Amirkabir University of Technology
(Tehran Polytechnic)

Cloud Computing

Hardware virtualization-Part1

Seyyed Ahmad Javadi

sajavadi@aut.ac.ir

Fall 2022

Introduction

Hardware-level Virtualization

- **An abstract execution environment in terms of computer hardware** on top of which a ***guest operating system*** can be run.

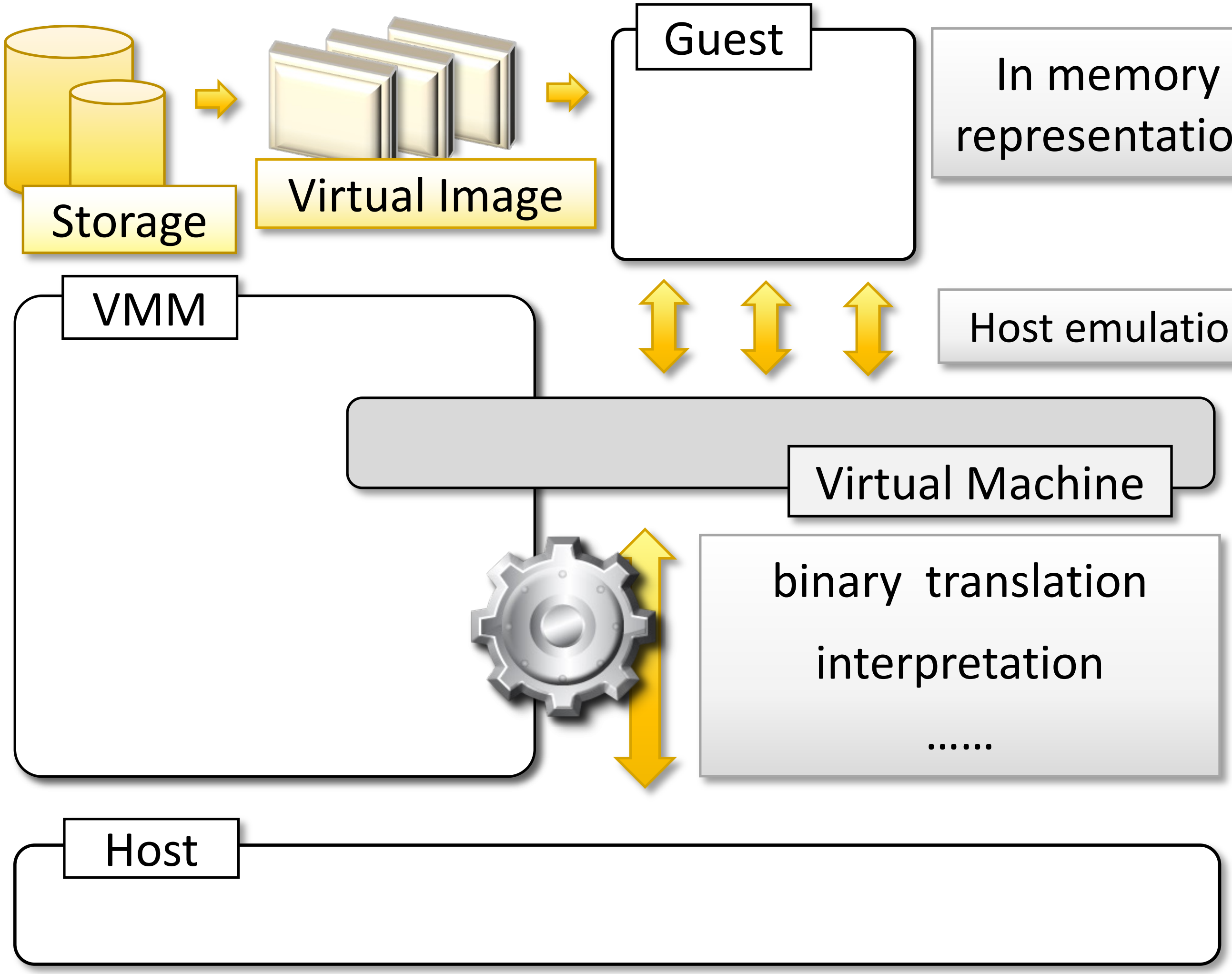
Concept	Represented by
Guest	Operating system
Host	Physical computer hardware
Virtual machine	Its emulation
Virtual machine manager	Hypervisor

What is Hypervisor?

Hypervisor is a program enabling the abstraction of the underlying physical hardware.

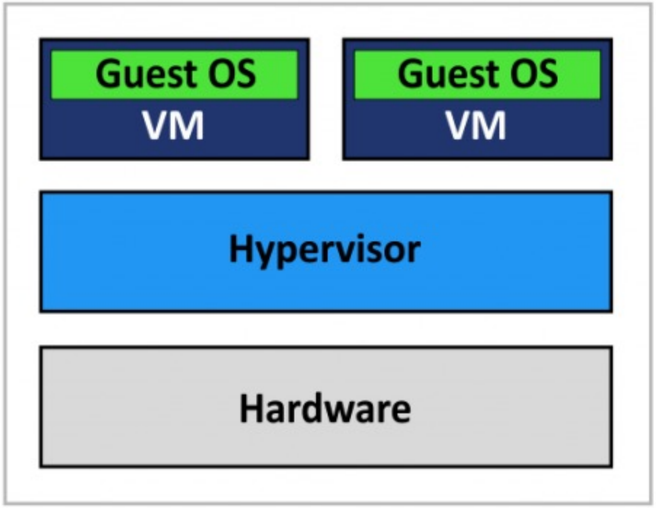
Hypervisor is also called Virtual Machine Manager (**VMM**)

Hardware-level Virtualization



Types of Hypervisor

- ***Type I hypervisors*** (native VM)
  - Run *directly* on top of the hardware.
  - *Take the place* of the operating systems
  - Interact directly with the ISA interface

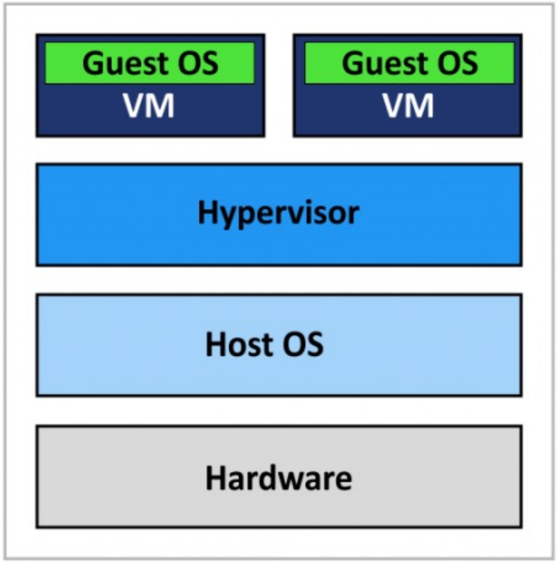


Type 1 Hypervisor
(Bare-Metal Architecture)

Source: http://: https://www.nakivo.com/blog/hyper-v-virtualbox-one-choose-infrastructure/

Types of Hypervisor (cont.)

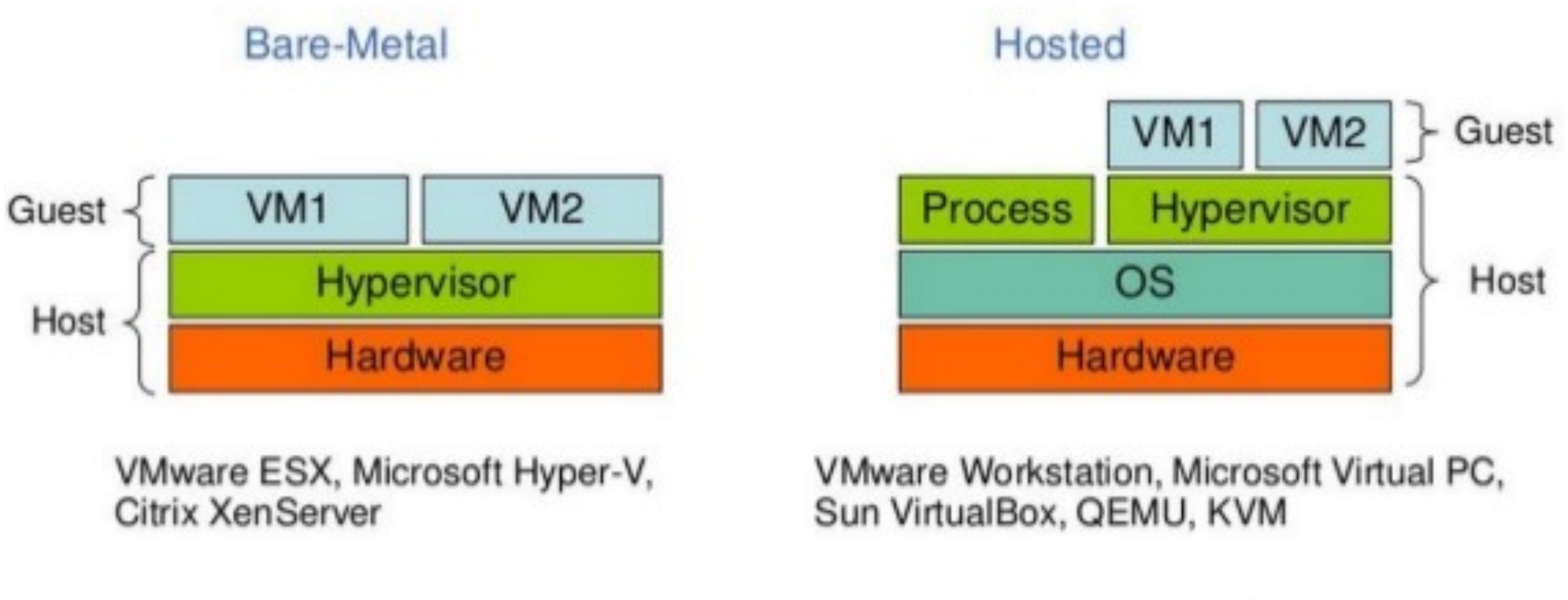
- ***Type II hypervisors*** (hosted VM)
  - Require the support of an operating system
  - Are programs **managed by the operating system**
  - Interact with operating system through the **ABI**.



Type 2 Hypervisor
(Hosted Architecture)

Source: http://: https://www.nakivo.com/blog/hyper-v-virtualbox-one-choose-infrastructure/

Type of Hypervisors (cont.)


Bare-Metal
Guest
VM1
VM2
Host
Hypervisor
Hardware
VMware ESX, Microsoft Hyper-V, Citrix XenServer
Hosted
VM1
VM2
Guest
Process
Hypervisor
OS
Host
Hardware
VMware Workstation, Microsoft Virtual PC, Sun VirtualBox, QEMU, KVM

Source: https://www.slideshare.net/PraveenHanchinal/virtualizationthe-cloud-enabler-by-inspiregroups/18-Types_of_hypervisors_VMM

Approaches of Executing Guest Instructions

Executing Guest Instructions

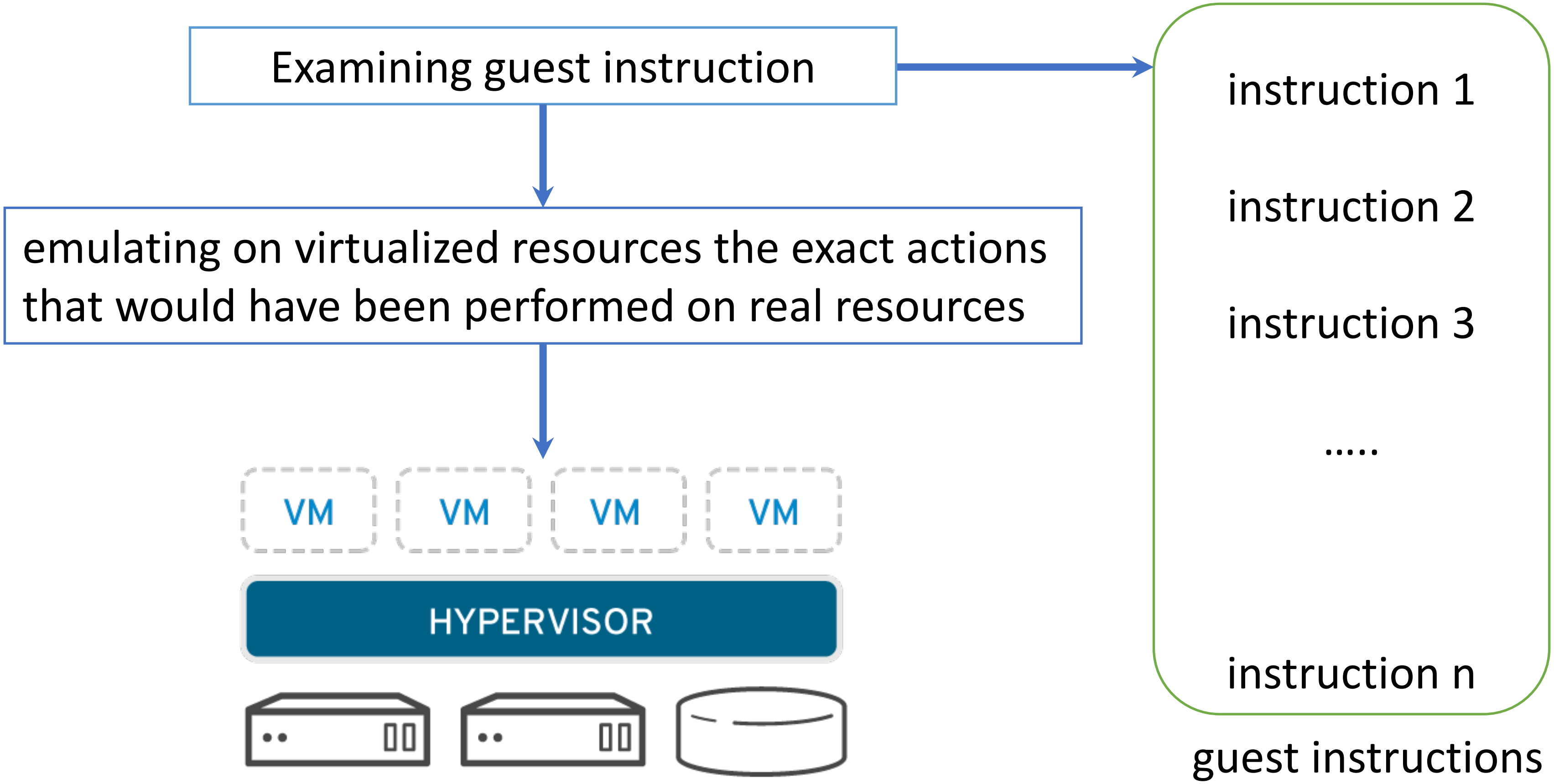
- **Emulation**

- **Direct native execution**

Emulation

“the process of implementing the interface and functionality of one system or subsystem on a system or subsystem having a different interface and functionality...”

Emulation (cont.)



Examining guest instruction

emulating on virtualized resources the exact actions that would have been performed on real resources

instruction 1

instruction 2

instruction 3

.....

instruction n

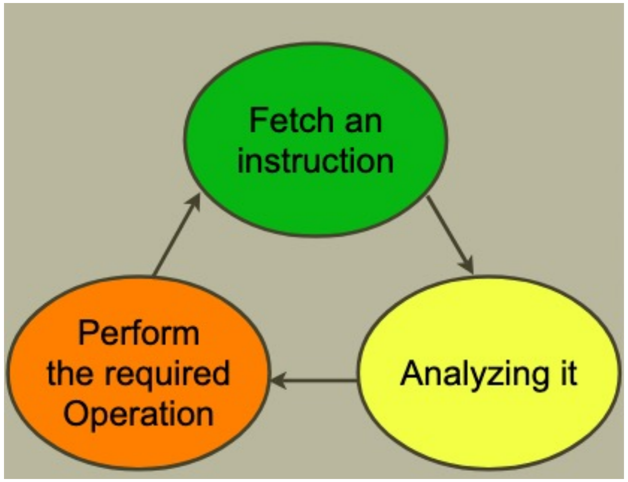
guest instructions

Only available mechanism when the **ISA of the guest** is ***different*** from the **ISA of the host**.

Emulation Approaches

- **Interpretation**
  - Done in software,

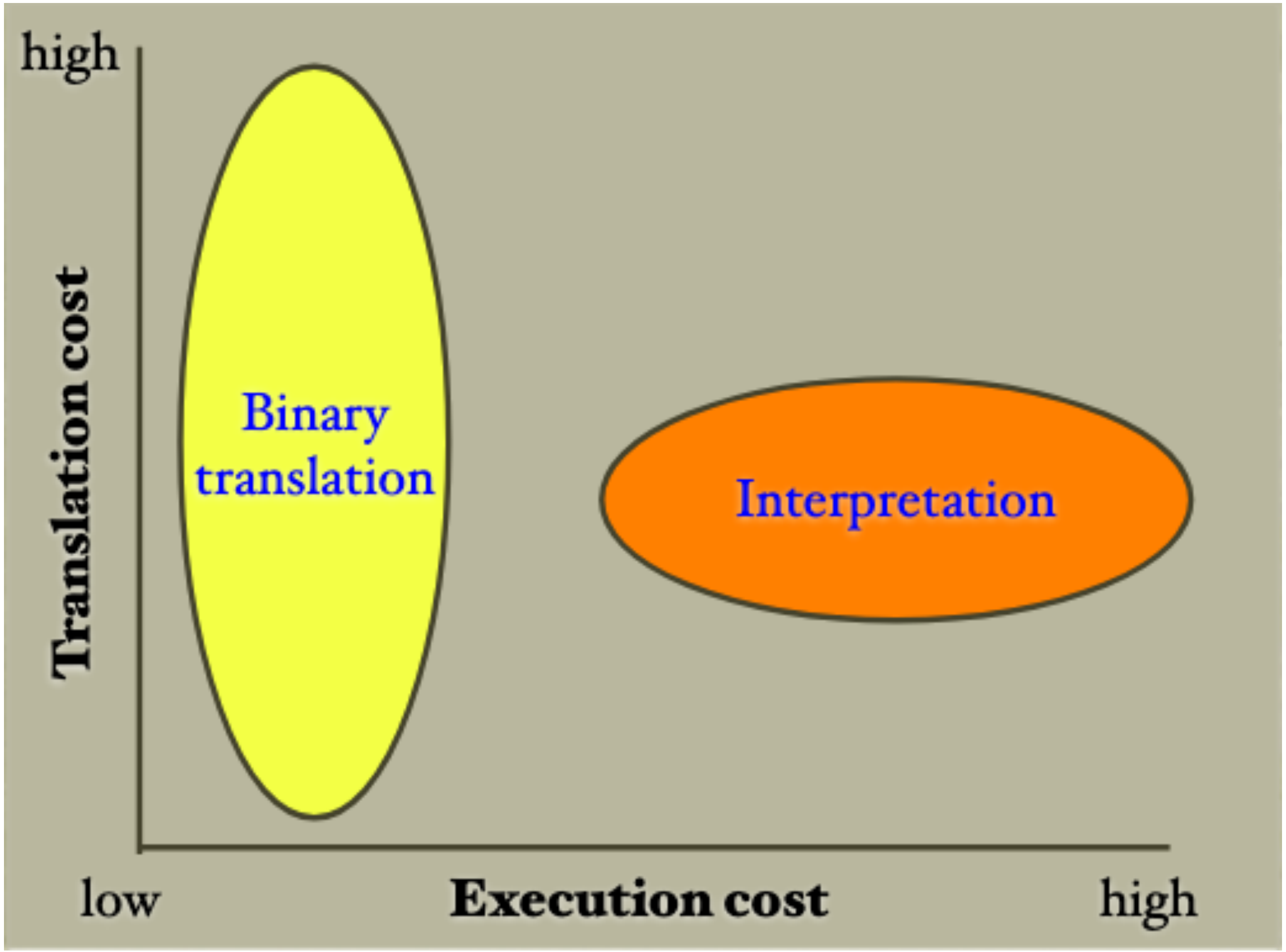
    one instruction at a time



http://cse.unl.edu/~witty/class/embedded/material/note/emulation.pdf

- **Binary translation**
  - Translating a block of source instructions to target instructions.
  - Saving the translated code for repeated use

Interpretation versus Binary Translation



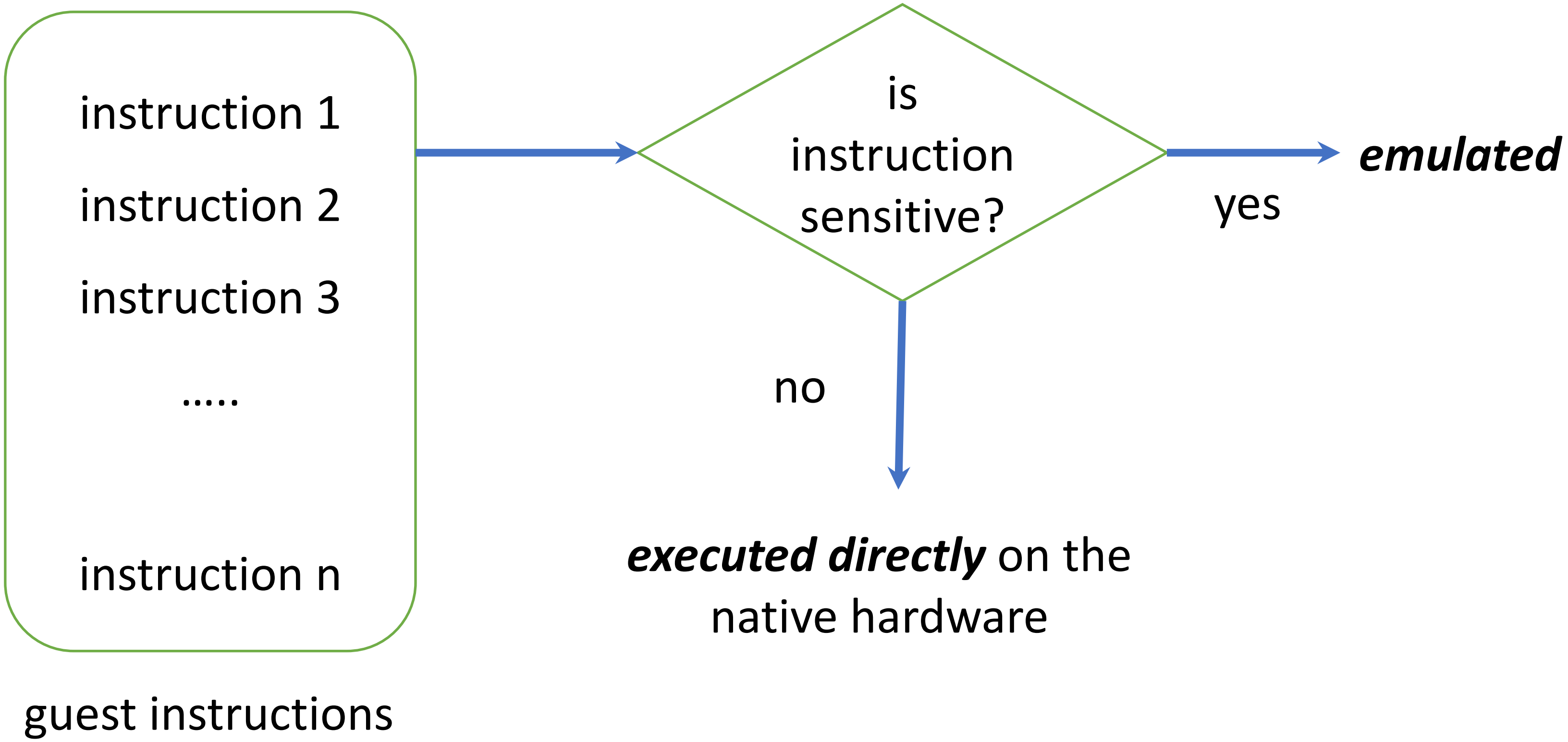
http://se.unl.edu/~witty/class/embedded/material/note/emulation.pdf

Interpretation versus Binary Translation (cont.)

	Implementation	Performance
Interpretation	simple and easy	low
Binary Translation	complex	***high initial*** translation cost, ***small execution*** cost

http://www.ittc.ku.edu/~kulkarni/teaching/EECS768/slides/chapter2.pdf

Direct Native Execution



Only if the ***ISA of the host is identical to the ISA of the guest***.

Hardware Virtualization Methods

Hardware Virtualization Methods

- **Full Virtualization**
  - Binary Translation
  - Hardware-assisted virtualization

- **Paravirtualization**

Full Virtualization

- Run a program **directly on top of a VM** and **without any modification**
  - The program thought it were run on the raw hardware.

- The principal advantage of full virtualization
  - Complete isolation → enhanced security
  - Ease of emulation of different architectures
  - Coexistence of different systems on the same platform.

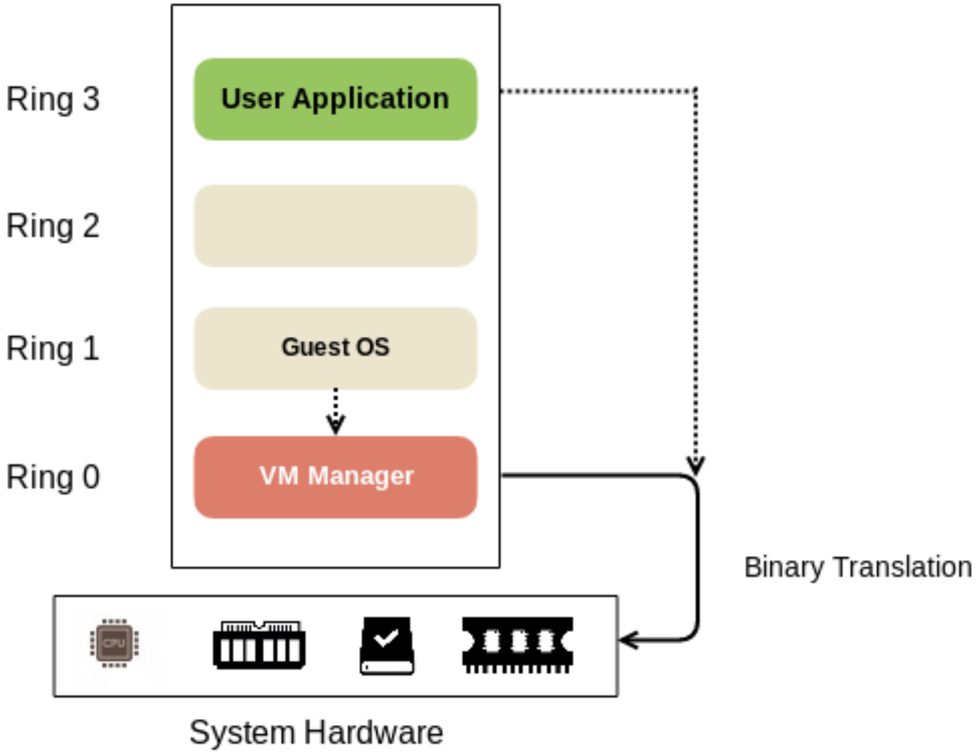
Full Virtualization (cont.)

- In some architectures, **some sensitive instructions are not privileged**
  - They cannot be virtualized in the classic way.
  - Like the non-hardware-assisted x86

- ***Two technologies:***
  - Binary translation
  - Hardware-assisted virtualization

Binary Translation

- Replaces the sensitive instructions that **do not generate traps** with a **trap into the VMM** to be **emulated in software**.



https://www.oreilly.com/library/view/enterprise-cloud-security/9781788299558/e301e0ae-2518-4b8f-82c3-b073b4ee8732.xhtml

Binary Translation (cont.)

- **Static Binary Translation**
  - On a full program
- **Dynamic Binary Translation**
  - Introduces an additional overhead.

Dynamic Binary Translation

- It is usually performed in small units called "***basic blocks***”.
- A basic block is a set of instructions that ends with a branch instruction but does not have any branch instructions inside.
  - Be executed start to finish by a CPU
  - An ideal unit for translation
- The translations of the basic blocks are ***cached***
  - Overhead of translating only happens the first time a block is executed.

https://blogs.oracle.com/ravello/nested-virtualization-with-binary-translation

Static vs Binary Translation

	Input type	Granularity	Translation time
Static binary translation	Binary program	Full program	Before running program
Dynamic binary translation	Binary program	Basic block	At runtime