Welcome to System Programming and Security

Dr. Carlos Moreno Garcia

c.moreno-garcia@rgu.ac.uk

@CarlosFMorenoG

Introduction to the Module

Aims of the module

- To review the design, implementation and functioning of operating systems.
- To provide the student with the ability to proficiently manage the resources provided by operating systems.
- To enable the student to deploy secure infrastructure using common operating systems.

Module Organisation

Activity	Details	Contact Hours
Lecture	Main principles and theory	1 hour per week
Labs	Practical techniques	2 hours per week
Directed Study	Reading and coursework preparation	3 hours per week
Private Study		3 hours per week

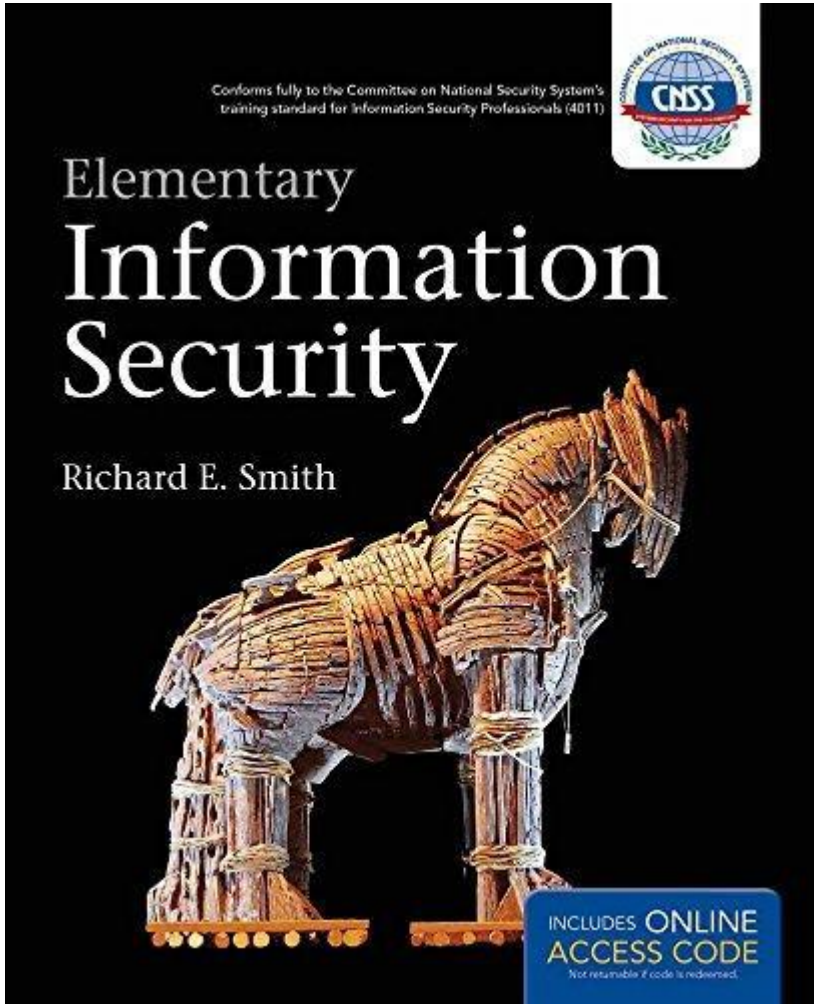
Course Assessment

- Coursework
  - Assignment (handed out later in the semester).
- Exam
  - Closed book exam at the end of the semester.
- Marking
  - 50 : 50 (Course Work : Exam)
- Moodle Quizzes (optional)
  - Used to check your understanding of topics throughout the semester.

Course Book (Weeks 2 - 5)

- ELEMENTARY INFORMATION SECURITY. Richard E. Smith. Jones & Bartlett Learning, 2011. ISBN 10: 1449648207 / ISBN 13: 9781449648206.

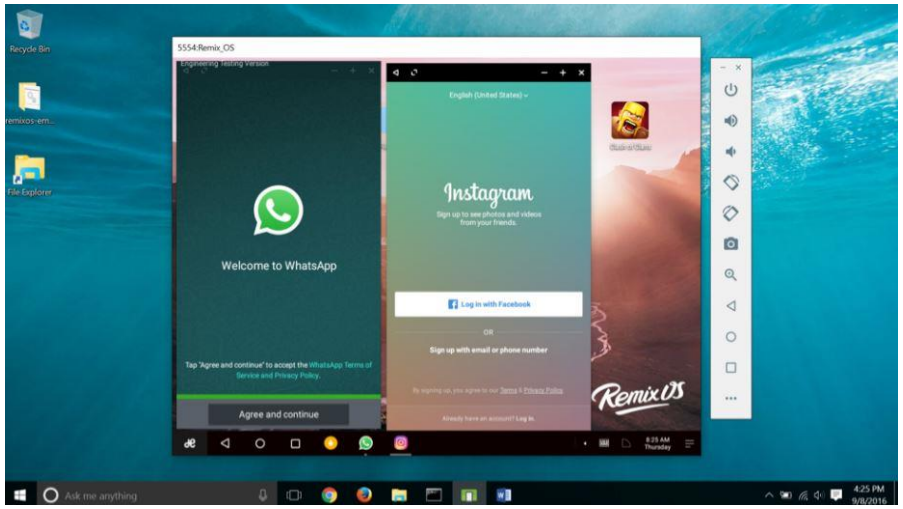
- Available through Amazon or at RGU‘s library.



Today’s Plan

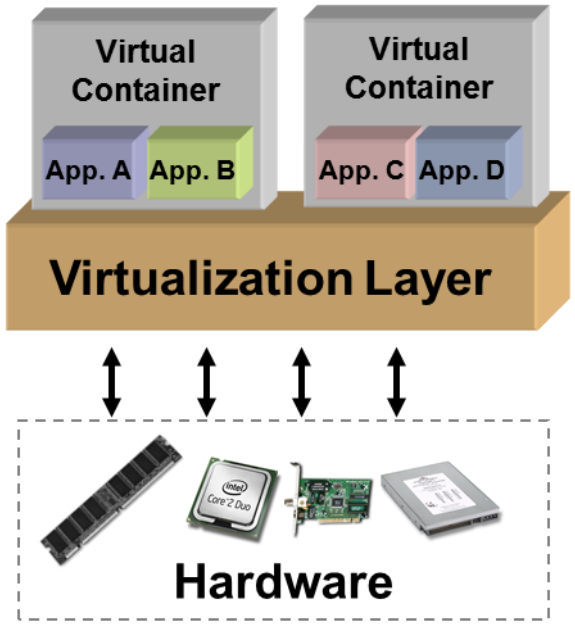
- What is virtualisation?
- Techniques to virtualise the x86 architecture
- Other uses of virtualisation
- Virtual Machines
- Lab: Creating a virtual machine

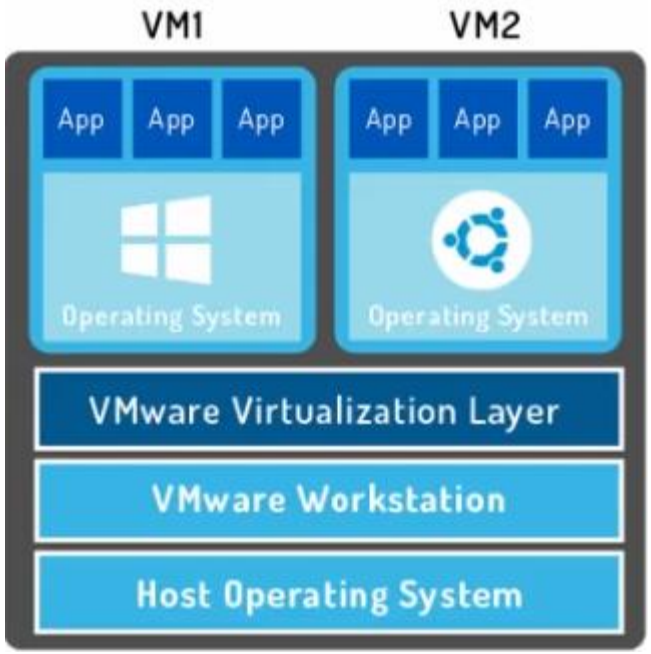
Virtualisation



Adding a layer… but where?

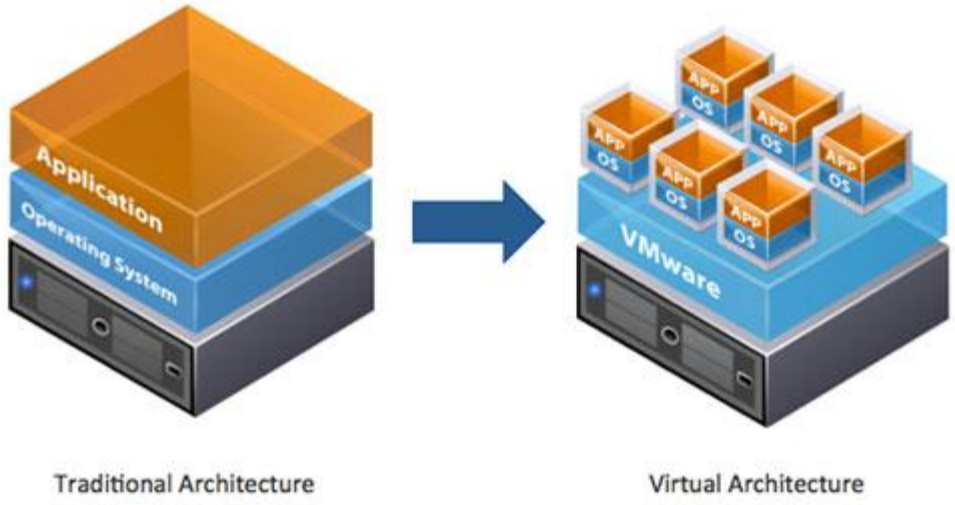
"Creation of a virtualisation layer which maps interfaces onto the resources of a real system".



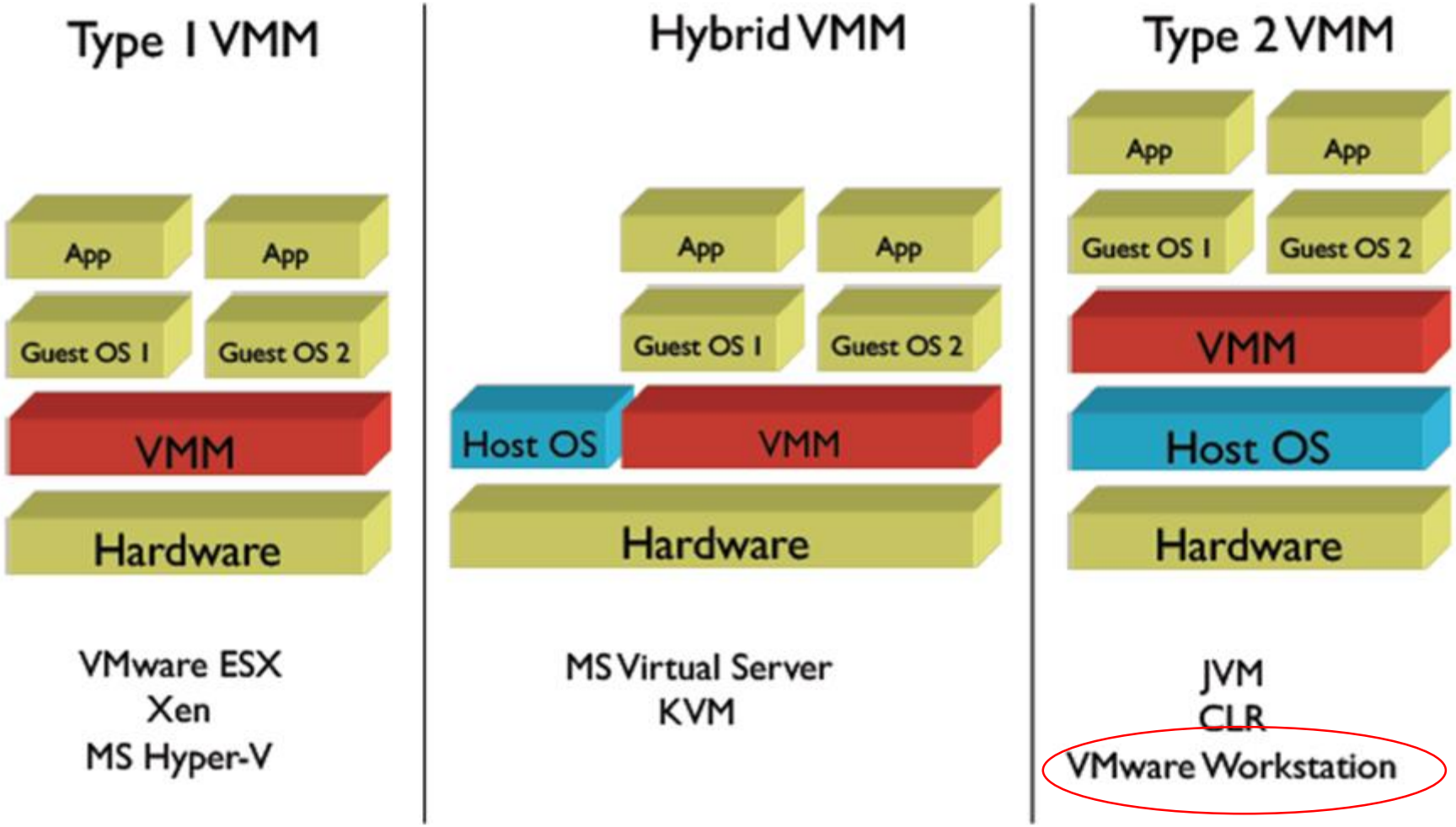


Why Hypervisor?

"Implementation of a *Virtual Machine Manager* (VMM), also called ***Hypervisor***".



Types of Virtual Machine Managers/Hypervisors



CSE443 (Spring 2012). Introduction to Computer and Network Security. Professor Jaeger. www.cse.psu.edu/~tjaeger/cse443-s12/

Benefits of virtualisation?

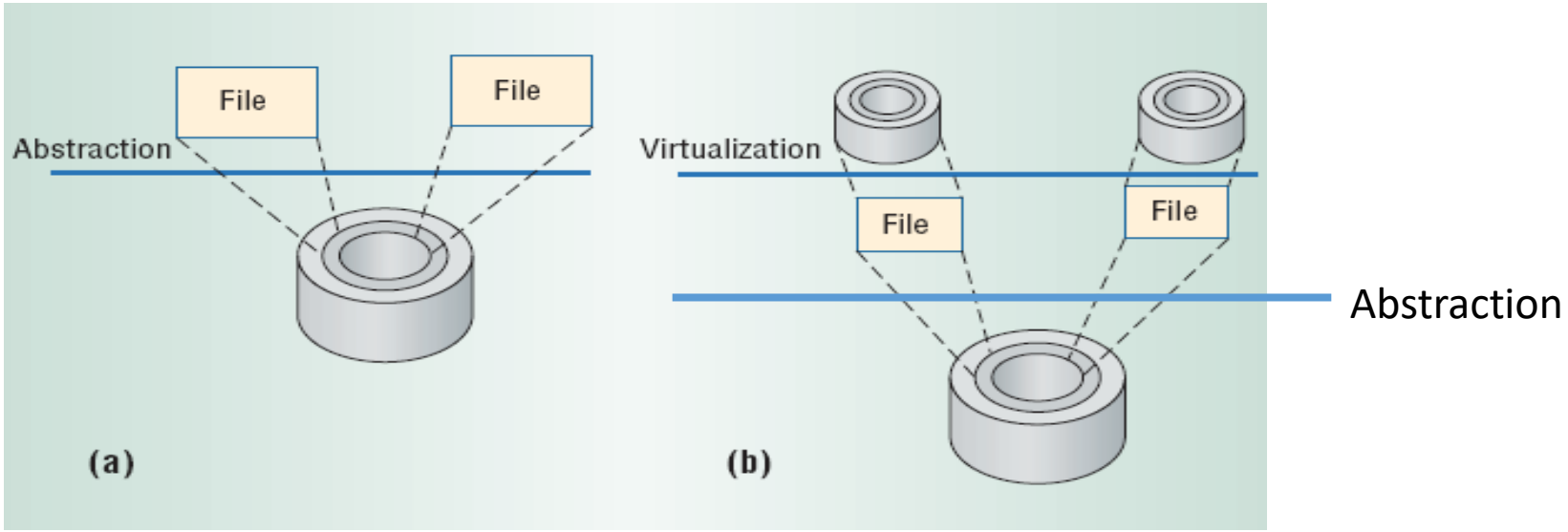
- **Abstraction**
- Replication
- Isolation
- Cross compatibility
- Efficiency of resources
- Recovery and Backup
- Legacy SW usage
- Testing
- Save £££

Disadvantages?

- More complexity
- Single physical point of failure
- Spend more £££

Abstraction vs Virtualisation

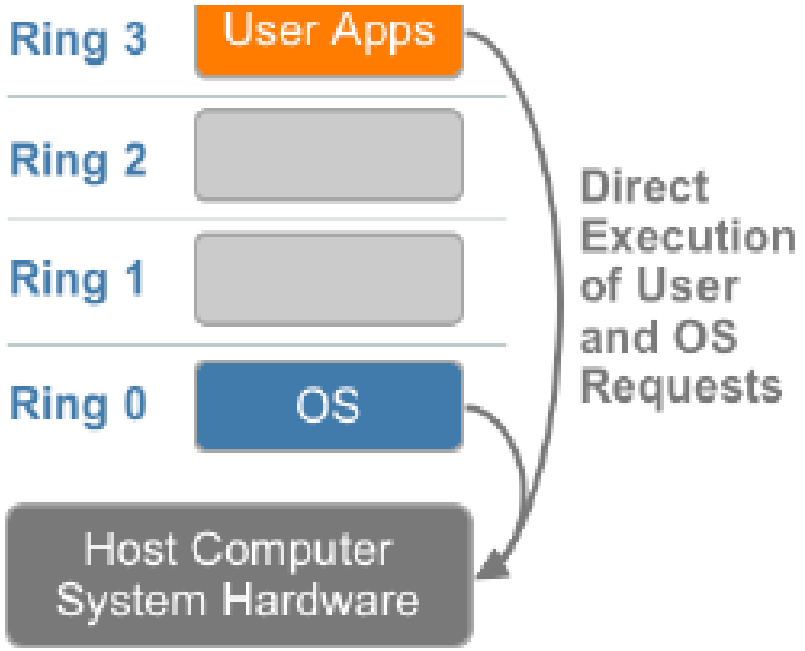
- A classical computer architecture benefits from *abstraction*.
  - Well-defined interfaces for HW and SW to use.
  - Limits are based on the HW implementation.
- In virtualisation, interfaces and resources may be mapped to different architectures.



Techniques to virtualise the x86 architecture

Requirements of x86 virtualisation

- The OS needs direct access to memory and HW, thus it runs in Ring 0 (kernel).
- The Rings 1 and 2 are usually for device drivers.
- User level applications run in Ring 3.
- Virtualisation of the x86 architecture required placing a virtualisation layer **under** the OS to create and manage the VMs.
- VMware resolved the challenge in 1998, developing **binary translation techniques**, also known as **full virtualisation**.



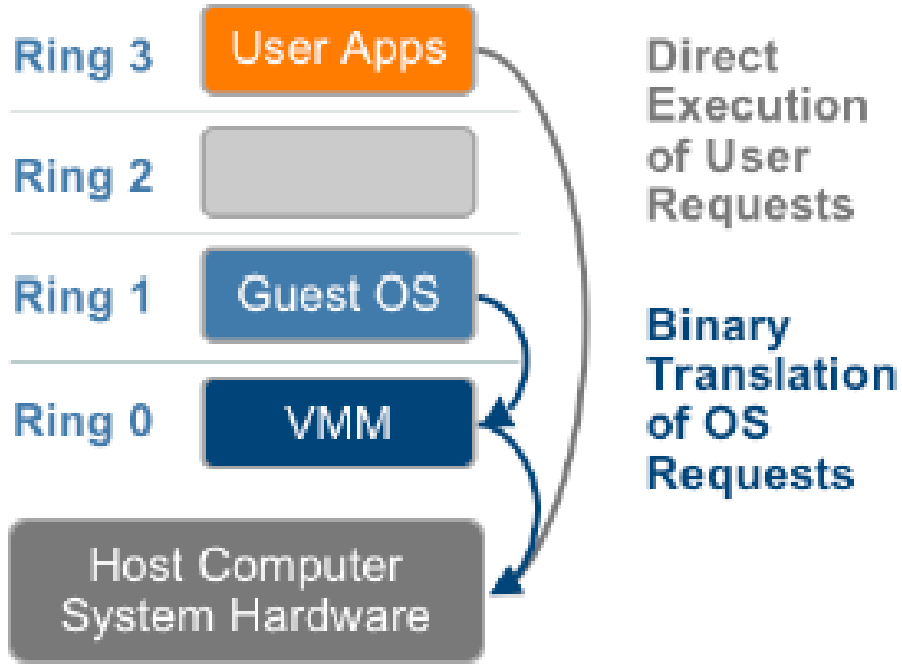
Which type of virtualisation does this resemble the most?

Full virtualisation

Full virtualisation using binary translation (VMWare)

- The hypervisor occupies Ring 0 and translates kernel code into Ring 3 instructions, replacing nonvirtualisable code with new sequences of instructions that have effect on the **virtual hardware**.

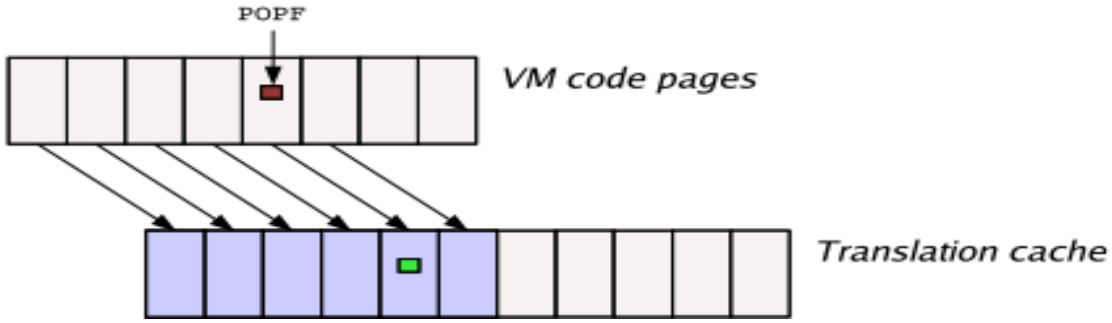
- The Guest OS is not aware that it is being virtualised, and thus requires **no modification**.



Which type of virtualisation does this resemble the most?

Full virtualisation using binary translation (VMWare)

- Code running in each VM is scanned “on-the-fly” for nonvirtualisable instructions and is rewritten into a safer form.

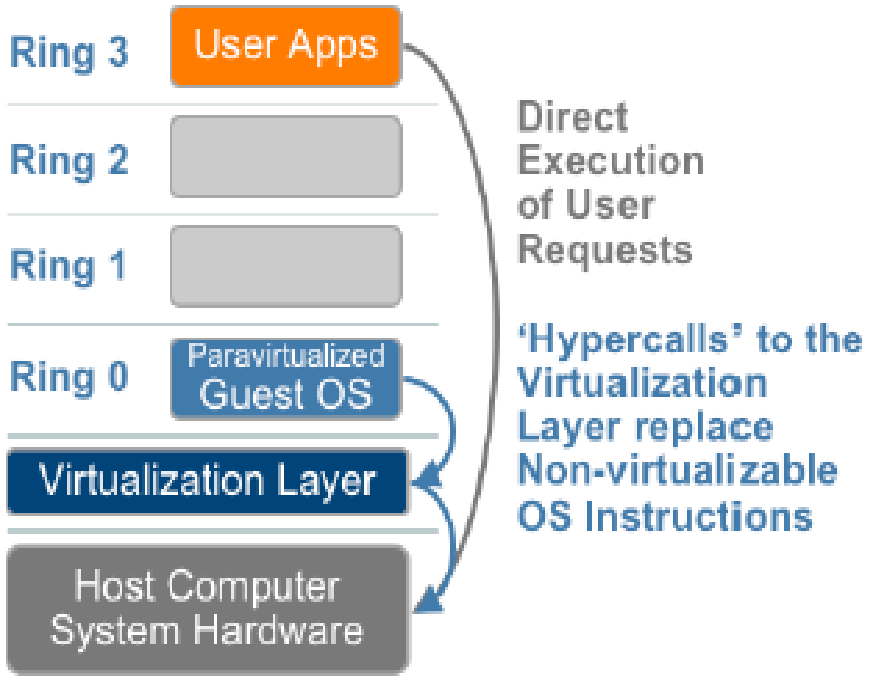


- Directly inline VMM functionality into the VM code – e.g., fast versions of I/O processing code.
- Maintains a translation cache of recently-rewritten code pages, avoiding high overhead for rescanning and rewriting on each execution pass.

Para-virtualisation/OS assisted virtualisation

Para-virtualisation/OS assisted virtualisation

- Modifying the OS kernel to replace nonvirtualisable instructions with **hyper calls** that communicate directly with the virtualisation layer.
- Para-virtualisation is different from full virtualisation, since OS knows it is being virtualized and sensitive OS calls are not trapped using binary translation.
- The performance advantage of para-virtualisation over full virtualisation can vary greatly depending on the workload.



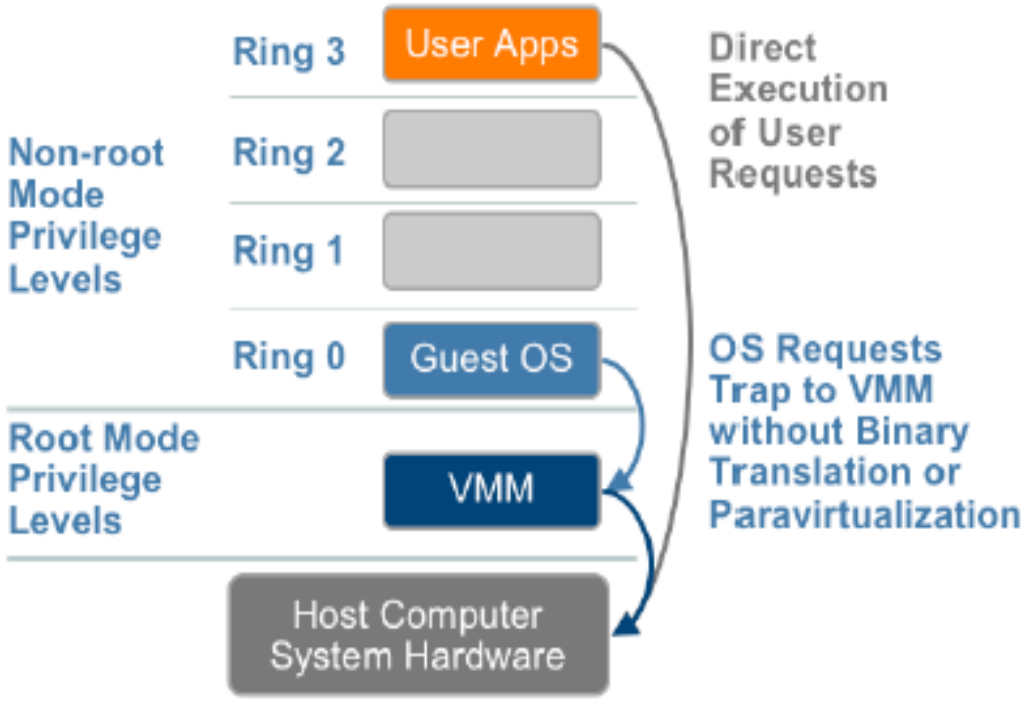
Para-virtualisation/OS assisted virtualisation

- Para-virtualisation cannot support unmodified OS (e.g. Windows 2000/XP).
- Examples: XEN virtualises the processor and memory using a modified Linux kernel and virtualizes the I/O using custom guest OS device drivers.
- Advantage: Easier to modify the host than to implement full virtualisation.

Hardware assisted virtualisation

Hardware assisted virtualisation

- In the mid 2000's, AMD and Intel made processors that support VMs (AMD-V and Intel VT-X), allowing a new CPU execution mode feature that allows the VMM to run in a new root mode below Ring 0.
- Privileged and sensitive calls are set to automatically "trap" to the hypervisor, removing the need for either binary translation or paravirtualisation.



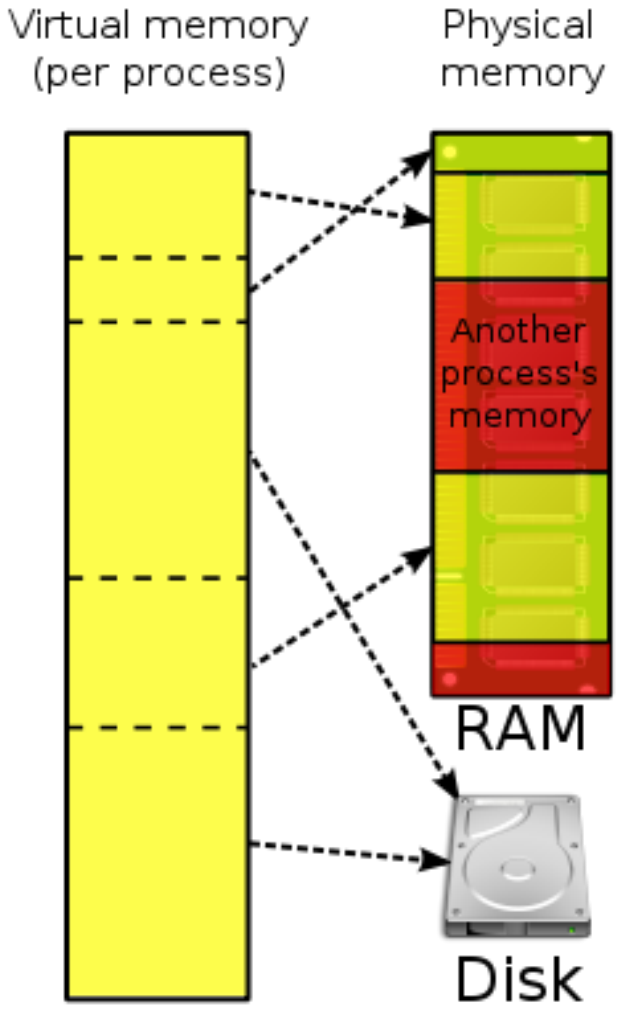
Relation between types of VMM and Techniques for x86 virtualisation

- The relation between both categorisations is quite strange!
- Full virtualisation seems Type 1 VMM, but in practice is simulated.
- An example of para-virtualisation is XEN, which is formally categorised as Type 1 VMM but in theory should be Type Hybrid.
- Hardware assisted techniques seem Type 1 VMM, but in practice VMWare Workstation is Type 2 VMM.
- Moreover, some literature mentions “OS assisted virtualisation” (i.e. FreeBSD) which resembles Type Hybrid.

Other uses of virtualisation

Memory virtualisation

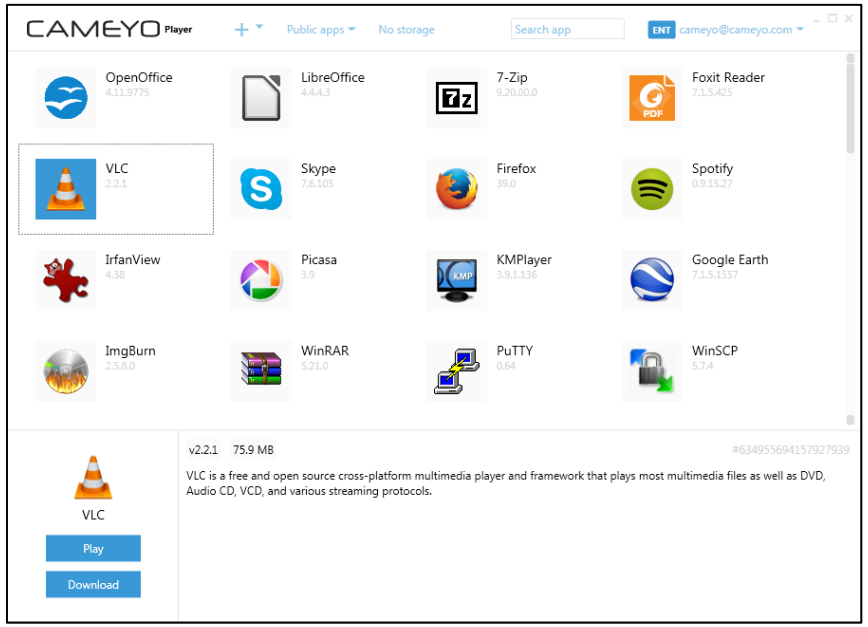
- The OS keeps **mappings** of virtual page numbers to physical page numbers stored in page tables.
- The guest OS continues to control the mapping of virtual addresses to the guest memory physical addresses, but the guest OS cannot have direct access to the actual machine memory.
- The hypervisor becomes responsible for mapping guest physical memory to the actual machine memory.
- Memory virtualisation in a single computer is called **virtual memory**.
- All modern x86 CPUs include a memory management unit (MMU) and a translation look-aside buffer (TLB) to translate virtual and physical memory access.



https://en.wikipedia.org/wiki/Virtual_memory

Application virtualisation

- Encapsulate a computer program from the OS in which it is executed.
- Sandbox for legacy applications.
- Allows applications to run in non-native OS in a “portable” way.
- The VM is created when the process is initiated, and is destroyed when the application is exited.
- Examples: Cameyo, Ceedo, Citrix XenApp.



https://www.cameyo.com/cameyo-offline/

http://bretty.me.uk/citrix-slow-web-interface-5-4/

Virtual Machines

- OS is abstract to the physical hardware. Each VM has its own…
  - “virtual CPU” (Thread)
  - “virtual memory” (Process and Address Space)
  - “virtual disk” (Filesystems)
- Programs in the VM have to use system calls to request services from the OS.
- Two types:
  - Process Virtual Machines (application virtualisation).
  - **System Virtual Machines** (VM created by VMWare).

M. Sudha, G. Harish, A. Nandan, and J. Usha, “Performance Analysis of Kernel-Based Virtual Machine,” *International Journal of Computer Science & Information Technology (IJCSIT)*, vol. 5, no. 1, pp. 137–144, 2013.

Lab 1: Creating a Linux Virtual Machine using VMware