Virtualization and the Cloud

Virtualization

- » Dates to the 1960's
- » VMM (Virtual Machine Monitor) creates the illusion of multiple (virtual) machines on the same physical hardware.
  - – Also known as a hypervisor
    - • Type 1 hypervisors run on the bare metal
    - • Type 2 hypervisors that may make use of an underlying operating system.
- – virtualization allows a single computer to host multiple virtual machines

Virtualization

- » Advantage of this approach is that a failure in one virtual machine does not bring down any others.
  - – Strong Isolation
- » BUT, If the server running all the virtual machines fails, the result is even more catastrophic than the crashing of a single dedicated server.
- » Only software running in the highest privilege mode is the hypervisor
  - – A couple orders of magnitude less lines of code than an OS

Virtualization Advantages

- Checkpointing
- Migrating virtual machines (e.g., for load balancing across multiple servers) is much easier than migrating processes running on a normal operating system.
  - Just move memory and disk images
- Cloud

History

- 1960s IBM experimented with two independently developed hypervisors:
  - SIMMON
  - CP-40
    - Reimplemented as CP-67 to form the control program of CP/CMS, a virtual machine operating system for the IBM System/360 Model 67
    - Reimplemented again and released as VM/370 for the System/370 series in 1972

History

- » 1974 two computer scientists at UCLA, Gerald Popek and Robert Goldberg, published“Formal Requirements for Virtualizable Third Generation Architectures”
  - – listed exactly what conditions a computer architecture should satisfy in order to support virtualization efficiently
- » 1990 researchers at Stanford developed Disco hypervisor
  - » Left to form VMWare
    - » Binary Translation

Requirements for Virtualization

- Safety: hypervisor should have full control of virtualized resources.
- Fidelity: behavior of a program on a virtual machine should be identical to same program running on bare hardware.
- Efficiency: much of code in virtual machine should run without intervention by hypervisor.

Safety

- » Execute each instruction in an interpreter
  - – Bochs
  - – Cannot allow the guest operating system to disable interrupts for the entire machine or modify the page-table mappings.
    - Make the OS think it has done that
  - – Performance sucks.
    - VMMs try to execute most code natively

Fidelity

- Virtualization has long been a problem on the x86 architecture due to defects in the Intel 386 architecture
  - Carried forward into new CPUs for 20 years in the name of backward compatibility.
    - Sensitive instructions
      - Instructions that do I/O, change the MMU settings
    - Privileged instructions
      - Instructions that cause a trap if executed in user mode

Fidelity

- » A machine is virtualizable only if the sensitive instructions are a subset of the privileged instructions
- » Some sensitive 386 instructions were ignored if executed in user mode or executed with different behavior.
  - – POPF instruction replaces the flags register, which changes the bit that enables/disables interrupts.
    - In user mode, it was ignored.

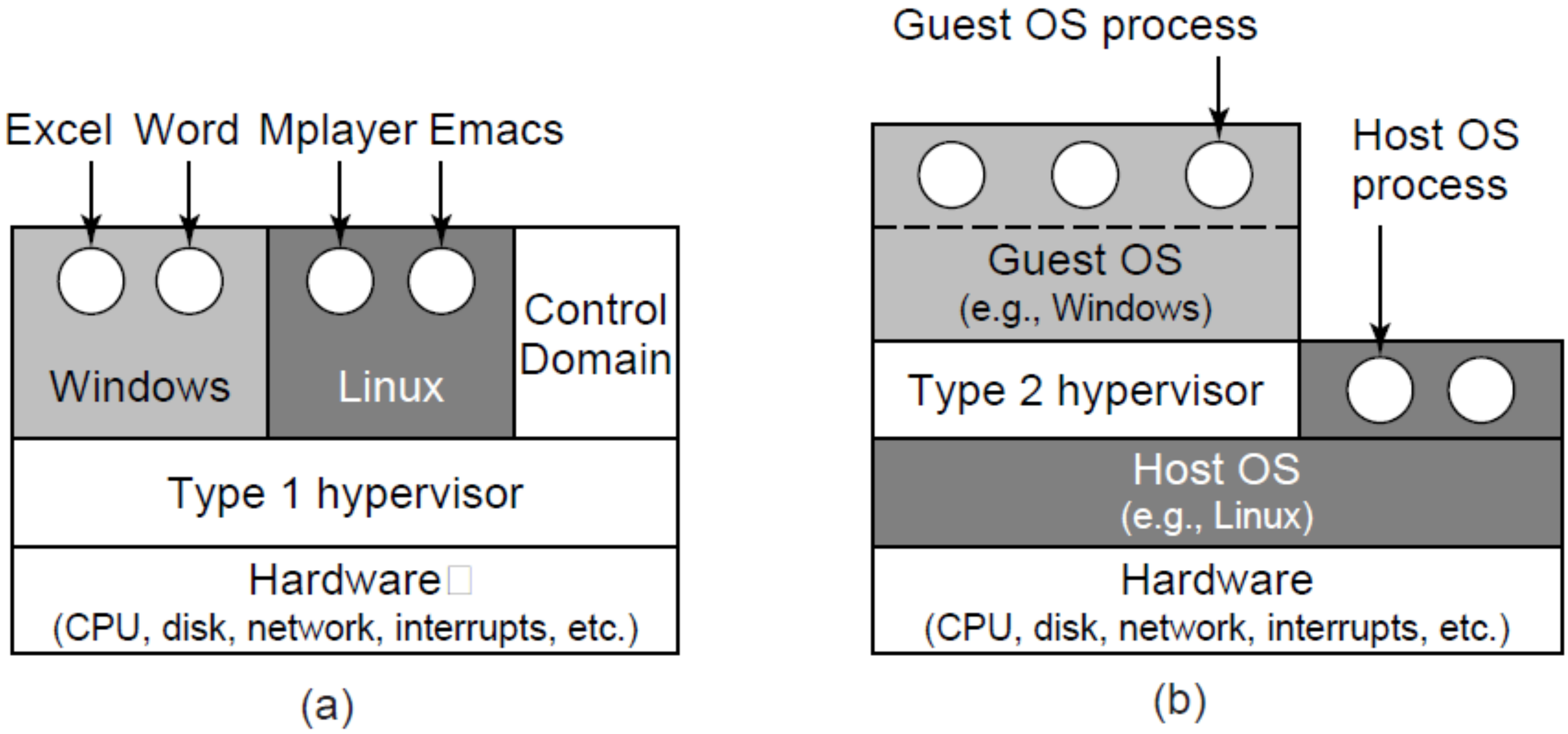
Fidelity

- » 2005 Intel released VT (Virtualization Technology); AMD called it SVM (Secure Virtual Machine).
- » When a guest operating system is started up in a VT container, it continues to run there until it causes an exception and traps to the hypervisor
- » The set of operations that trap is controlled by a hardware bitmap set by the hypervisor.
  - – Classical trap-and-emulate becomes possible.

Paravirtualization

- Never even aims to present a virtual machine that looks just like the actual underlying hardware.
- Presents a machine-like software interface that explicitly exposes the fact that it is a virtualized environment.
  - Offers a set of hypercalls, which allow the guest to send explicit requests to the hypervisor
  - Guests use hypercalls for privileged sensitive operations like updating the page tables

Type 1 and Type 2

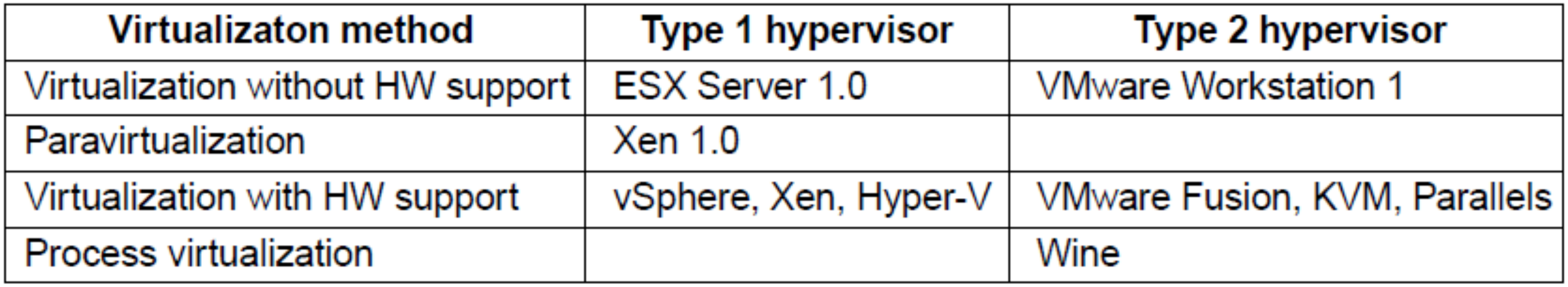


Location of type 1 and type 2 hypervisors.

Type 1 and Type 2

- Type 1 hypervisor - only program running in ring 0
- Type 2 hypervisor relies on host OS to allocate and schedule resources, very much like a regular process.
  - Also called hosted hypervisors
- Guest Operating System: operating system running on top of the hypervisor
- Host Operating System: operating system running on the hardware

Type 1 and Type 2

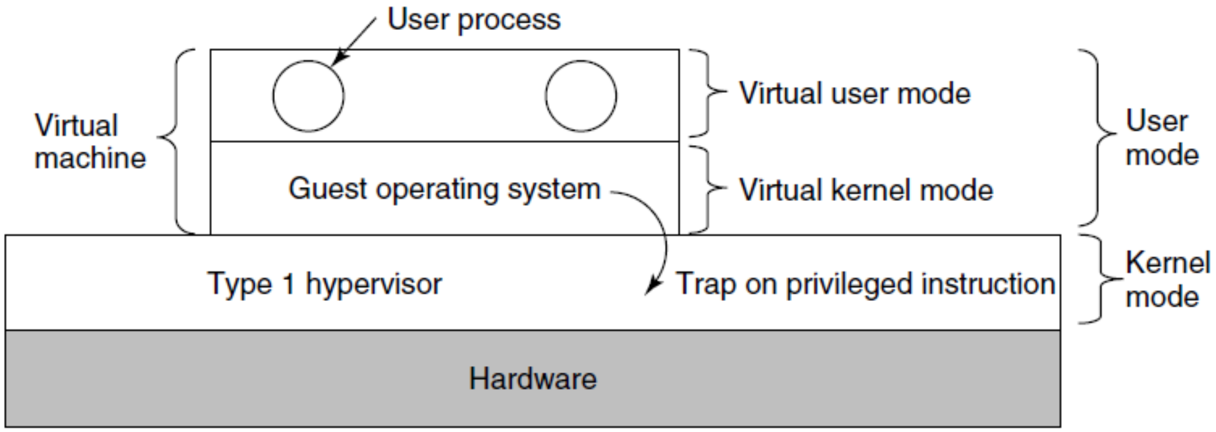


Virtualizaton method	Type 1 hypervisor	Type 2 hypervisor
Virtualization without HW support	ESX Server 1.0	VMware Workstation 1
Paravirtualization	Xen 1.0	
Virtualization with HW support	vSphere, Xen, Hyper-V	VMware Fusion, KVM, Parallels
Process virtualization		Wine

Type 1 hypervisors always run on the bare metal

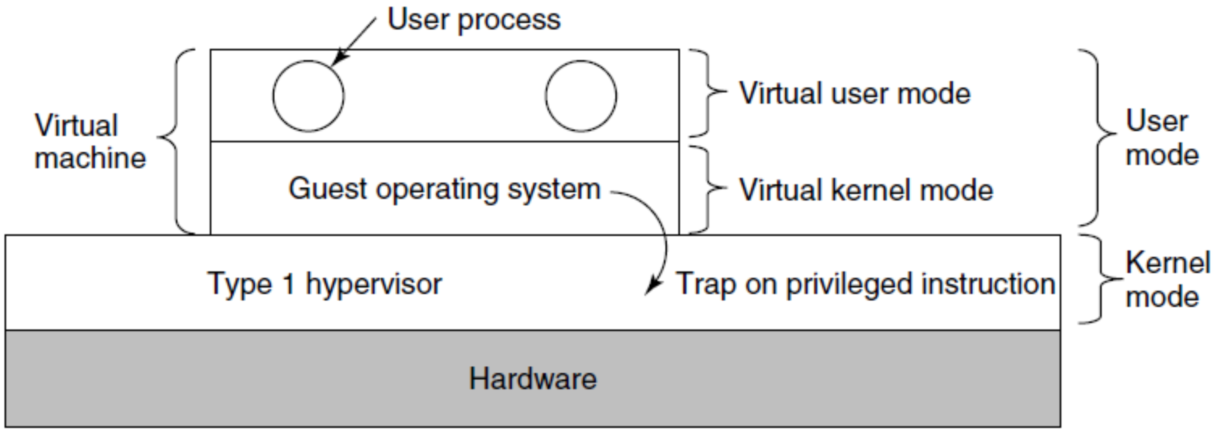
Type 2 hypervisors use the services of an existing host operating system.

Techniques for Efficient Virtualization



- Virtual machine runs as a user process in user mode, and not allowed to execute sensitive instructions
- Virtual machine runs a guest operating system that thinks it is in kernel mode
  - It is not.
  - Virtual kernel mode.

Techniques for Efficient Virtualization

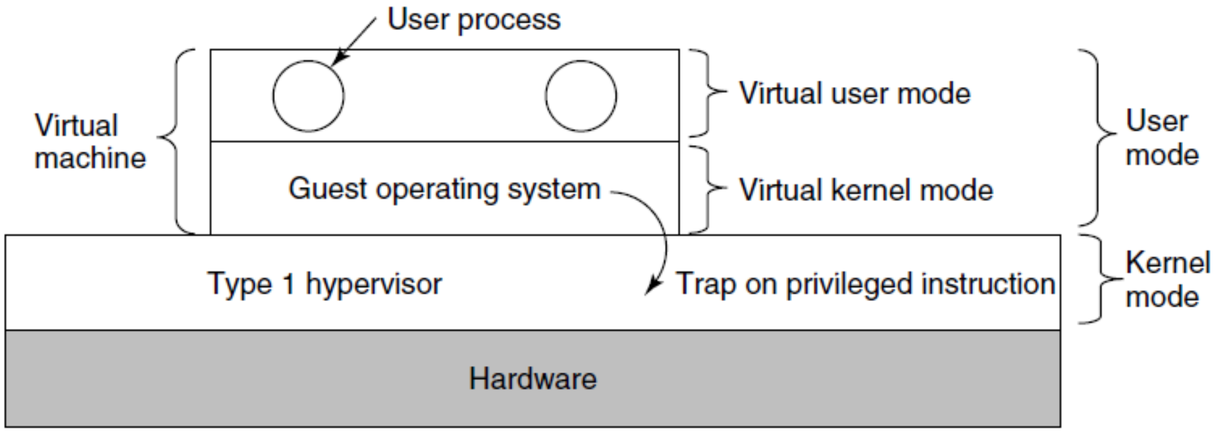


What happens when the guest operating system executes an instruction that is allowed only when the CPU really is in kernel mode?

On CPUs without VT, the instruction fails and the operating system crashes.

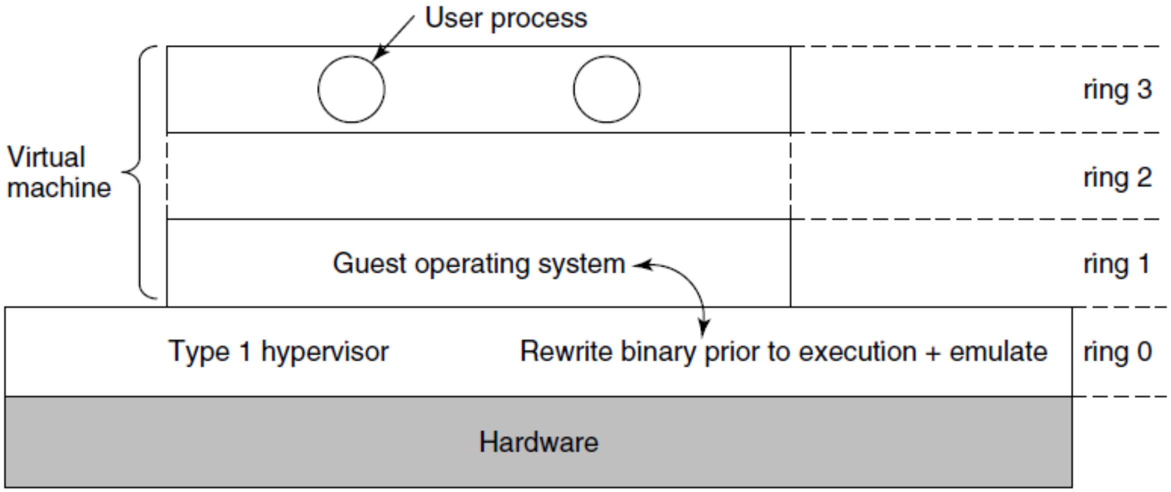
On CPUs with VT, when the guest operating system executes a sensitive instruction, a trap to the hypervisor occurs

Techniques for Efficient Virtualization



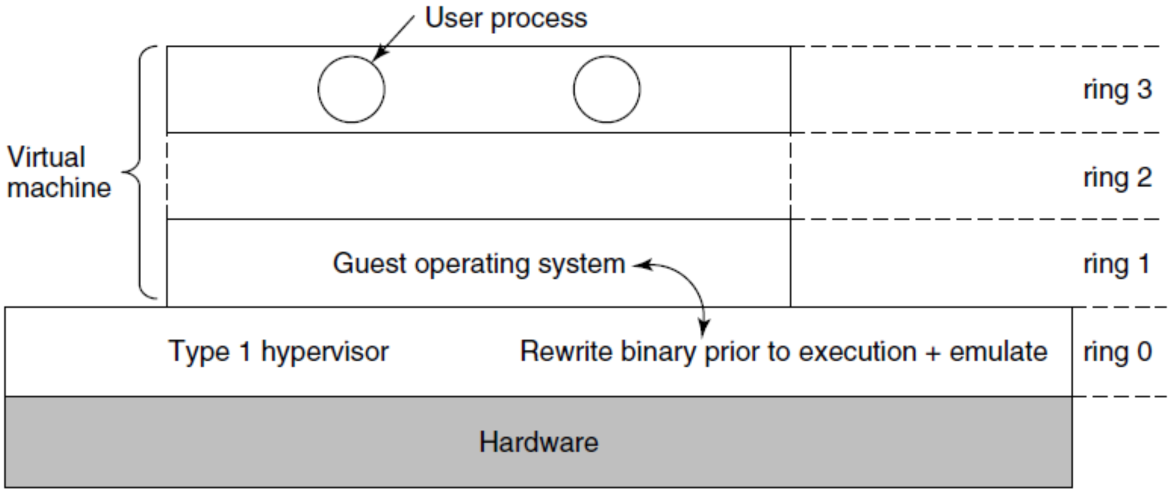
- The hypervisor can then inspect the instruction to see if it was issued by the guest operating system in the virtual machine or by a user program in the virtual machine.
    - In the former case, it arranges for the instruction to be carried out;
    - In the latter case, it emulates what the real hardware would do when confronted with a sensitive instruction executed in

Virtualizing the Unvirtualizable



- Virtualizing with VT is straight forward.
- Pre VT:
  - Binary translation
  - Use rings 1 and 2

Virtualizing the Unvirtualizable



- The kernel is privileged relative to the user processes and any attempt to access kernel memory from a user program leads to an access violation.

- At the same time, the guest operating system's privileged instructions trap to the hypervisor. The hypervisor does some sanity checks and then performs the instructions on the guest's behalf.

Binary Translation

- Basic block: a short, straight-line sequence of instructions that ends with a branch.
  - By definition, a basic block contains no jump, call, trap, return, or other instruction that alters the flow of control, except for the very last instruction
- Prior to executing a basic block, the hypervisor first scans it to see if it contains sensitive instructions and replaces them with a call to a hypervisor procedure that handles them
  - Most code blocks don’t contain sensitive instructions

World Switch

- » Going from a hardware configuration for the host kernel to a configuration for the guest operating system is known as a world switch
  - – Interrupts in the guest move the guest kernel mode.
  - – Guest kernel expects to be the only kernel in kernel space

Cost of Virtualization

- Trap-and-emulate approach used by VT hardware generates a lot of traps, and traps are very expensive on modern hardware
  - Ruin CPU caches, TLBs, and branch prediction tables internal to the CPU.
  - When sensitive instructions are replaced by calls to hypervisor procedures within the executing process, none of this context-switching overhead is incurred

Cost of Virtualization

- The translated code itself may be either slower or faster than the original code.
  - CLI (Clear Interrupts Instruction)
    - Does not mean the hypervisor should turn them off.
    - Dedicated IF (Interrupt Flag) in the virtual CPU data structure per guest OS.
- Guest operating system modifies its page tables
  - Not cheap.