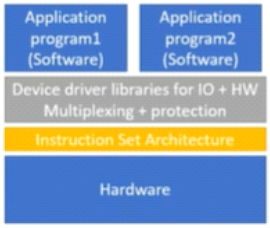
Ai accelerators 4

29 July 2025 00:18

OS → manage CPU, memory, I/O devices.
→ read / write disk.
→ access hardware.

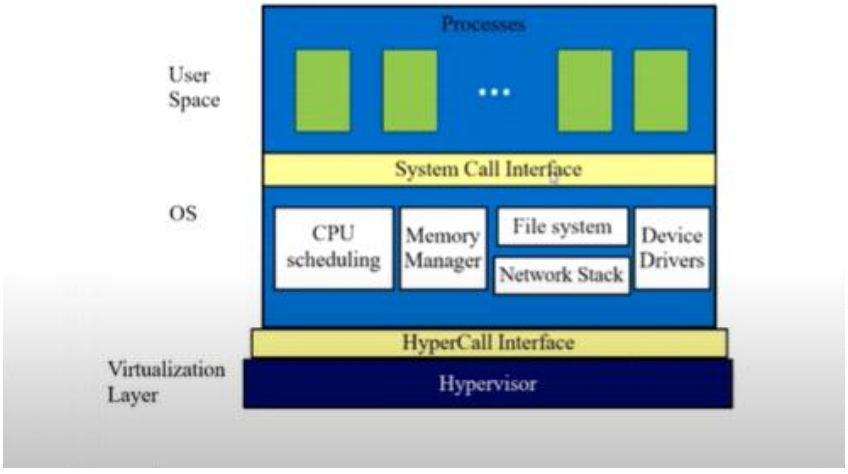
why need ?

⇒ if multiple software needs to run, then we need multiplexing to access resources available.

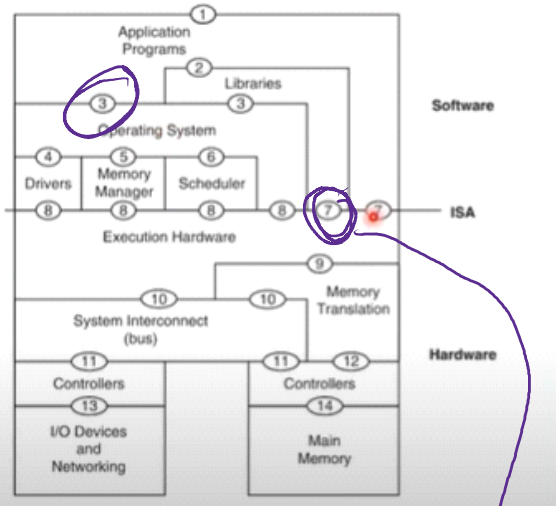


- Two programs does not trust each other
- OS does not trust programs
- HW does not trust programs

⇒ most of the time OS will not know ∃ Hypervisor, but if there's need to talk to it then it can via Hypercall interface.



Interfaces available in a computing system



User ISA : 7
System ISA : 8
Syscalls : 3
ABI : 3, 7
API : 2,7

ISA = Instruction Set Architecture
ABI = Application Binary Interface
API = Application Programming Interface

Application Program has 2 sections open to make system calls

Virtualization

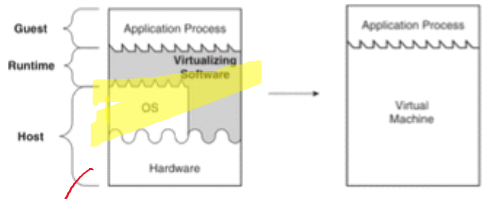
- Makes a real system appear to be a set of virtual systems
- One-to-many virtualization
  - E.g. one physical machine may appear as multiple virtual machines
  - one physical disk may look like multiple virtual disk
  - one physical network may look like multiple virtual networks
- Many-to-one virtualization
  - Many physical machines/disks/networks may appear to look like one virtual machine/disk/network etc
- Many-to-many virtualization

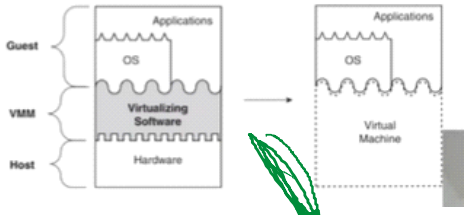
Virtual Machine (VM)

- Logical/Emulated representations of full computing system environment
  - CPU + memory + I/O
  - Implemented by adding layers of software to the real machine to support the desired VM architecture.
- Uses:
  - Multiple OSes on one machine, including legacy OSes
  - Isolation
  - Enhanced security
  - Live migration of servers
  - Virtual environment for testing and development
  - Platform emulation
  - On-the-fly optimization
  - Realizing ISAs not found in physical machines

Different types of VMs

- Process VM
  - Virtualizes the ABI
  - Virtualization software => Runtime
  - Runs in non-privileged mode (user space)
  - Performs binary translation.
  - Terminates when guest process terminates.
- System VM
  - Virtualizes the ISA
  - Virtualization software => Hypervisor
  - Runs in privileged mode
  - Traps and emulates privileged instructions





interface b/w hardware & OS = ISA (x86)

⇒ we want to test a new ISA, virtualization software will do conversion from this ISA to new one.

eg. JVM → translates ISA byte code to your x86 ISA, it can run it.

Java byte code → Run on JVM

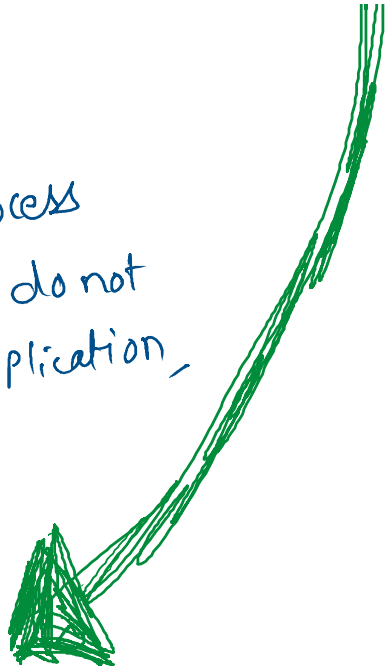
⇒ through VMs we are virtualizing (ABI)

interface combination of system, call interface & user ISA.

advantages of virtualizations.

Virtualizations.

⇒ if we want to run application Process into another system (ARM), arm we do not need to change anything inside application, VM 'll convert bytecode



System VMs → virtualizes ISA itself.

(x86 to ARM)

OS don't know unaware of underlying hardware that application runs on.