer underneath guest OSs
  - Runs in “ring 0”. See “ring -1” discussion later
  - Hypervisor controls and isolates guest OSs from one another
  - Guests are ‘promoted’ to ring 1

IaaS vs CaaS

APPLICATION VIRTUALIZATION

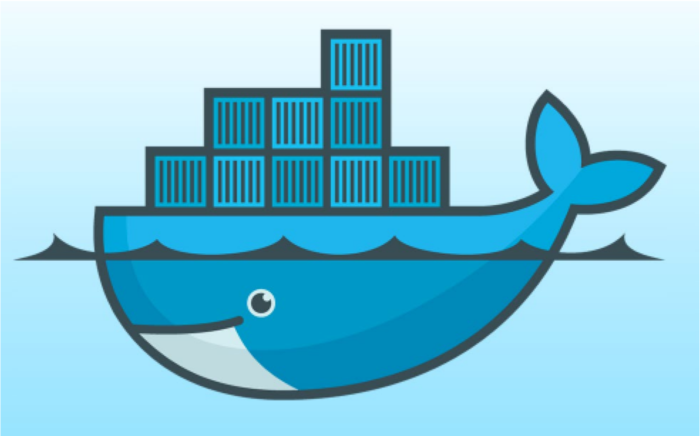
Challenges for IaaS

- Relies on *hardware* virtualisation
  - Heavyweight and complex hypervisor
  - (performance, resource consumption)
  - Virtualises an entire guest OS
  - (kernel, device drivers, filesystem, network protocol stacks, standard middleware and applications, …)

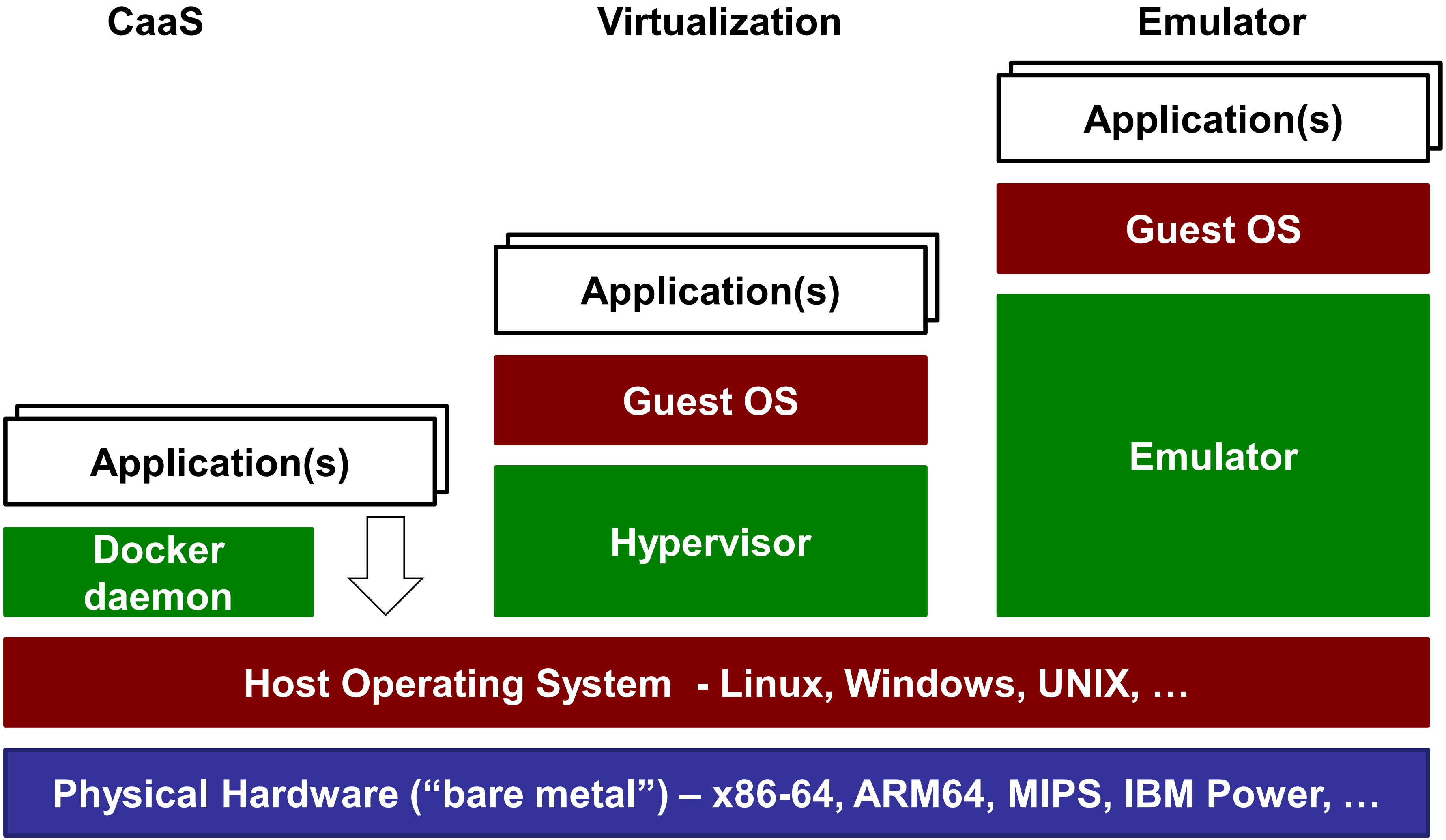
- Replication schemes:
  - Cloning: copies entire OS state – disk + main memory image
  - Redo (or “infrastructure mgmt”):
  - Requires additional powerful (and complex) tools
  - (“Chef” and “Puppet”)

CaaS

- Container as a Service combines *both:*
  1. Application virtualisation
  2. Infrastructure management



CaaS vs Virtualization vs Emulators



CaaS

Virtualization

Emulator

Application(s)

Guest OS

Application(s)

Guest OS

Emulator

Application(s)

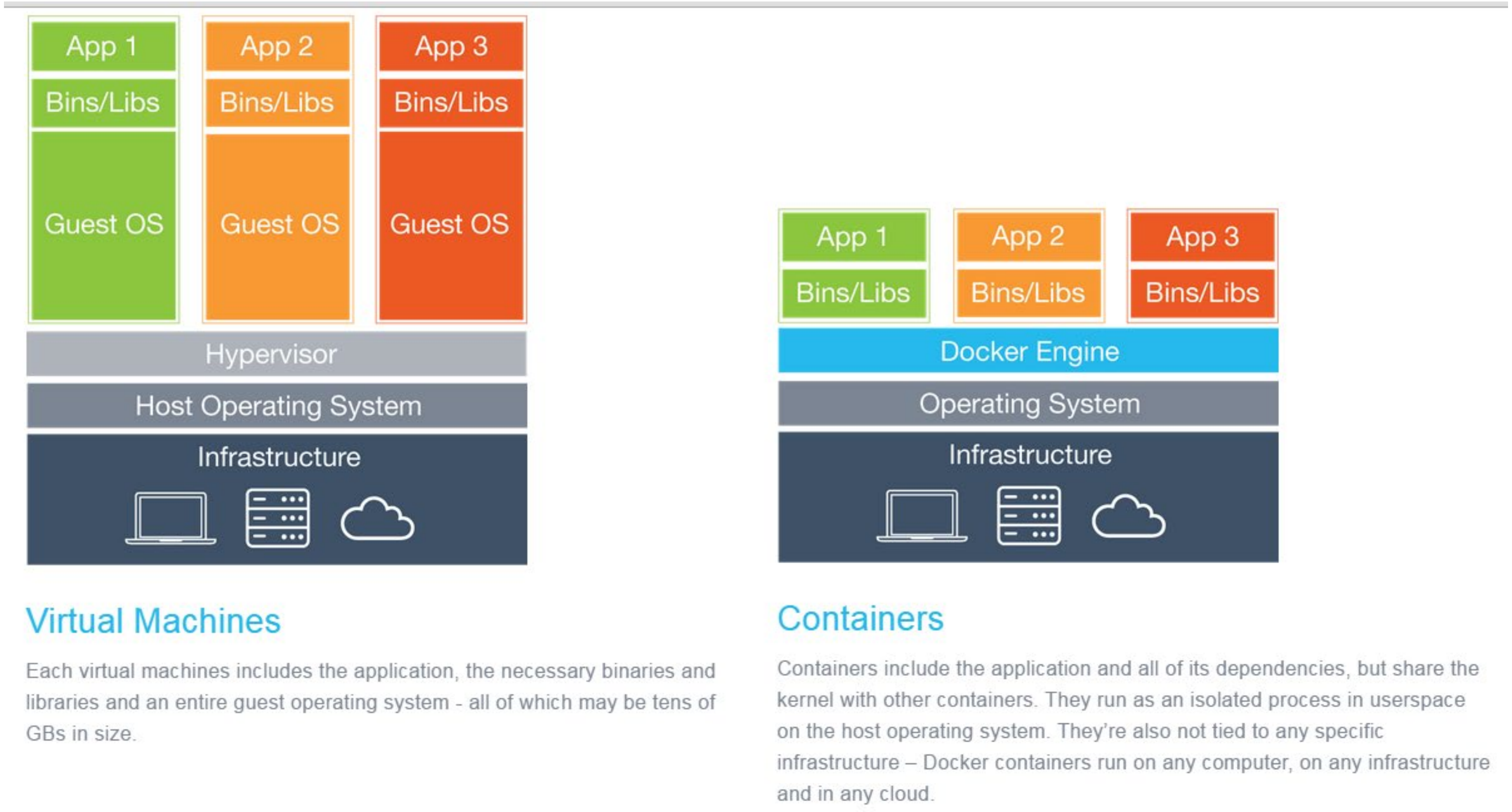
Docker daemon

Hypervisor

Host Operating System - Linux, Windows, UNIX, …

Physical Hardware (“bare metal”) – x86-64, ARM64, MIPS, IBM Power, …

CaaS = Simplified Virtualisation



Virtual Machines

Each virtual machines includes the application, the necessary binaries and libraries and an entire guest operating system - all of which may be tens of GBs in size.

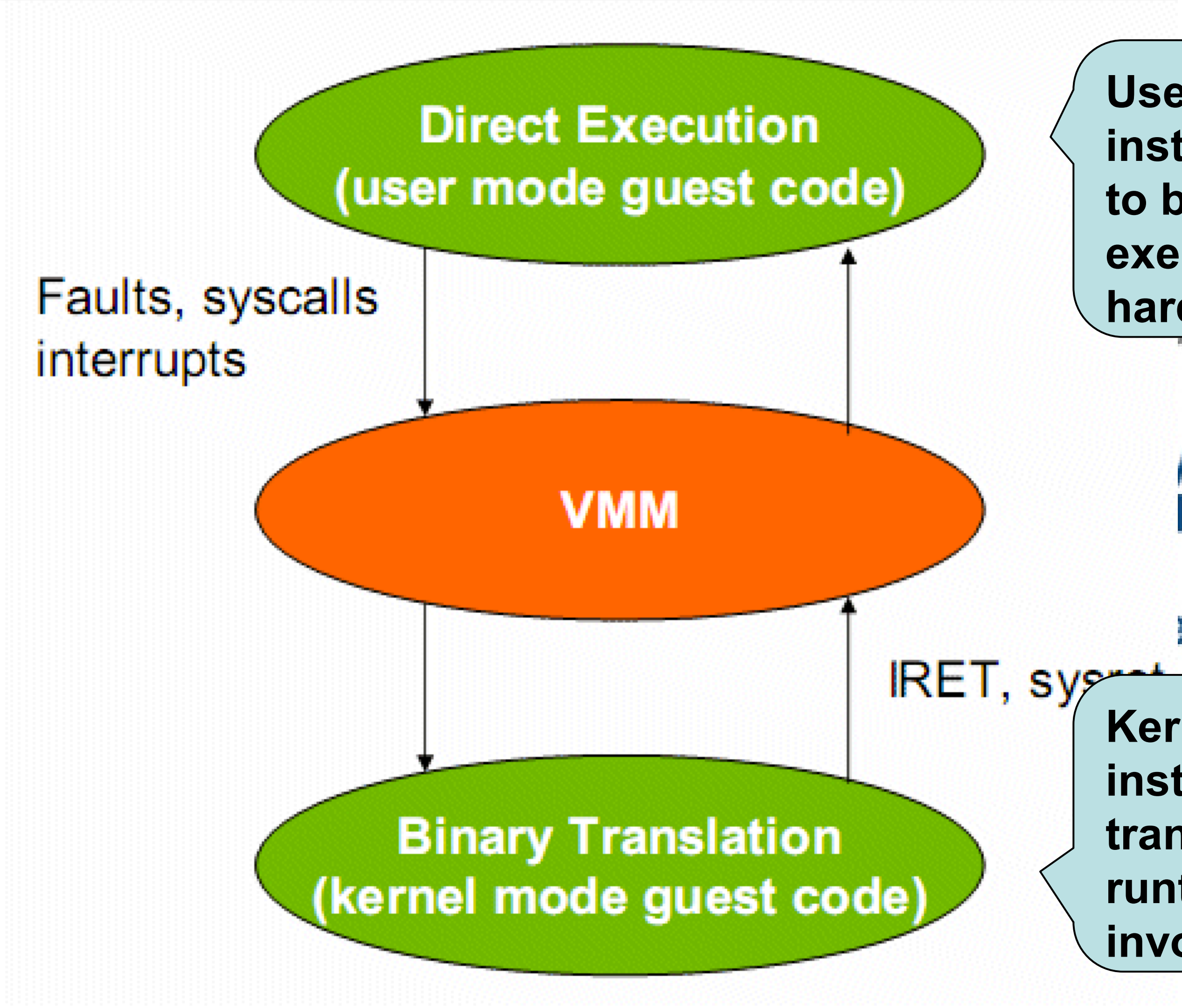
Containers

Containers include the application and all of its dependencies, but share the kernel with other containers. They run as an isolated process in userspace on the host operating system. They're also not tied to any specific infrastructure – Docker containers run on any computer, on any infrastructure and in any cloud.

Emulation, Paravirtualisation, Assisted Virtualisation

A DEEP DIVE

Hardware Emulation



User mode (ring 3) instructions are safe to be directly executed on host hardware

Kernel mode (ring 0) instructions must be translated (at runtime) into VMM invocations

”):

/

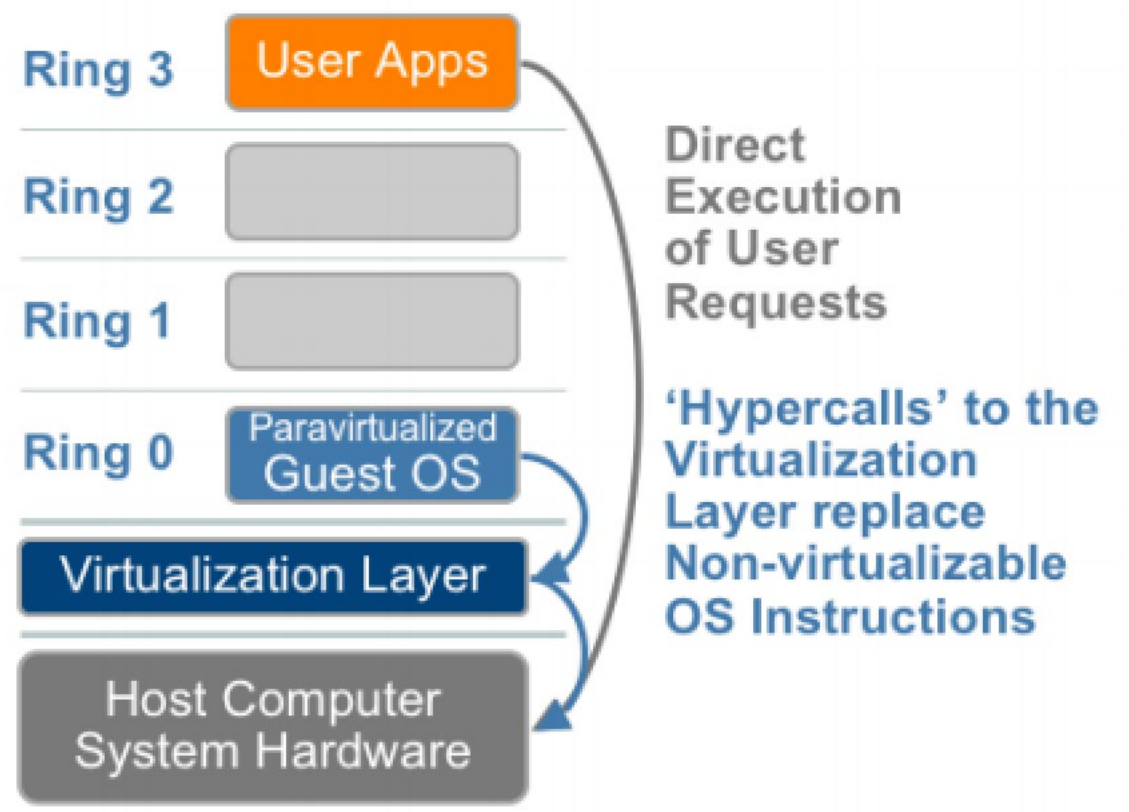
lation

ests

.

Paravirtualisation

- Mc
- VI
  - tions
- Pr
- Cc
- Ex


Ring 3
User Apps
Ring 2
Ring 1
Ring 0
Paravirtualized Guest OS
Virtualization Layer
Host Computer System Hardware
Direct Execution of User Requests
'Hypercalls' to the Virtualization Layer replace Non-virtualizable OS Instructions

Hardware Assisted-Virtualisation

- “Fixes” a flaw in the x86 instruction set

- Dynamically “trap” unsafe operations
  - Again avoids binary translation
  - Introduces a new “ring -1” (root mode) for VMM
  - VMM kicks in after unsafe ring 0 instruction is “trapped”,

Hardware Assisted-Virtualisation

- Pros:
  - Guest OS continues in privileged kernel mode (ring 0)
  - Most ring 0 instructions do not “VMExit” to “ring -1”
  - Faster than full virtualisations,
  - Sometimes faster than paravirtualisation, sometimes not
  - Does not require guest OS modifications

- Cons:
  - Requires hardware support (Intel VT-x, AMD-V)
  - But this is now common