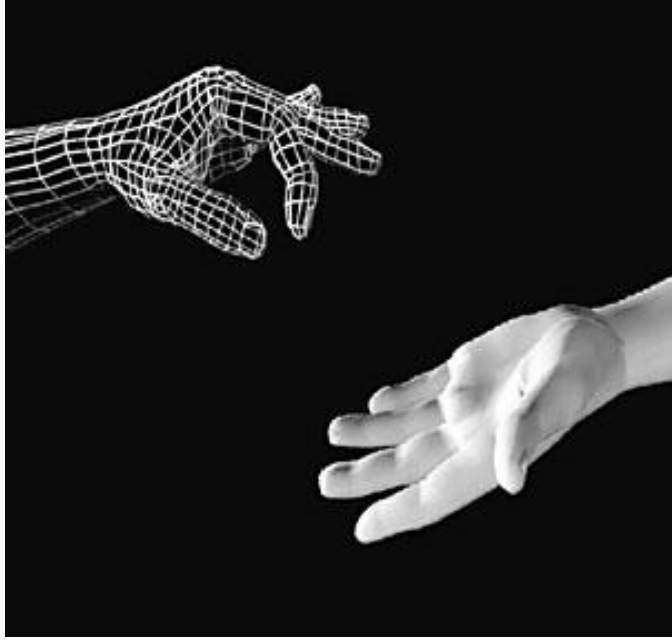
ENCE360 Operating Systems

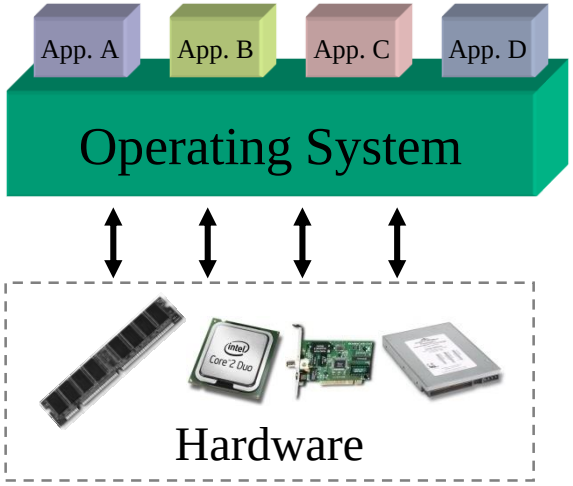


Virtualization

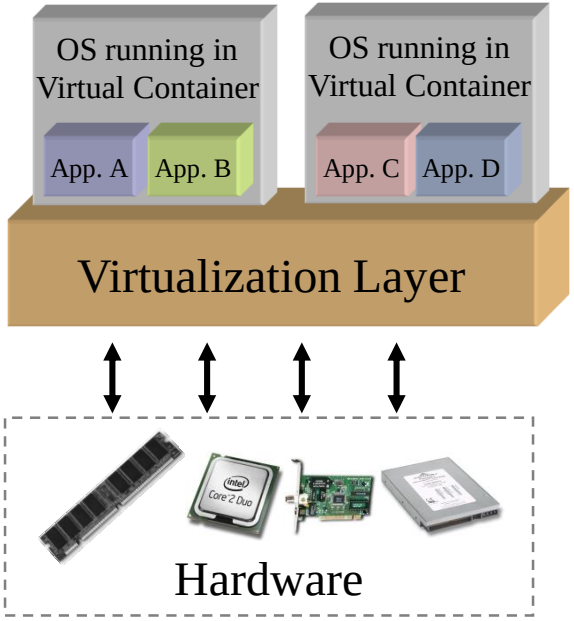
What is virtualization?

Virtualization is a broad term (virtual memory, storage, network, etc)

Virtualization basically allows one computer to do the job of multiple computers, by sharing the resources of a single hardware across multiple operating systems



'Nonvirtualized' system
A single OS controls all hardware platform resources



Virtualized system
It makes it possible to run multiple Virtual Containers on a single physical platform

Virtualization: Why?

- Server consolidation
- Application Consolidation
- Sandboxing
- Multiple execution environments
- Virtual hardware
- Debugging
- Software migration (Mobility)
- Appliance (software)
- Testing/Quality Assurance

Definitions

- **Virtualization**
  - **A layer mapping its visible interface and resources onto the interface and resources of the underlying layer or system on which it is implemented**
  - **Purposes**
    - Abstraction – to simplify the use of the underlying resource (e.g., by removing details of the resource's structure)
    - Replication – to create multiple instances of the resource (e.g., to simplify management or allocation)
    - Isolation – to separate the uses which clients make of the underlying resources (e.g., to improve security)

- **Virtual Machine Monitor (VMM)**
  - **A virtualization system that partitions a single physical "machine" into multiple virtual machines.**
  - **Terminology**
    - Host – the machine and/or software on which the VMM is implemented
    - Guest – the OS which executes under the control of the VMM

Virtualization Requirements

Popek and Goldberg describe in their "Formal Requirements for Virtualizable Third Generation Architectures – 1974":

- Properties for a Virtual Machine Monitor
  - Equivalence: A program running under the VMM should exhibit a behavior essentially identical to that demonstrated when running on an equivalent machine directly.
  - Resource control: The VMM must be in complete control of the virtualized resources.
- Formal analysis described through 2 theorems:

  Theorem 1:

  For any conventional third generation computer, a VMM may be constructed if the set of sensitive instructions for that computer is a subset of the set of privileged instructions. This guarantees the resource control property. Non privileged instructions must instead be executed natively. The holding of the equivalence property also follows.

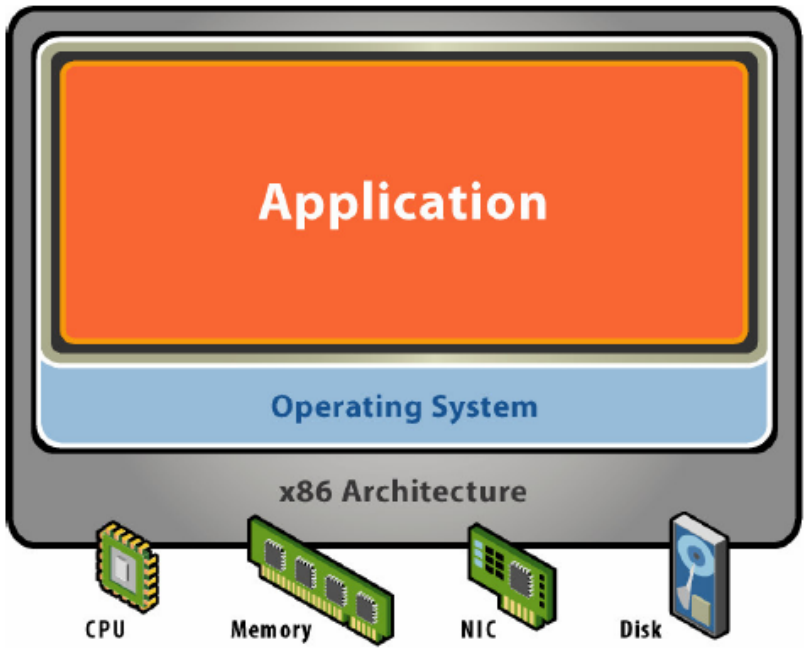
  Theorem 2:

  A conventional third generation computer is recursively virtualizable if it is virtualizable and a VMM without any timing dependencies can be constructed for it.

How did it start?

- Server virtualization has existed for several decades
  - Initially commercialised in 1970's by IBM's VM370 for specialized, proprietary, high-end server and mainframe systems
- By the 1990s, server virtualization instead used
  - Inexpensive x86 hardware platforms
  - Windows/Linux adopted as server Oss
- Now, we have:
  - VirtualBox
  - VMware suite
  - MS virtual PC, Hypervisor
  - QEMU (Quick Emulator): open source
  - XEN
  - OpenVZ
  - KVM, etc.

Starting Point: A Physical Machine



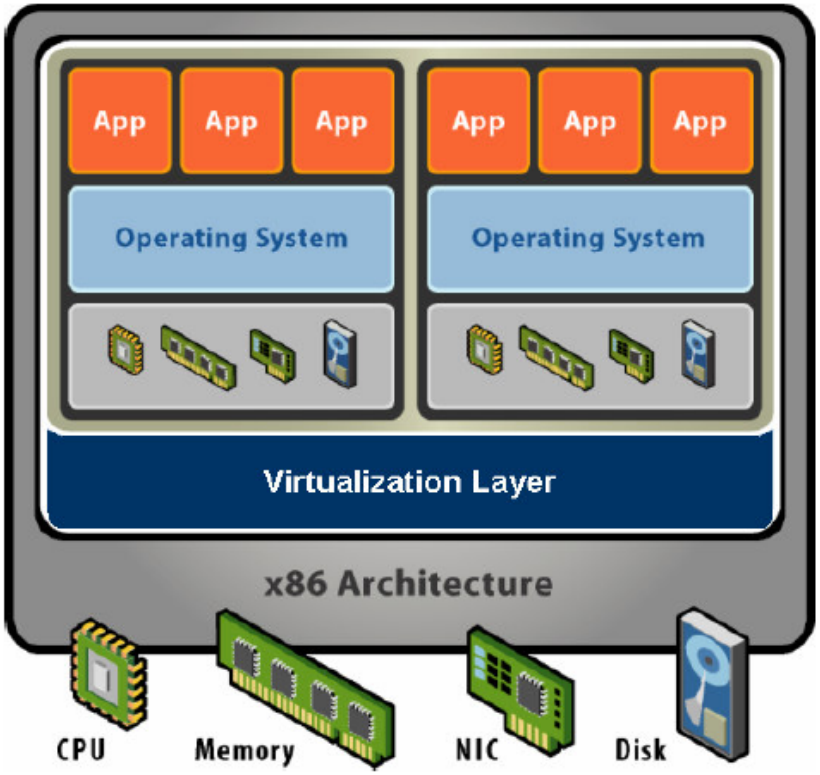
Physical Hardware

- Processors, memory, chipset, I/O bus and devices, etc.
- Physical resources often underutilized

Software

- Tightly coupled to hardware
- Single active OS image
- OS controls hardware

What is a Virtual Machine?



Hardware-Level Abstraction

- Virtual hardware: processors, memory, chipset, I/O devices, etc.
- Encapsulates all OS and application state

Virtualization Software

- Extra level of indirection decouples hardware and OS
- Multiplexes physical hardware across multiple “guest” VMs
- Strong isolation between VMs
- Manages physical resources, improves utilization

VM Isolation

Secure Multiplexing

- Run multiple VMs on single physical host
- Processor hardware isolates VMs, *e.g.* MMU

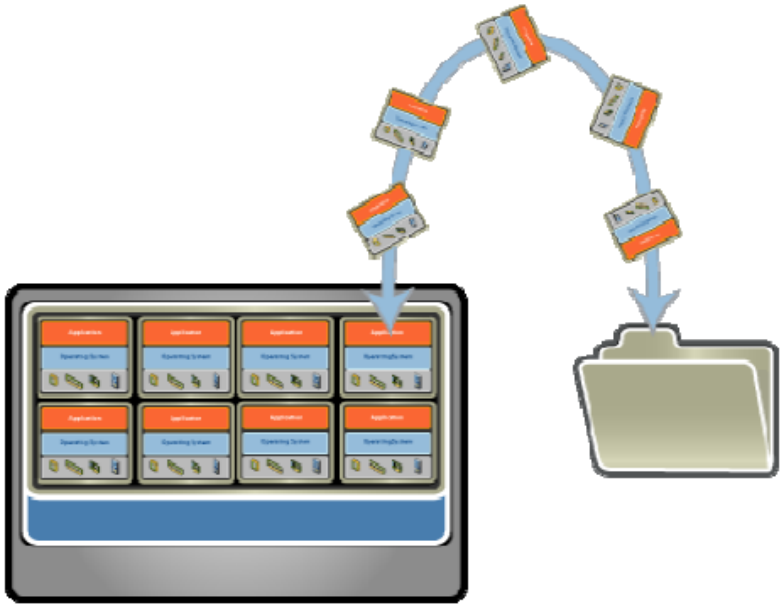
Strong Guarantees

- Software bugs, crashes, viruses within one VM cannot affect other VMs

Performance Isolation

- Partition system resources
- Example: VMware controls for reservation, limit, shares

VM Encapsulation



Entire VM is a File

- OS, applications, data
- Memory and device state

Snapshots and Clones

- Capture VM state on the fly and restore to point-in-time
- Rapid system provisioning, backup, remote mirroring

Easy Content Distribution

- Pre-configured apps, demos
- Virtual appliances

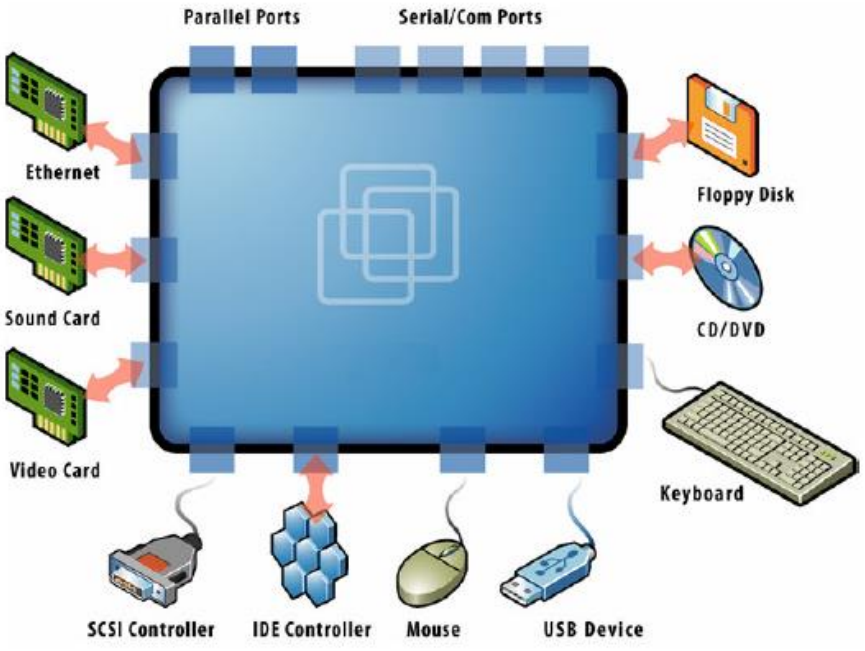
Snapshots & Migration

- *Snapshot:* freeze a copy of virtual machine
  - Identify all pages in disk files, VM memory
  - Use copy-on-write for any subsequent modifications
  - To revert, throw away the copy-on-write pages
- *Migration:* move a VM to another host
  - Take snapshot (fast)
  - Copy all pages of snapshot (not so fast)
  - Copy modified pages (fast)
  - Freeze virtual machine and copy VM memory
    - Very fast, fractions of a second

Cloning

- *Simple clone:*
  - Freeze virtual machine
  - Copy all files implementing it
  - Use copy-on-write to speed up
- *Linked clone:*
  - Take snapshot
  - Original and each clone is a copy-on-write version of snapshot

VM Compatibility



Hardware-Independent

- Physical hardware hidden by virtualization layer
- Standard virtual hardware exposed to VM

Create Once, Run Anywhere

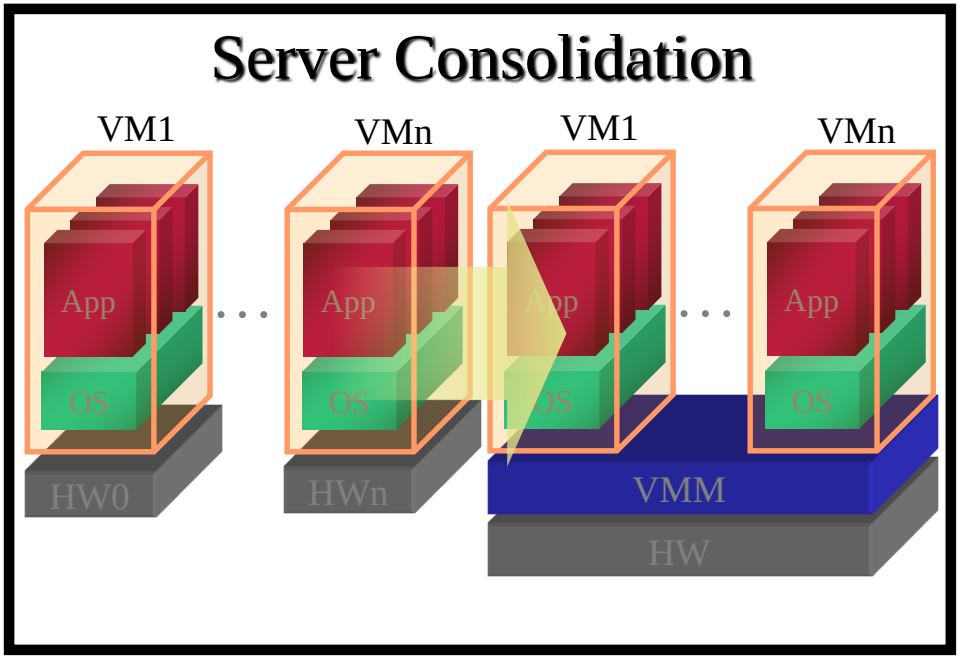
- No configuration issues
- Migrate VMs between hosts

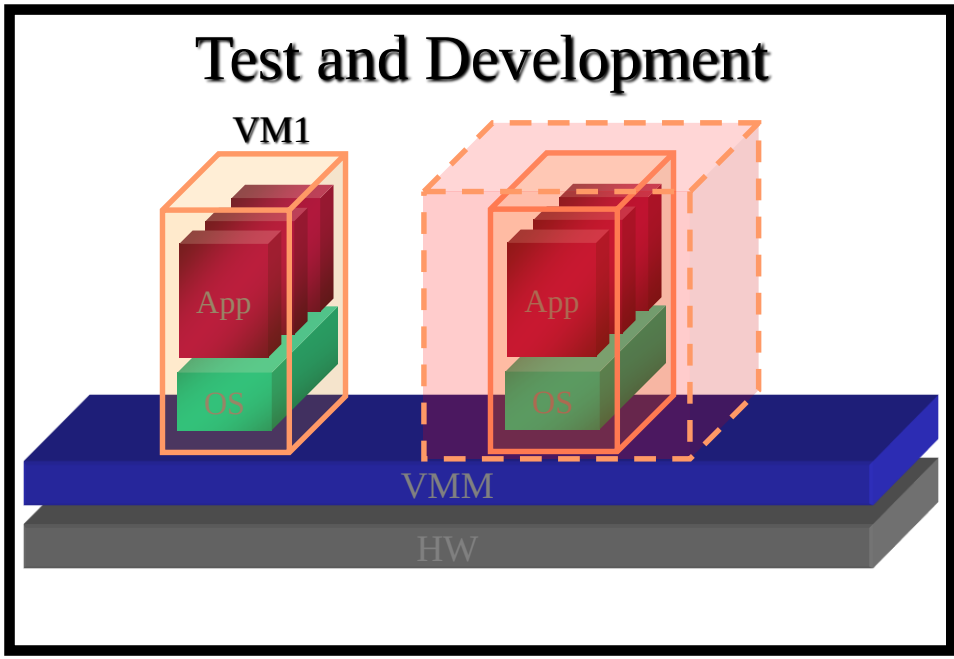
Legacy VMs

- Run ancient OS on new platform

The trajectory of virtualisation

Evolution from mainframes to x86 platforms



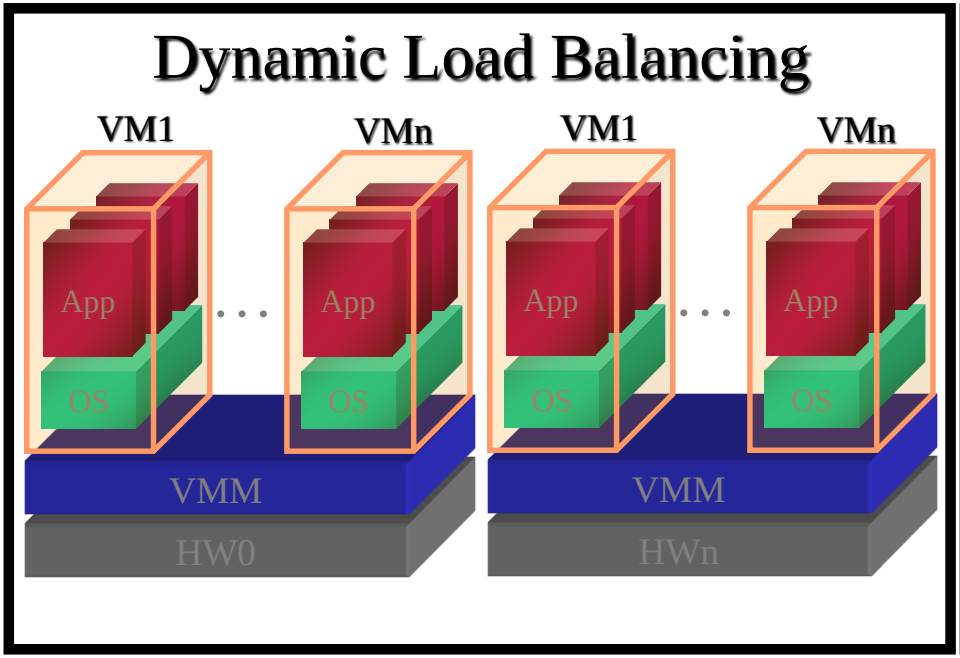


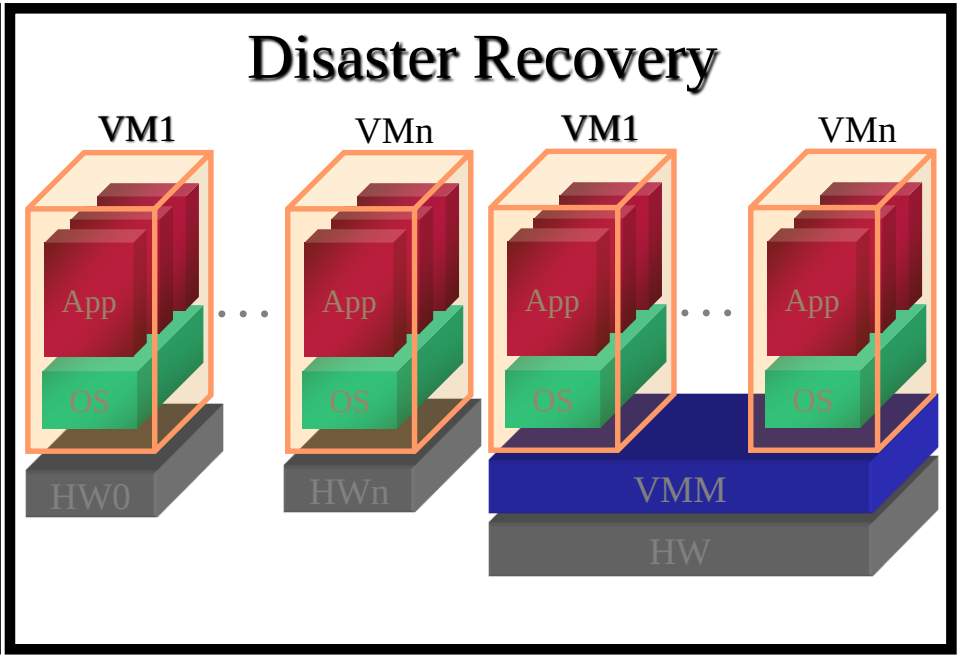
10:1 in many cases

Enables rapid deployment

Server Virtualization: 80% of companies are now using virtualization for consolidation

Emerging Usage Models





Goal: True “Lights Out” Datacenter

Instantaneous failover

Dynamic load balancing

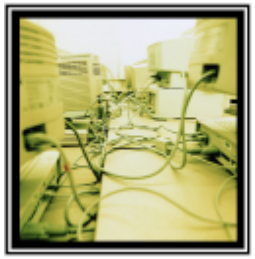
Autonomics

Self healing

Common Virtualization Uses Today

Test and Development – Rapidly provision test and development servers; store libraries of pre-configured test machines

Business Continuity – Reduce cost and complexity by encapsulating entire systems into single files that can be replicated and restored onto any target server



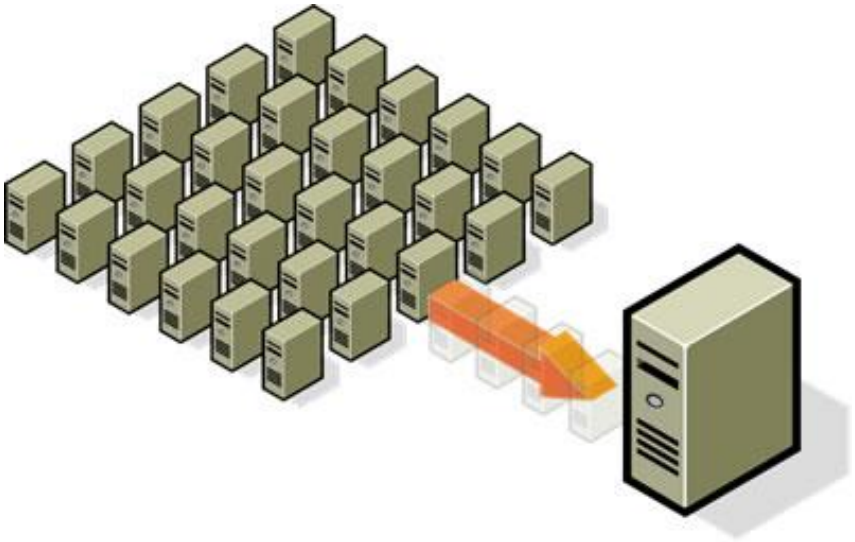
Enterprise Desktop – Secure unmanaged PCs without compromising end-user autonomy by layering a security policy in software around desktop virtual machines

Common Virtualization Uses Today

- Run legacy software on non-legacy hardware
- Run multiple operating systems on the same hardware
- Create a manageable upgrade path
- Manage outages (expected and unexpected) dynamically

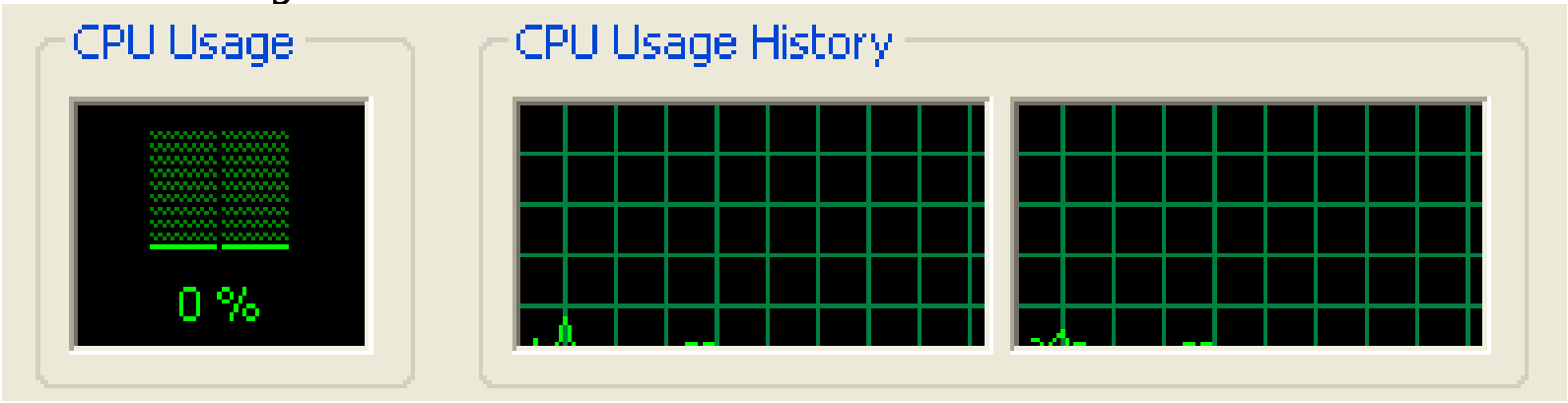
Common Virtualization Uses Today

- Reduce costs by consolidating services onto the fewest number of physical machines



Non-virtualized Data Centers

- Too many servers for too little work

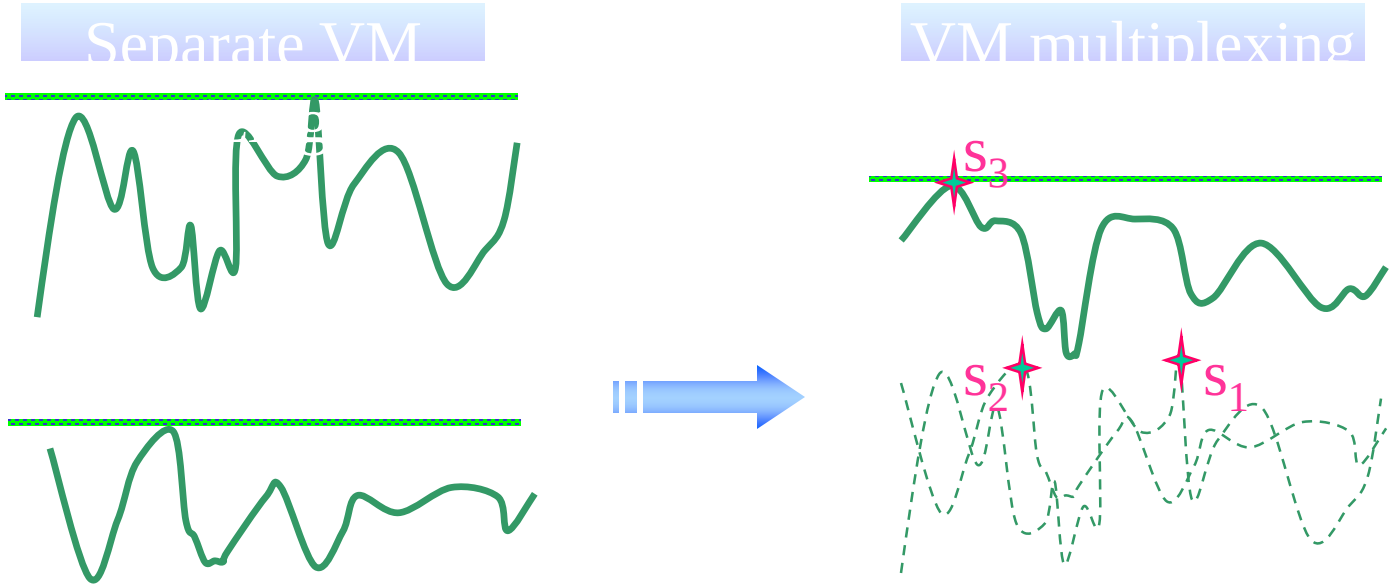


- High costs and infrastructure needs
  - Maintenance
  - Networking
  - Floor space
  - Cooling
  - Power
  - Disaster Recovery

Dynamic Data Center

- Virtualization helps us break the “one service per server” model
- Consolidate many services into a fewer number of machines when workload is low, reducing costs
- Conversely, as demand for a particular service increases, we can shift more virtual machines to run that service
- We can build a data center with fewer total resources, since resources are used as needed instead of being dedicated to single services

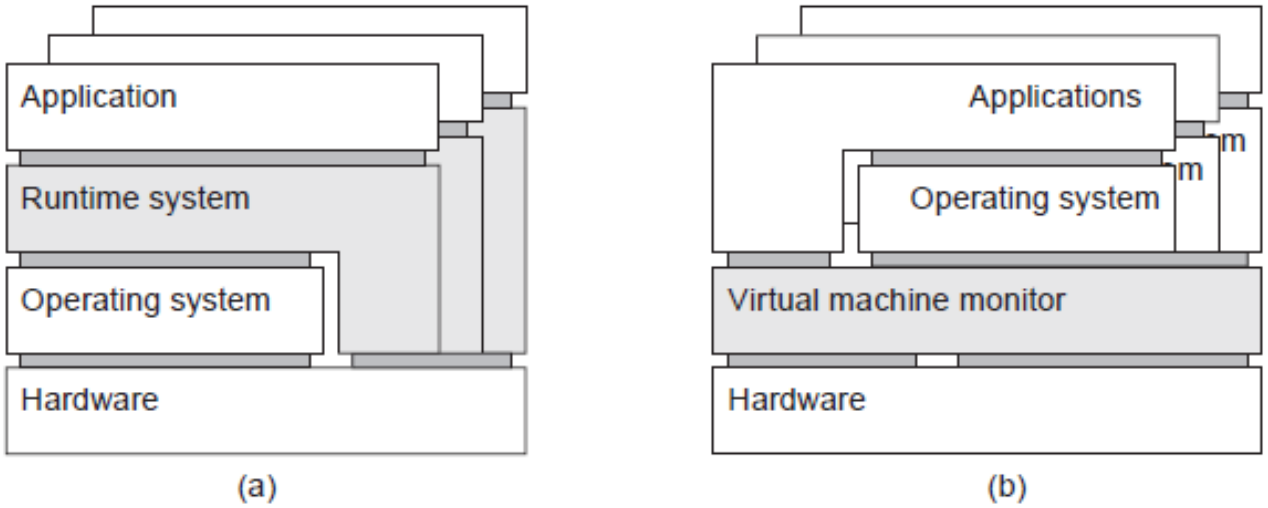
VM workload multiplexing



We expect $s_3 < s_1 + s_2$. Benefit of multiplexing !

- Multiplex VMs' workload on same physical server
  - Aggregate multiple workload. Estimate total capacity need based on aggregated workload
  - Performance level of each VM be preserved

Process VMs versus VM Monitors



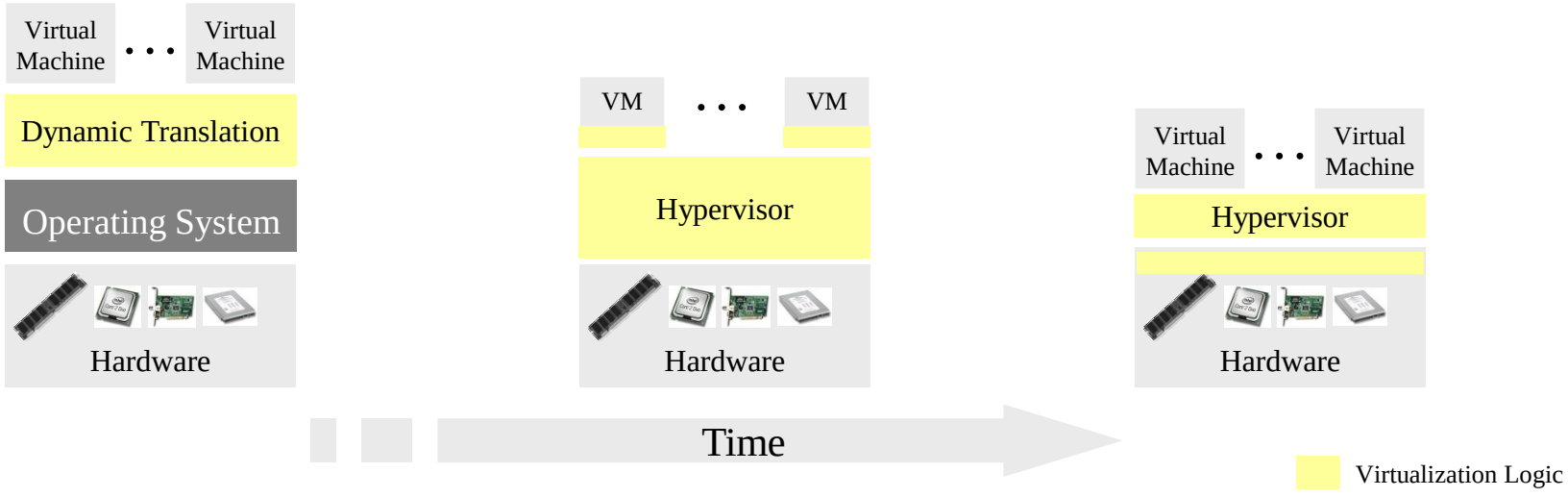
- Process VM: A program is compiled to intermediate (portable) code, which is then executed by a runtime system (Example: Java VM).
- VM Monitor: A separate software layer mimics the instruction set of hardware $\Rightarrow$ a complete operating system and its applications can be supported (Example: VMware, VirtualBox).

Thee Virtualization Approaches

1. Full Virtualization
2. Paravirtualization
3. Hardware-assisted Virtualization

Evolution of virtualization solutions

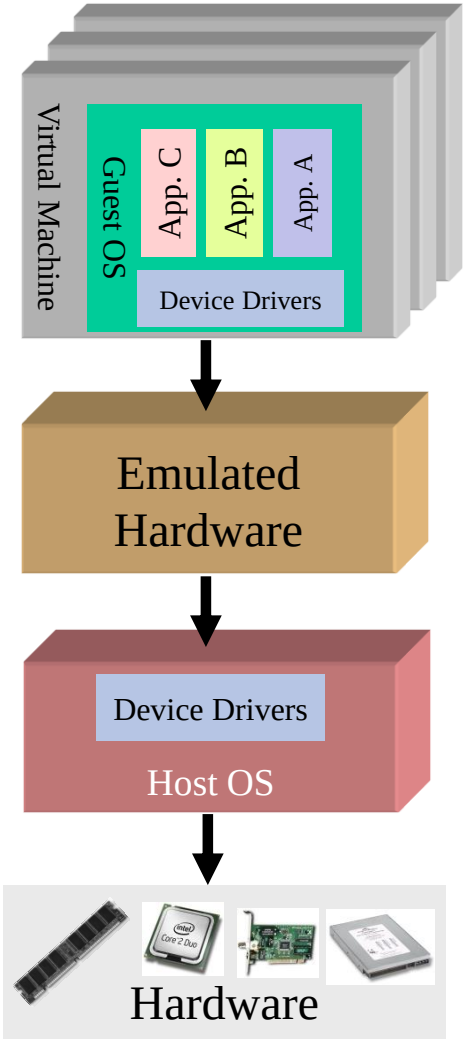
- 1st Generation: Full virtualization (Binary rewriting)
  - Software Based
- 2nd Generation: Paravirtualization
  - Cooperative virtualization
  - Modified guest OS
- 3rd Generation: Silicon-based (Hardware-assisted) virtualization
  - Unmodified guest on virtualization-aware hardware platforms



A **hypervisor** or virtual machine monitor (VMM) is computer software, firmware or hardware that creates and runs virtual machines. A computer on which a **hypervisor** runs one or more virtual machines is called a host machine, and each virtual machine is called a guest machine.

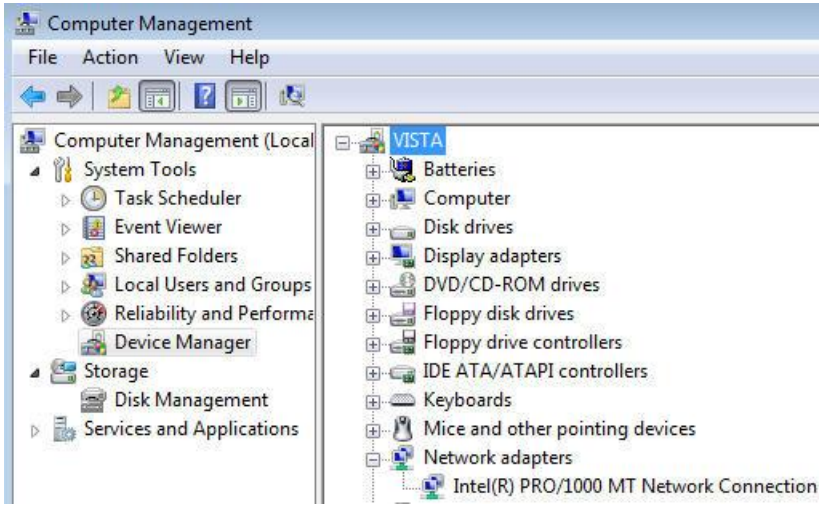
Full Virtualization

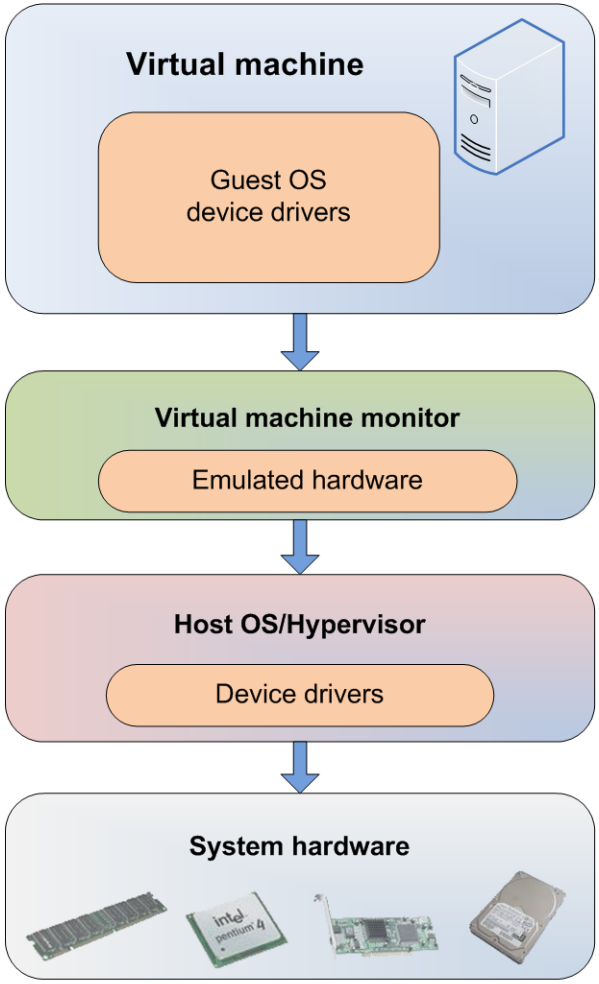
- 1st Generation offering of x86/x64 server virtualization
- Dynamic binary translation
  - The emulation layer talks to an operating system which talks to the computer hardware
  - The guest OS doesn't see that it is used in an emulated environment
- All of the hardware is emulated including the CPU
- Two popular open source emulators are QEMU and Bochs



Full Virtualization

- Everything is virtualized
- Full hardware emulation
- Emulation = latency





Pros and Cons – Full Virtualization

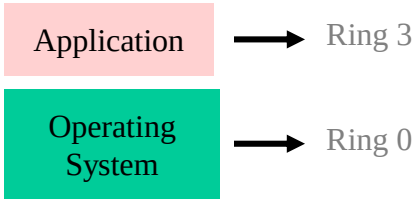
- **Pros**
  - Disaster recovery, failover
  - Virtual appliance deployment
  - Legacy code on non-legacy hardware
- **Cons** – LATENCY of four core resources:
  - RAM performance reduced 25% to 75%
  - Disk I/O degraded from 5% to 20%
  - Network performance decreased up to 10%
  - CPU privileged instruction dings nearing 1% to 7%

Full Virtualization - Advantages

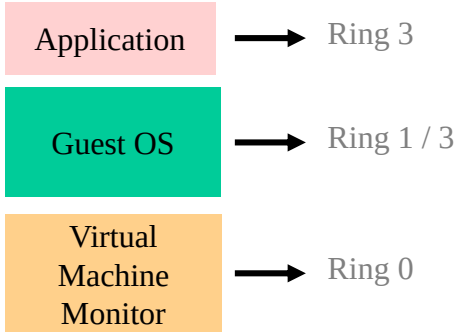
- The emulation layer
  - Isolates VMs from the host OS and from each other
  - Controls individual VM access to system resources, preventing an unstable VM from impacting system performance
- Total VM portability
  - By emulating a consistent set of system hardware, VMs have the ability to transparently move between hosts with dissimilar hardware without any problems
    - It is possible to run an operating system that was developed for another architecture on your own architecture

Full Virtualization - Drawbacks

- Hardware emulation comes with a performance price
- In traditional x86 architectures, OS kernels expect to run privileged code in Ring 0
  - However, because Ring 0 is controlled by the host OS, VMs are forced to execute at Ring 1/3, which requires the VMM to trap and emulate instructions
- Due to these performance limitations, paravirtualization and hardware-assisted virtualization were developed



Traditional x86 Architecture



Full Virtualization

Thee Virtualization Approaches

- Full Virtualization
- **Paravirtualization**
- Hardware-assisted Virtualization

Paravirtualization

–OS or system devices are virtualization aware

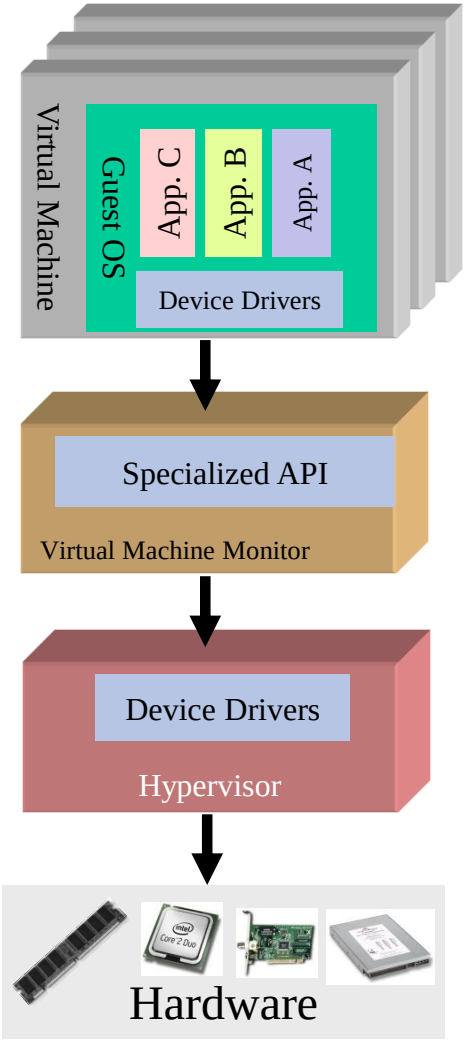
Requirements:

–OS level – recompiled kernel

–Device level – paravirtualized or “enlightened” device drivers

Paravirtualization

- The Guest OS is modified and thus run kernel-level operations at Ring 1
  - the guest is fully aware of how to process privileged instructions
  - thus, privileged instruction translation by the VMM is no longer necessary
  - The guest operating system uses a specialized API to talk to the VMM and, in this way, execute the privileged instructions
- The VMM is responsible for handling the virtualization requests and putting them to the hardware



Paravirtualization

Guest operating systems are paravirtualized by:

- **Recompiling the OS kernel**
  - Paravirtualization drivers and APIs must reside in the guest operating system kernel
  - You do need a modified operating system that includes this specific API, requiring a compiling operating systems to be virtualization aware

- **Installing paravirtualized drivers**
  - In some operating systems it is not possible to use complete paravirtualization, as it requires a specialized version of the operating system
  - To ensure good performance in such environments, paravirtualization can be applied for individual devices
  - For example, the instructions generated by network boards or graphical interface cards can be modified before they leave the virtualized machine by using paravirtualized drivers

Paravirtualization

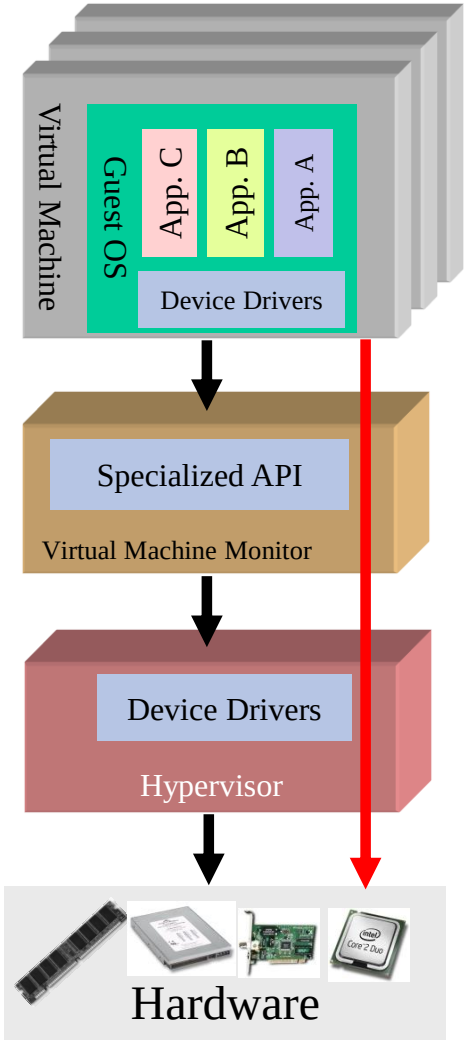
- **Pro:** faster than full virtualisation
- **Con:** requires a specially modified guest OS, thus precludes the ability to run off-the-shelf and legacy OS in paravirtual environments

Thee Virtualization Approaches

- Full Virtualization
- Paravirtualization
- **Hardware-assisted Virtualization**

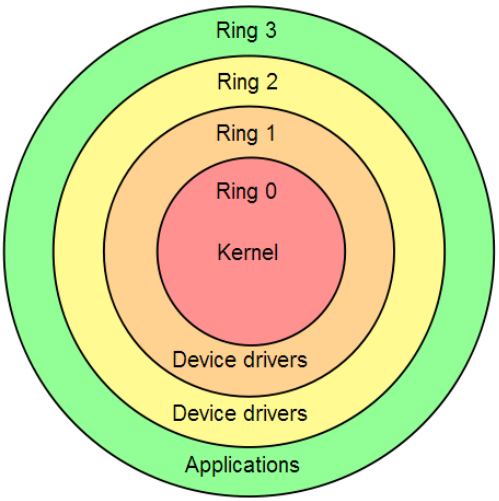
Hardware-assisted virtualization

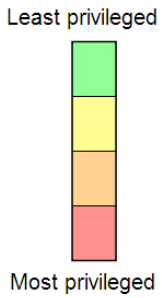
- The VMM uses processor extensions (such as **Intel-VT** or **AMD-V**) to intercept and emulate privileged operations in the guest
- Hardware-assisted virtualization removes many of the problems that make writing a VMM a challenge
- The guest OS runs at ring 0
- The VMM runs in a more privileged ring than 0, a virtual -1 ring is created

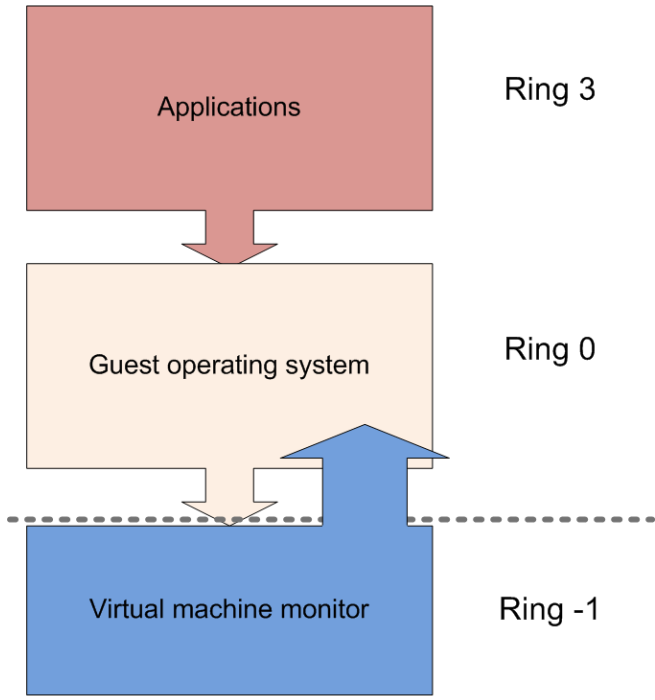


Hardware-assisted Virtualization

- Server hardware is virtualization aware
- Hypervisor and VMM load at privilege Ring -1 (firmware)
- Removes CPU emulation bottleneck
- Memory virtualization in quad core AMD and Intel CPUs







Hardware-assisted virtualization

Hardware-assisted virtualization

- Pros
  - It allows to run unmodified OS (so legacy OS can be run without problems)
- Cons
  - Speed and Flexibility
    - An unmodified OS does not know it is running in a virtualized environment and so, it can't take advantage of any of the virtualization features
      - It can be resolved using paravirtualization partially

Hardware-assisted virtualization: Page tables

- Suppose *guest OS* has its own page tables Then *virtualization layer* must
  - Copy those tables to its own
  - Trap every reference or update to tables and simulate it
- During page fault
  - *Virtualization layer* must decide whether fault belongs to *guest OS* or self
  - If *guest OS*, must simulate a page fault
- Likewise, *virtualization layer* must trap and simulate *every* privileged instruction in machine!

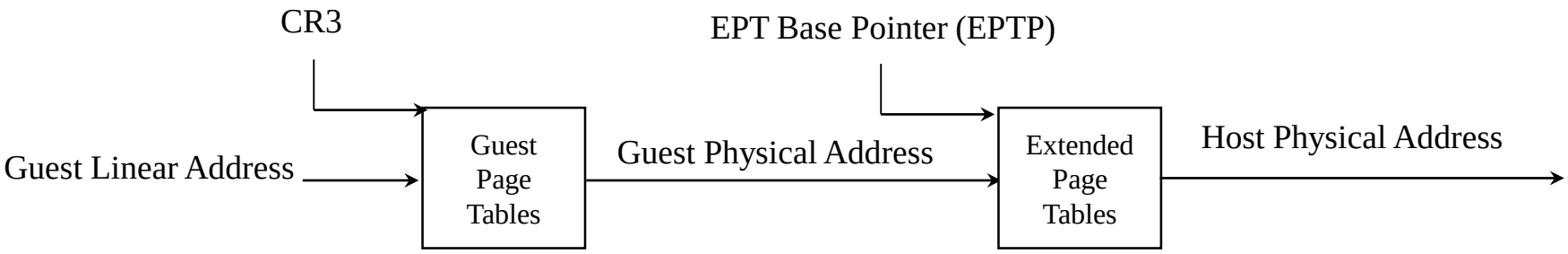
Hardware-assisted virtualization: Extended Page Tables (EPT)

- A VMM must protect host physical memory
  - Multiple guest operating systems share the same host physical memory
  - VMM typically implements protections through "page-table shadowing" in software

- Page-table shadowing accounts for a large portion of virtualization overheads

EPT reduces these overheads

Hardware-assisted virtualization: Extended Page Tables (EPT)

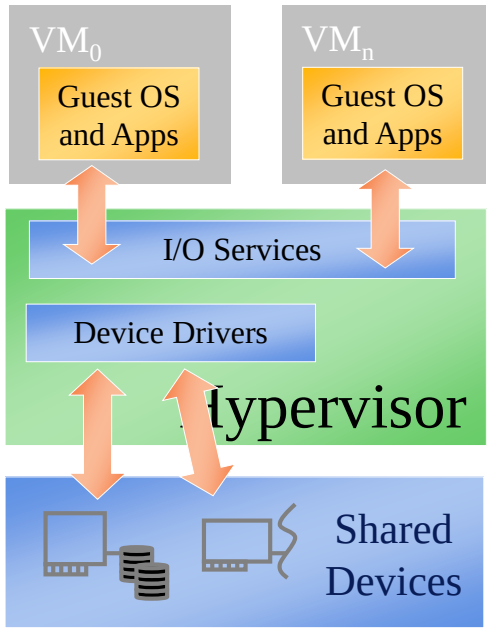


(* Control Register number 3 enables x86 processors to translate virtual addresses into physical addresses by locating the page directory)

- Extended Page Table
- A new page-table structure, under the control of the VMM
  - Defines mapping between guest- and host-physical addresses
  - EPT base pointer (new VMCS field) points to the EPT page tables
    (VMCS: Virtual Machine Control Structure shadowing)
  - EPT (optionally) activated on VM entry, deactivated on VM exit
- Guest has full control over its own page tables
  - No VM exits due to guest page faults

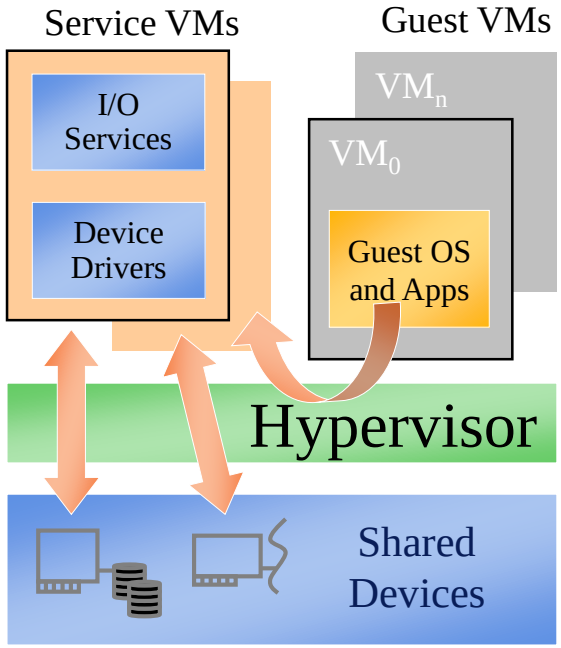
Hardware-assisted virtualization: Options For I/O Virtualization

Monolithic Model



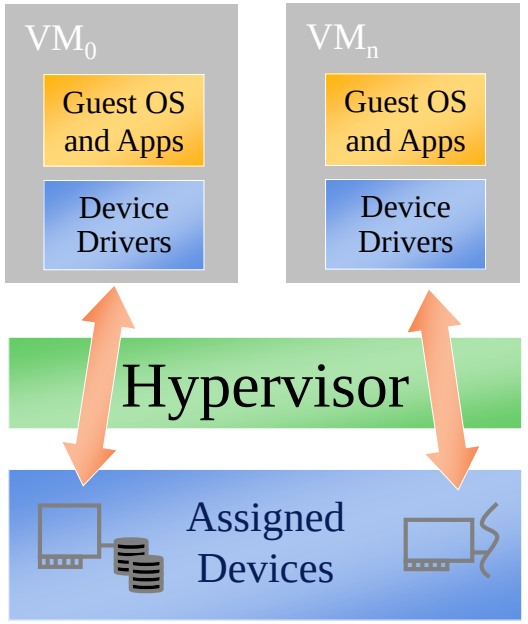
- Pro: Higher Performance
- Pro: I/O Device Sharing
- Pro: VM Migration
- Con: Larger Hypervisor

Service VM Model



- Pro: High Security
- Pro: I/O Device Sharing
- Pro: VM Migration
- Con: Lower Performance

Pass-through Model



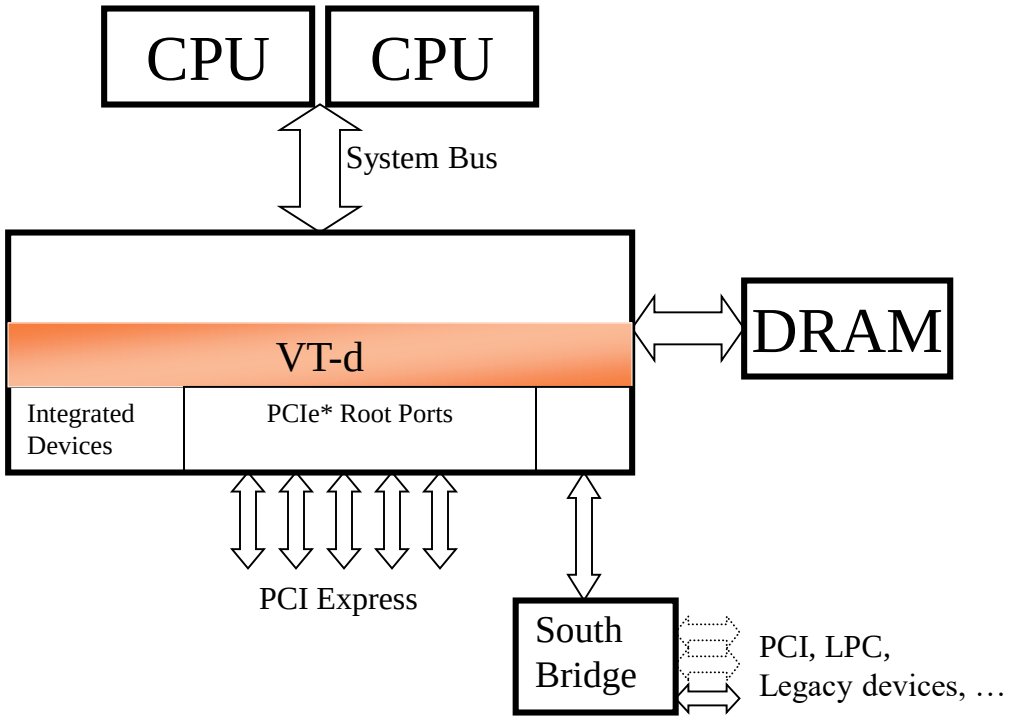
- Pro: Highest Performance
- Pro: Smaller Hypervisor
- Pro: Device assisted sharing
- Con: Migration Challenges

VT supports all Models

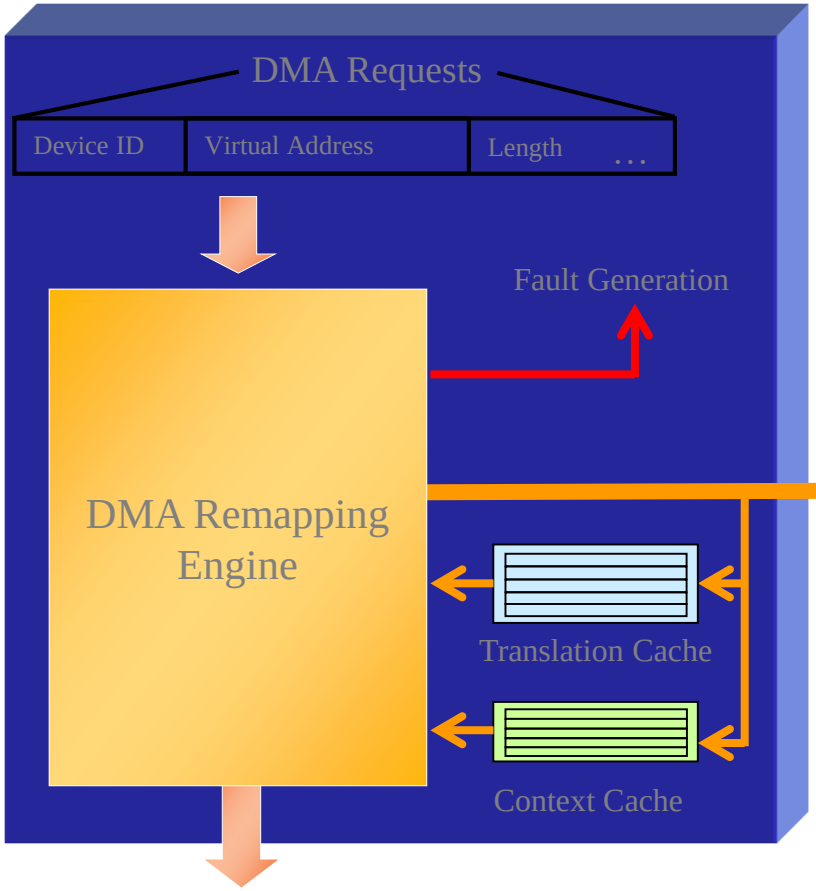
(VMM uses processor extensions such as Intel-**VT** or AMD-V to intercept and emulate privileged operations in the guest)

Hardware-assisted virtualization: I/O Virtualization

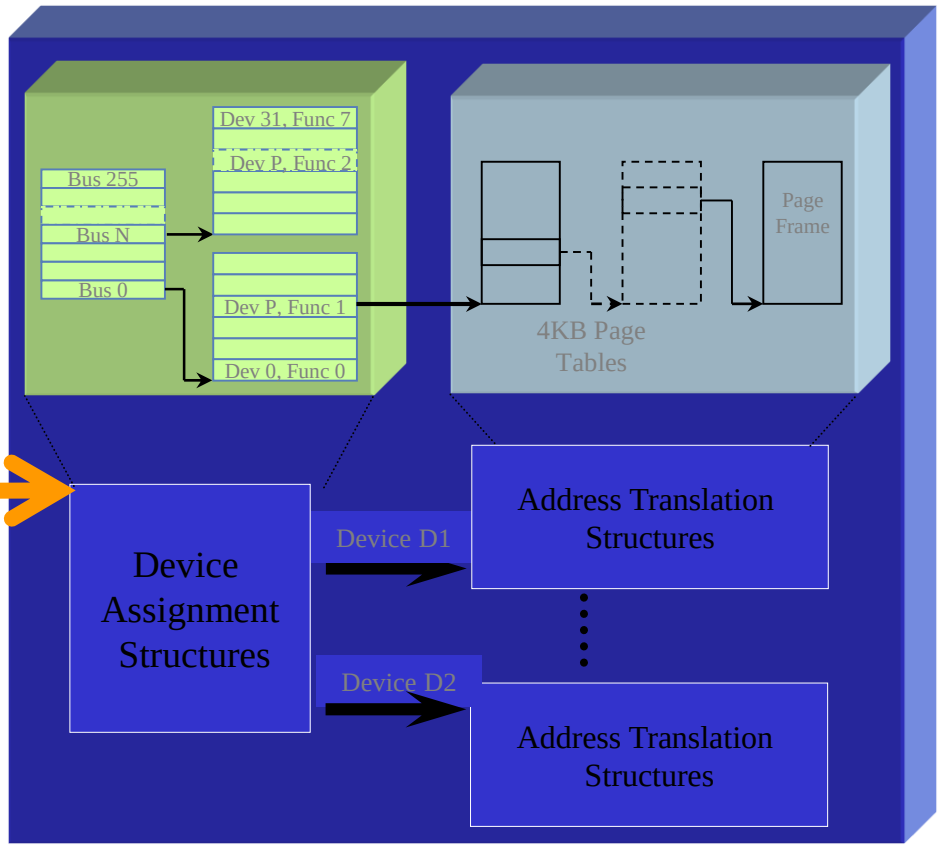
- Infrastructure for I/O virtualization
  - Defines architecture for DMA remapping
  - Implemented as part of platform core logic
  - Supported in Intel server and client chipsets



Hardware-assisted virtualization: Architecture Detail

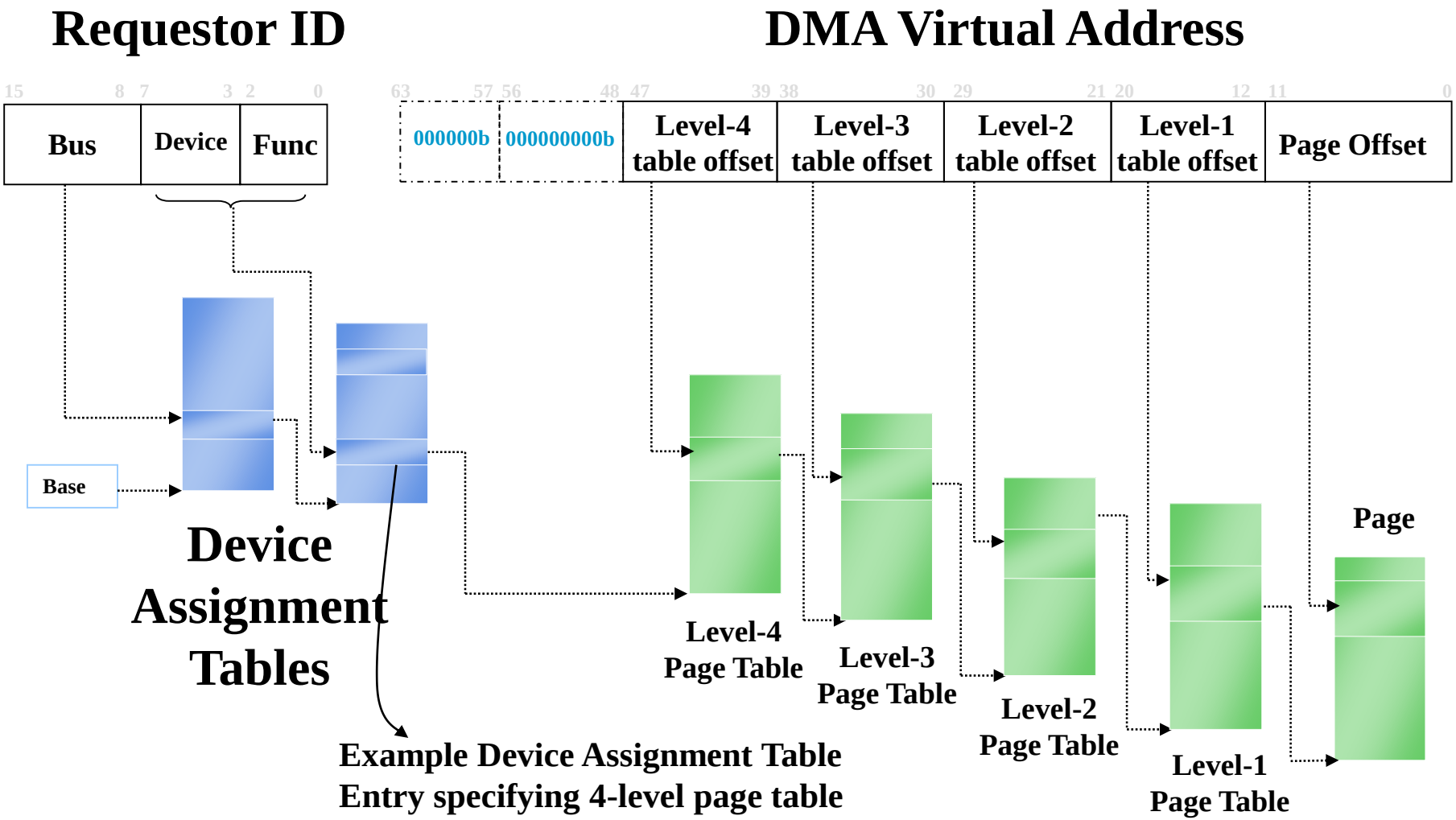


Memory Access with System Physical Address



Memory-resident Partitioning and Translation Structures

Hardware-assisted virtualization: Hardware Page Walk

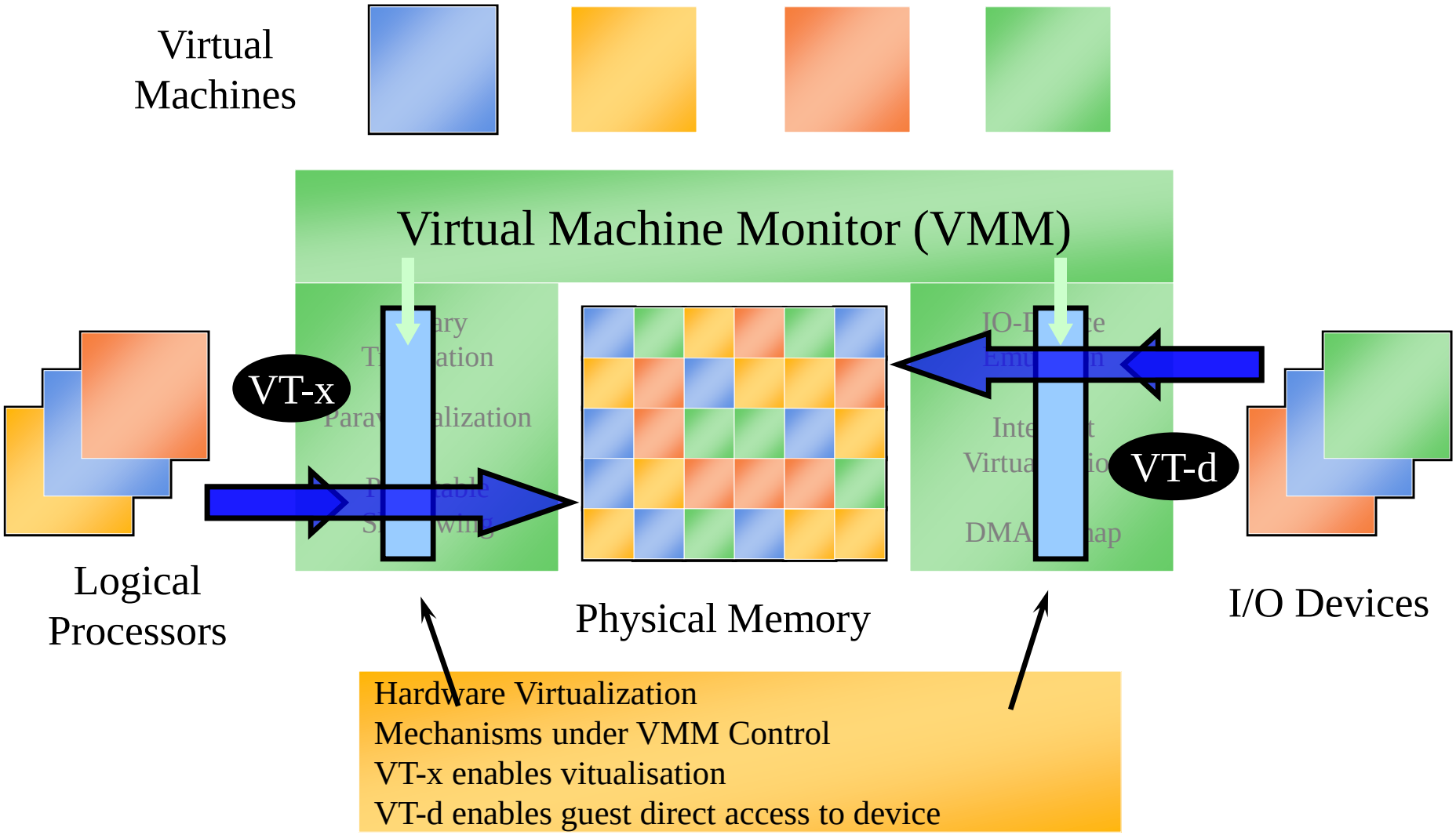


Hardware-assisted virtualization: Extended Features

- PCI Express protocol extensions for Address Translation Services (**ATS**)
  - Enables scaling of translation caches to devices
  - Devices may request translations from root complex and cache
  - Protocol extensions to invalidate translation caches on devices

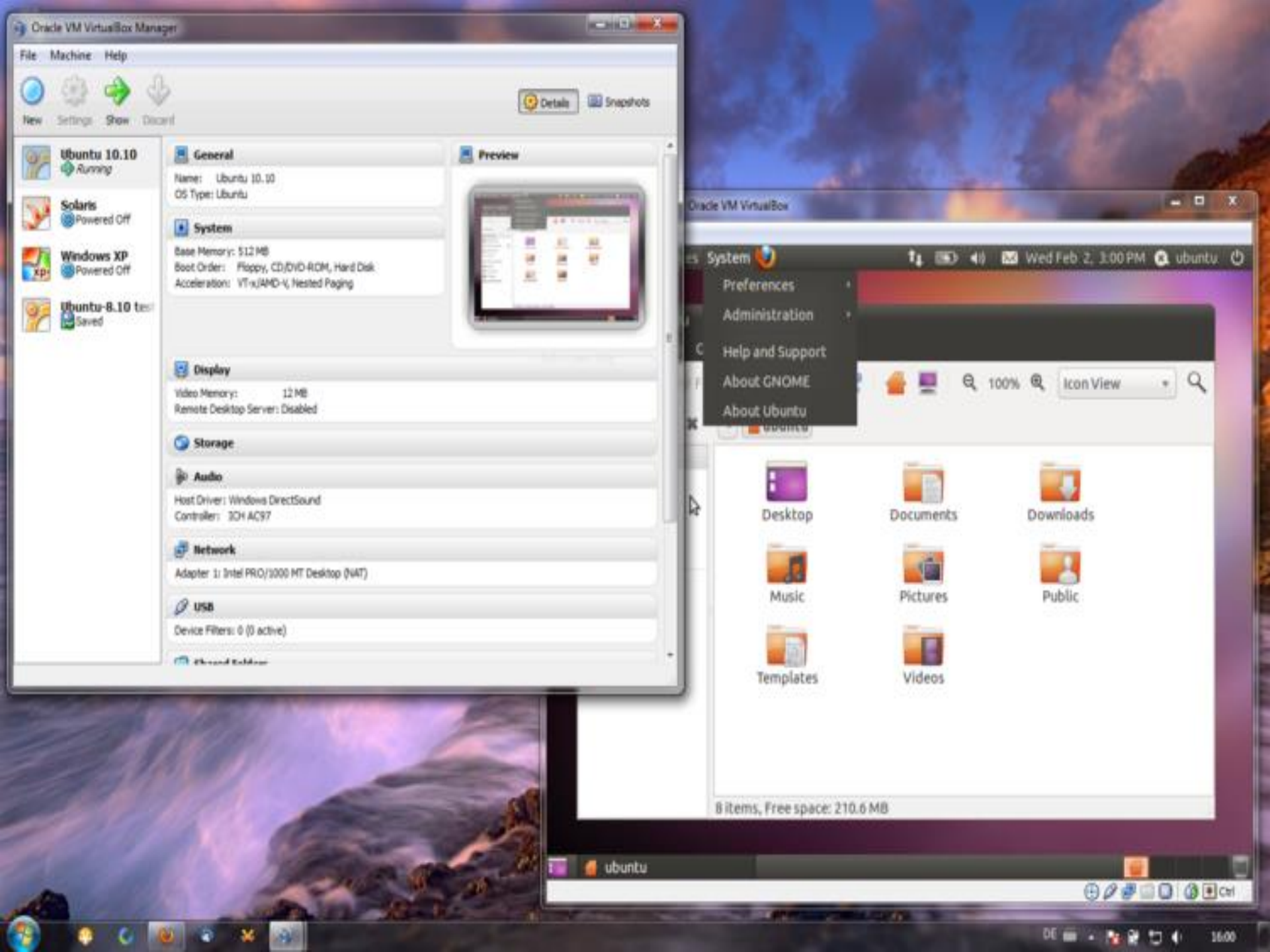
- **VT extended capabilities**
  - Enables VMM software to control device participation in ATS
  - Returns translations for valid ATS translation requests
  - Supports ATS invalidations
  - Provides capability to isolate, remap and route interrupts to VMs
  - Support device-specific demand paging by ATS capable devices

Hardware-assisted virtualization: Working Together



How Hardware-assisted virtualization Address Virtualization Challenges

- Reduced Complexity
  - VT removes need for binary translation / paravirtualization
  - Can avoid I/O emulation for direct-mapped I/O devices
- Improved Functionality
  - 64-bit guest OS support, remove limitations of paravirtualization
  - Can grant Guest OS direct access to modern physical I/O devices
- Enhanced Reliability and Protection
  - Simplified VMM reduces “trusted computing base”
  - DMA errors logged and reported to software
- Improved Performance
  - Hardware support reduces address-translation overheads
  - No need for shadow page tables (saves memory)


Oracle VM VirtualBox Manager
File Machine Help
New Settings Show Discard
Details Snapshots
Ubuntu 10.10
Running
Solaris
Powered Off
Windows XP
Powered Off
Ubuntu-8.10 test
Saved
General
Name: Ubuntu 10.10
OS Type: Ubuntu
System
Base Memory: 512 MB
Boot Order: Floppy, CD/DVD-ROM, Hard Disk
Acceleration: VT-x/AMD-V, Nested Paging
Preview
Display
Video Memory: 12 MB
Remote Desktop Server: Disabled
Storage
Audio
Host Driver: Windows DirectSound
Controller: ICH AC97
Network
Adapter 1: Intel PRO/1000 MT Desktop (NAT)
USB
Device Filters: 0 (0 active)
Oracle VM VirtualBox
System
Wed Feb 2, 3:00 PM
ubuntu
Preferences
Administration
Help and Support
About GNOME
About Ubuntu
100%
Icon View
Desktop
Documents
Downloads
Music
Pictures
Public
Templates
Videos
8 items, Free space: 210.6 MB
ubuntu
Ctrl

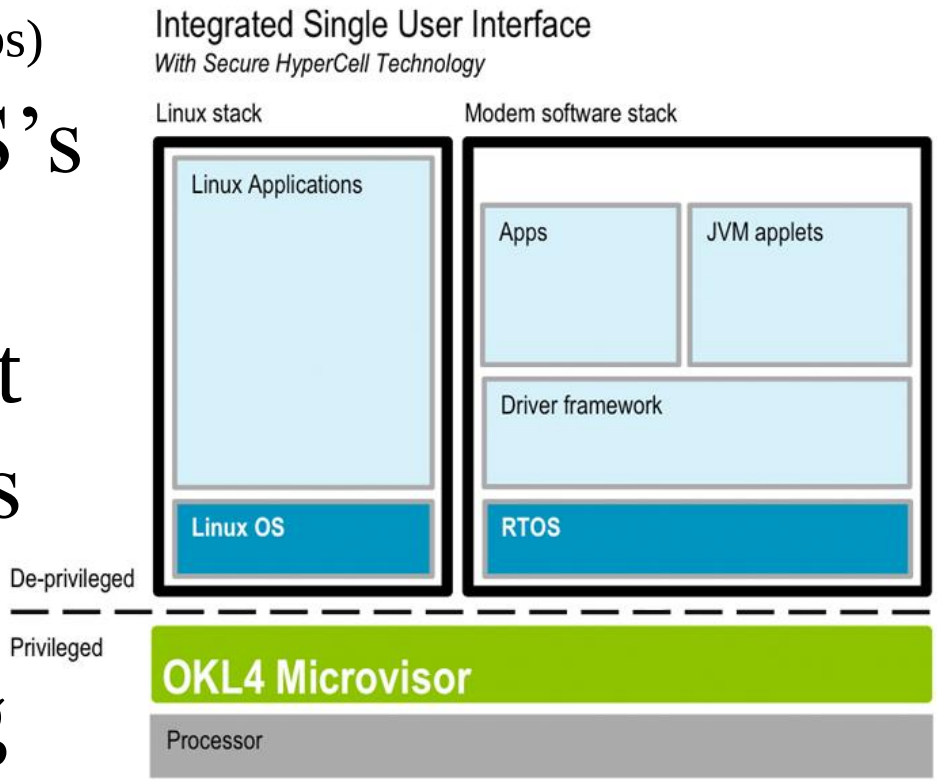

VirtualBox File Machine Window Help
Oracle VM VirtualBox Manager
New Settings Show Discard
Details Snapshots
Windows-7 Running
Ubuntu-8.10 Powered Off
01_test_1 Saved
02_test Powered Off
02_test_2 Powered Off
Windows-XP-Dev Saved
Fedore-14 Saved
ubuntu Powered Off
Solaris Powered Off
MSDOS (Snapshot 2) Powered Off
Win-XP-SP3 Powered Off
ee Powered Off
General
Name: Windows-7
OS Type: Windows 7
System
Base Memory: 1024 MB
Boot Order: Floppy, CD/DVD-ROM, Hard Disk
Acceleration: VT-x/AMD-V, Nested Paging
Preview
Display
Video Memory: 128 MB
Remote Desktop Server: Disabled
Storage
Audio
Host Driver: CoreAudio
Controller: ICH AC97
Network
Adapter 1: Intel PRO/1000 MT Desktop (NAT)
Serial Ports
USB
Device Filters: 0 (0 active)
Windows-7 [Running]
DE German (Germany) Help
Donnerstag
27
Januar 2011
Libraries
Open a library to see your files and arrang...
Documents
Library
Music
Library
Pictures
Library
Videos
Library
New library
Left

Mobile Virtualisation

- Horizon Mobile runs a hypervisor on the Android phone to create a guest operating system—a second instance of Android—to isolate a user's work or personal apps and data. The user's employer can then manage the corporate side of the phone without having any visibility into the user's personal space. The corporate side of the device is encrypted and can be integrated with "standard enterprise directory services."

Mobile Virtualisation

General Dynamics Broadband (was Open Kernel Labs) paravirtualizes guest OS's for use with OKL4 Microvisor - has support for a wide range of OS's and application environments, including Android, multiple Linux distributions, RTOS's and Windows Mobile.



Example Exam Question

- The first, second and third generations of virtualization were

  “full virtualisation”, “paravirtualization” and “hardware assisted virtualization”

- Briefly describe one pro and one con for each of these three generations of virtualization.

Example Exam Question

- **Full Virtualisation** (first generation):
- **Pros**
  - Run an unmodified operating system - legacy code on non-legacy hardware
    - Total VM portability : By emulating a consistent set of system hardware, VMs have the ability to transparently move between hosts with dissimilar hardware without any problems
    - It is possible to run an operating system that was developed for another architecture on your own architecture

Example Exam Question

- **Full Virtualisation** (first generation):
- **Cons**
  - Latency of four core resources (RAM, Disk, Network, CPU)
  - Hardware emulation comes with a performance price
    - RAM performance reduced 25% to 75%
    - Disk I/O degraded from 5% to 20%
    - Network performance decreased up to 10%
    - CPU privileged instruction dings nearing 1% to 7%
    - In traditional x86 architectures, OS kernels expect to run privileged code in Ring 0. However, because Ring 0 is controlled by the host OS, VMs are forced to execute at Ring 1/3, which requires the VMM to trap and emulate instructions

Example Exam Question

- **Paravirtualisation** (second generation):
- **Pro:**
  - faster than full virtualisation
- **Con:**
  - requires a specially modified guest OS, thus precludes the ability to run off-the-shelf and legacy OS in paravirtual environments

Example Exam Question

- **Hardware assisted virtualisation** (third generation):
- **Pro:**
  - It allows to run unmodified OS (so legacy OS can be run without problems)
- **Con**
  - Speed and Flexibility
    - An unmodified OS does not know it is running in a virtualized environment and so, it can’t take advantage of any of the virtualization features
      - It can be resolved using paravirtualization partially