An Introduction to Virtual Machine

虚拟机技术简介

诸葛建伟

ERCIS, ICST, PKU

内容

1. 虚拟化技术基本概念与发展历程
2. 虚拟机技术分类
3. 虚拟机技术实现机制
4. 虚拟机技术引领者—**VMware**

   Dr. Mendel Rosenblum: What's ahead?

关于“虚拟 / 虚拟机 / 虚拟化”的词汇

- 你可能听说 / 接触过的相关词汇
  - 虚拟内存、虚拟光驱、虚拟网卡、虚拟磁盘、虚拟**CPU**
  - 虚拟机: **VMware, Virtual PC, Java**虚拟机, **Bochs, Qemu, UML, Xen, KVM, etc...**
  - 虚拟主机**(Virtual hosting)**
  - 虚拟网络: 虚拟局域网**(VLAN)**、虚拟专有网**(VPN)**、虚拟内网**(VNN)**
  - 虚拟货币、虚拟资产、虚拟世界、虚拟现实、虚拟人生
- **Everything can be virtual? Virtual Brain?**

什么是虚拟化?

- 虚拟化(Virtualization)
  - 创建某种事物的虚拟(非真实)版本的方法和过程.
  - the creation of a virtual (rather than actual) version of something. [Whatis.com]
- 虚拟(Virtual)
  - 通常用于区分纯粹概念上的事物和拥有物理实体的事物.
  - In general, it distinguishes something that is merely conceptual from something that has physical reality.
  - 反义词: 真实的, 实际的, 物理的, 绝对的
  - Opposite: real, actual, physical, absolute

什么是计算领域中的虚拟化?

- 计算领域中的虚拟化**[whatis.com]** **[webopedia]**
  - 创建某种计算资源的虚拟版本的方法和过程.
  - 某种事物 → 某种计算资源
  - 示例: 处理器, 内存, 磁盘, 完整的计算机, 网络, 等
- 计算领域中的虚拟化**[wikipedia]**
  - 在计算领域中，虚拟化是一个含义广泛的词汇，指的是对计算机资源的抽象化(abstraction)，虚拟化对计算机资源的用户(应用程序或终端用户)隐藏了它们的物理特性。[1]
  - [1] IBM White paper: Virtualization in education

Abstraction, Interface, Architecture, Implementation

- 抽象化（**Abstraction**）
  - In computer science, abstraction is a mechanism and practice to reduce and factor out details so that one can focus on a few concepts at a time.
- 接口（**Interface**）
  - Separates levels of abstraction, facilitates independent subsystem development by both hardware and software teams.
- 体系框架（**Architecture**）
  - Refers to a formal specification of an interface in the system, including the logical behavior of resources managed via the interface.
- 实现（**Implementation**）
  - Describes the actual embodiment of an architecture.

Abstraction对于CS发展的巨大意义

- 克服复杂性
  - 计算机系统结构已无比复杂
  - 能够很好地工作
  - 有条不紊地继续发展新的计算机理论和技术
- 促进多样性
  - 接口规范和标准化
  - 有效的社会分工，全才→专业人才
  - 生态环境的多样性→稳定性
- 示例
  - **Intel, AMD**设计和开发**CPU**实现**Intel IA-32(x86)**指令集规范接口
  - **M$**设计和开发**Windows**操作系统,提供**API**接口和**Visual Studio**等开发环境,编译二进制指令符合**x86**指令集规范接口
  - **Win32**平台应用软件开发者使用**API**接口**+VS**编写应用软件

Architected Interfaces

- **ISA: Instruction Set Architecture**
  - Interface 3: system ISA, visible only to OS for managing HW
  - Interface 4: user ISA, visible to an application program
- **ABI: Application Binary Interface**
  - Interface 2: system call interface
  - Interface 4: user ISA
- **API: Application Programming Interface**
  - Interface 1: high-level language (HLL) library calls
  - Interface 4: user ISA

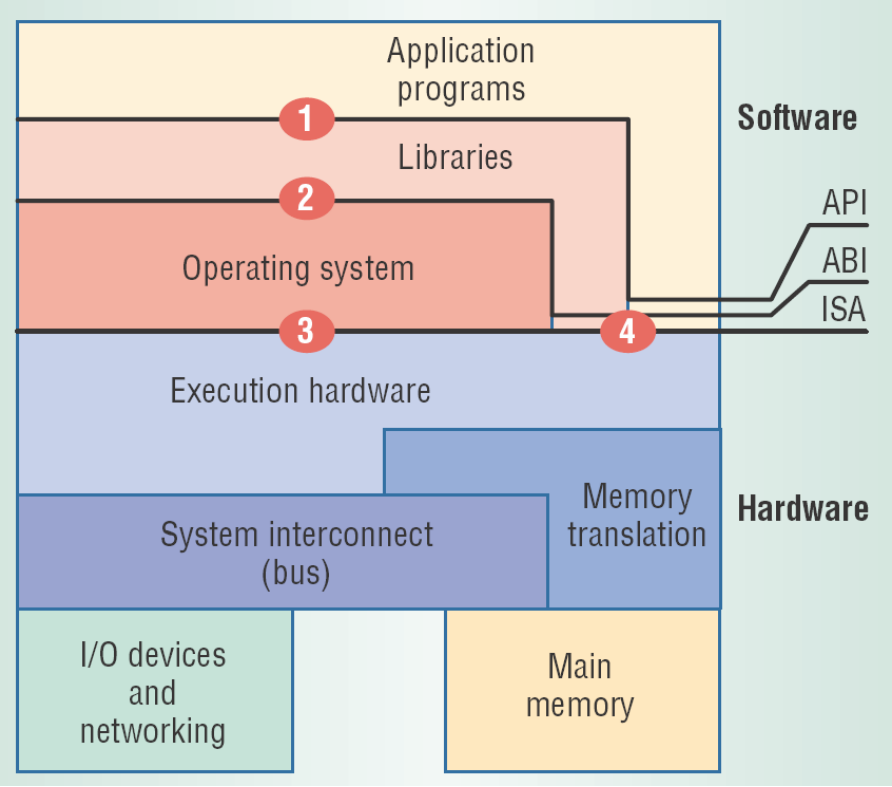


Figure: Computer System Architecture

抽象化 vs. 虚拟化

- 抽象化带来的问题
  - Lack of interoperability（互操作性）
  - 根据特定接口和体系框架设计实现的系统和组件无法直接在另外接口上运行
- 虚拟化
  - 虚拟化可以从真实系统构建虚拟版本
  - 可以将一种抽象接口转换成另外一种抽象接口
  - 与抽象化不同:虚拟化并不一定隐藏细节

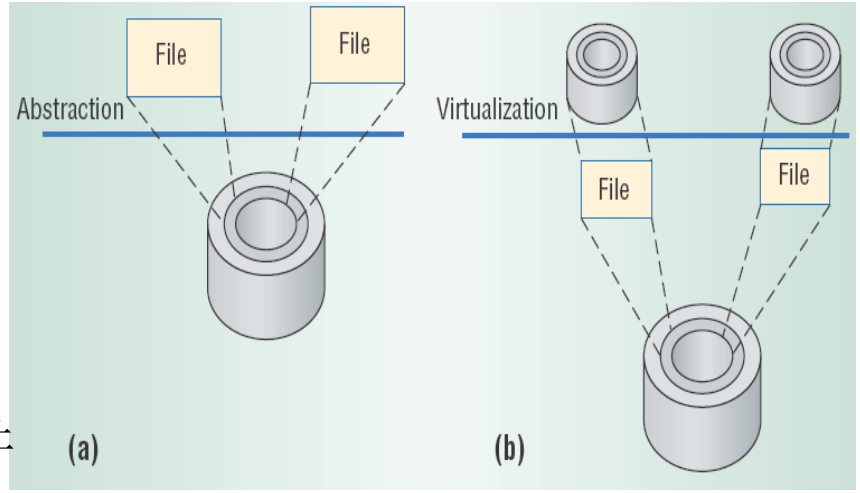
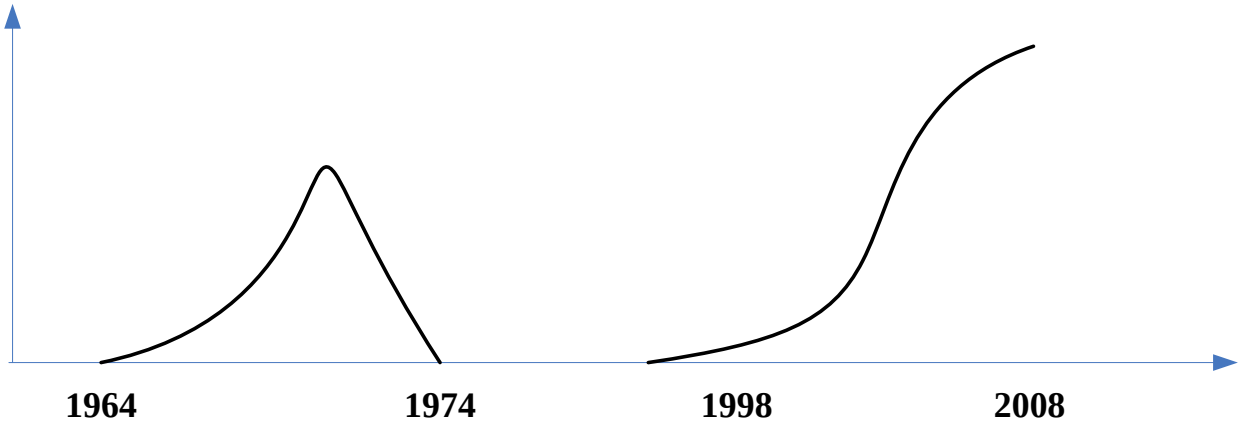


Figure: Abstraction and virtualization applied to disk storage.

虚拟化技术的提出与复兴



- “Zero generation”: Mainframe Virtualization
  - 1964-1973: IBM mainframes (CP-40/44/45, S/360, CP-67, CP/CMS, S/370, VM/370) – time-sharing, virtual memory, run VM under VM, to multiplex such a scarce resource among multiple applications.
  - 1974: IEEE Computer “Survey of Virtual Machine Research”
- 80s, 90s: VMMs disappeared due to modern multitasking operating systems / cheaper hardware
- 1997-: Revival of virtualization – Java VM (95), Virtual PC (Connectix, 97), VMware(98)

Why the Revival?

- 多任务操作系统/硬件价格下降导致问题
  - 机器数量的泛滥导致资源利用率低
  - 操作系统的复杂性导致其脆弱性
- **one application running per machine**
  - 减少系统崩溃和入侵造成的影响
  - 带来硬件需求增加，资源利用率低
  - 增加了硬件费用、管理和能源负担
- **Why the Revival of Virtual Machine?**
  - Multiplexing hardware: server consolidation and utility computing
  - more a solution for: ease of management, security, reliability

Virtualization Benefits: Decoupling HW and SW

- *tremendous control* over how GuestOSs use HW resource
- *uniform view* of underlying hardware
  - 将硬件视为物理资源池，可以按需运行任意服务
- complete *encapsulation*
  - map/remap, suspend/resume/checkpoint/revert, migrate/replicate
  - load balancing
  - robust model for dealing with hardware failures
  - supports a very general mobility model
- *total mediation* of all interactions btw HW and SW
  - Multiplexing of many virtual machines on a single hardware platform
  - strong isolation: valuable for reliability and security

虚拟化技术的复兴: first generation

- **The first Generation: x86 virtualization (1997-2005)**
  - 1997: Virtual PC for Macintosh by Connectix
  - 1998: Diane Greene和Mendel Rosenblum利用Stanford研究成果创建VMware公司，申请专利技术
  - 1999: VMware Virtual Platform (Workstation) for x86
  - 2001: VMware GSX Server product (Server, 06 free release)
  - 2003: M$ acquired Connectix (Virtual PC & Virtual Server), EMC acquired VMware for $635 million
  - Full Virtualization with Binary Translation

虚拟化技术的复兴: second generation

- **The second Generation: Hardware/OS assisted Virtualization(05-)**
- **Hardware assisted virtualization**
  - 2005: Intel's IVT (Vanderpool/Silvervale)
  - 2006: AMD's AMD-V (Pacifica)
  - Native Virtualization
- **OS assisted virtualization (paravirtualization)**
  - 2002: Denali by Washington U.
  - 2003: Xen by XenSource (from U. of Camb.)
  - 2005: Virtual Machine Interface by VMware
  - 2008: XenSource is also developing a compatibility layer for M$ Windows Server 2008
  - paravirtualization
- **Virtual Infrastructure: third generation?**
  - 2006-: Virtual Infrastructure by VMware

内容

1. 虚拟化技术基本概念与发展历程
2. 虚拟机技术分类
3. 虚拟机技术实现机制
4. 虚拟机技术引领者—VMware

   Dr. Mendel Rosenblum: What's ahead?

What is a Virtual Machine

- **Virtual Machine**
  - virtual version of a “machine”
- **What is a “Machine” then?**
  - Consider the meaning from both a process and system perspective
- **From the process’s perspective**
  - A logical memory address space; User-level instructions and registers; I/O (only visible through the operating system calls).
  - Thus, ABI defines the machine as seen by a process, API specifies the machine as seen by HLL program
- **From the OS’s perspective**
  - the underlying hardware’s characteristics alone define the machine.
  - ISA provides the interface between the system and machine.

Process VM & System VM

- **Process VM**
  - A *process* VM is a virtual platform that executes an individual process.
  - Java VM, FVM Sandbox, etc.
- **System VM**
  - a *system* VM provides a complete, persistent system environment that supports an operating system along with its many user processes.
  - VMware, Qemu, etc.
- **Basic concepts**
  - guest, host, runtime, VMM

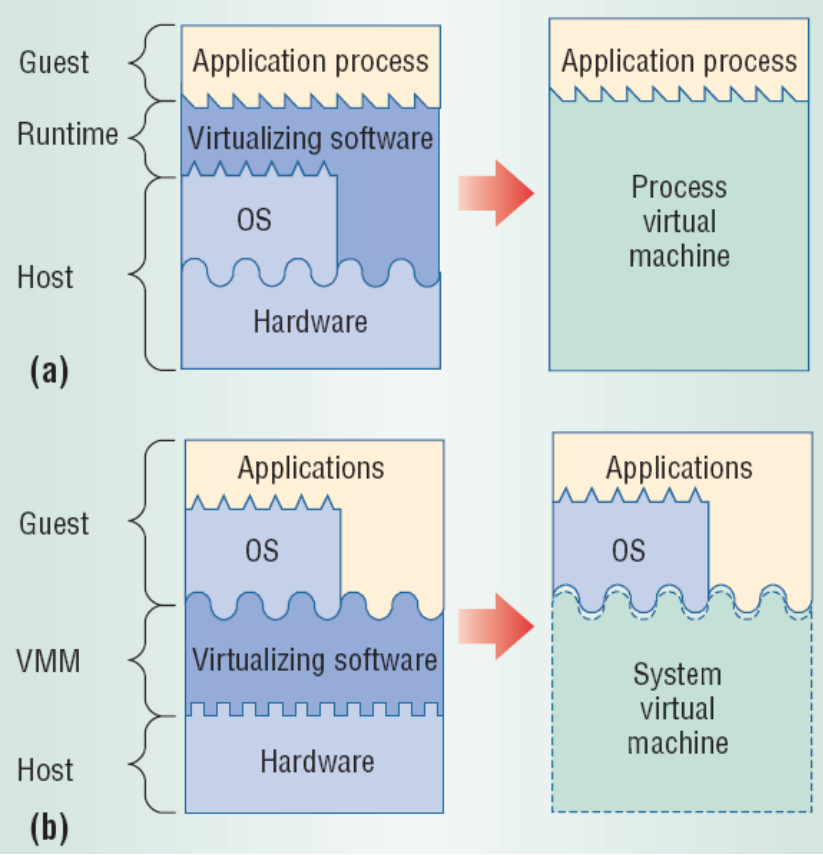


Figure: Process and system VMs

Process VMs

- **Multiprogrammed systems** 多任务操作系统
  - Most OS simultaneously support multiple user processes through multiprogramming
  - most common process VM
- **Emulators and Dynamic Binary Translators**
  - 支持针对特定ISA编译的二进制程序在其他ISA上执行
  - 解释执行(*interpretation*)
  - Dynamic Binary Translator: 动态二进制代码翻译 (*dynamic binary translation*)
- **Same-ISA binary optimizers / translators**
  - 在二进制代码解释执行/翻译过程进行代码优化: Dynamo
  - 相同ISA, 不同ABI的代码解释/翻译: Wine
- **High-level-language VMs** 高级语言虚拟机
  - cross-platform portability: Java VM, .NET framework

HLL VMs—高级语言虚拟机

- **HLL VMs**
  - Full cross-platform portability
  - 在高级编程语言开发环境中直接考虑对process VM的支持
- **传统的平台相关编译环境 VS. 高级语言虚拟机环境**
  - 传统平台相关编译环境：发布生成二进制文件，限于特定的ISA/OS
  - 高级语言虚拟机环境：发布虚拟ISA可移植的代码和元数据，通过VM解释器/编译器生成运行平台的二进制指令

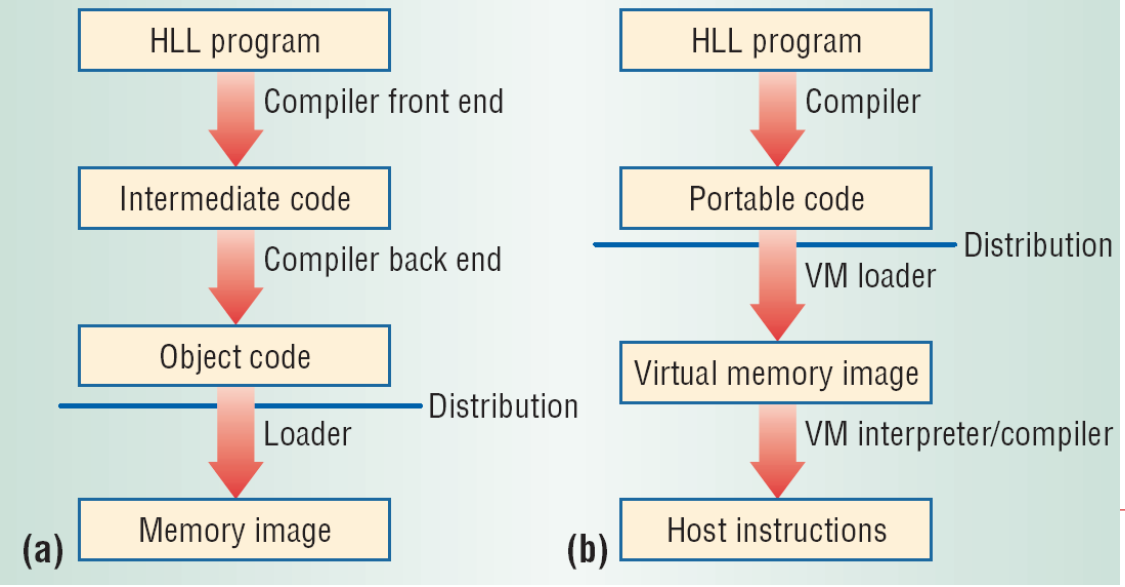
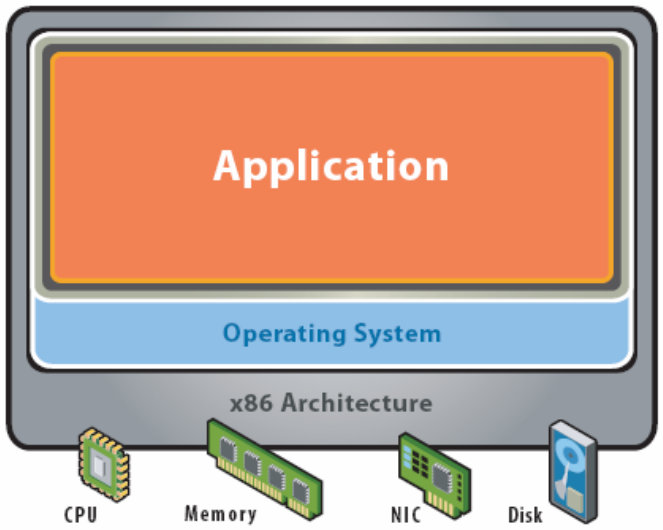


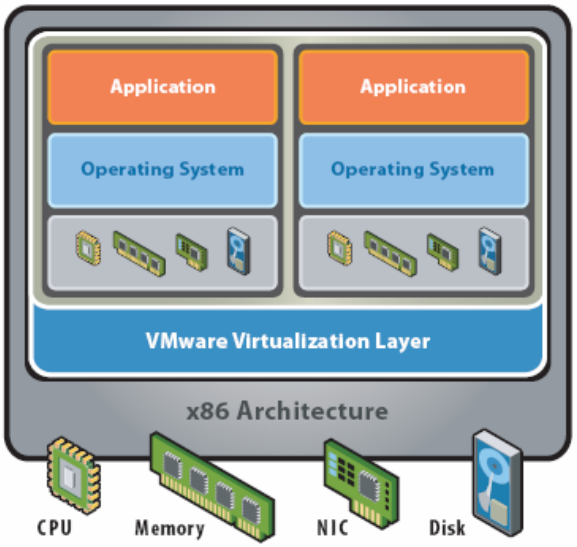
Figure: high-level-language environments

System VM



Before Virtualization:

- Single OS image per machine
- Software and hardware tightly coupled
- Running multiple applications on same machine often creates conflict
- Underutilized resources
- Inflexible and costly infrastructure



After Virtualization:

- Hardware-independence of operating system and applications
- Virtual machines can be provisioned to any system
- Can manage OS and application as a single unit by encapsulating them into virtual machines

System VMs

- **Classic system (hypervisor) VMs**
  - 60s-70s, VMM直接运行于物理硬件之上
  - VMM运行于最高权限,guest OS运行于受限权限
  - VMM 劫持和仿真guest OS操作，访问/操纵关键硬件资源
  - 回归: VMware ESX Server
- **Hosted VMs**
  - VMM运行在宿主操作系统上
  - 优势
    - VMM安装-普通应用软件
    - 可依赖于宿主操作系统提供设备驱动和其他底层服务
  - **VMware Workstation/Server, Xen …**

System VMs (2)

- **Whole-system VMs**
  - host and guest systems do not have a common ISA
  - ISA: x86, PowerPC, ARM, IA-64(Itanium)
  - VMM需要仿真应用程序和操作系统代码
  - Example: Virtual PC, QEMU, etc.
- **Codesigned VMs**
  - codesigned VMs使用新的、专属的ISA, 以提升性能/降低能源使用等为目标
  - VMM一般和硬件协同设计, 实现紧耦合
  - Example
    - IBM AS/400 (1996): provide support for an object-based instruction set
    - Transmeta Crusoe (2000): Host ISA- very-long instruction word architecture; Guest ISA – x86; Adv: power-saving

Virtual Machine Taxonomy

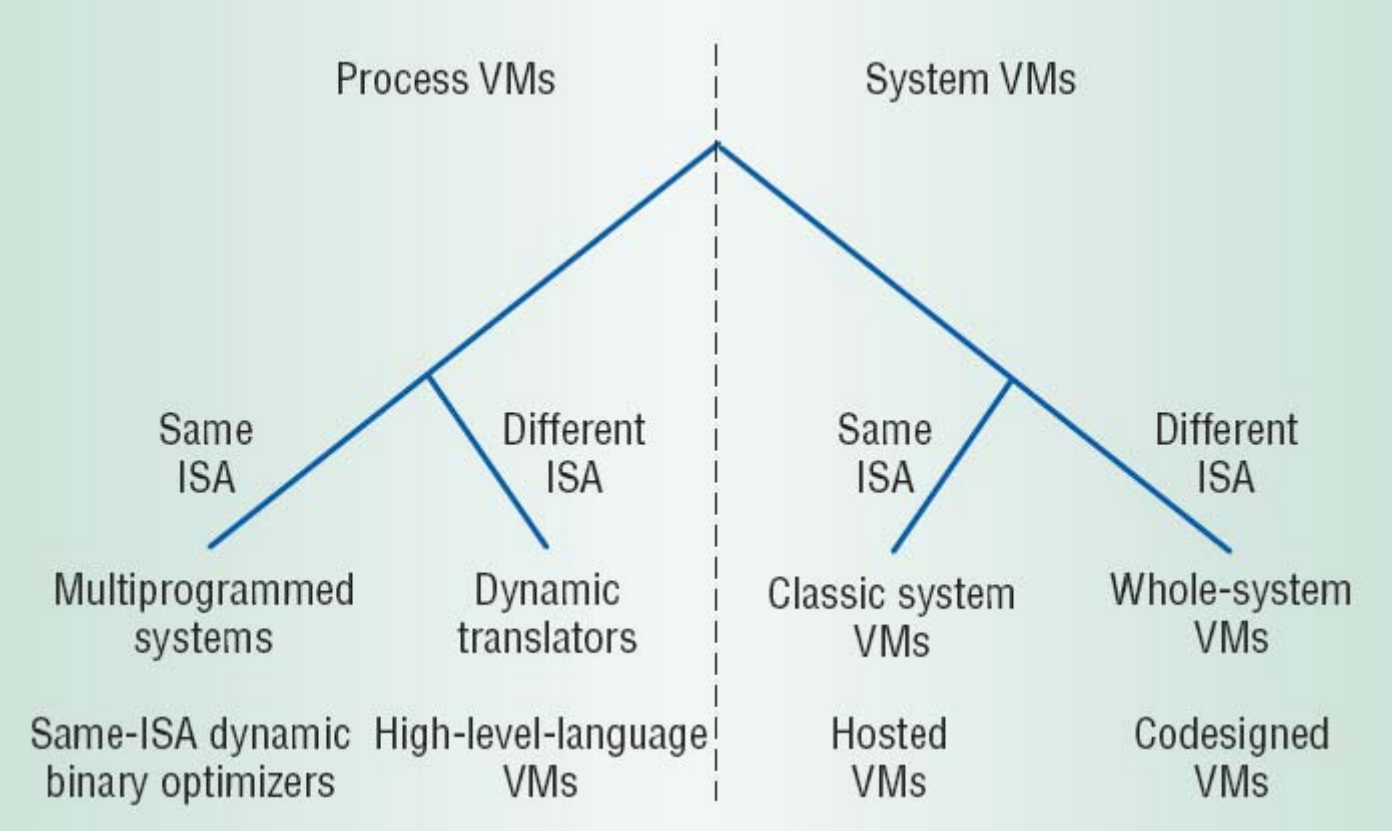


Figure: Virtual machine taxonomy

内容

1. 虚拟化技术基本概念与发展历程
2. 虚拟机技术分类
3. 虚拟机技术实现机制
4. 虚拟机技术引领者—VMware

   Dr. Mendel Rosenblum: What's ahead?

System VMM 实现需求和目标

- **System VMM实现需求**
  - 向Guest OS提供与真实硬件相类似的硬件接口
  - 硬件接口: CPU, Memory, I/O (Disk, Network, Peripheral equipment)
- **System VMM实现目标**
  - Compatibility: ability to run legacy software
  - Performance: low virtualization overhead
  - Simplicity: secure isolation (be free of bugs), reliability (without VMM failure)
  - Various techniques, each offering different design tradeoffs

CPU Virtualization

- **A CPU architecture is *virtualizable***
  - if it supports the basic VMM technique of *direct execution*
- **direct execution**
  - 在VMM保持对CPU的最终控制权前提下，能够让虚拟机中的指令直接在真实主机上运行
  - 实现direct execution需要:
    - virtual machine's privileged and unprivileged code: CPU's unprivileged mode
    - VMM: CPU's privileged mode
    - virtual machine performs privileged operation: CPU traps into the VMM, emulates the privileged operation on the virtual machine state
- 提供可虚拟化的CPU体系框架的关键
  - 提供trap semantics,使得VMM可以安全的、透明地、直接的使用CPU执行虚拟机.

CPU Virtualization Challenge

- 大部分modern CPU 并不支持可虚拟化，如x86
- 需直接访问内存和硬件的操作系统特权代码必须在Ring 0执行
- CPU虚拟化必须在Guest OS下面添加VMM(Ring 0)
- 一些关键指令在非Ring 0权限级执行具有不同语义：不能有效虚拟化，如POPF指令
- 非特权级指令可以查询CPU的当前特权级，x86并不trap这些指令

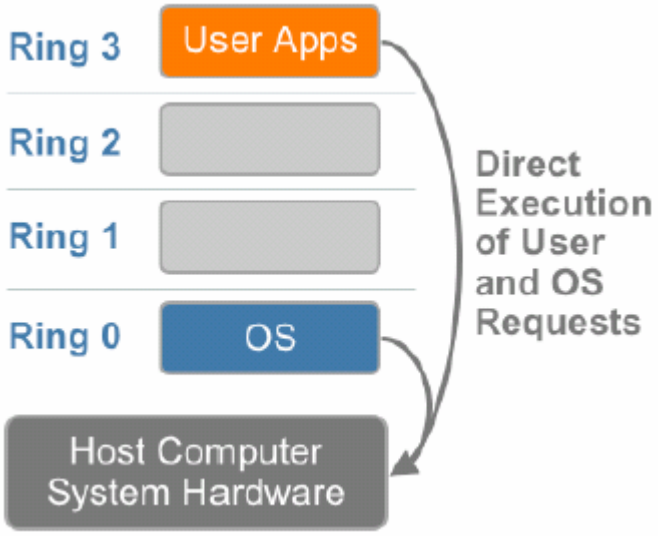


Figure: x86 privilege level architecture without virtualization

CPU Virtualization Tech: Full virtualization using binary translation

- **direct execution combined with fast binary translation**
  - 运行普通程序代码的CPU模式可虚拟化:直接运行
    - High performance
  - 不可虚拟化的特权级CPU模式: binary translator
    - fast: same ISA, negligible overhead
    - dynamic/on-the-fly: opposite - paravirtualization
    - Compatible: guest OS is not aware, run legacy software - main adv.
- **Full virtualization是无需硬件和OS支持实现对特权指令虚拟化唯一选择.**

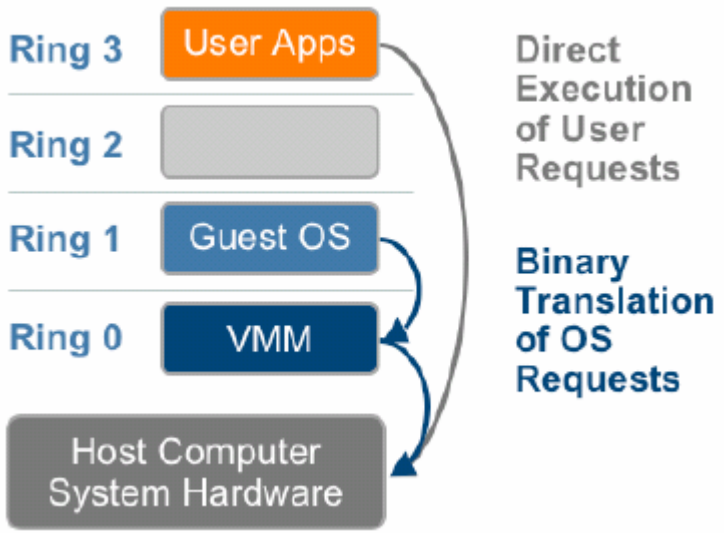


Figure: the binary translation approach to x86 virtualization

CPU Virtualization Tech: Paravirtualization

- **Paravirualization**
  - alongside virtualization
  - OS-assisted virtualization
  - VMM设计者需要定义虚拟机接口，将不可虚拟化的指令替换为可虚拟化/可高效直接执行的等价指令
  - Adv: 消除trap等虚拟化overhead, 性能高效
  - Disadv: 不兼容, 需要修改操作系统, 对商业OS第三方无法移植
- **Xen, Windows Svr 2008, VMware Virtual Machine Interface**

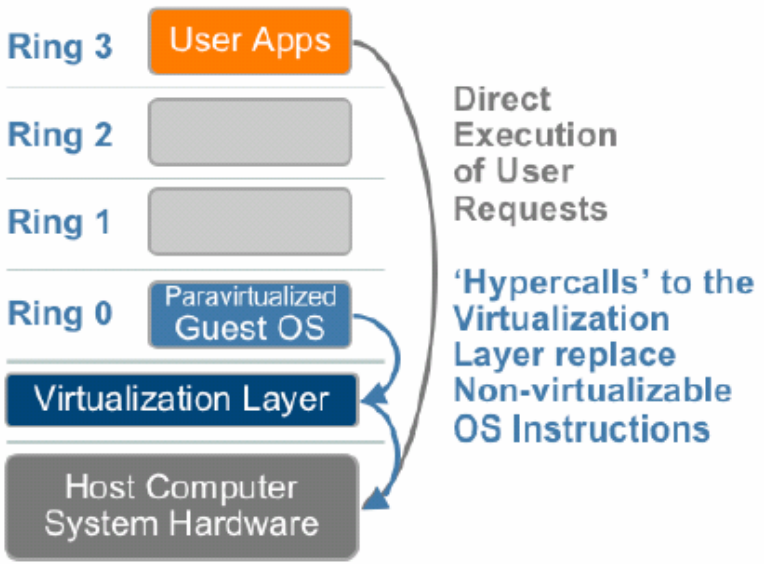


Figure: the paravirtualization approach to x86 virtualization

CPU Virtualization Tech: Native Virtualization / hardware-assisted

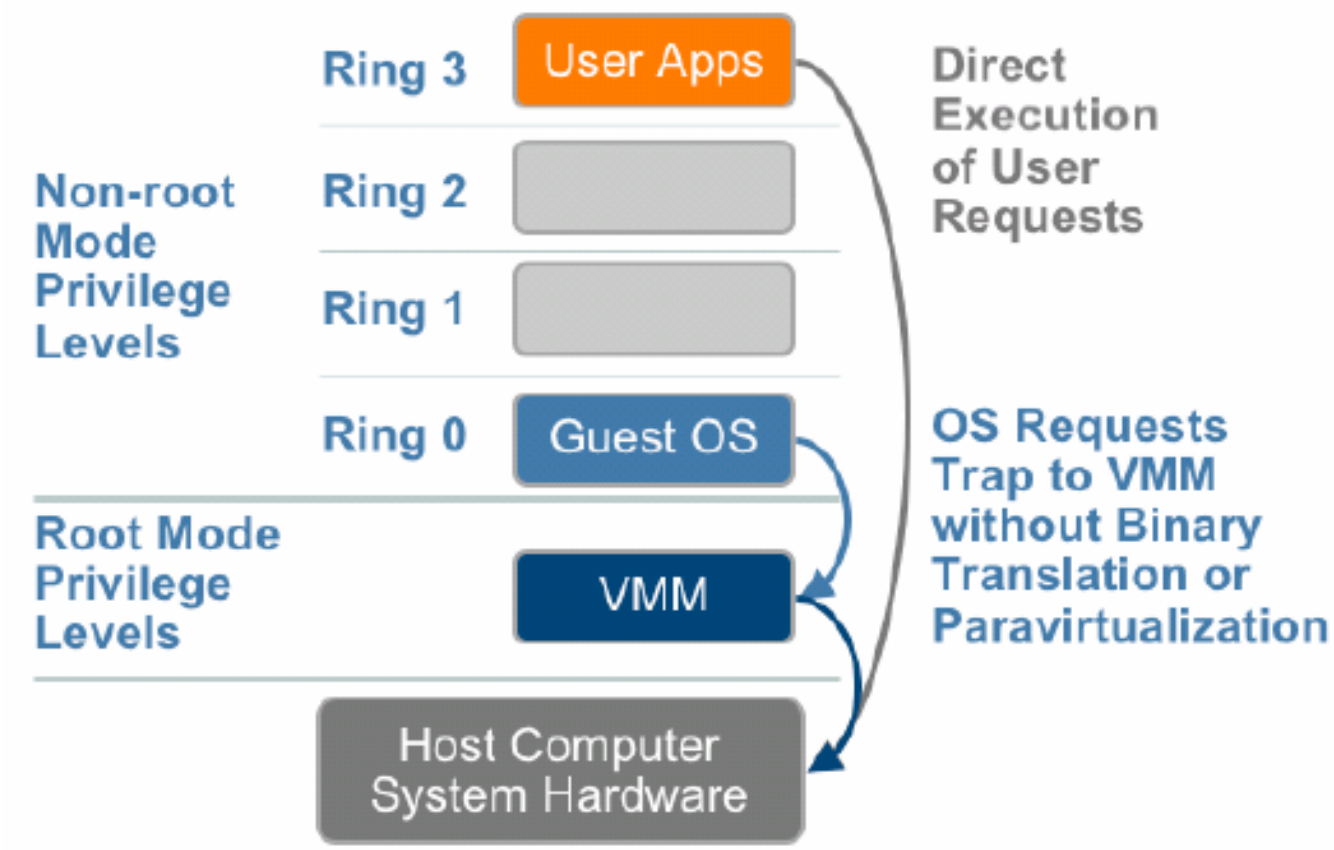


Figure: the native virtualization approach to x86 virtualization

CPU Virtualization Tech: Native Virtualization / hardware-assisted

- Virtualizable CPU
  - 随着虚拟化技术复兴, 硬件厂商快速跟进并推出简化虚拟化技术的CPU新特性
  - Intel Virtualization Technology (VT-x)
  - AMD's AMD-V
- 针对特权代码的虚拟化
  - 在Ring 0之下增加一个新的root mode (VMM)
  - 特权代码会自动trap至hypervisor, 无需paravirtualization或dynamic translation
  - guest state存储于Virtual Machine Control Structures (VT-x) / Virtual Machine Control Blocks (AMD-v)
  - Disadv: high hypervisor to guest transition overhead
  - VMware limited cases (64-bit guest support), Xen 3.0

CPU Virtualization: Summary

	Dynamic Translation	Paravirtualization	Native Virtualization
Compatibility	Excellent	Poor	Excellent
Performance	Good	Excellent	Average
Simplicity	Low	Average	Average

- **Three Current CPU Virtualization Techs**
  - Dynamic Translation, Paravirtualization, Native Virtualization
  - Unique Advs. and Disadvs.
  - Trends
    - More, better hardware assistance: better performance
    - More, better OS assistance: standard virtual machine interface, improve Compatibility for paravirtualization
    - Multi-mode, flexible architecture, select best mode for the workload

More reading: Keith Adams, Ole Agesen, A Comparison of Software and Hardware Techniques for x86 Virtualization, ASPLOS'06.

Memory Virtualization Requirements

- **sharing the physical system memory and dynamically allocating it to virtual machines**
- **Tradeoff between isolation and sharing**
  - Isolation for security and reliability
  - Sharing for performance
- **Page the virtual machine to disk**
  - Like a traditional operating system's virtual memory subsystems

Memory Virtualization Technique

- **similar to the virtual memory support by modern OS**
  - page tables: mappings of virtual memory to physical memory
  - Memory management unit (MMU), translation lookaside buffer (TLB)
- ***shadow pagetables* for memory translation: shadow physical memory → actual machine memory**

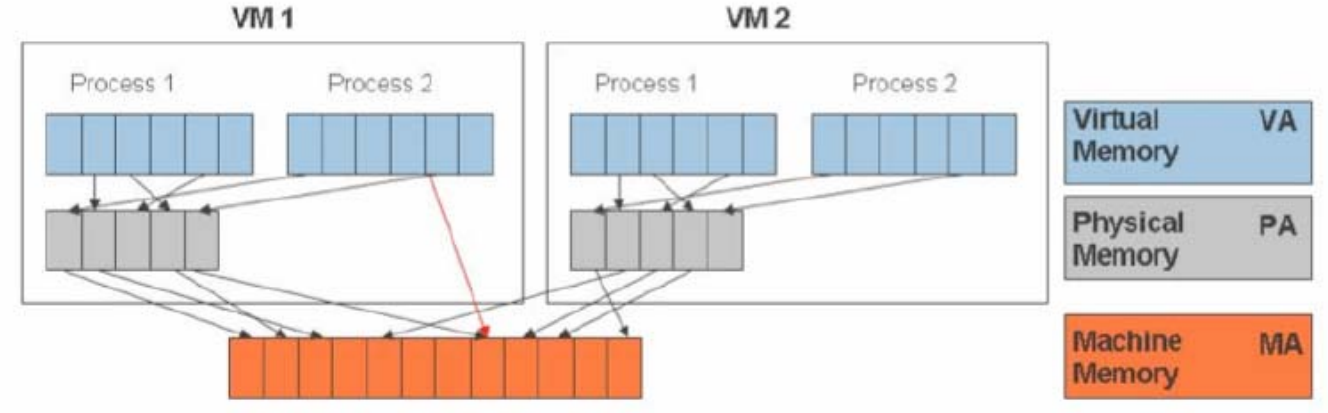


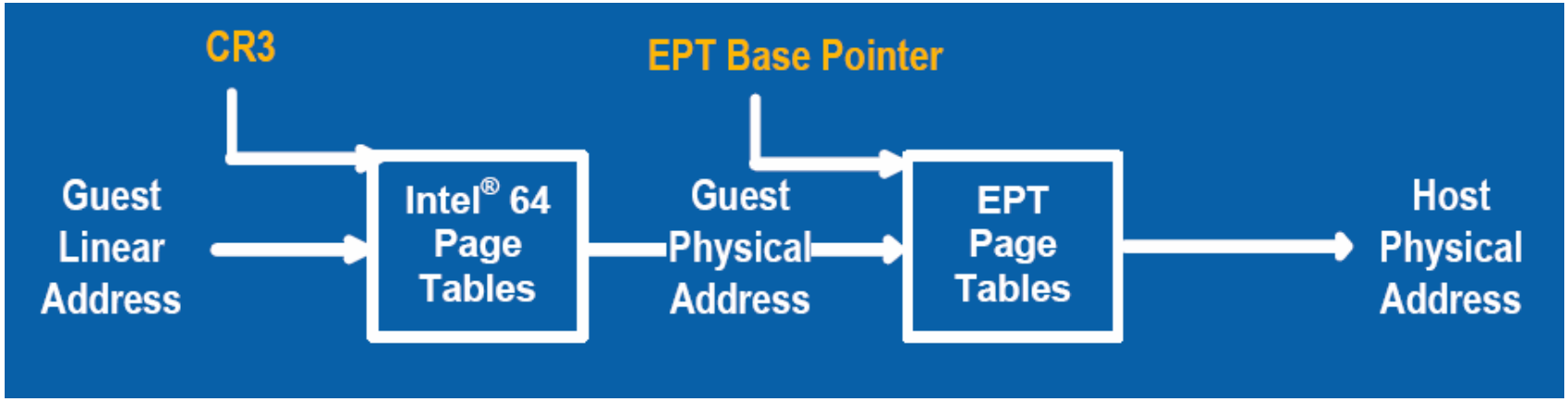
Figure: memory virtualization technique

Memory Virtualization Challenges & Advanced Techniques

- **Two levels of memory translation causes performance overhead**
  - From shadow pagetables to direct pagetables
    - VMM uses TLB to map the virtual memory directly to the machine memory
    - maintaining MMU updates due to mapping changes
- **paging out memory pages to disk**
  - Guest OS have much info about which pages are good candidates
  - paravirtualization-like approach: *balloon process*
- **Running multiple virtual machines waste considerable memory due to redundant**
  - content-based page sharing
  - normal copy-on-write page-sharing scheme

Memory Virtualization New Feature

- **Hardware assisted paging**
  - Intel Extended Page Tables (EPT): March 2008
  - AMD Nested Page Tables (NPT): March 2008
  - Xen and the Art of Virtualization Revisited, Ian Pratt, NSDI 08 Keynote, Apr 2008
    - Current implementations seem to do rather worse than shadow PTs (e.g. 15%)
    - HW will improve: TLBs will get bigger, caching more elaborate, prefetch more aggressive



I/O Virtualization

- **I/O Virtualization Requirements**
  - I/O: Network, Disk, other devices
  - managing the routing of I/O requests between virtual devices and the shared physical hardware
- **Classic Approach in 70s**
  - Native I/O: a channel-based architecture
  - a separate channel processor: safely export I/O device access directly to the virtual machine
  - Advs: very low virtualization overhead for I/O
  - Worked well for the I/O devices of that time: text terminals, disks, card readers, and card punches

I/O Virtualization Challenge

- **Richer and more diverse collection of I/O devices**
  - make virtualizing I/O much more difficult.
- **some devices have extremely high performance requirements**
  - graphics subsystem or network interface
  - even more critical prerequisite: low-overhead virtualization

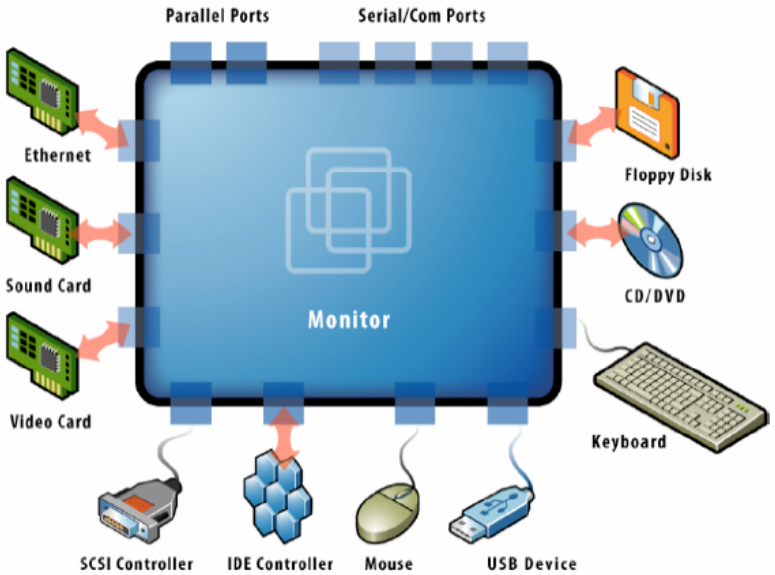


Figure: I/O virtualization

I/O Virtualization Tech: Software based I/O Virtualization

- **Uses device drivers of HostOS**
- **Example:**
  - GuestOS: reads/writes blocks from virtual disk
  - VMM: translates to file reads/writes
  - HostOS: native disk reads/writes
- **Advs:**
  - Simple to install VMM
  - fully accommodates the rich diversity of I/O devices
  - VMM can use the scheduling, resource management, and other services the HostOS environment offers
  - offers rich set of features
- **Disadvs:**
  - Just for hosted VM
  - Performance overhead
  - Lack of resource management support

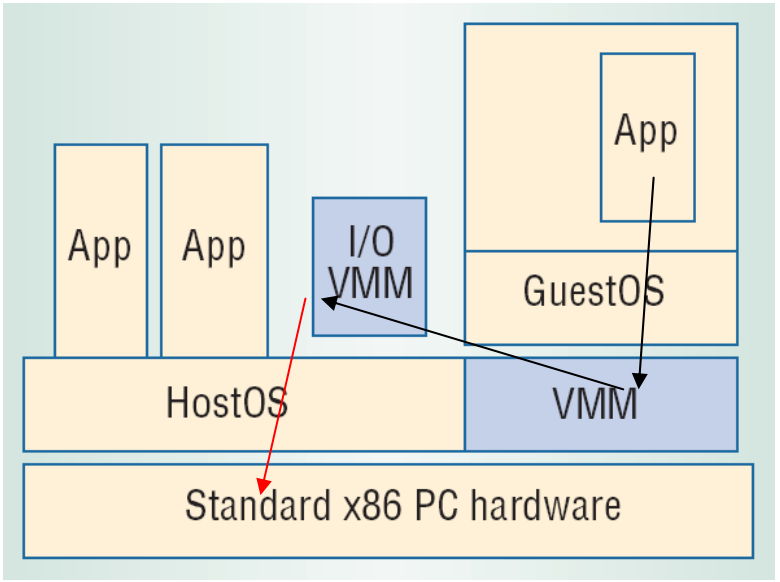


Figure: software based I/O virtualization

I/O Virtualization Tech: Hypervisor Direct I/O Virtualization

- **Requirements**
  - Sophisticated scheduling and resource management
  - highly optimized I/O performance for network and storage devices
- **Running directly on the hardware**
  - VMware ESX Server: use device drivers from the Linux kernel
  - Paravirtualization: the ability to export special highly optimized virtual I/O devices that don't correspond to any existing I/O devices.

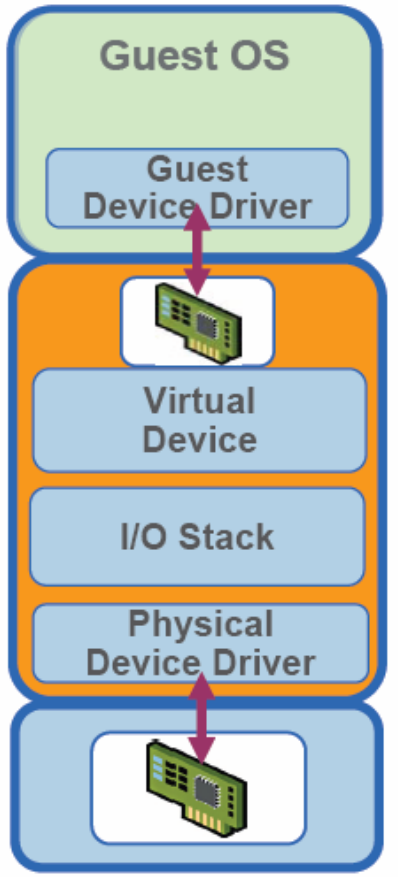


Figure: hypervisor direct I/O virtualization

I/O Virtualization Tech: Pass-through I/O Virtualization

- **Guest drives device directly**
  - Without CPU, Effectively eliminating I/O virtualization overhead
  - achieve high-performance I/O device virtualization
  - Discrete I/O devices → channel-like I/O devices
    - support multiple virtual interfaces
    - PS2 → USB, IDE → SCSI, SATA
- **I/O devices perform DMA requires memory remapping**
- **routing device completion interrupts to the correct virtual machine**
- **Benefits: Performance, improved security and reliability gained from removing device drivers**
- **Future Hardware feature supports**
  - I/O MMU for DMA address translation and protection (Intel VT-d, AMD I/O MMU)
  - Partitionable I/O device for sharing (PCI-SIG IOV SR/MR specification)

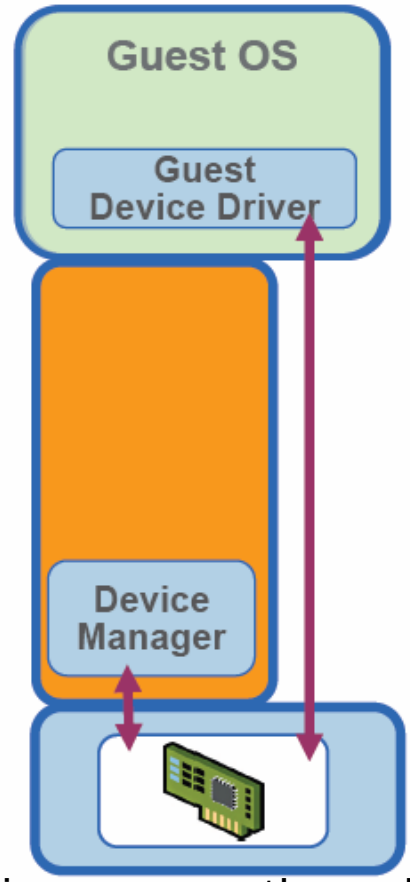


Figure: pass-through I/O virtualization

I/O Virtualization Summary

- **Tradeoff between functionality and efficiency**
  - Virtualized I/O provides rich functionality
    - Software-based Split
    - Dypervisor Direct
  - Pass-through I/O minimizes CPU utilization

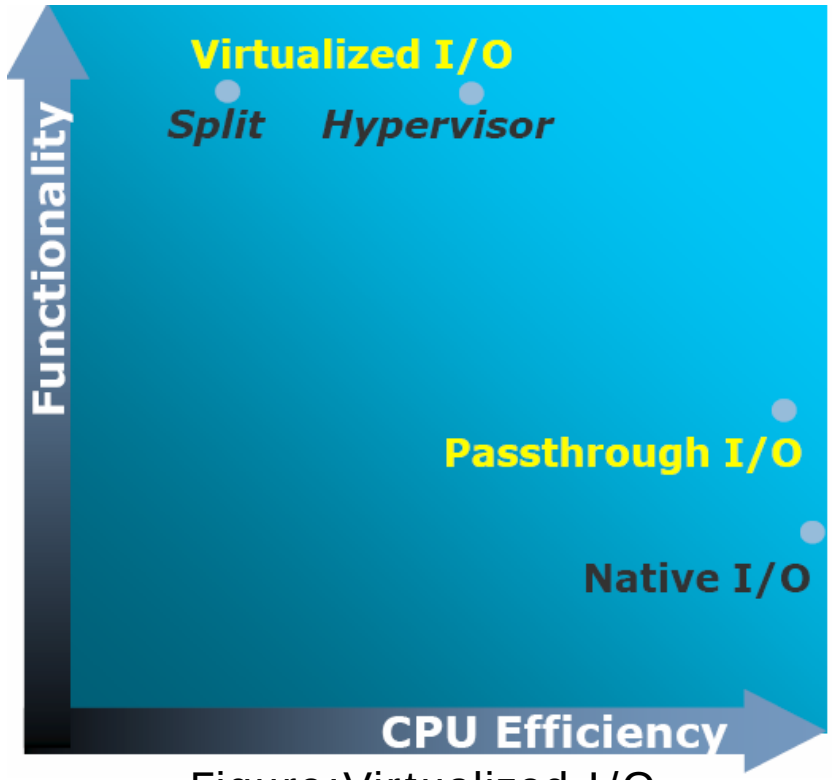


Figure:Virtualized I/O VS. pass-through I/O

内容

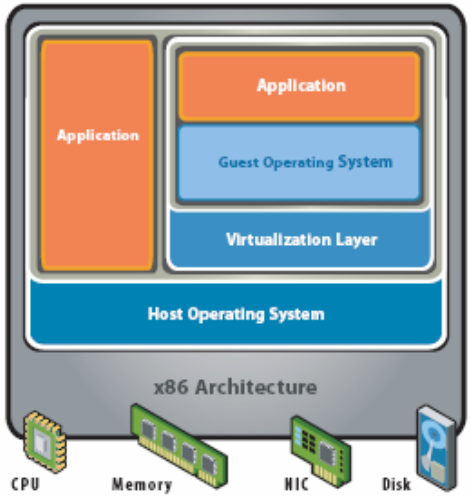
1. 虚拟化技术基本概念与发展历程
2. 虚拟机技术分类
3. 虚拟机技术实现机制
4. 虚拟机技术引领者—VMware

   Dr. Mendel Rosenblum: What's ahead?

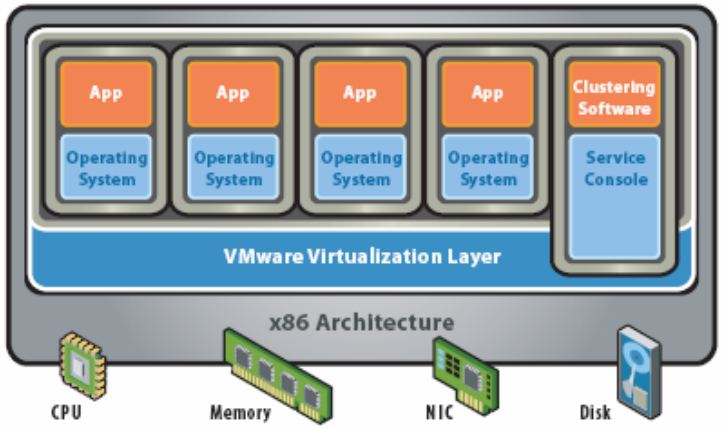
VMware Inc.

- 虚拟化技术引领者－**VMWare Inc.**
  - Founded at 1998, from Stanford
  - 2003年被EMC公司收购
  - 全球虚拟化市场－EMC/VMware过半市场份额, M$/Connectix, Citrix/Xen, Sun, Oracle
- **VMware**大佬们
  - President & CEO: Diane Greene, Berkeley
  - Chief Scientist: Dr. Mendel Rosenblum, Standford

VMware的虚拟机技术



Hosted Architecture



Bare-Metal (Hypervisor) Architecture

- Hosted Architecture: ACE, Workstation, Server
- Hypervisor Architecture: ESX Server
- CPU Virtualization: Full Virtualization with BT / ParaVirtualization / Native Virtualization; Multi-Mode VMM Architecture
- Memory Virtualization: shadow pagetables
- I/O Virtualization: software-based, hypervisor direct for storage / net

VMware's paravirtualization

- **Xen's paravirtualization**
  - Paravirtualized operation system
    - major Linux distributions are starting to bundle paravirtualization into the OS kernel
    - Windows Server 2008
  - strong dependency between paravirtualized OS with hypervisor
  - Delivers performance benefits with maintenance costs and incompatibility
- **VMware's Transparent Paravirtualization**
  - proposed standard communication mechanism between GuestOS and hypervisor: Virtual Machine Interface
  - Balances performance benefits with maintenance costs

Virtual Machine Interface

- **VMI standard**
  - layer between the hypervisor and the paravirtualized GuestOS
  - Transparent paravirtualization
  - Same GuestOS can run either natively or virtualized on any compatible hypervisor
- **Implementation**
  - paravirt-ops into kernel since 2.6.20
  - VMI backend since 2.6.22
  - Raise compatibility from poor to good

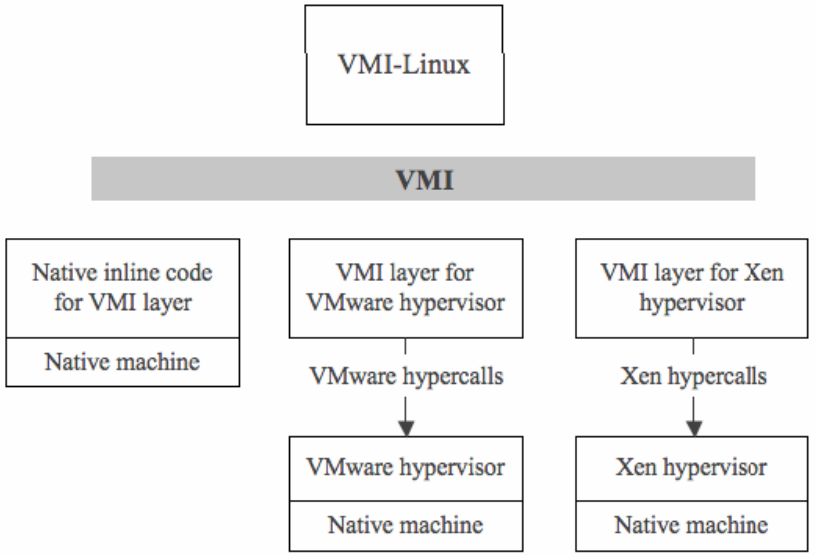
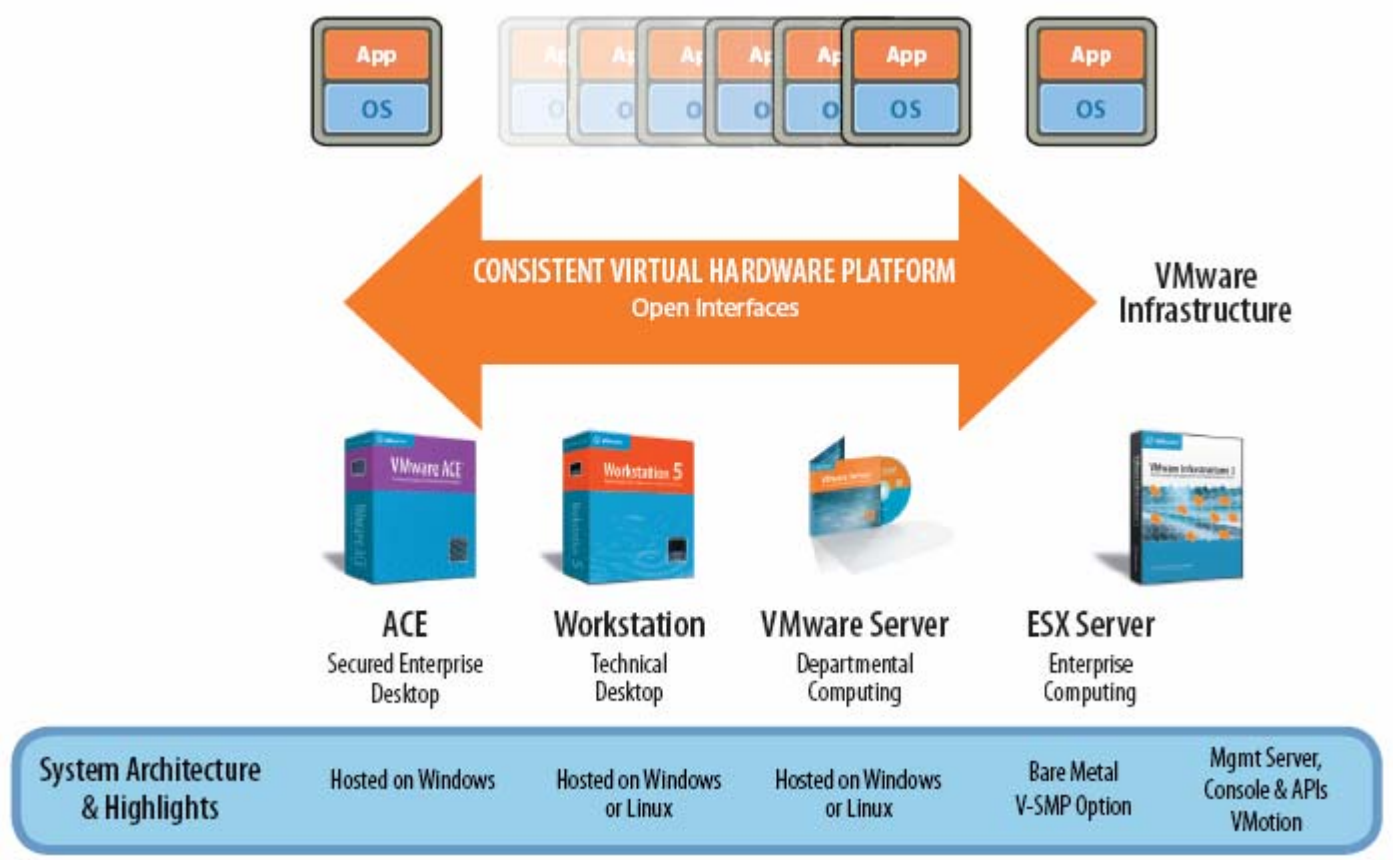


Figure:Virtual Machine Interface

VMware的产品线和技术解决方案

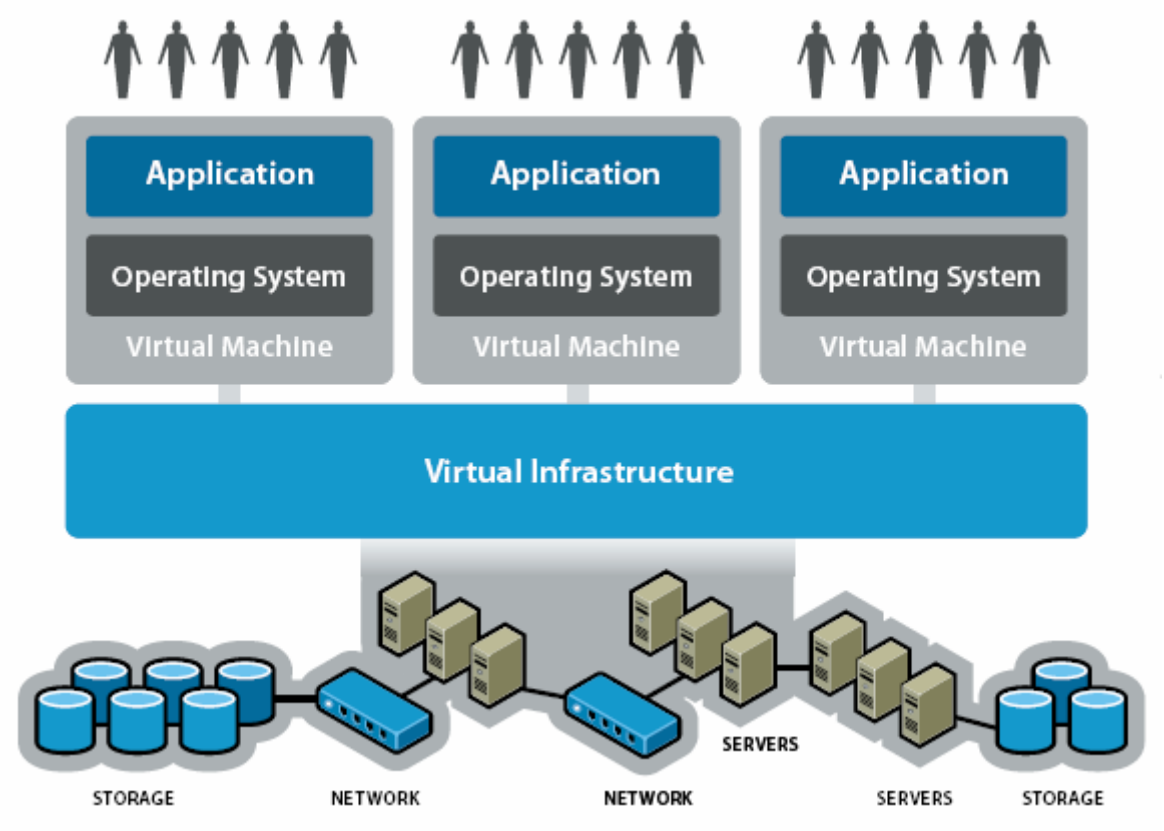


What's ahead? – Server side

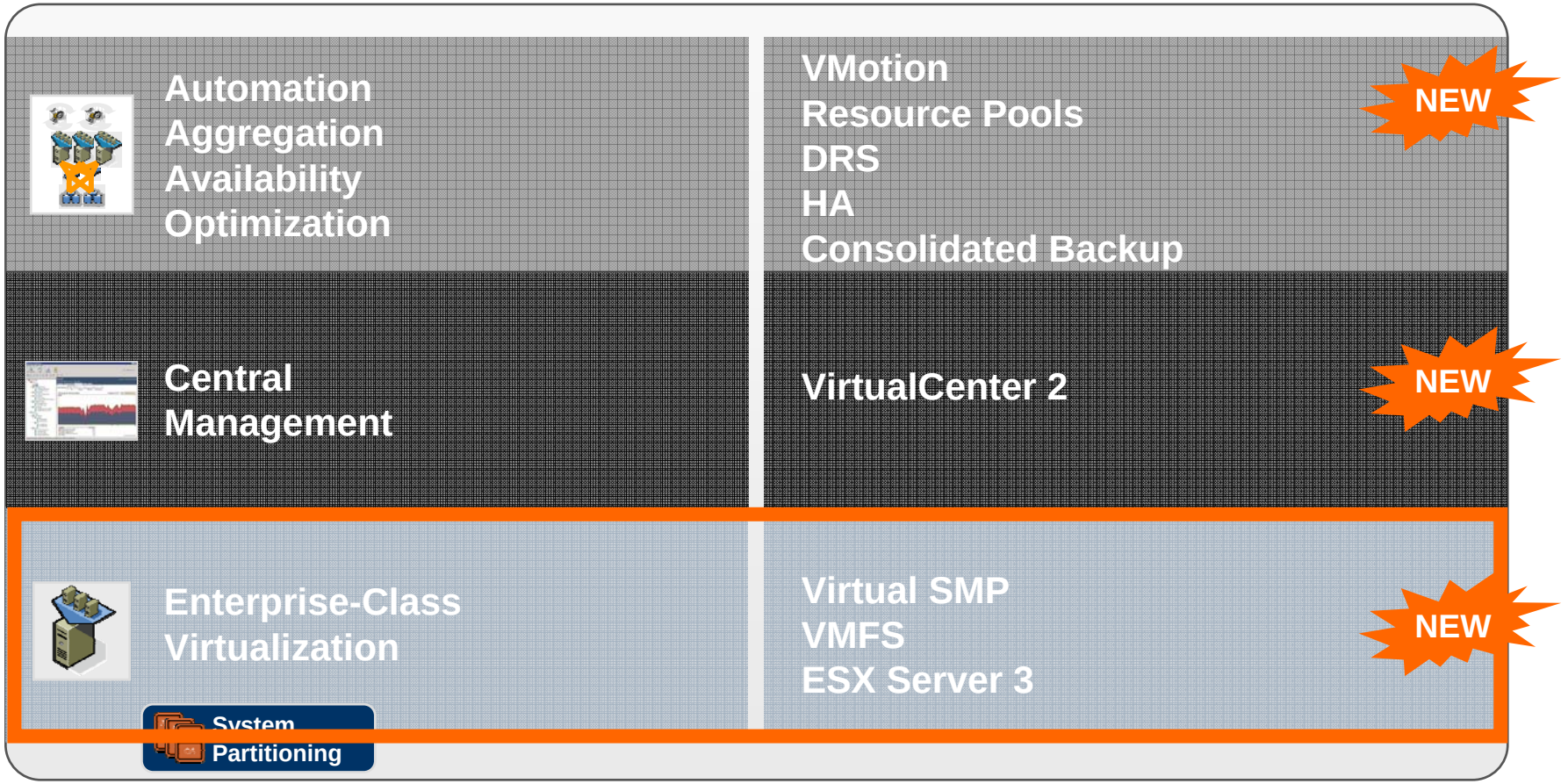
- What's ahead? - Dr. Mendel Rosenblum, Virtual Machine Monitors: Current Technology and Future Trends. IEEE Computer, May 2005.

- Quickly provision, monitor, and manage VMs from a single console
- create new servers from an existing template, view computers simply as part of a resource pool
- Hot migration: move rapidly between physical machines with continuous service availability
- Handle hardware failures, maintenance, upgrade easily without service interruptions
- creates and destroys virtual machines on demand

VMware Virtual Infrastructure



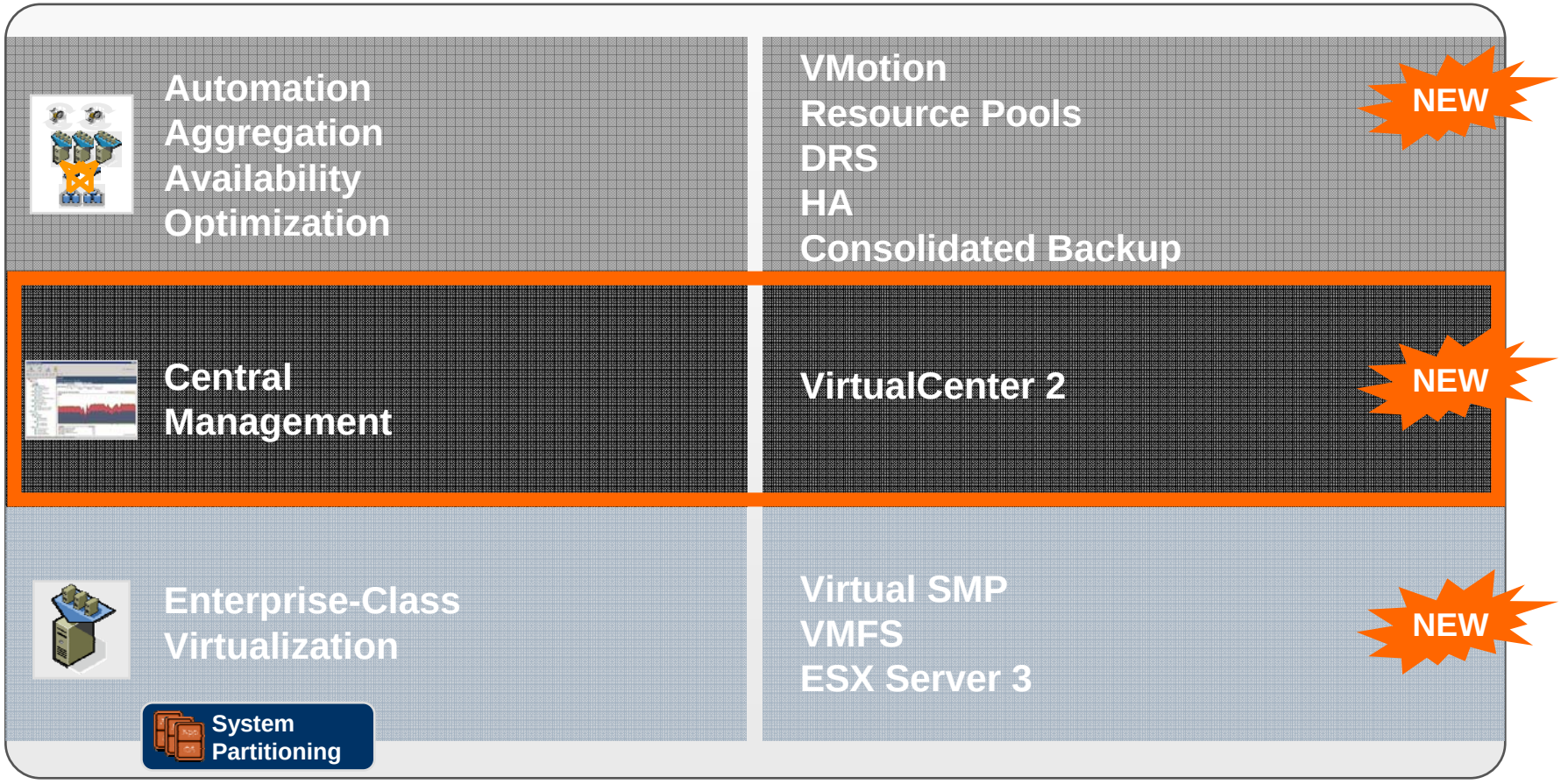
VMware Infrastructure 3



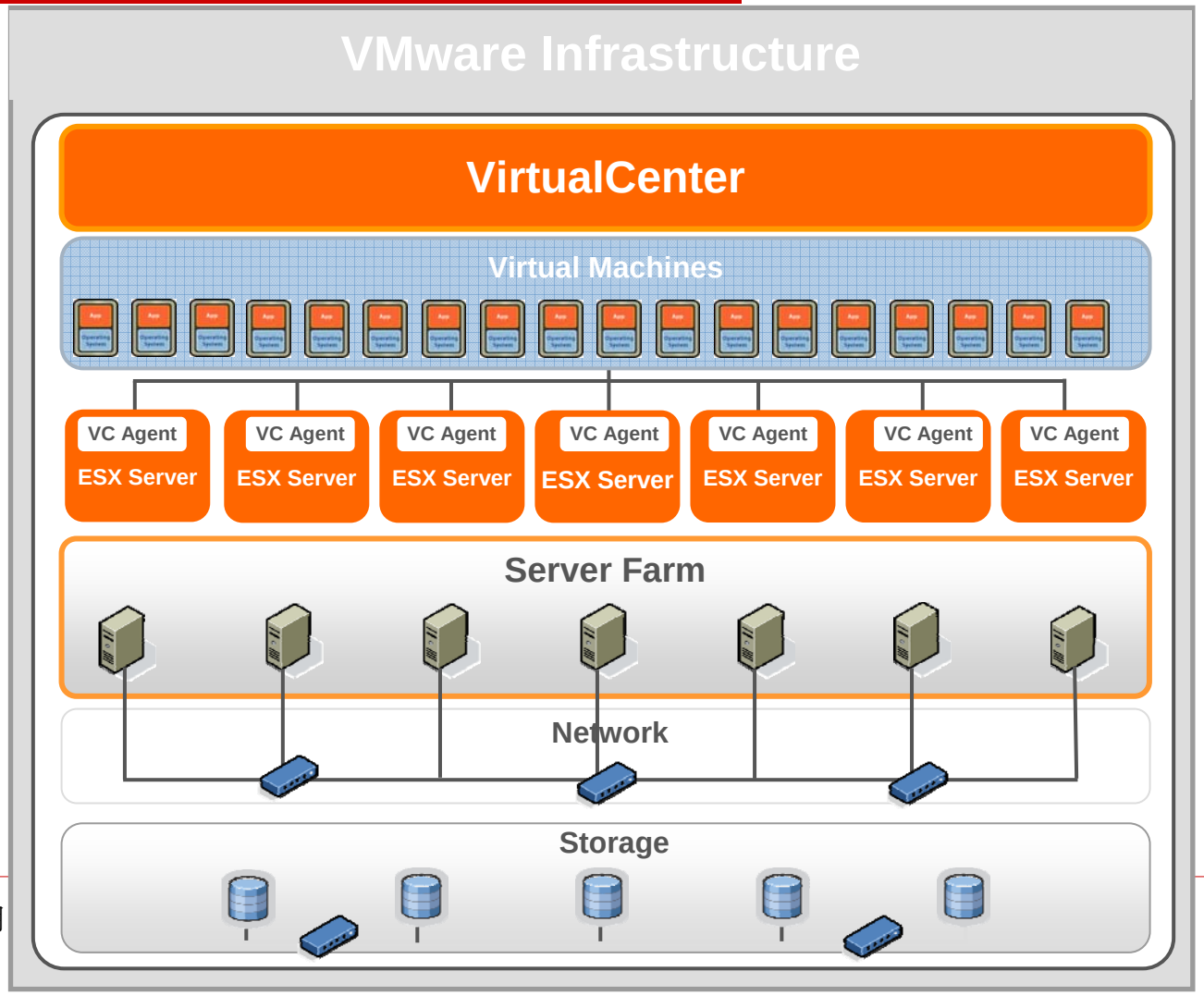
Requirements for enterprise-class virtualization

- **Virtualization of servers, storage and network**
  - Not just servers, storage and network virtualization as well!
- **Reliability, scale and performance**
  - ESX Server at a customer site: 800 days continuous uptime!
- **Interoperability and certification**
  - Extensive certification, testing and interoperability throughout the stack
- **Support for enterprise workloads**
  - Up to 16GB RAM and 4 virtual CPUs per VM

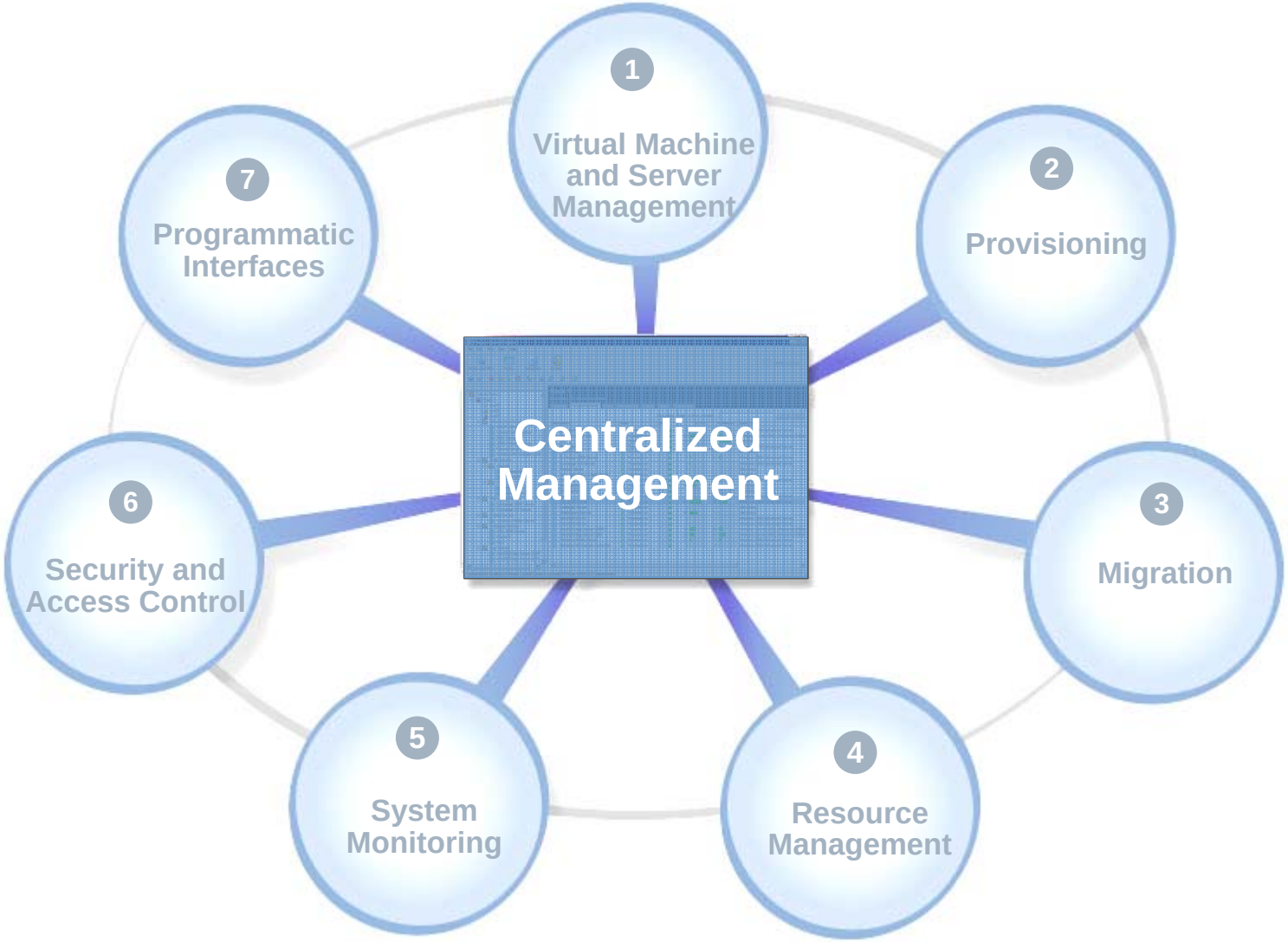
VMware Infrastructure 3



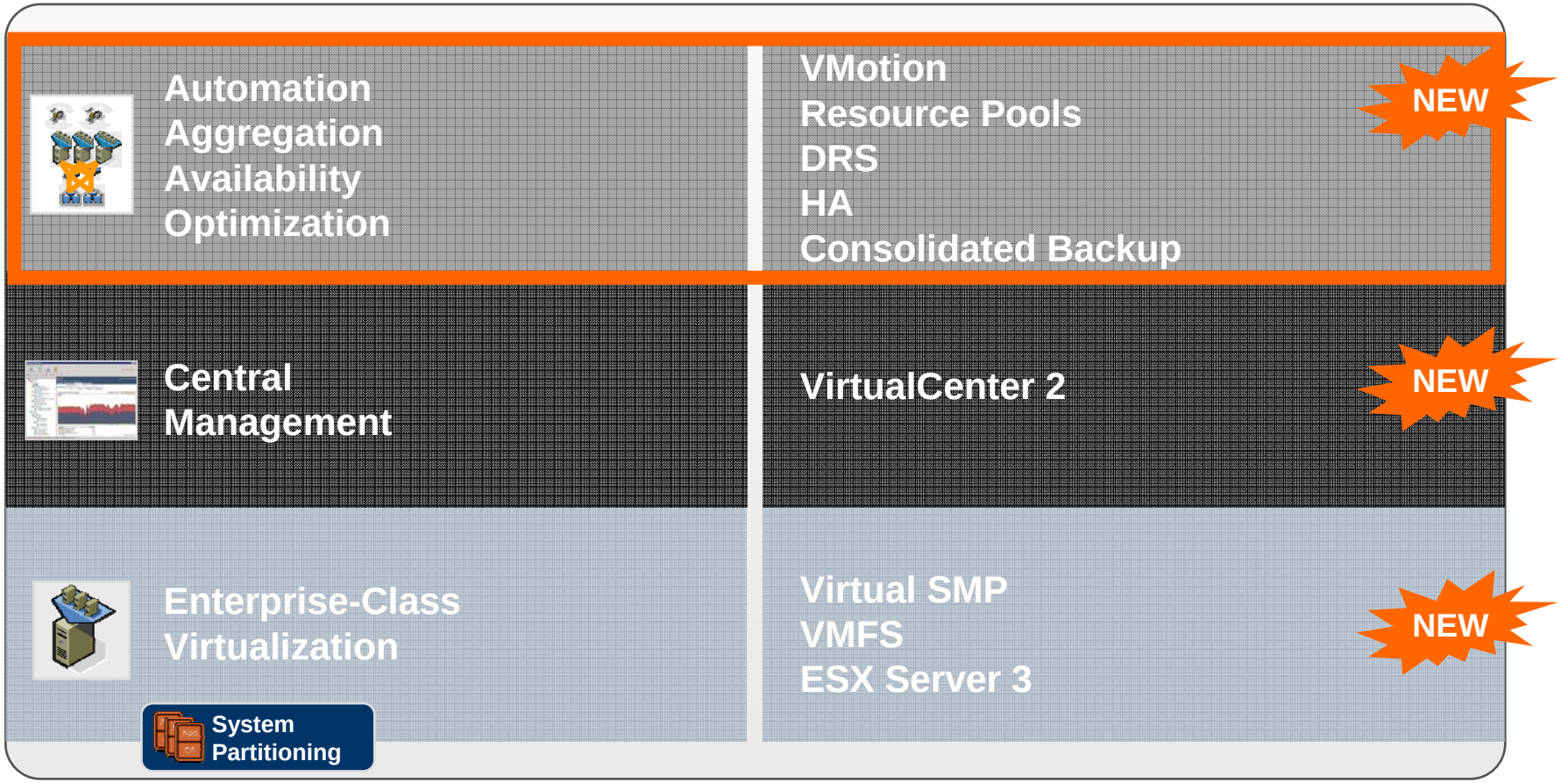
VMware Infrastructure management



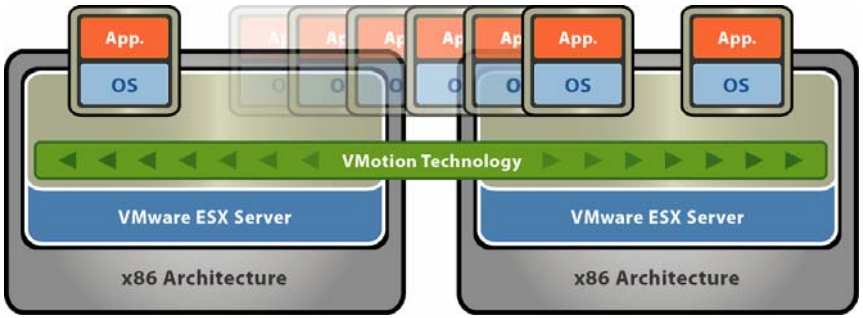
VirtualCenter—Key functionality



VMware Infrastructure 3



Live migration of virtual machines with VMotion



What is it?

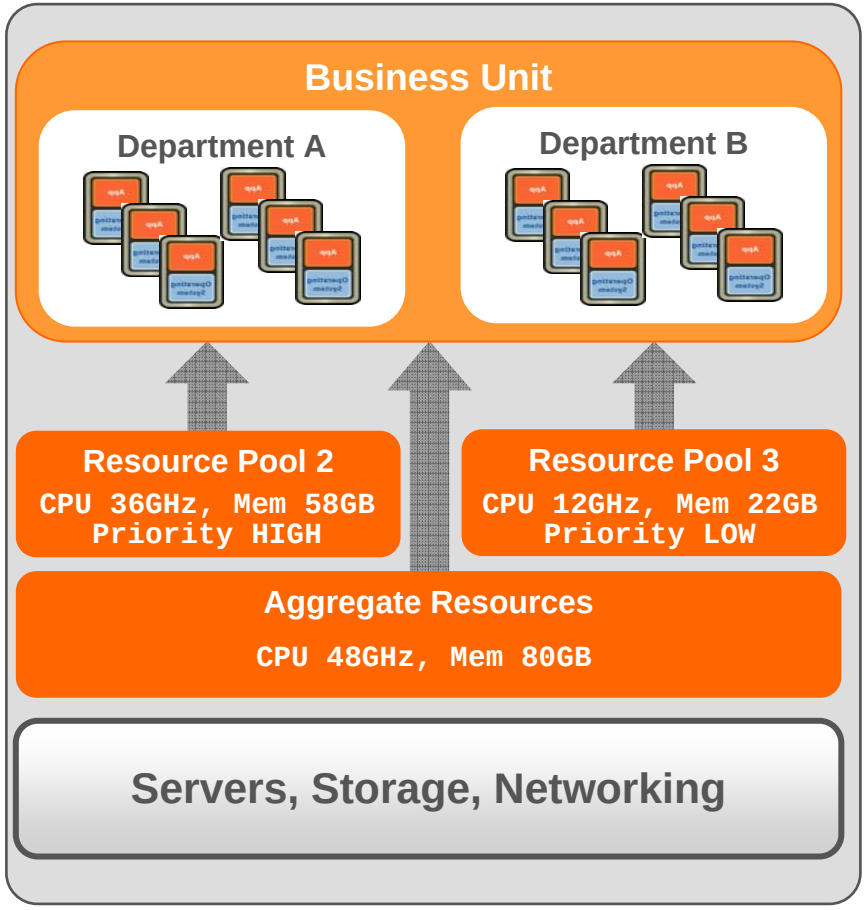
- Live migration of virtual machines with VMware VMotion

Customer Impact

- Zero downtime maintenance
- Continuous service availability
- Complete transaction integrity
- Supported on Fibre Channel and iSCSI SAN and NAS

More Reading: Michael Nelson, Beng-Hong Lim, and Greg Hutchins, Fast Transparent Migration for Virtual Machines, Proceedings of USENIX '05, Anaheim, California, USA, April 2005

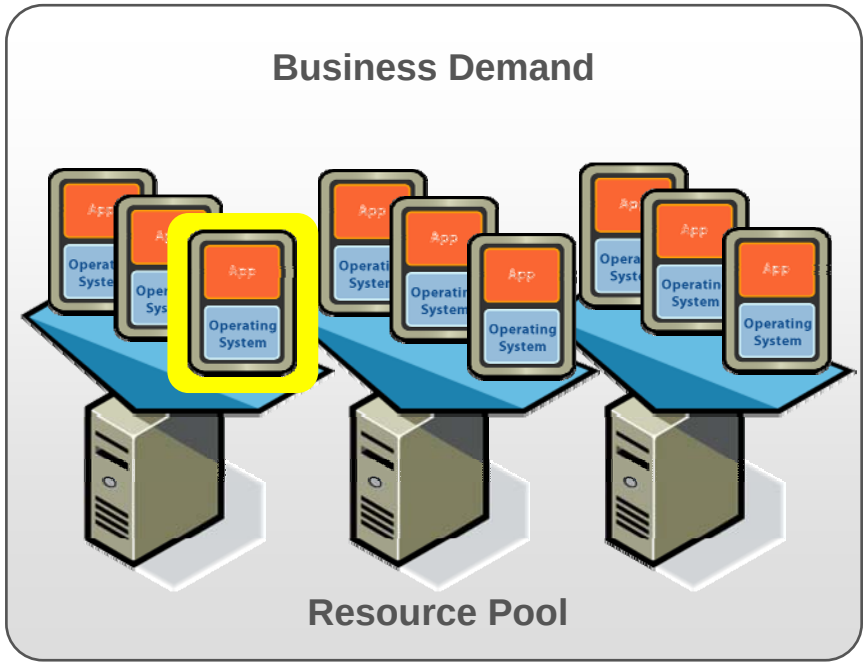
Resource pools



Customer Impact

- Failed server means fewer resources not a failed application
- Enables high availability across the infrastructure
- Provides service level assurance
- Dedicated (virtual) infrastructure for each business unit; central IT retains control over hardware
- Delegation of resource and virtual machine management down to the business unit

Resource optimization with VMware Distributed Resource Scheduler



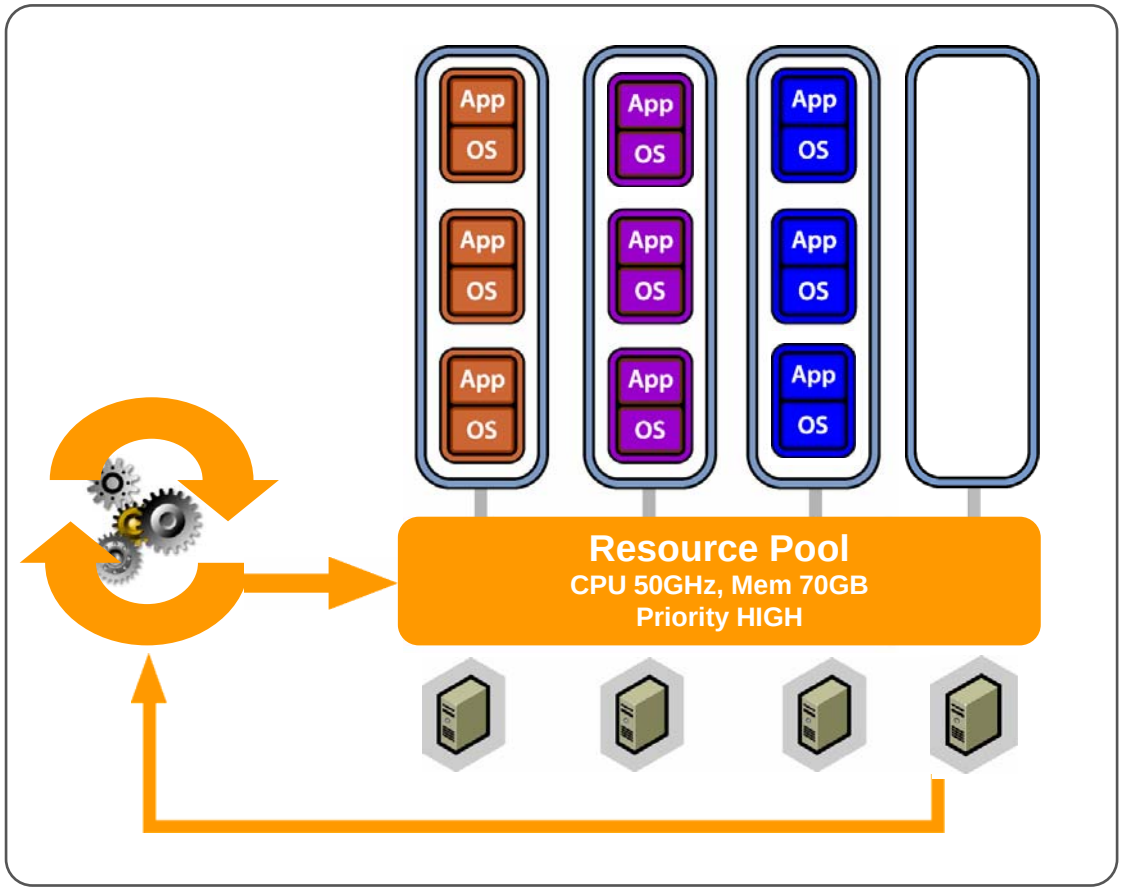
What is it?

- Dynamic balancing of computing resources across resource pools
- Intelligent resource allocation based on pre-defined rules

Customer Impact

- Align IT resources with business priorities
- Operational simplicity; dramatically increase system administrator productivity
- Add hardware dynamically to avoid over-provisioning to peak load
- Automate hardware maintenance

Capacity on demand with VMware DRS



- Provisioning is “fire and forget”
- Easily add more capacity
- Avoid over-provisioning to peak load

Zero-downtime maintenance using VMware technology

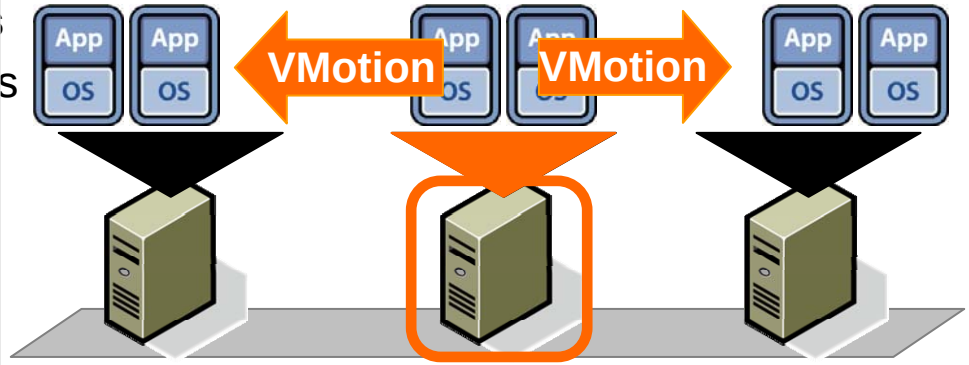
Use VMotion to evacuate hosts

- Move running applications to other servers without disruption
- Perform maintenance at any time of day

NEW

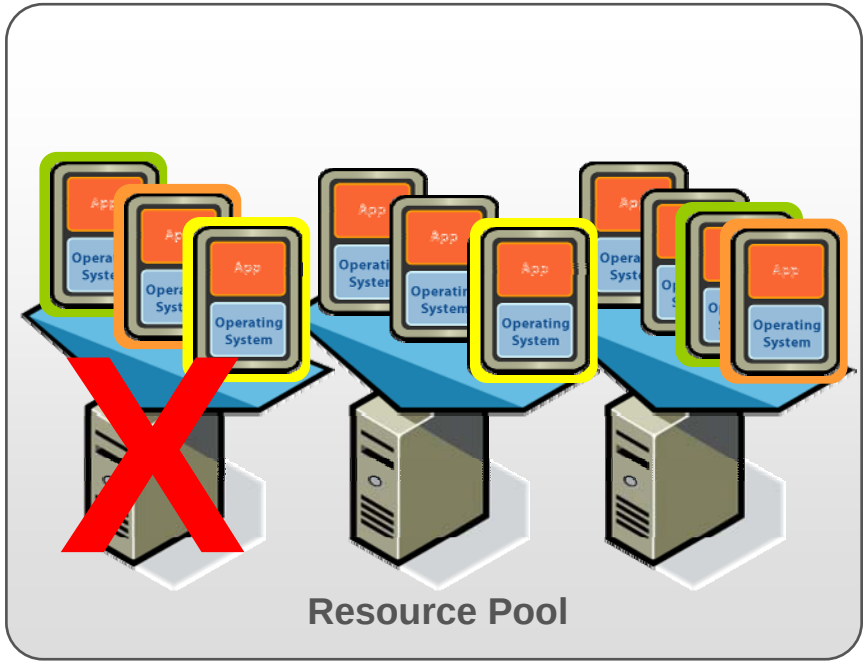
Automate with DRS maintenance mode

- Automates moving virtual machines to other hosts
- Automates re-balancing after maintenance complete



1. Activate Maintenance Mode for physical host
   - Shut down idle host and perform maintenance
2. DRS migrates running virtual machines to other hosts
   - Restart host; DRS automatically rebalances workloads

Ensure high availability with VMware HA



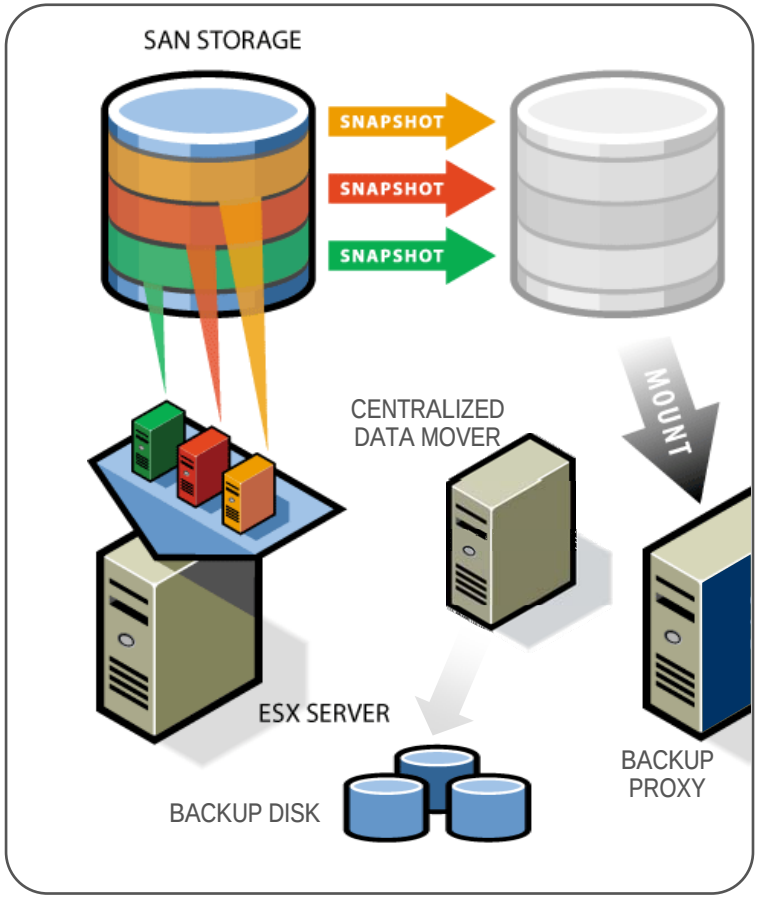
What is it?

- Automatic restart of virtual machines in case of server failure

Customer Impact

- Cost effective high availability for all applications
- No need for dedicated stand-by hardware
- None of the cost and complexity of clustering

Protect data with VMware consolidated backup



What is it?

- Centralized agentless backup for virtual machines
  - Move backup out of the virtual machine
  - Eliminate backup traffic on the local area network
- Pre-integrated with major 3rd-party backup products

Customer Impact

- Perform backup in the middle of the day

What's ahead? – Beyond the machine room

- From server room to desktop
- provide a powerful unifying paradigm for restructuring desktop management
- Virtual machines could also significantly change how users think about computers
- increased mobility: migrating a user's entire computing environment over the local and wide area
- increasingly dynamic character: require more dynamic network topologies, Virtual switches, virtual firewalls, and overlay networks

VMware Virtual Desktop Infrastructure

- **An Integrated Desktop Virtualization Solution**
- **VMware VDI, 2007**
  - VMware Infrastructure 3
  - Virtual Desktop Manager
  - Thin clients (RDP)
- **End-to-end Virtual Desktop Infrastructure Functionality**
- **Simplified Desktop Management & Secure Provisioning with VMware VDM**
- **Familiar End-user Experience**
- **Seamless Integration with VMware Infrastructure 3**

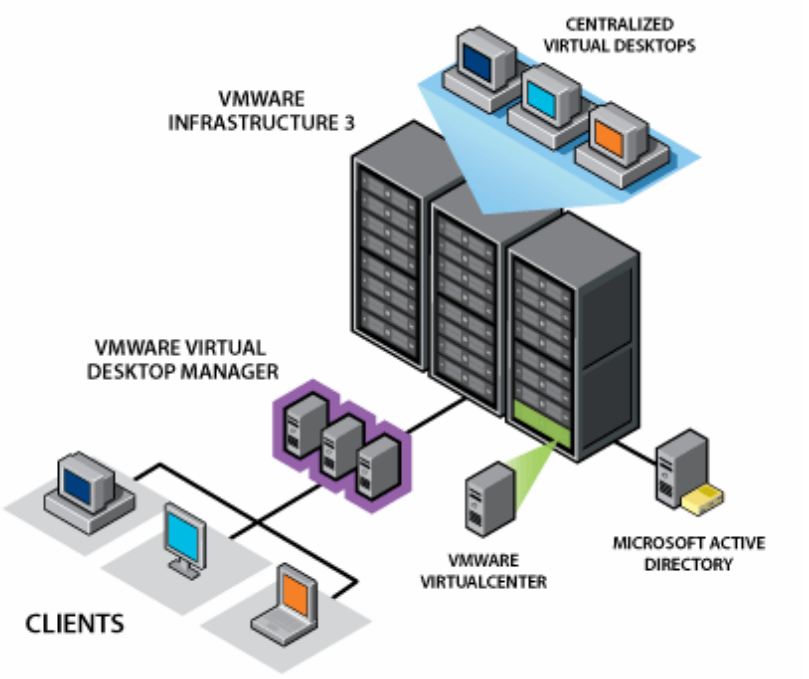


Figure: VMware Virtual Desktop Infrastructure

What's ahead? – software distribution

- VMM层支持软件厂商发布包含复杂软件环境的整个虚拟机
  - Oracle VM已发布超过10,000份
- 新软件发布方式 - Virtual appliances
  - VMware ACE: 支持生成virtual appliances, 并动态设置使用策略
  - 需要新的软件license模式
    - 传统软件发布license模式: 只能在特定硬件上运行
    - 以虚拟应用设备模式发布软件: 较严格受限license模式，用户期望更便捷、便宜的VM license模式

VMware ACE & Virtual Appliances

- **VMware ACE**
  - Workstation 6 with ACE Option Pack for create VM package
  - ACE Management Server
- **VMware Server (free) / VMware Workstation**
- **Virtual Appliances**
  - VMware Virtual Appliance Marketplace
  - http://www.vmware.com/appliances/

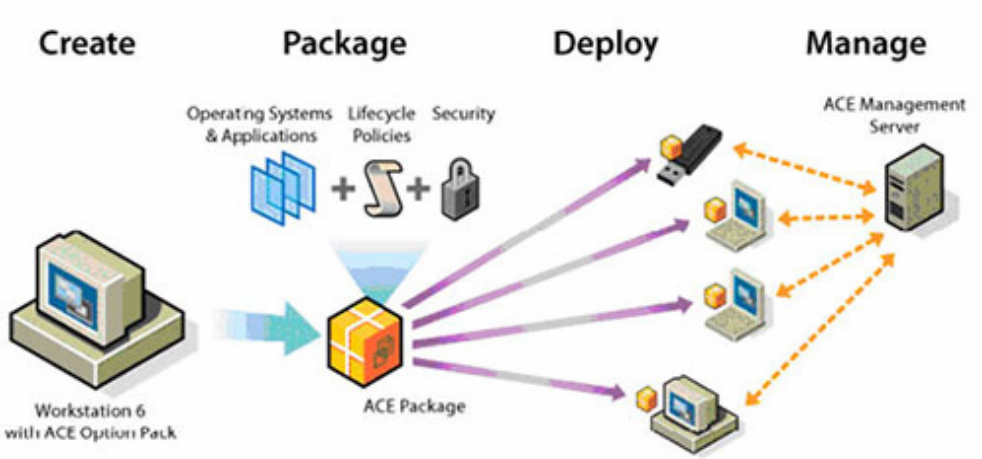


Figure: VMware ACE

Figure: Virtual Appliance

Benefits for Vendors

- **Reduced supported OS platforms**
  - First of choice – Ubuntu JeOS (“Juice”)
- **Expand footprint of certified hardware**
- **Focus purely on developing and optimizing your application**
- **Reduced the length and cost of your sales cycle**
  - VMware Virtual Appliances Marketplace
  - VMware Certified Virtual Appliances Program

What's ahead? – Security Improvements

- **VMMs offer**
  - the potential to restructure existing software systems to provide greater security
  - facilitating new approaches to building secure systems
- **Host-based security mechanisms**
  - Current OS provides poor isolation, subject to attack
  - VMM provides strong isolation
    - same? (greater) functionality but with much stronger resistance to attack
    - Research examples: Livewire, ReVirt, TTAnalyzer, VMWatcher(CCS'07), VMscope(RAID'07), ...
- **Network-based security mechanisms**
  - VMM provides an attractive way to quarantine the network: virtual firewall, virtual IDS, etc.

What's ahead? – Security Improvements (2)

- **Well suited for constructing high-assurance systems**
  - NSA's NetTop architecture, uses VMM to isolate multiple environments
  - run multiple software stacks with different security levels
- **Well suited for building trusted computing, example**
  - Terra system: running multiple virtual machines of different security levels simultaneously
  - VMM can authenticate software running inside a virtual machine to remote parties, in a process called attestation.
- **flexible resource management that VMMs provide can make systems more resistant to attack**
  - rapidly replicate virtual machines and dynamically adapt to large workloads
  - dealing with the scaling demands that flash crowds and distributed denial-of-service attacks can impose.

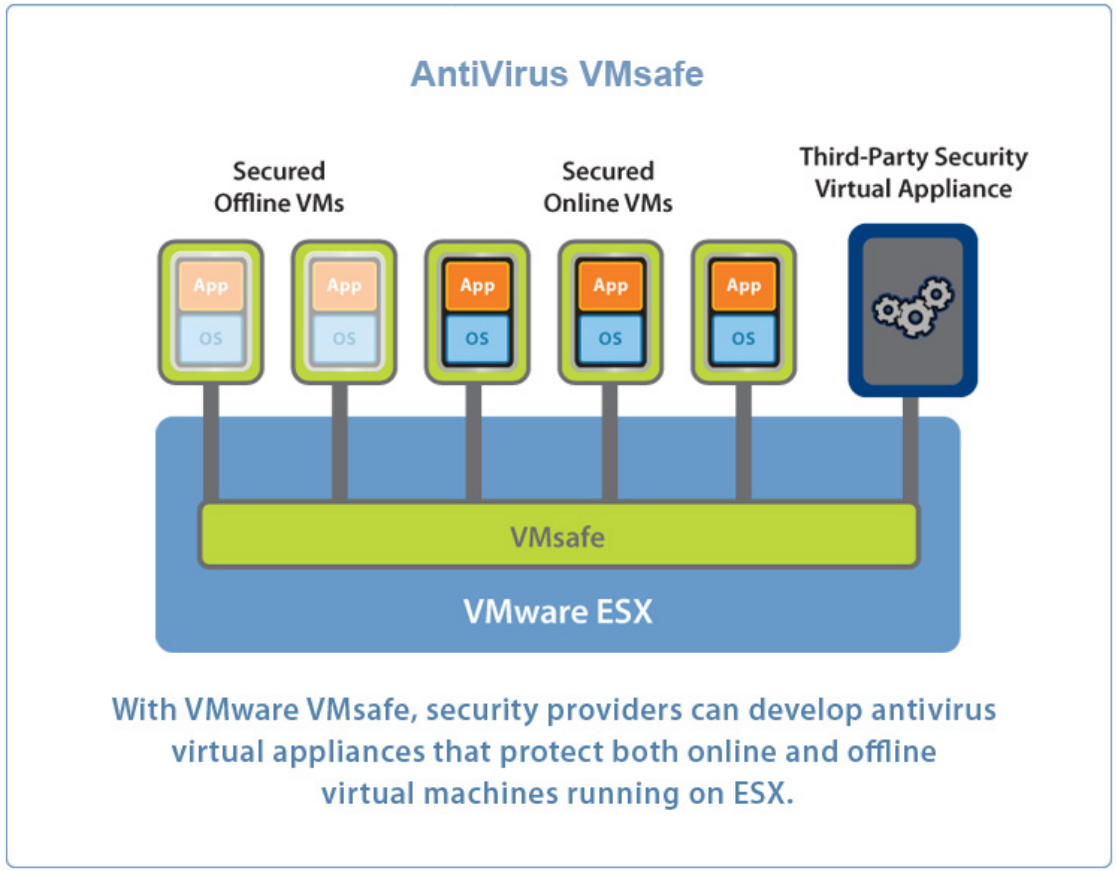
VMware VMSafe

- **A Security Technology for Virtualized Environments**
- **enable a rich ecosystem of third-party security solutions, by providing a set of security APIs**
  - fine-grained visibility over virtual machine resources
  - monitor every aspect of the execution of the system
  - stop previously undetectable viruses, rootkits and malware before they can infect a system
  - introspection of virtual machine memory pages and CPU states
  - filtering of network packets inside hypervisors, or VM
  - in-guest, in-process APIs that enable complete monitoring and control of process execution
  - Guest VM disk files can be mounted, manipulated and modified as they persist on storage devices.

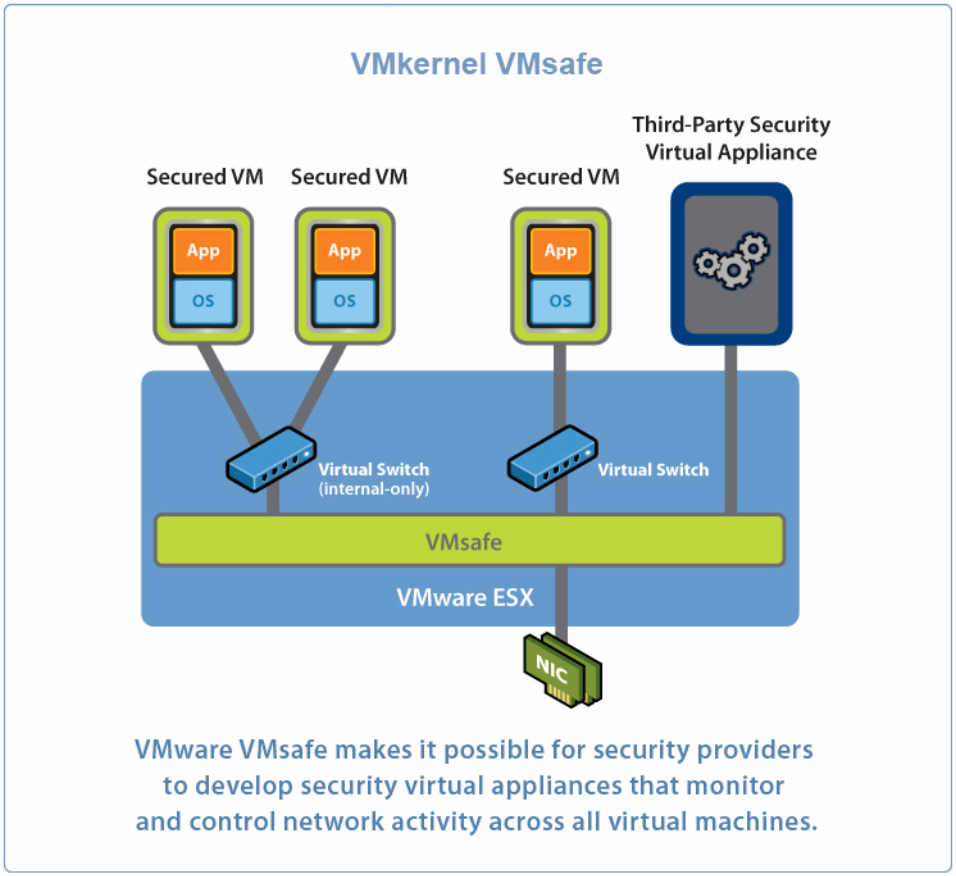
VMware VMSafe Benefits

- **an open, interoperabl**
  set of technologies wi
  can provide innovativ
  for virtual machines.
  - Stronger security
  - greater visibility, mana
    enforcement of security
  - Better isolation
  - Closer correlation
  - Better scalability: hook
    Infrastructure
- **VMsafe Partner Program**

VMsafe use case - antivirus



VMSafe use case - Scalable, Integrated Network Security



VMware VMsafe makes it possible for security providers to develop security virtual appliances that monitor and control network activity across all virtual machines.

Summary

- **Virtual Machine: interconnection technology decoupling HW/SW**
  - enable interoperability between HW, OS, and application
  - provide a backward-capability path for deploying innovative operating system solutions
- **Altering the complex HW/SW environments**
  - From server to desktop
  - Software distribution model
- **Provide more features**
  - Interoperability, mobility, high-availability, ease-of-management, reliability, security
- **New environment (chance) for security researchers and vendors.**

Q & A

Thanks

zhugejianwei@icst.pku.edu.cn