Ciência de Dados em Larga Escala

Inês Dutra and Zafeiris Kokkinogenis

DCC-FCUP
room 1.31
ines@dcc.fc.up.pt
zafeiris.kokkinogenis@gmail.com

23/24

Virtualization

based on material available
https://www.cl.cam.ac.uk/teaching/2223/CC/materials.htmlhere

Abstracts the underlying resources (processors, memory, communication links); simplifies their use; isolate users from one another; supports replication and migration.
Allows for:

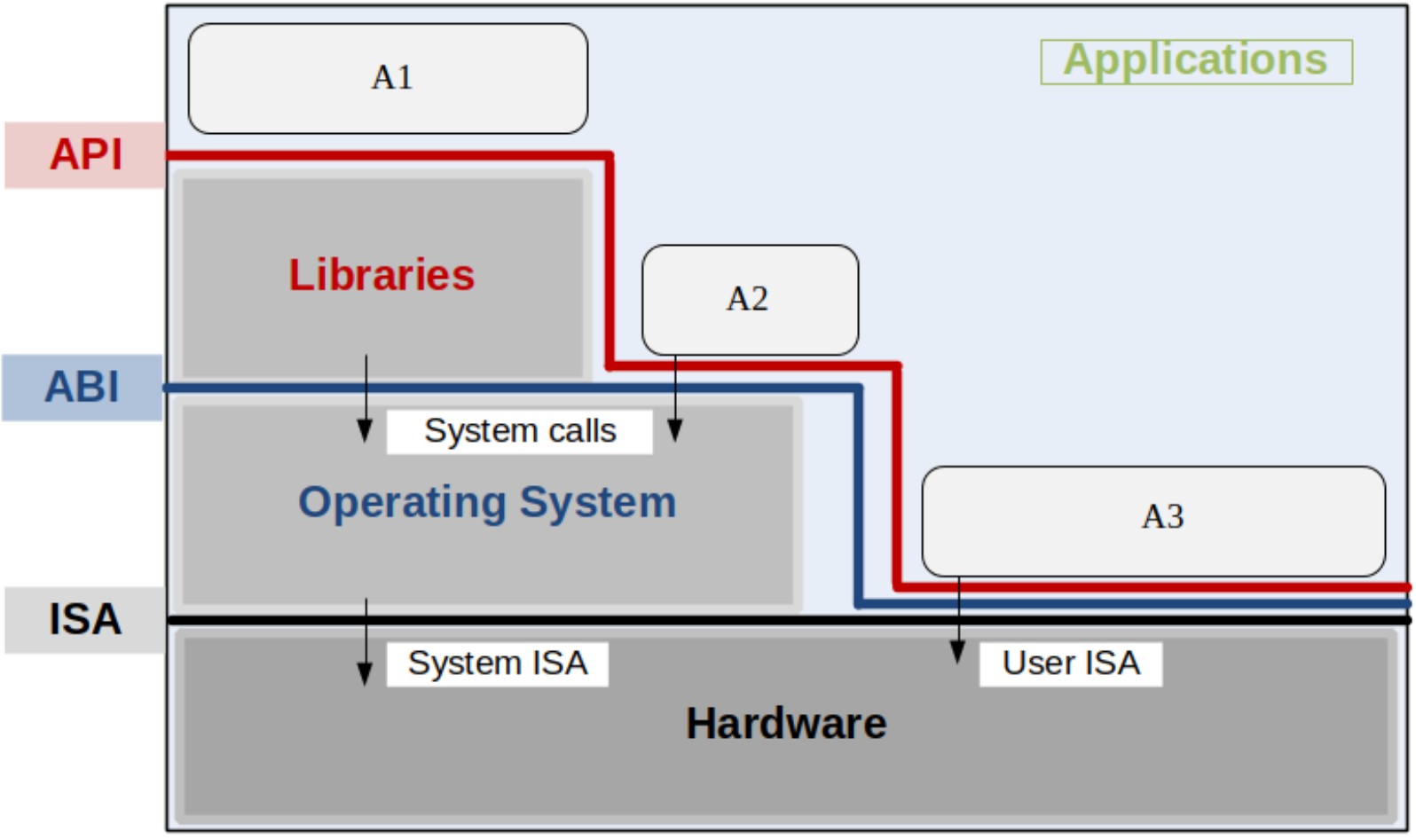
- performance isolation: dynamically assigns and accounts for resources across different applications
- system security: allows isolation of services running on the same hardware
- performance and reliability: allows apps to migrate among different platforms

Virtualization

Simulates the interface to a physical object by:

- multiplexing: creates multiple virtual objects from one instance of a physical object
- aggregation: creates one virtual object from multiple physical objects
- emulation: builds a virtual object of a certain type from a different type of a physical object
- multiplexing and emulation: examples: virtual memory with paging multiplexes real memory and disk; virtual address that emulates a real address etc.

Layering and interfaces



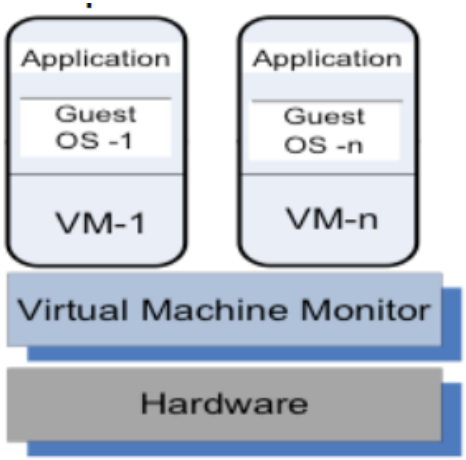
API - Application Programming Interface, ABI - Application Binary Interface, ISA - Instruction Set Architecture. An application uses library functions (A1), makes system calls (A2), and executes machine instructions (A3)

Code portability

- it is possible to compile a program written in a Higher Level Language (HLL) to a virtual machine and make the code run in any physical host by using binary translators
- a dynamic binary translator converts blocks of guest instructions from the portable code to the host ISA, such that blocks can be cached and reused

Virtual machine monitor (VMM)

- VMM or **hypervisor** partitions the resources of a computer system into one or more **virtual** machines (VMs)
- allows several OS to run concurrently on a single hw platform
- a VM is an execution environment that runs an OS
- a **guest OS** is an OS that runs in a VM under the control of a VMM

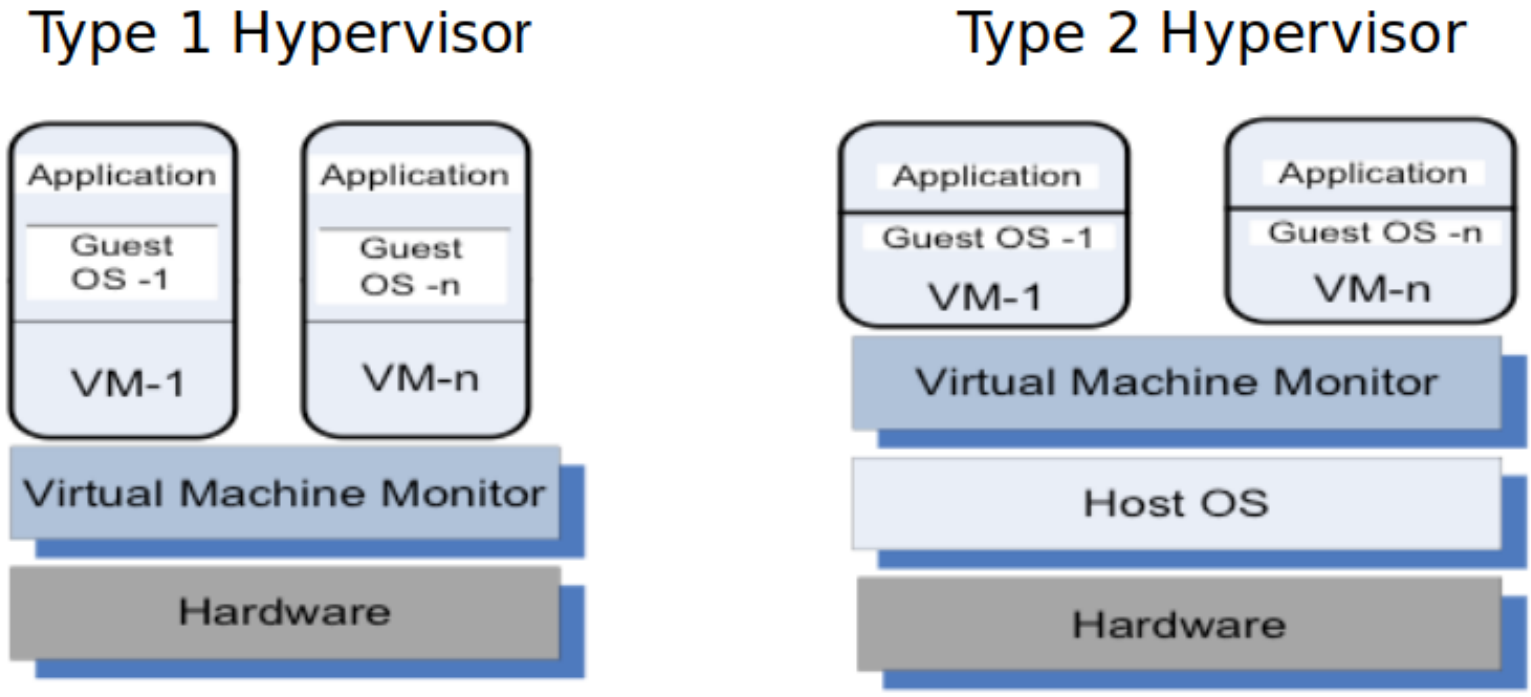


Virtual machine monitor (VMM)

VMM:

- traps privileged instructions executed by a guest OS and enforces correctness and safety
- traps interrupts and dispatches them to the individual guest OS
- controls the virtual memory managament
- maintains a shadow page table for each guest OS
- monitors system performance (may swap out a VM to avoid trashing)

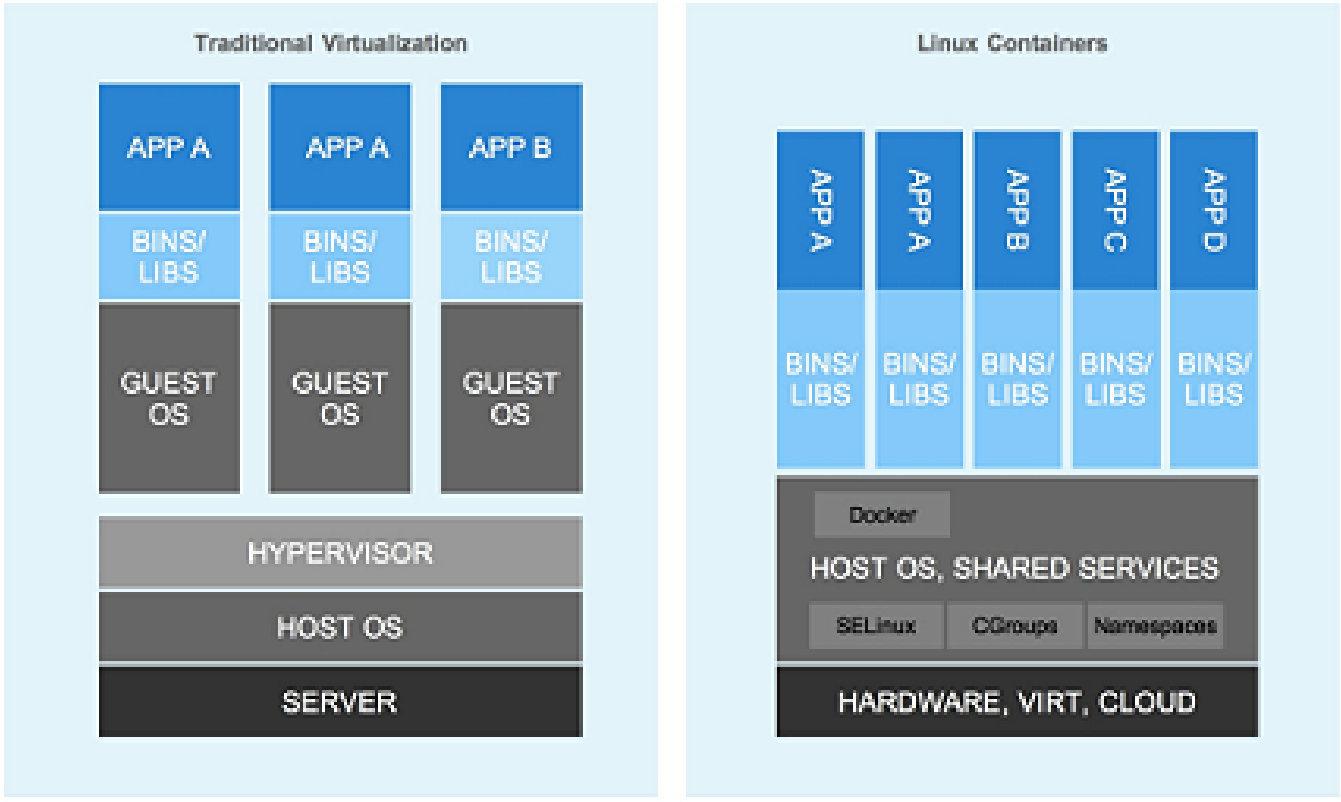
Type 1 and type 2 hypervisors



Type 1 (bare metal, native): VMWare ESX, Xen, Denali
Type2 (hosted): user-mode Linux

Linux container

It is a Linux process (or processes) that is a virtual environment with its own process network space (lightweight process virtualization)



Linux containers

Containers share portions of the host kernel
Use:

- namespaces: per-process isolation of OS resources (file system, network and user ids)
- cgroups: resource management and accounting per process