計劃名稱: 99-100年度教育部資訊軟體人才培育推廣計畫
跨校資源中心:雲端計算與服務（國立中山大學）
課程名稱: 虛擬化技術

Part1-課程教材

教材名稱:虛擬化技術及應用

國立高雄大學資訊工程學系 張保榮教授

大綱

1. 虛擬化技術介紹
2. 虛擬化技術原理
3. 虛擬化技術演進
4. **Hypervisor型式**
5. 虛擬化實現層次和方式
6. 虛擬化技術的產品
7. 虛擬化的優勢及顧慮
8. 虛擬化的安全問題
9. 虛擬化的未來展望
10. 參考文獻

1. 虛擬化技術介紹

- 虛擬化技術漸漸普遍化，從伺服器到桌上型電腦都呈現急切需求導入其虛擬化之應用，因此相信大家都不會懷疑虛擬化技術的可用性和研究其技術的必要性。

- 一般而言，虛擬化就是把實體資源轉變爲邏輯上可以管理的資源，以打破實體結構間的不可切割的障礙。

- 虛擬化技術本質就是一種資源管理技術，它將硬體、軟體、存儲、網絡等硬體設備分離開來,讓使用者能更合理、更充分的控制與管理這些資源 。

虛擬化技術介紹(續)

術語介紹

- **寄宿機 (Host)**

  即虛擬機管理程序所在的主機系統 。
- **客戶機 (Guest)**

  即運行在虛擬化管理器之上的虛擬機系統 。
- **VMM (Virtual Machine Monitor)**

  虛擬機監視器可以監視虛擬機的運作。
- **Hypervisor**

  虛擬機管理程序（算是高階VMM） 。

虛擬化技術介紹(續)

虛擬化方法

- 基本上虛擬化解決方案是要進行將實體機器虛擬化，這台機器可能直接支持虛擬化，也可能不會直接支持虛擬化。

- 若硬體不會直接支持虛擬化，那麼就需要使用虛擬化管理程序層的支持虛擬化。虛擬機管理程序稱爲VMM或Hypervisor，可以看作是平台上硬體及作業系統的抽象化。

虛擬化技術介紹(續)

- 在某些情況中，這個虛擬機管理程序(VMM)等同於一個作業系統，可以稱爲主機作業系統 (Host OS)。
- 架在虛擬機管理程序之上是虛擬機（VM），其內部運行的是客戶機作業系統(Guest OS) 。
- 這些VM都是一些相互隔離的客戶機作業系統，它們將底層硬體平台視爲自己所擁有。但是實際上，是虛擬機管理程序爲它們製造了這種假象(虛擬化)。

2. 虛擬化技術原理

- 有非常多有關於虛擬化名詞:

  Java VM、Dalvik VM、Apple Rosetta、Transmeta Crusoe、VMWare ESXi、VMWare Workstation、Xen、KVM、Parallels Desktop ... etc.

- 以上名詞都是屬於虛擬機(Virtual Machine, VM), 差別在於不同工作性質，有些屬於 Process VM，有些屬於 System VM！

不同工作性質的虛擬機

- Process VM: 只虛擬一個 Process 的行爲，可能是爲了支援舊架構上的應用程式 ， e.g., Apple Rosetta；也可能是爲了跨平台上執行應用程式e.g., Java 。
- System VM:虛擬的是一個完整的 OS, 需要更複雜的處理 ，e.g., VMWare ESX Server, Hyper-V R2, Xen, KVM, and Citrix XenServer。
- 本課程只探討System VM而已!

系統虛擬機架構分類

- System VM 又區分爲Hosted Architecture及Bare Metal Architecture 兩種架構，如Fig. 2.1.所示。

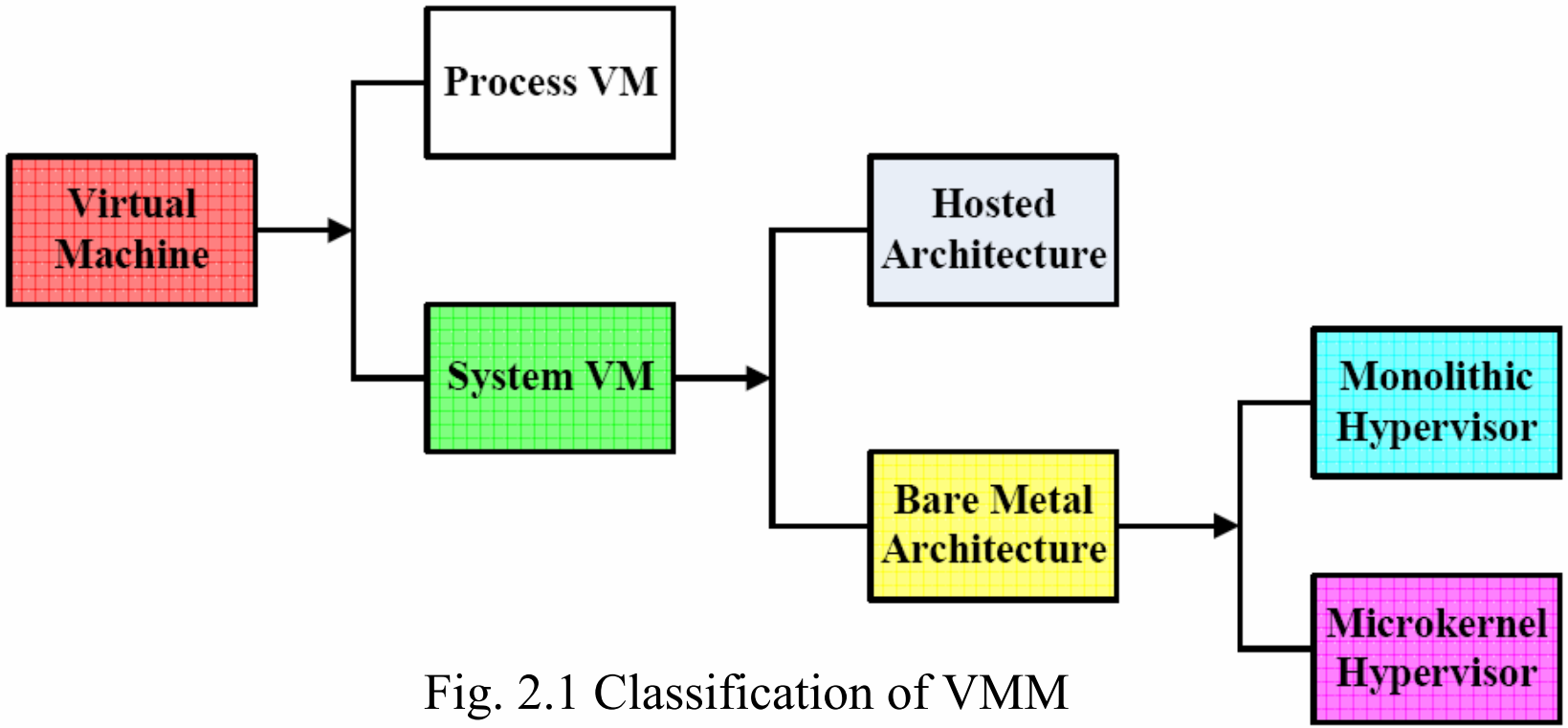


Fig. 2.1 Classification of VMM

系統虛擬機架構分類

- Hosted Architecture : 將 VMM 建立在一個 Native OS 上。
  - e.g., Vitual PC, Virtual Box, VMware Workstation 7, KVM, OpneNebula, OpenStack, and Proxmox Virtual Server。
- Bare Metal Architecture : 直接將 VMM (或Hypervisor) 建立在硬體之上。
  - e.g., VMware ESX, Hyper-V R2, Citrix XenServer, Xen, and KVM(後期)。

虛擬化理論

- **Popek and Goldberg Theorem**
- Gerald J. Popek 跟 Robert P. Goldberg 在 1974 年發表了一篇文章[1]

  Formal requirements for virtualizable third generation architectures, Published in: Communications of the ACM (Magazine), Vol. 17, No. 7, July 1974.

  提出 Virtualization 需要具有三個特性:
- Equivalence – 執行結果需與直接在機器上執行相同
- Resource control – 需要控管所有 VM 的 Resource
- Efficiency – 要有效率, 盡量 Native Execution

CPU指令分類

- 說到Efficiency性質, 要盡量讓程式碼在機器上 Native Execution,可是又要做到 Resource Control, 這時會探討一個 Architecture 是否 Virtualizable 探討這個問題，我們將 Instruction 分成四類，如Fig. 2.2所示。
- Privileged Instruction – 需要權限才能執行，若在 user mode 會發生 trap
- Sensitive Instruction – 會控制到硬體狀態的指令，e.g., 更換硬體 mode
- Critical Instruction – Sensitive but non-privileged instruction
- Others – 不屬於以上三種

CPU指令分類圖

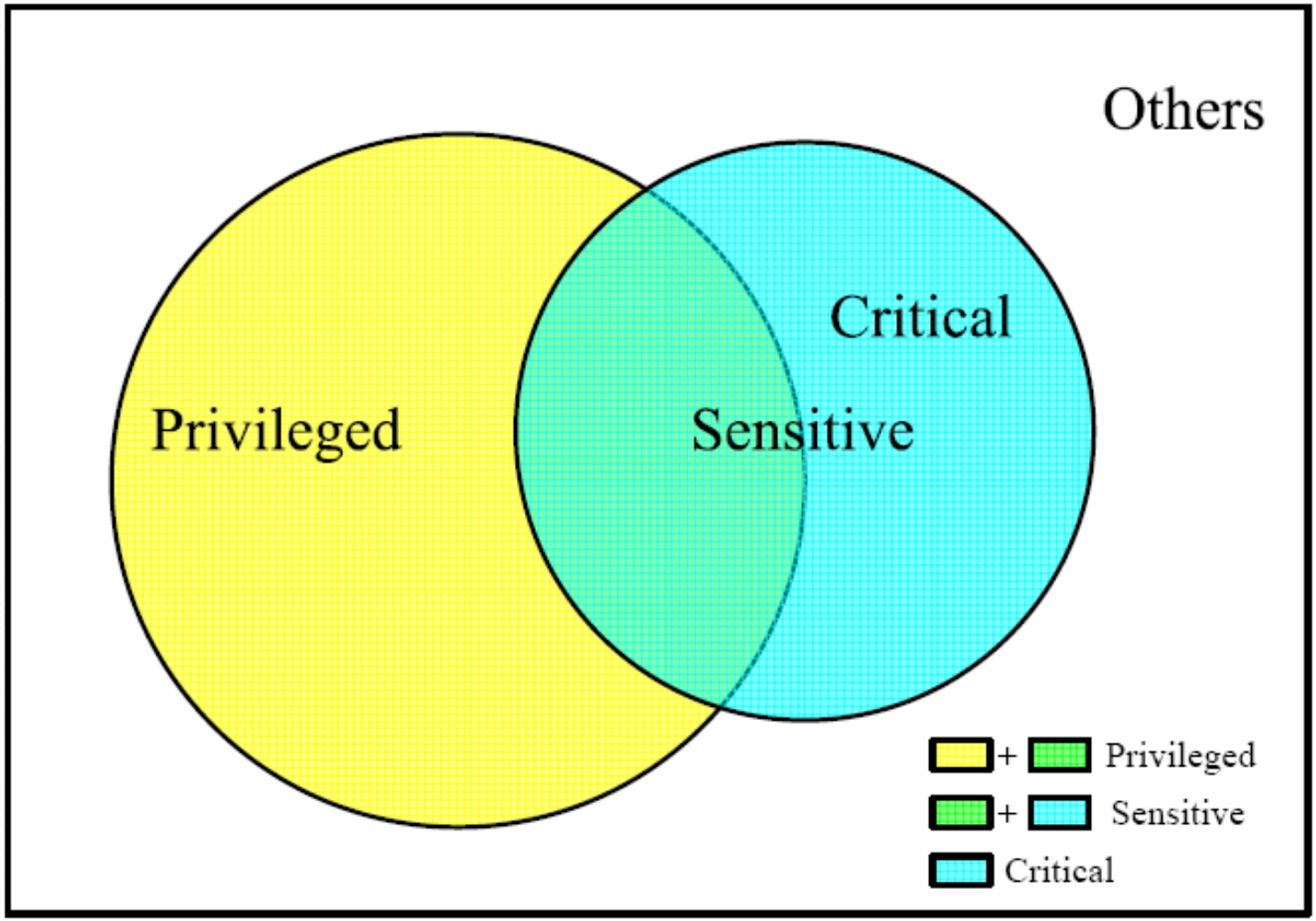


Fig. 2.2 Categories of CPU instruction

虛擬化技術原理(續)

- 在上面分類簡圖中，最需要在意的是粉紅色的 Critical Instructions，倘若所有 Sensitive Instruction 皆爲 Privileged Instruction，那 Virtualization 的問題會簡單很多，因爲所有要控制硬體的指令皆會發生 Trap，此時從硬體架構來看我們就可稱爲Architecture Virtualizable。

- 若VM所發出的Critical Instruction無法給虛擬機管理程序所Trap做損害管制，而是直接使用硬體那將造成系統無法管控的損害。

虛擬化技術原理(續)

- 但不巧的是 X86、ARM機器皆不是此種類型，所以 Critical Instruction Handling，就成爲Virtualization 時需要煩惱的問題。

- Critical Instruction Handling方式
  - **Dynamic Binary Translation** (Full Virtualization技術)
  - **Hyper Call** (Para Virtualization技術)
  - **Virtualization Hardware Support** (Hardware-Assisted Virtualization技術)

虛擬化技術原理(續)

- Virtualization Hardware Support技術
  - Intel CPU 提出了 Intel-VT 技術。
  - AMD CPU提出了AMD-V技術。
  - ARM Cortex A15 (2012年) 也將支援 Virtualization Hardware Support。
- 不難看出CPU擁有Virtualization 的重要性越來越高！

3. 虛擬化技術演進

- 在1960年代晚期IBM開發System/360™Model67大型主機時，就使用VMM（Virtual Machine Monitor）對所有的硬體界面都進行了虛擬化。

- 隨著x86平台CPU性能越來越增強，x86平台下的虛擬化技術同樣得到了快速發展，特別是支持虛擬化技術的晶片輔助（Chip-Assisted）技術出現以後，改變x86平台對虛擬化支持效能，因此x86平台已經成爲了虛擬化技術發揮作用的重要平台之一。

3. 虛擬化技術演進

- 支持虛擬化技術的晶片輔助（Chip-Assisted）技術，即是CPU、ChipSet、Network 的支持虛擬化技術。
  - Intel
    - VT-x (Virtualization technology for x86 platform)
    - VT-d (Virtualization Technology for Directed I/O)
    - VT-c (Virtualization Technology for Connectivity)
    - VMDq (Virtual Machine Data Queue)
  - AMD
    - AMD-V (AMD Virtualization)
    - AMD-Vi (I/O Virtualization Technology, IOMMU and PCI-SR-IOV)

系統虛擬機架構

系統虛擬機架構的發展經歷了兩個階段。

- 初級階段：寄宿架構(Hosted Architecture)
- 進階階段：裸金屬架構(Bare Metal Architecture or Hypervisor)

初級階段---寄宿架構

- **寄宿架構**：採用模擬軟體技術模擬出計算機硬體和軟體。模擬層與作業系統對話，而作業系統與計算機硬體對話。在模擬層上虛擬機器安裝的作業系統並不知道自己是被安裝在模擬環境下的，你可以按照一般的方法安裝該作業系統。**這種虛擬化需要付出很大的性能代價**。(Hosted Architecture)

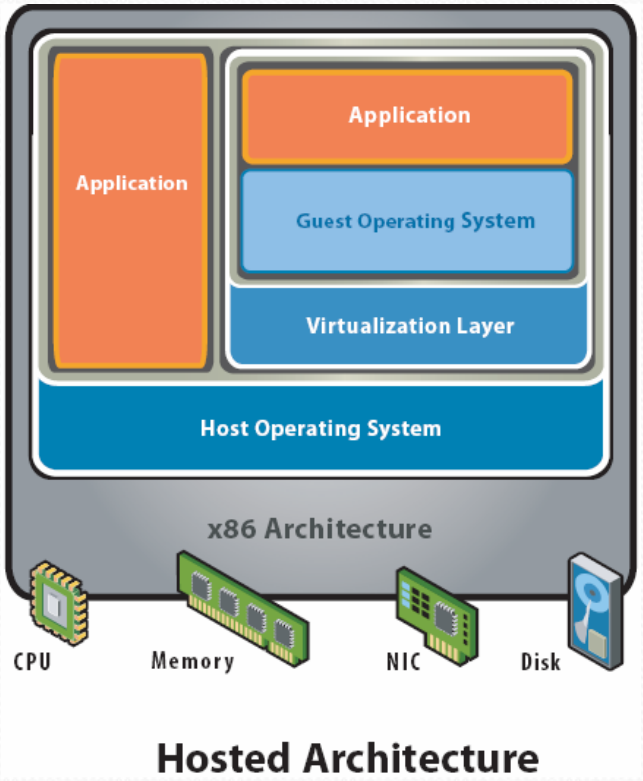


Fig. 3.1 Hosted architecture (source: VMware [2])

進階階段---裸金屬架構

- **裸金屬架構**：虛擬化技術不斷的在進步，虛擬化被發展到一個更進階的架構。在模擬層（負責虛擬機器的指令翻譯）和硬體之間不需要任何原型主機作業系統運行在硬體上的一種新虛擬技術。**虛擬機監視器直接運行在硬體上，因此虛擬化變得更加地高效率。**(Hypervisor or Bare Metal Architecture)

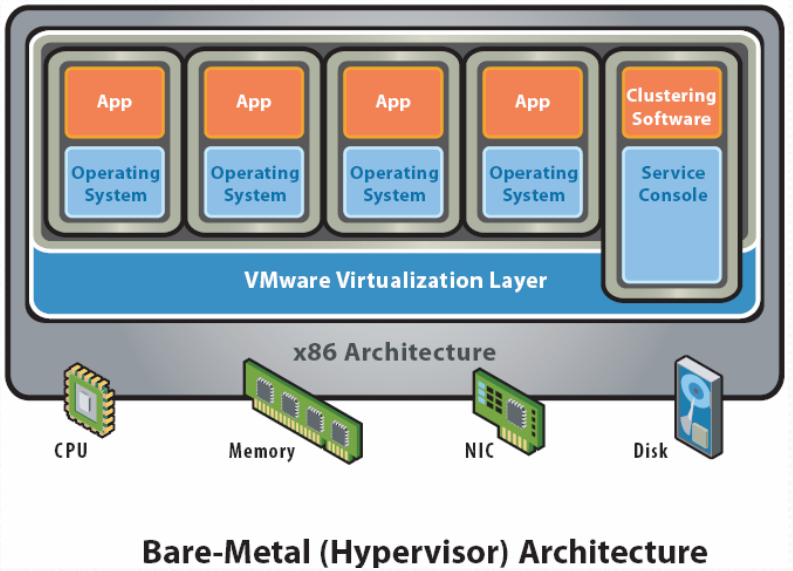


Fig. 3.2 Bare metal architecture (source: VMware)[2]

4. Hypervisor型式

虛擬化技術的核心元件是VMM，而VMM具體的結構可以分爲三類：

- **Hosted Architecture**
  - **OS-hosted Hypervisor** (寄宿型)
- **Bare Metal Architecture**
  - **Monolithic Hypervisor** (單石型)
  - **Microkernel Hypervisor** (微核型)

OS-hosted Hypervisor

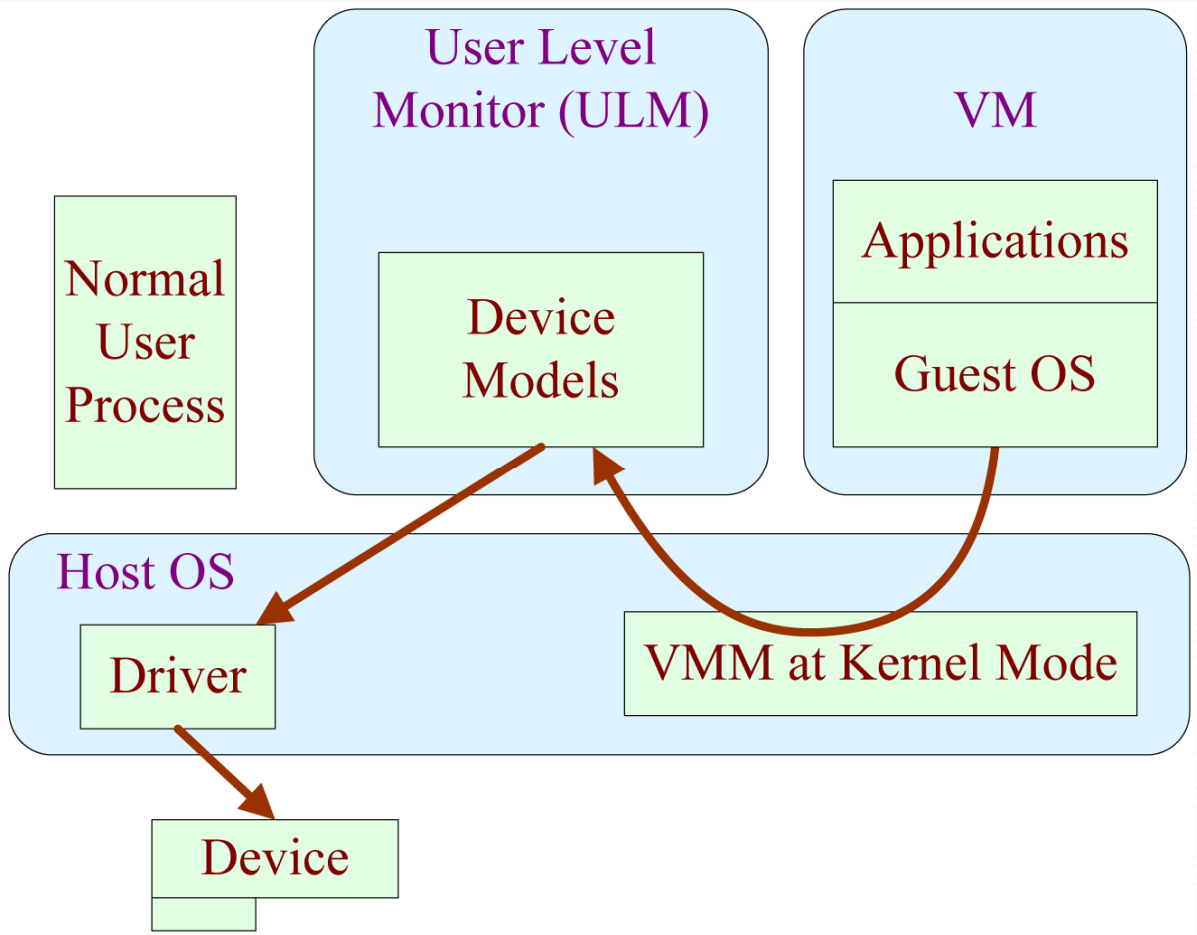


Fig. 4.1. Diagram of OS-hosted Hypervisor

OS-hosted Hypervisor (Cont.)

- 寄宿型

  在圖3.1中的VMM，實體資源由Host OS（ Windows or Linuxetc)來管理，實際的虛擬化功能由VMM提供，它通常是Host OS 核心的一個獨立模組（有的VMM還包含了用戶行程，具有負責I/O虛擬化的User Mode Device Model），VMM通常透過Host OS的服務來獲得實體資源，以便實現CPU、Memory和I/O設備的虛擬化。

  VMM創建出VM後，通常將VM作爲Host OS的一個行程並參與調度，如圖3.1所示，VMM模組負責CPU和Memory虛擬化，除此之外它經由ULM呼叫Host OS設備的驅動程式實現了I/O設備的虛擬化。

OS-hosted Hypervisor (Cont.)

- 寄宿型
  - **優點**：可以充分利用現有OS的設備驅動程式，VMM無需自己實現大量的設備驅動程式，輕鬆實現I/O設備的虛擬化。安全性方面，VM的安全依賴於VMM及Host OS的安全。
  - **缺點**：因資源受Host OS控制，VMM需透過Host OS的服務來獲取資源進行虛擬化，其效率和功能會受到一定影響。
  - **產品**：採用該VMM結構的有VMware Workstation、VMWare Server(GSX)、Virtual PC。
  - **對象**：桌上層級虛擬化、PC虛擬化。

Monolithic Hypervisor

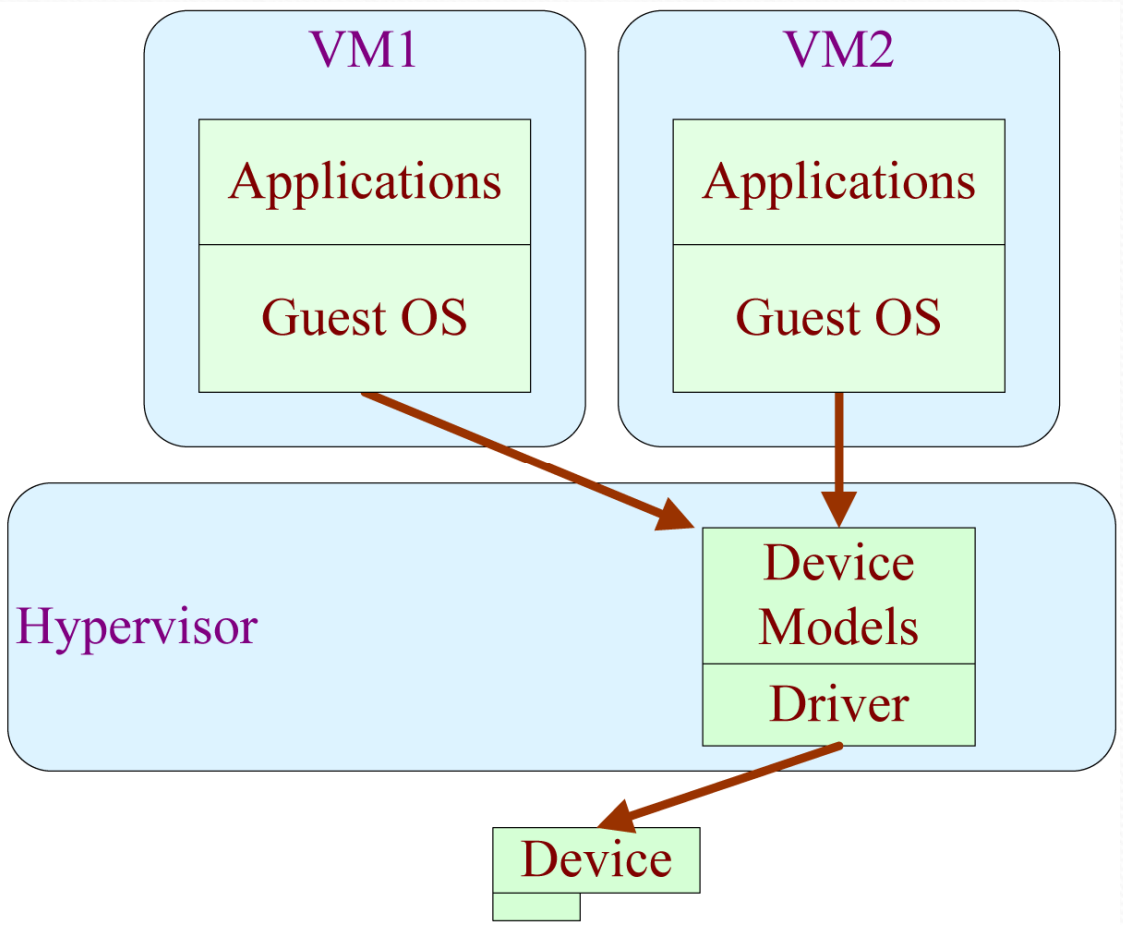


Fig. 4.2. Diagram of Monolithic Hypervisor

Monolithic Hypervisor (Cont.)

- 單石型

  在圖3.2中的VMM可以看作爲了虛擬化而設計出來的一個完整OS，它掌控有所有硬體資源（CPU，Memory，I/O Devices）並管理所有硬體資源。

  VMM還需向上建立VM，讓Guest OS運行在VM中，因此VMM還負責虛擬環境的創建和管理。

Monolithic Hypervisor (Cont.)

- 單石型
  - **優點**：因VMM同時具有硬體資源的管理功能和虛擬化功能，故虛擬化的效率會較高；安全性方面，VM的安全只依賴於VMM的安全。
  - **缺點**：因VMM完全擁有實體資源，因此，VMM需要進行實體資源的管理，包括設備的驅動程式，而IO設備驅動程式的開發負擔是很重的，這對VMM是個很大的挑戰。
  - **產品**：採用該結構的VMM有VMWare ESX Server、Wind River Hypervisor、KVM（後期）。
  - **對象**：企業層級虛擬化、大型伺服器虛擬化。

Microkernel Hypervisor

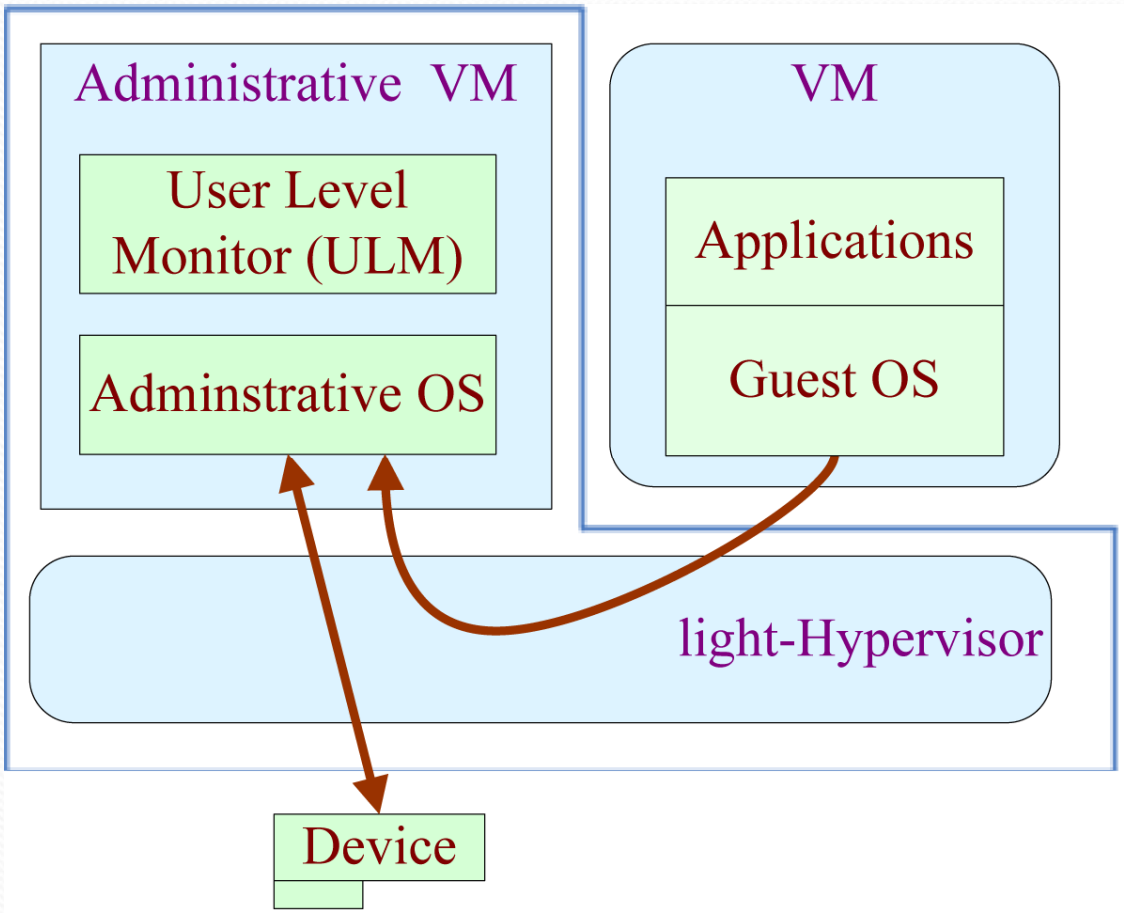


Fig. 4.3. Diagram of Microkernel Hypervisor

Microkernel Hypervisor (Cont.)

- 微核型

  在圖3.3中的VMM是上述兩種模式的混合體，VMM依然位於最底層，擁有所有實體資源，但VMM是一個輕量型Hypervisor，因此會主動讓出大部分I/O設備的控制權，將它交由一個運行在管理VM (Administrative VM)內的管理OS (Administrative OS)來控制。

  VMM只負責CPU和Memory的虛擬化，I/O設備的虛擬化由VMM和管理OS共同完成。

Microkernel Hypervisor (Cont.)

- 微核型
  - **優點**：可利用現有OS的I/O設備驅動程式，避免在VMM上開發I/O設備驅動程式；VMM直接控制CPU和Memory等實體資源，虛擬化效率較高，無需透過Host OS的服務；若對管理OS的權限控制得當，虛擬機的安全性只依賴於VMM。
  - **缺點**：因管理OS運行於VM上，當需要管理OS提供服務時，VMM需要切換到管理OS，這裡面就產生上下文切換的時間浪費。μ-Hypervisor無法trap由VM發出的kernel mode 的Ring-0指令，必須修改Guest OS才能hypercall到light-Hypervisor。

Microkernel Hypervisor (Cont.)

- 微核型
  - 產品：採用該結構的VMM有Citrix XenServer、Microsoft Hyper-V R2。
  - 對象：研發或小型辦公室層級虛擬化、一般伺服器或高階PC虛擬化。

5. 虛擬化實現層次和方式

- 虛擬化的實現層次和方式直接決定真正虛擬化解決方案的效率和功能。

虛擬化實現層次

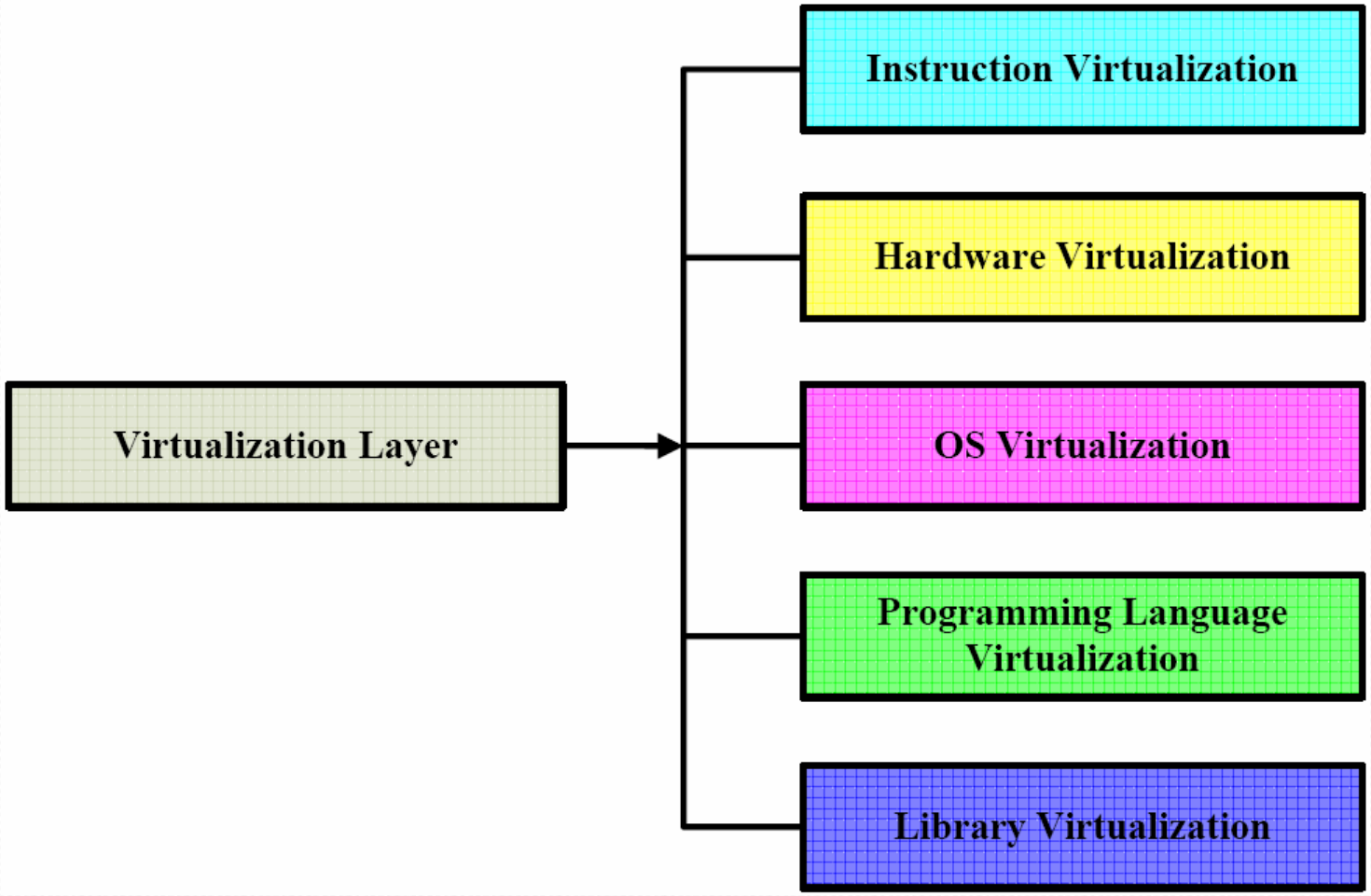


Fig. 5.1 Virtualization layer

虛擬化實現層次

- 本課程討論:
  1. 指令層級
  2. 硬體層級
  3. 作業系統層級

- 本課程不討論:
  4. 程式語言層級
  5. 函式庫層級

指令層級虛擬化

1. **指令層級虛擬化**，它是通過純軟體的方式模擬**各種不同 CPU(x-86、PowerPC、MIPS、ARM...)** 硬體的指令集來達到虛擬化的目的，這種方式的優點是可以完全的模擬出所需要虛擬設備的所有特性，這樣必然會帶來的一個缺點就是性能太差，而且實現複雜。

   代表軟體有QEMU，是模擬處理器的自由軟體，具高速度及跨平台的特性。

硬體層級虛擬化

2. **硬體層級虛擬化**，硬體層級虛擬化是基於虛擬機的CPU硬體指令集一般與原本主機實際CPU硬體指令集非常相似，在這個假設前提下來實現。因爲有上面的前提，通過硬體虛擬化可以把虛擬機的大多數操作指令通過映射方式(hypercall)在主機上直接執行，從而大大提供虛擬化的效率。它的優點是簡單、高效，缺點是所能虛擬的硬體平台範圍有限，且可能需要硬體對虛擬化的支持。代表軟體有Xen，guest OS必須進行明顯地修改才可以在Xen上運行，因此Xen無需特殊硬體支持。KVM是一種 linux kernel 模組，可以修正 QEMU program 使之用於硬體虛擬化等。

作業系統層級虛擬化

3. **作業系統層級虛擬化**，有時也被稱爲作業系統層級或共享式作業系統的虛擬化。作業系統虛擬化是虛擬化實體機器的作業系統（核心）層，這將可創建一些獨立的容器(類似程序)在一台實體機器以及它的實例作業系統上，並以最高的效率方式去使用硬體、軟體、數據中心和管理等資源，在圖3.4顯示了作業系統虛擬化和管理程序或硬體虛擬化技術之間相比較。

作業系統層級虛擬化(續)

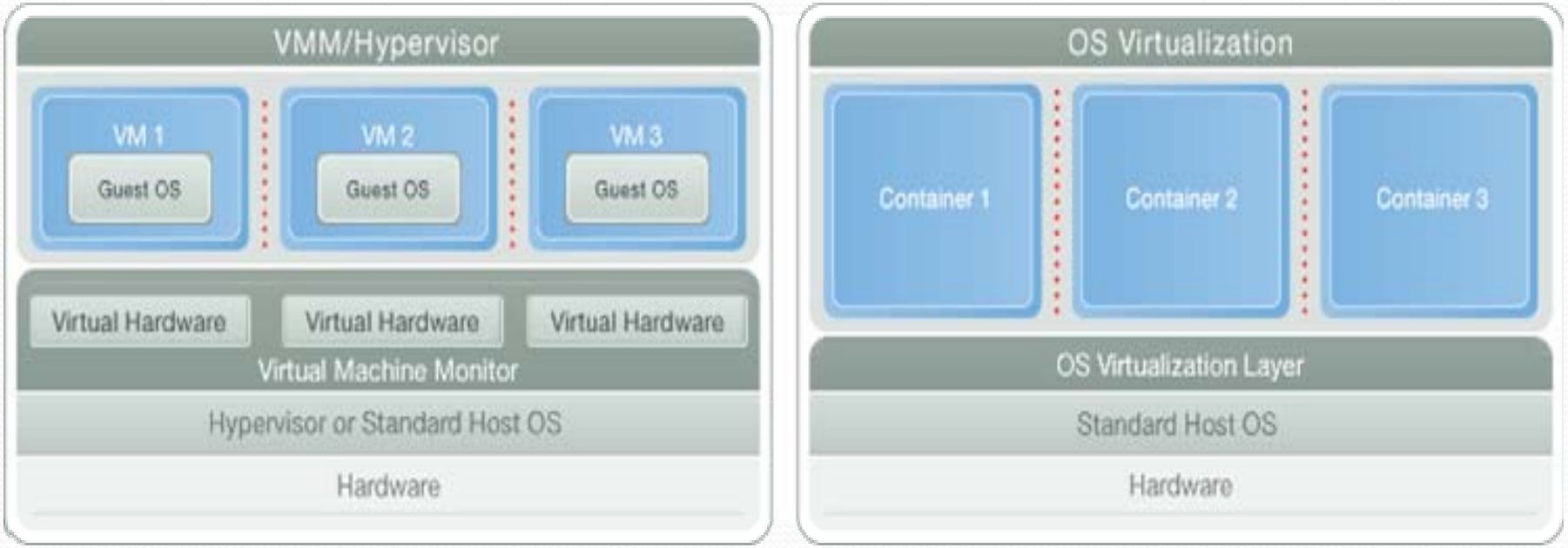


Fig. 5.2 Comparison of Hypervisor to OS-level Virtualization (source: VMware [2])

程式語言、函式庫層級虛擬化

4. **程式語言層級虛擬化**，代表有JVM (Java虛擬機）。
5. **函式庫層級虛擬化**，代表有Wine庫（在linux下模擬windows的運行環境），cygwin庫（在windows下模擬linux運行環境）。

- 這裡特別要強調一下的是，我們一般提到的虛擬技術僅包括上面前三種類別，後兩種屬於一種廣義上的虛擬化技術的範疇。在此我們主要討論的是狹義的虛擬化技術範疇。

虛擬化過程的問題

- 虛擬化的過程有二個問題!!!

  問題一：VM處於user mode不具kernel mode，所以Guest OS之kernel所發出指令碼(低特權態部件發出的敏感指令)，不能自動被hypervisor偵測而截獲，以進行“無害化”處理。

  問題二：不同VM有不同的Guest OS和AP，其架構與底層硬體架構不一致，因此所有由VM所發出之指令碼必須經由Hypervisor做dynamic binary translation轉成可接受的執行碼，才能再交給底層CPU去解碼執行。

解決的方法

- 解決的方法有下面三種：

  (1) 修改Guest OS利用Hypercall強迫呼叫Hypervisor。

  (2) 採用Hypervisor接受Guest OS kernel指令並做(動態)指令碼轉譯(Dynamic) Binary Translation。

  (3) 修改x86CPU硬體結構支持虛擬化，讓Guest OS的kernel發出指令具有guest mode特權模式，讓主機CPU皆能直接執行各類型Guest OS發出的虛擬機指令碼，所以無需再經hypervisor轉譯或使用hypercall呼叫hypervisor 。

虛擬化實現方式

- 按技術分類，實現虛擬化的方式:

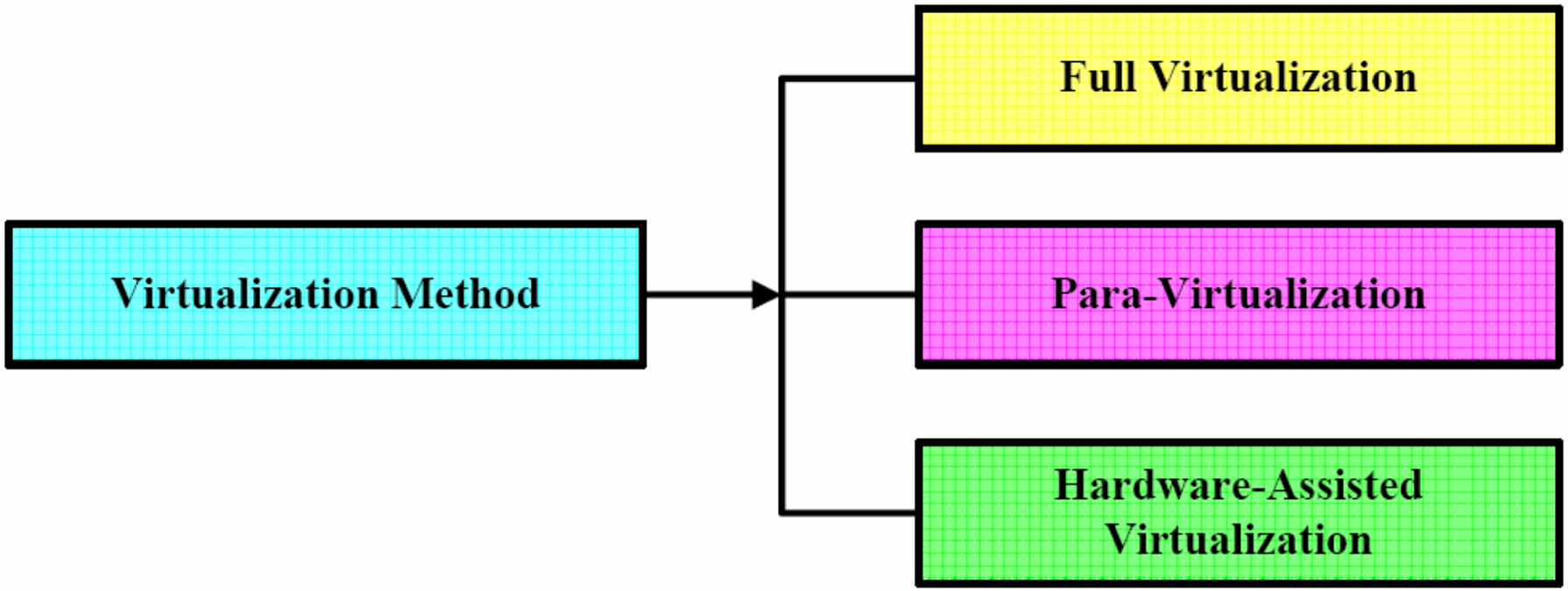


Fig. 5.3 Virtualization method

全虛擬化

1. **全虛擬化**（Full-Virtualization）- 全虛擬化就是完全虛擬出一個主機所需要的所有環境，因爲運行於其上的虛擬機來說，它感覺不到Hypervisor的存在，所以此種方案是對Guest OS的支持類型最廣泛的一種。缺點是Hypervisor接受Guest OS kernel指令必須做**動態指令碼轉譯**(Dynamic Binary Translation)成爲x86架構下CPU可以解碼執行的機械碼。

   e.g., 可以在指令層級如QEMU；可以在作業系統層級如VirtualBox。

性質

- **全虛擬化直接使用hypervisor分享底層硬體**，也稱爲原始虛擬化技術，如圖5.2。

- **hypervisor在guest VM和硬體之間用於工作協調，讓虛擬機感覺像實體機一般地完全自由的操作**，一些受保護的指令必須由Hypervisor來捕獲和處理。

- 管理者是透過管理虛擬機Admin來控管hypervisor運行，本結構屬VMM分類中之Hypervisor模型。

全虛擬化架構

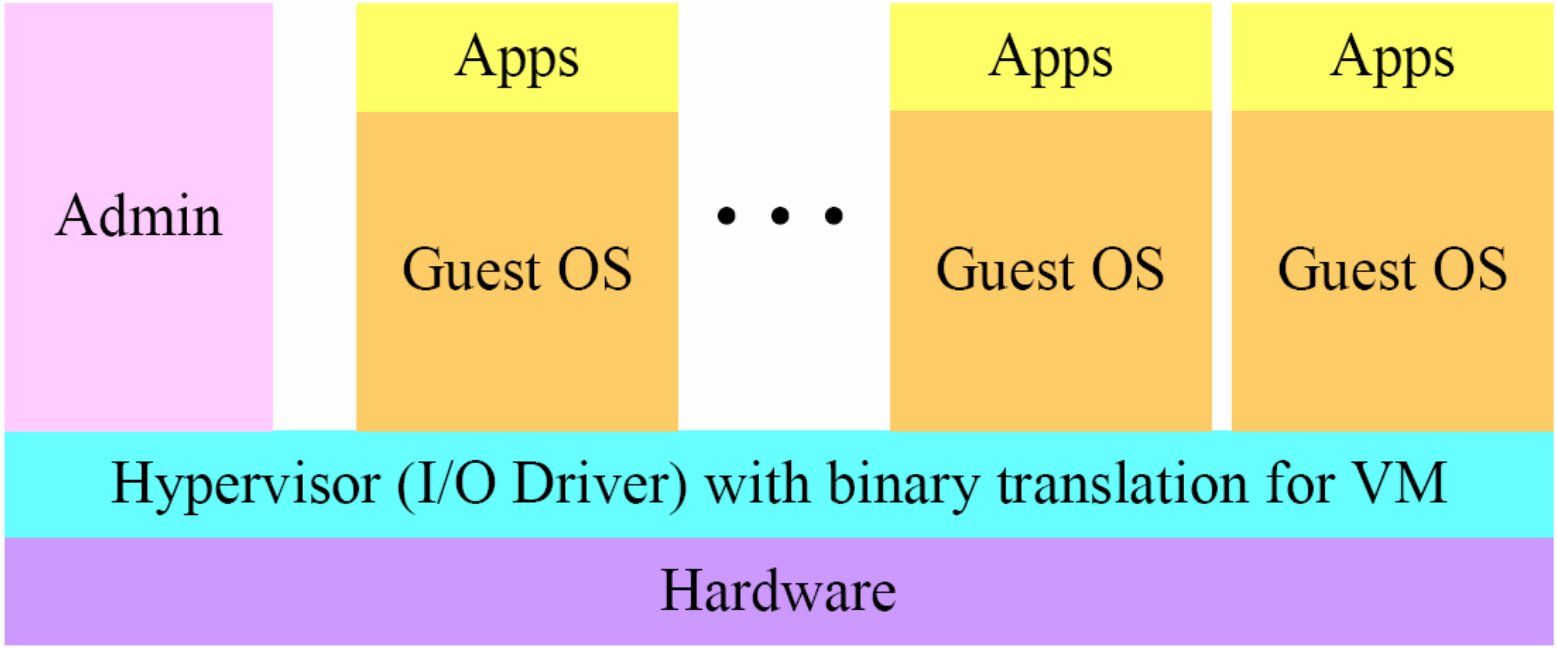


Fig. 5.4 Diagram of full virtualization.

旁虛擬化

2. 旁虛擬化（Para-Virtualization）技術是爲了改善傳統x86架構下，硬體對虛擬化無法支持，而提出的一種改進技術，實現硬體層級的虛擬化層。爲了提高虛擬化的效率，需要讓虛擬機的操作指令直接給hypervisor執行，以減少中間轉換造成的性能損失和時間延遲。在傳統x86架構中，處於kernel mode 狀態下的Kernel和Driver使用Ring0高特權CPU指令，而處於user mode 狀態下AP使用Ring3低特權CPU指令，做爲區隔保護系統的運作。如果guest OS一些Ring0高特權指令在低特權級模式(user mode)下運行，將會遭遇到不可預測的運行結果。

現象

換句話說，Guest OS 所發出的低特權態部位(user mode)的敏感指令(Ring0)，這一類指令是不能自動被hypervisor (kernel mode)偵測而截獲(trap)，這是既有的環境中做不到的。
因此修改Guest OS增加hypercall強迫呼叫hypervisor。
e.g., Citrix XenServer and MS Hyper-V R2 。

缺點: 除了必須修改Guest OS以外，另外若採用非自由軟體的作業系統做Guest OS時，除非原廠要修改，不然它是無法被修改的像Windows OS，因此就無法實現旁虛擬化的目標。

性質

- 旁虛擬化修改guest OS配合hypervisor分享底層硬體。
- 旁虛擬化的modified guest OS集合了虛擬化方面的指令碼，並且自己知道是在虛擬模式下運行，如圖5.3。
- 因爲modified guest OS運用hypercall虛擬行程與hypervisor進行很好的合作，所以旁虛擬化的性能接近於真實系統。

旁虛擬化架構

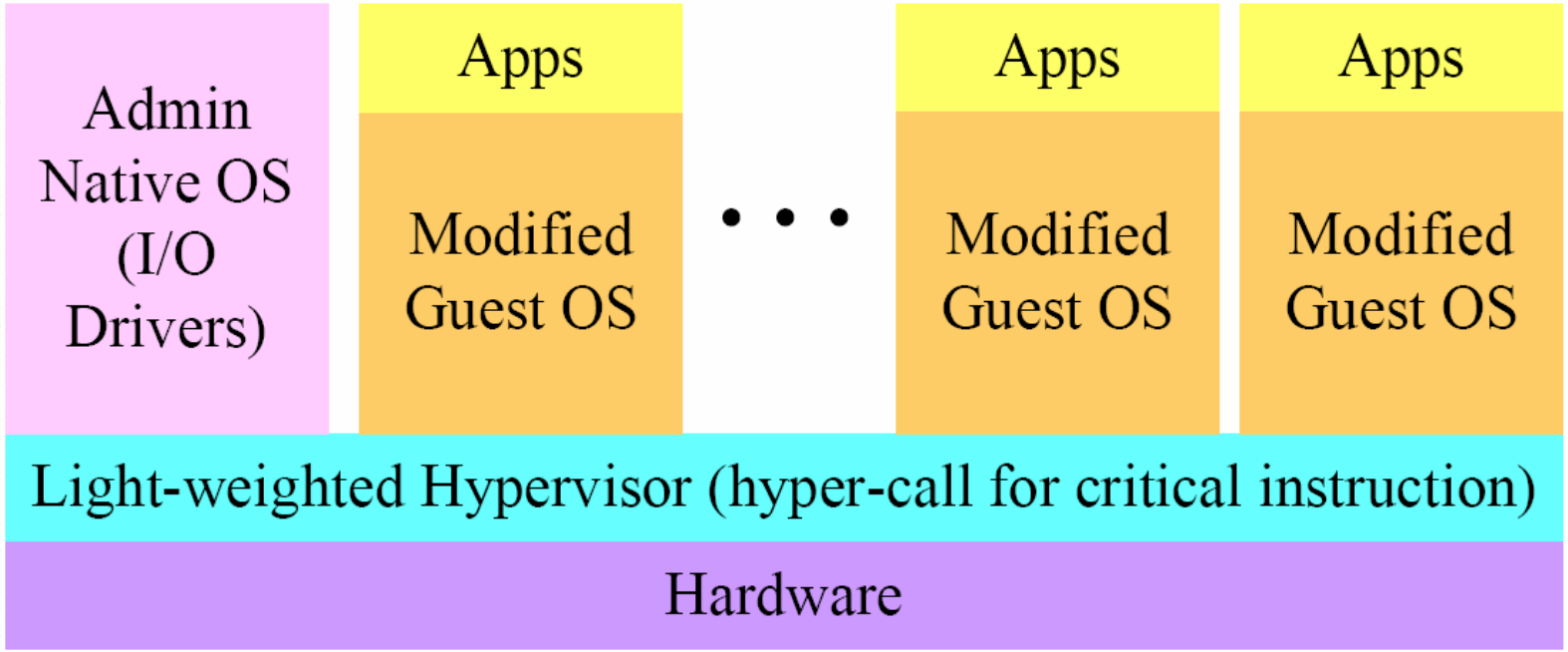


Fig. 5.5 Diagram of Para-vrtualization

性質

- Admin VM採用Native OS具有I/O Device Driver，提供實體機I/O Device Driver的服務。 guest VM借用Admin VM存取I/O Device 的硬體，guest VM中modified guest OS不具I/O Device Driver。
- hypervisor只虛擬CPU及Memory運作(稱爲半虛擬化)也不具I/O Device Driver 。
- 管理者透過admin VM來控管hypervisor運行以及管理guest VM，本結構屬於VMM分類中之混合模型。

硬體支持虛擬化

3.硬體支持虛擬化（Hardware-Assisted Virtualization ）是指硬體直接支持虛擬化操作，包括CPU虛擬化、Memory虛擬化、以及IO設備虛擬化等。硬體支持虛擬化技術是專門改善X86架構在虛擬化技術方面的缺失。

在CPU特權指令方面，新技術VT（Virtualization Technology）新增了一個客戶機模式guest mode，guest OS 就運行於guest mode (Ring 0)這一特權級模式，而guest AP的運行user mode（Ring3）保持不變，而hypervisor則運行在root mode (Ring -1)。

性質

- 新的Memory技術VT-d支持guest VM直接轉換爲硬體地址來對實體Memory存取，提高虛擬機memory性能。
- 新的網路技術VT-c支持guest VM直接連線實體NIC上網，提高虛擬機網路性能。
- 硬體支持虛擬化技術(如x86平台下VT-x、AMD-V)的這一系列發展，可同時得到全虛擬化的良好支持性和半虛擬的高效性，這是硬體支持虛擬化技術的優勢。

性質

- 硬體直接虛擬化 了CPU、Memory、IO Devices，如圖5.4。 e.g., *KVM and VMware ESX Server*

- 硬體支持虛擬化技術與hypervisor結合可以極大地提高虛擬化的效率。例如， KVM使用支持VT技術的CPU配合，可以使guest VM的性能損失降低到一個很小範圍內，本結構屬於 VMM分類中之hypervisor模型。

硬體支持虛擬化架構

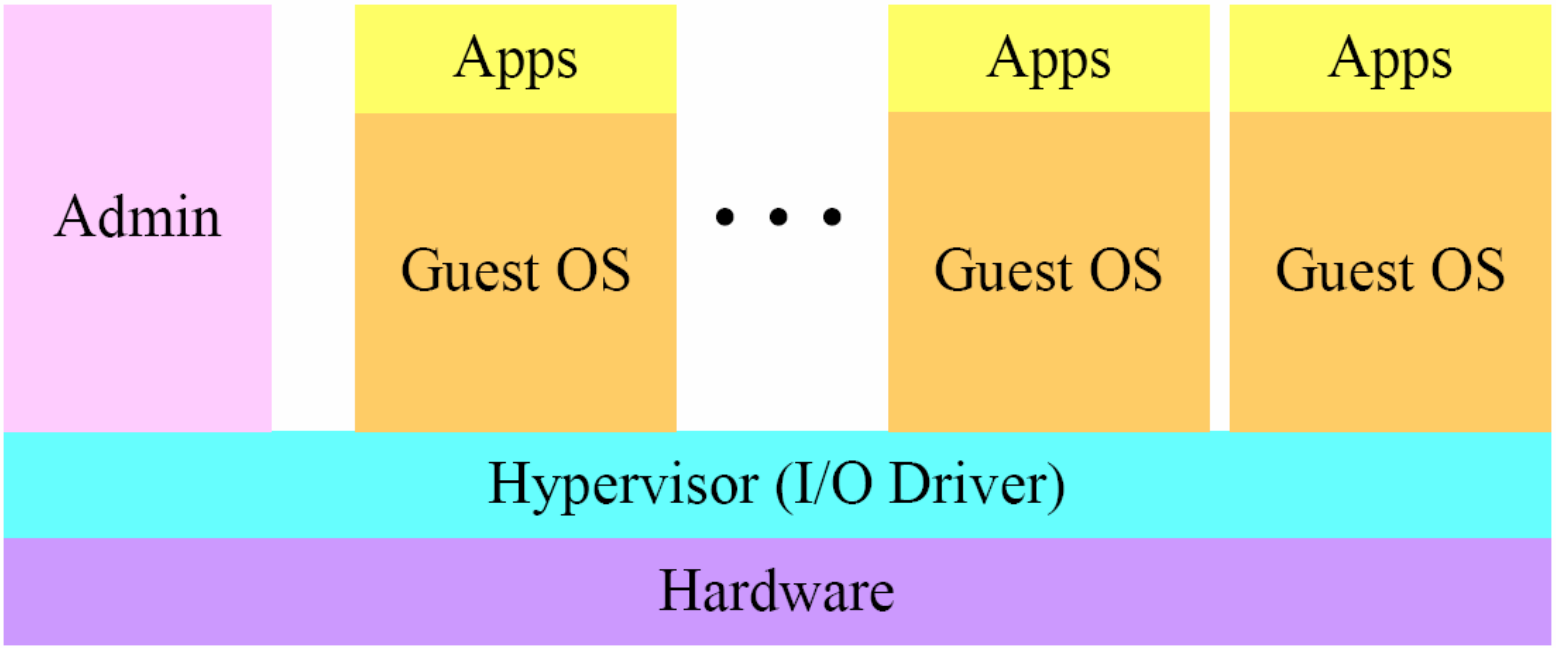
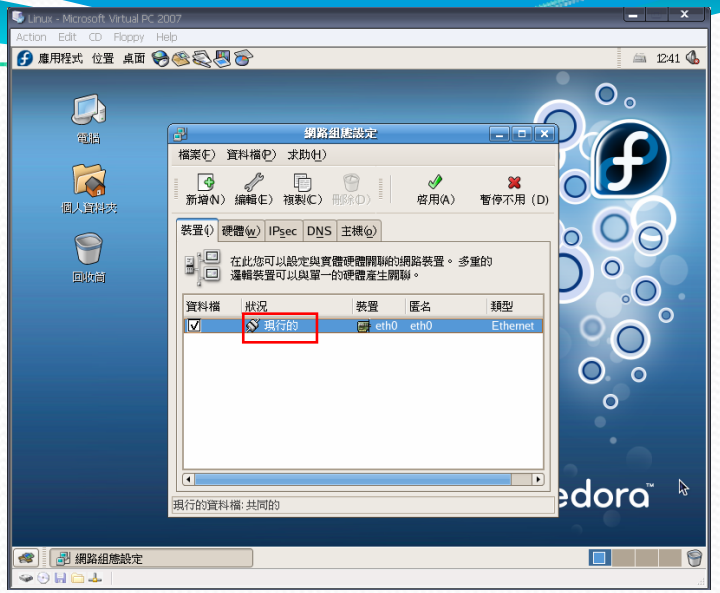


Fig. 5.6 Diagram of hardware-assisted virtualization.

6. 虛擬化技術的產品

- **PC虛擬機**
  - Virtual PC (see another topic)
  - Virtual Box (see another topic)
  - VMware Workstation 7 (see another topic)
- **伺服器虛擬機**
  - Xen (Xen Hypervisor)
  - KVM (Kernel-Based Virtual machine)
  - XenServer (Citrix XenServer)
  - VMware vSphere 4 (VMware ESX Server) (see another topic)
  - MS Hyper-V R2 (Windows Server 2008 R2) (see another topic)
  - OpenNebula (OpenNebula.org)
  - OpenStack (Nova-Swift-Glance-Keystone-Horizon)
  - Proxmox Virtual Environment (Proxmox VE)

Virtual PC

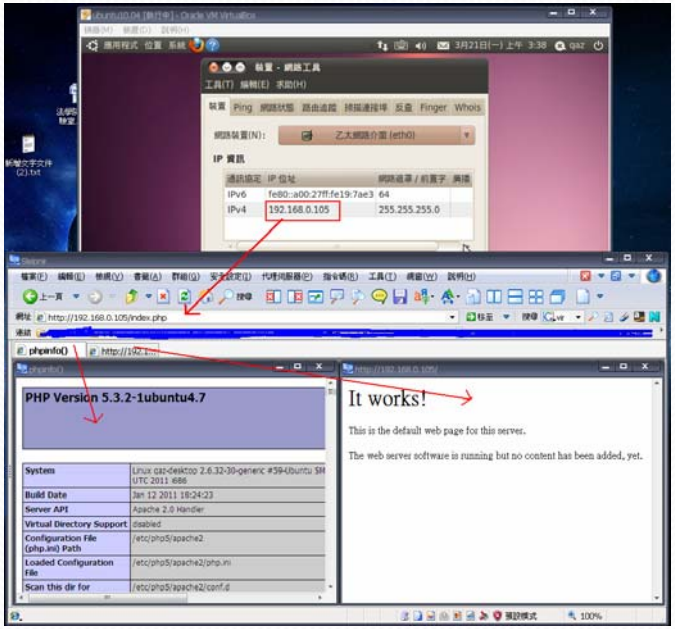


- 原本「Virtual PC」跟「VMWare」一樣，都是要錢的付費軟體，不過自從微軟將Virtual PC買下之後，就變成免費軟體了，可能是微軟在虛擬化這塊有什麼布局吧，雖然買下Virtual PC後，微軟看起來並沒有未Virtual PC軟體作太多的功能改進或調整，不過用起來還算穩定。
- 官方網站：
  http://www.microsoft.com/windows/virtual-pc/

Virtual PC (cont.)

- Virtual PC這類虛擬電腦的好處是，我們只需提供硬碟空間並與主電腦（現在正在用的這台）共享CPU、記憶體與其他相關資源，不需要重新分割硬碟、不需要重開機，主電腦與虛擬電腦可以同時操作、執行，還可直接與虛擬電腦連線、使用共用資料夾等...。看起來就像是台區域網路中的獨立電腦一樣，可以讓我們用來測試、學習Linux、Ubuntu..等各種作業系統，或者作一些架站練習、測試不同作業環境等等，相當方便！

Virtual Box



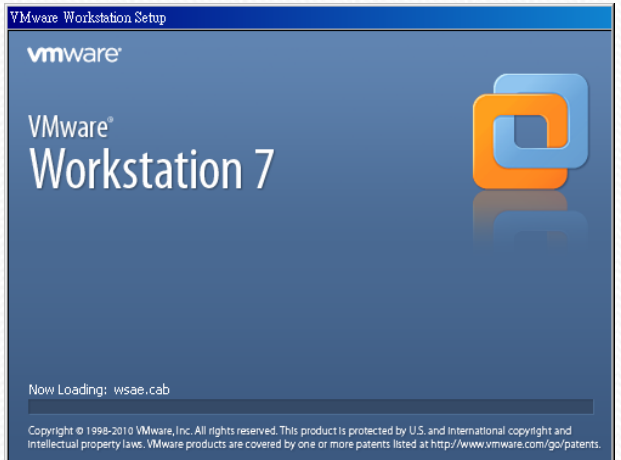
- 軟體名稱：VirtualBox
  軟體版本：4.0.4
  軟體語言：中文（內建多國語言）
  軟體性質：免費軟體
  檔案大小：62.9MB
  系統支援：Windows XP/2003/Vista/Mac OS /Linux/Solaris
  官方網站：http://www.virtualbox.org/

Virtual Box (cont.)

- 目前VirtualBox可支援的客戶端系統（可安裝在VirtualBox裡面的作業系統）有十幾種，幾乎各種常見的x86作業系統都可以支援，如：
  - Windows NT 4.0
  - Windows 2000/XP/Server 2003/Vista
  - DOS/Windows 3.x/95/98/ME
  - Linux 2.4
  - Linux 2.6
  - Solaris 10, OpenSolaris
  - FreeBSD/OpenBSD
  - OS/2 Warp 4.5

VMware Workstation 7

- 經由 VMware Workstation，可以發現PC或Notebook的真正功能和靈活性。通過一台 PC 上同時運行多個作業系統，可將系統所需的硬體總成本降低 50% 或更低，另外使用自動執行和任務優化，可以節省時間並提高執行效率。VMware Workstation具有四項優點。
  1. 代管舊版應用程式並克服平台遷移問題
  2. 在隔離的環境中配置和測試新軟體或修補程式
  3. 自動執行開發軟體和測試任務
  4. 在單台 PC 上展示多層配置



VMware Workstation 7 (cont.)

- VMWare Workstation 7新增多項功能包括支持Win7 Aero 效果，除了支持 Aero 效果，VMWare Workstation 7還新增了很多功能，下面列出部分新增功能：
  - 完整地對3D的支持
  - 支持最新Windows 7 WDDM驅動
  - 支持vSphere 4.0和ESX
  - 可直接使用虛擬機進行列印
  - AutoProtect
  - 支持對虛擬機進行加密
  - 支持IPv6、ALSA
  - 可擴展虛擬磁碟，無需使用另外軟體

KVM

- 見圖5.1 ， Linux將KVM包入kernel之中成爲其中一個模組，使kernel自身成爲一個Hypervisor，也因此在 KVM下的 Guest OS對於Linux kernel而言都只是一個正規的Process(行程)。

- KVM Driver的 user-space 介面是 '/dev/kvm' (即許多process 的共同介面)，因此每個process 可以執行自己的virtual machine，在一台電腦上也能執行多個 virtual machine，見圖5.2 。

- 爲了解決Guest OS全虛擬化，KVM必須搭配具支撐虛擬化之CPU (需Intel VT或AMD V處理器) ，並且對KVM的操作模式增加了一個新的執行模式Guest Mode (guest mode有自己的kenrel和user mode) 運行在Guest OS上。

KVM Model

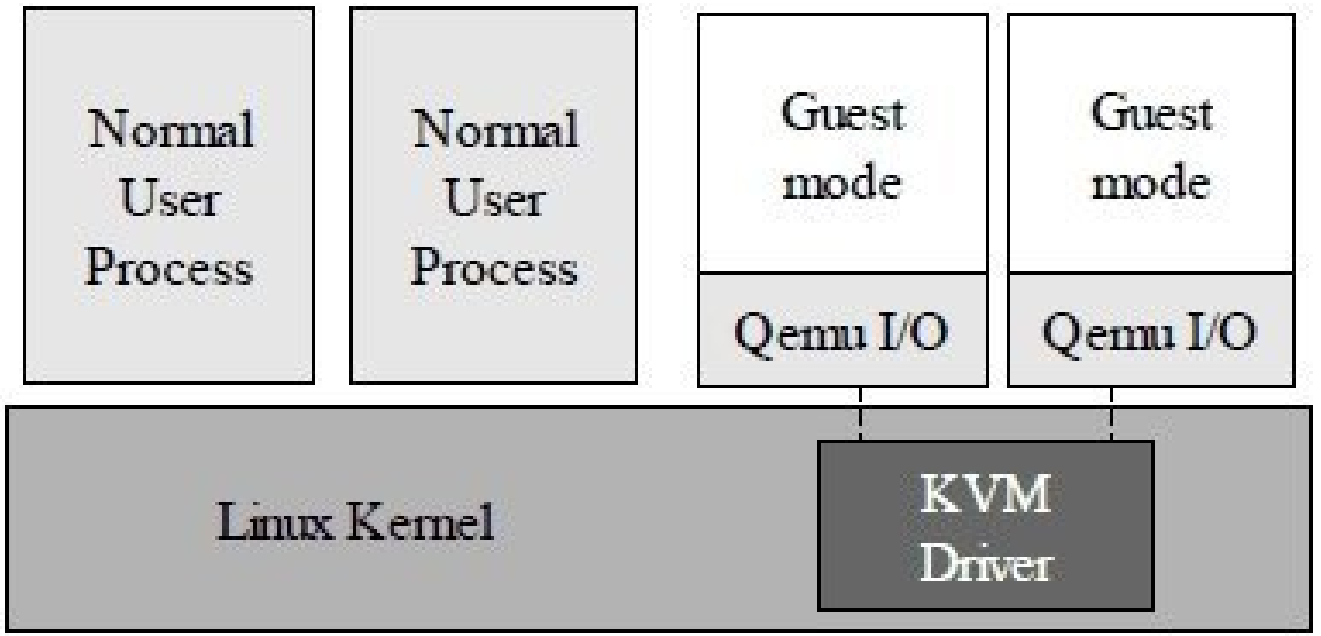


Fig. 6.1. KVM Model (source: Qumranet [3])

History

- 2006年10月以色列的Qumranet開放原始碼組織，開發一個Linux Kernel層級的全虛擬化Hypervisor，稱之爲Kernel-based Virtual Machine (KVM)，採GPL授權。
- KVM技術表現優異，獲得Linus Torvalds和Andrew Morton青睞，將其納入Linux核心當中，Linux 2.6.20 kernel已經將其包裝在內。
- RedHat在2008年前買下Qumranet，其Linux核心也由原本的Xen解決方案轉移至KVM，而2009年所推出的RHEL5.4便是內含KVM之作業系統。

Mode

- Guest Mode透過Modified QEMU至user-space 介面'/dev/kvm' 連至KVM Driver (@kernel mode)執行所有非I/O的Guest Code，見圖5.3。

- User Mode透過QEMU到Linux kernel上執行Guest I/O ，見圖5.3 。

- KVM 是由兩個主要的元件(components)所組成:
  1. Kernel Device Driver (managing the virtualization hardware) - 用來管理和模擬 Virtual Machine 的硬體。
  2. User Space Process - qemu 是一個 PC 硬體模擬器，經過 KVM 的修改過而成爲 kqemu屬於user space元件。

Framework

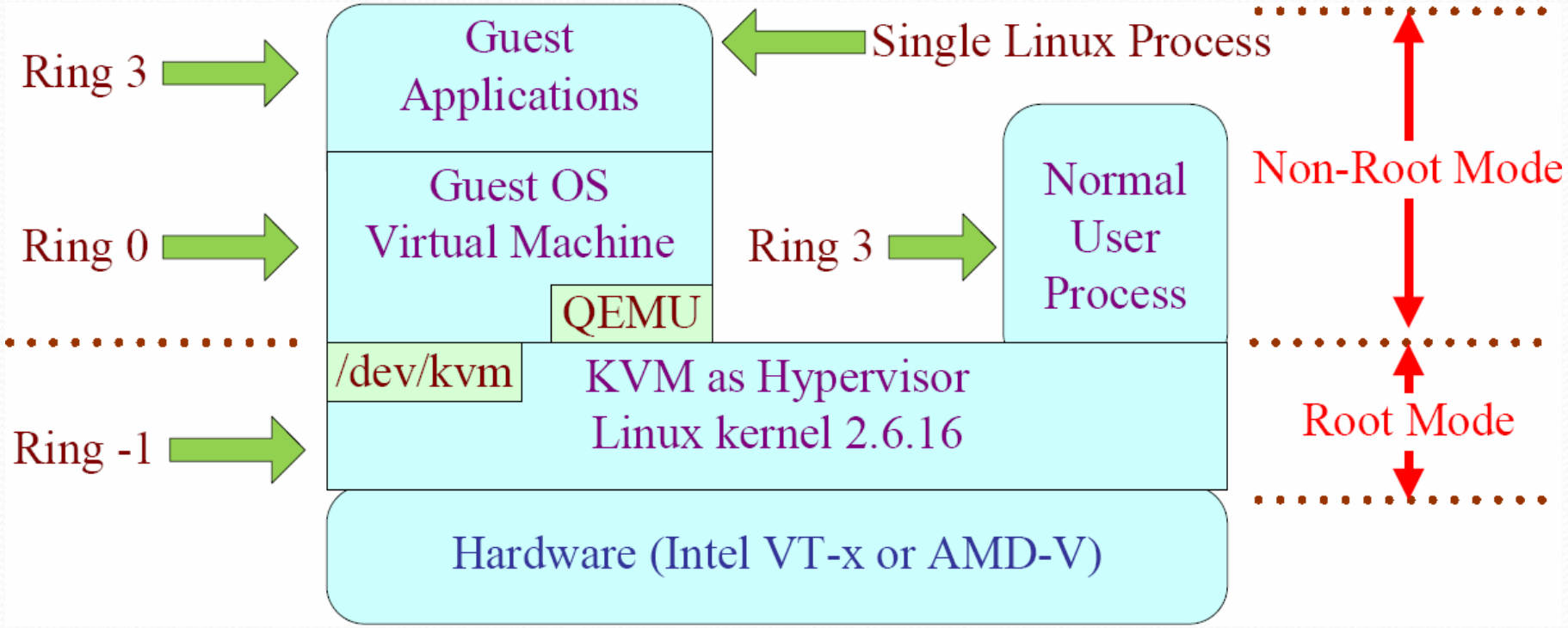


Fig. 6.2 KVM Framework

Operation Mode

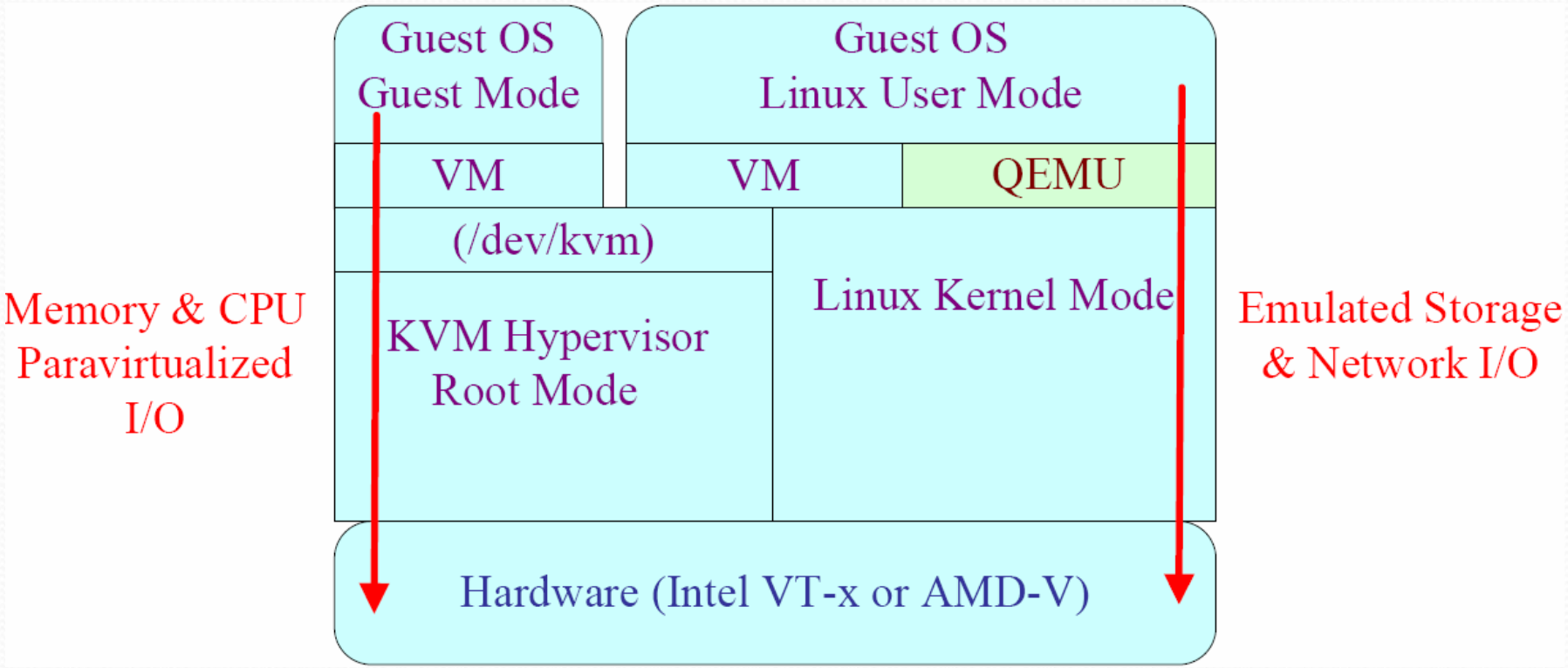


Fig. 6.3 KVM Operation Modes

User Space Tool for KVM

Linux ABI (application binary interface) : The Linux abi is a patch to the linux kernel that allows a linux system to run foreign binaries. [4]

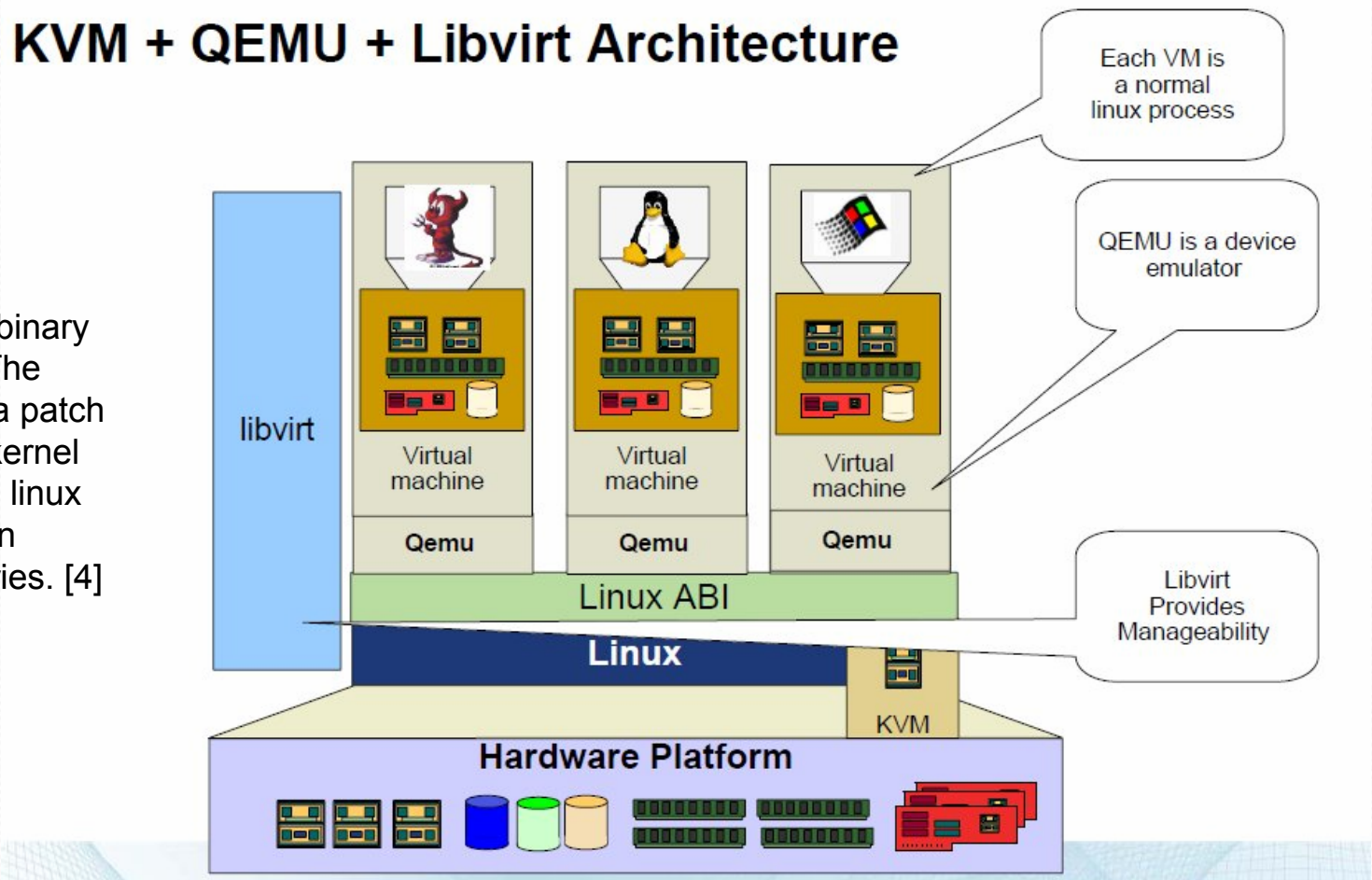


Fig. 6.4 KVM+QEMU+Libvirt Architecture (source: hagmann [4])

User Space Tools

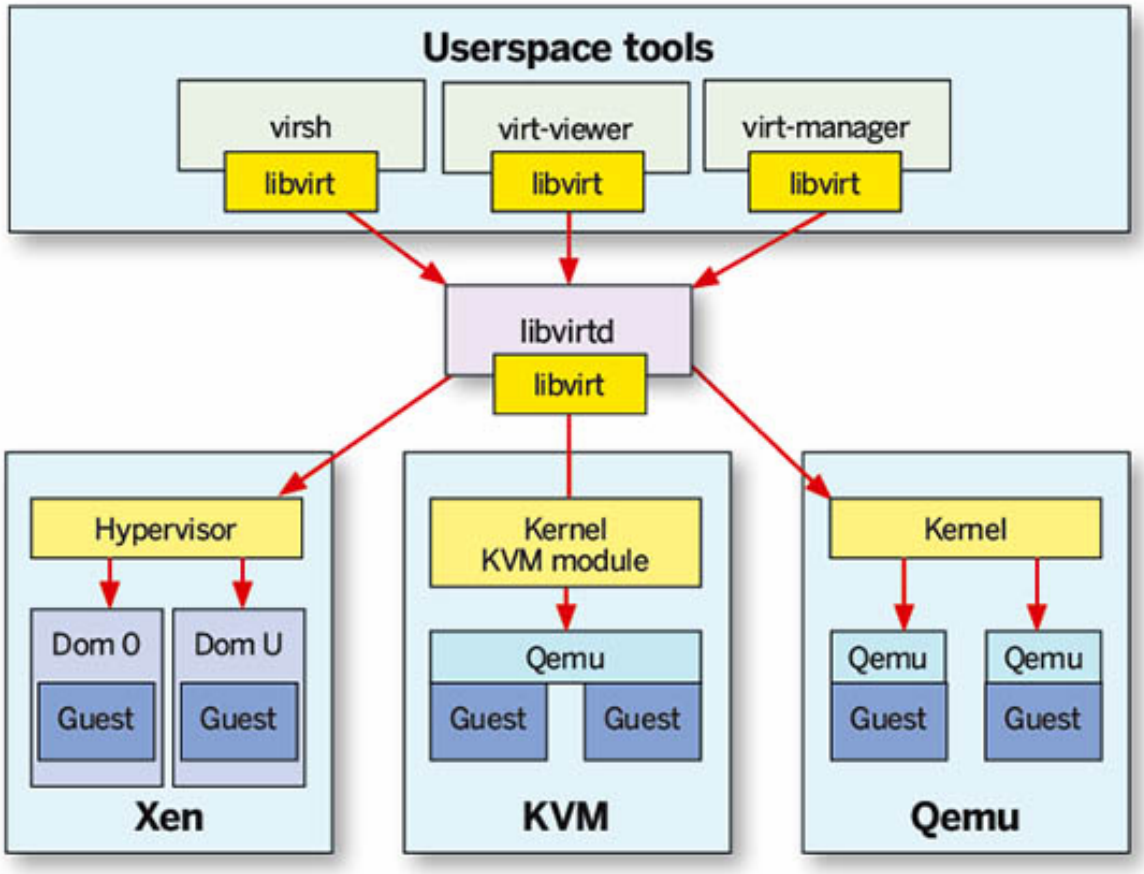


Fig. 6.5 User Space Tools (source: TuxRadar[6])

QEMU

- Fabrice Bellard公司針對 x86 有硬體支持虛擬化CPU的系統,修改自由軟體QEMU成爲一個只能使用在這樣系統上的CPU指令模擬器，它被稱爲 KQEMU 或是 QEMU Accelerator，最主要就是爲了提升 x86硬體支持虛擬化CPU對於虛擬化效能的提升 。

- 當運行在x86硬體支持虛擬化的Intel VT-x或AMD-V處理器上，KVM支持Linux(32和64位元)和Windows(32位元)的客戶機。

Analysis (1)

- KVM 主要的功能就只是一個 VMM (hypervisor) ， 他並沒有去模擬硬體的裝置。
- 模擬硬體的動作全都是交給 QEMU，不過 QEMU 只是存在 user space 的程式，它必須透過 /dev/kvm 來和 Linux kernel 與硬體溝通 。
- 先進的CPU架構(Intel VT-x或AMD-V)上也增加了硬體paravirtualization的支援(Xen後來利用了這樣的技術來達到full virtualization性能，支援Windows的VM) 。

Analysis (2)

- Paravirtualization的主要意義在予Light Hypervisor能力不足，需藉助外力以達成完整功能，因比有Administrative OS (Privileged Guest)用以增加使用IO Devices paravirtualization。

- KVM要作的就是
  - 在Guest OS之下建構paravirtualization層(硬體旁虛擬化)。
  - 在Guest OS之上提供VM的framework。
  - 這樣做的好處是即使沒有特別的硬體也能夠達到paravirtualization的效能(即 No Binary Translation)。

Analysis (3)

- 核心KVM模組，在Linux kernel中僅有3個內核模組：kvm、kvm_intel、kvm_amd，kvm_intel和kvm_amd分別對應Intel和AMD的CPU。它們提供來核心的虛擬機管理功能。

- QEMU 在User Mode之下，在Guest OS上執行I/O Device 存取的功能。QEMU是一個平台虛擬化解決方案，允許對一個完整的PC環境進行虛擬化（包括硬碟、顯示卡、網絡卡）。Guest OS所生成的任何I/O請求都會被中途截獲，並重新發送到QEMU行程模擬的用戶模式中。QEMU行程中提供了在User Mode對客戶機進行控制和管理的工具。

Example

- 下面使用KVM虛擬化技術創建一個虛擬客戶機實例的步驟，此步驟在fedora11下測試通過。
- 1）首先確認下面檔案存在於系統中，核心模組kvm,kvm_intel或者kvm_amd。用戶層程序qemu_kvm，沒有的話使用包裹管理器安裝qemu。
- 2）確保核心KVM模組已載入，可以使用qemu-user軟體包中提供的qemu服務自動載入，或者選擇手動載入。
- 3）如果不使用實體分區，請使用dd命令手動建立一個足夠大的影像檔案（也可以通過qemu_img命令實現此步，具體參考手冊）。
- 4）啓動客戶機，使用光碟影像檔案安裝系統，命令如下:
- kvm-no-acpi-m512-cdromguestos iso-hdavm-disk img
- 說明：-m指定Memory大小，-cdrom指定使用光碟機或者是影像光碟檔案，-hda指定作爲客戶機第一磁盤啓動器的實體磁盤或者影像檔案。
- 5）使用如下命令啓動當前實體磁盤上的系統:
- kvm-m512-hda/dev/sda
- 注意：同時在同一分區啓動兩個系統可能會導致分區數據出錯。

Performance

- 使用qemu_kvm命令啓動KVM虛擬機，如果當前主機CPU不支持VT技術，那麼qemu_kvm將使用傳統qemu模擬底層設備的方式提供底層虛擬支持，這樣虛擬機的性能將打打折扣。
- 在支持VT技術的主機系統上，qemu_kvm將會自動開啓CPU硬體虛擬化支持功能，在這種模式下客戶機在使用CPU和Memory時性能的損耗特別低，幾乎接近於真實係統。
- 但基於QEMU實現的KVM用戶層系統在圖形方面的性能目前表現較差，表現在圖像顯示效果不佳，鼠標移動有延遲。安裝kqemu增強工具可以一定程度的改善這個情況，但要實現接近於主機系統的圖形性能和流暢的用戶體驗，KVM還需要很大的進步。

Xen

- Xen是在2002年前英國劍橋大學計算機實驗室開發的一個旁虛擬化Para-virtualization自由軟體項目。

- 在Xen環境中，主要有兩個組成部分。一個是虛擬機管理程序Hypervisor，它是虛擬層位於硬體與虛擬機之間，主機開機後最先被載入到硬體之上，載入完成後就可部署虛擬機。Xen不包含任何與硬體對話的驅動程式。在Xen中，虛擬機叫做Domain ，其中有一個具特權的Guset 叫Domain0是Xen環境第二部分。

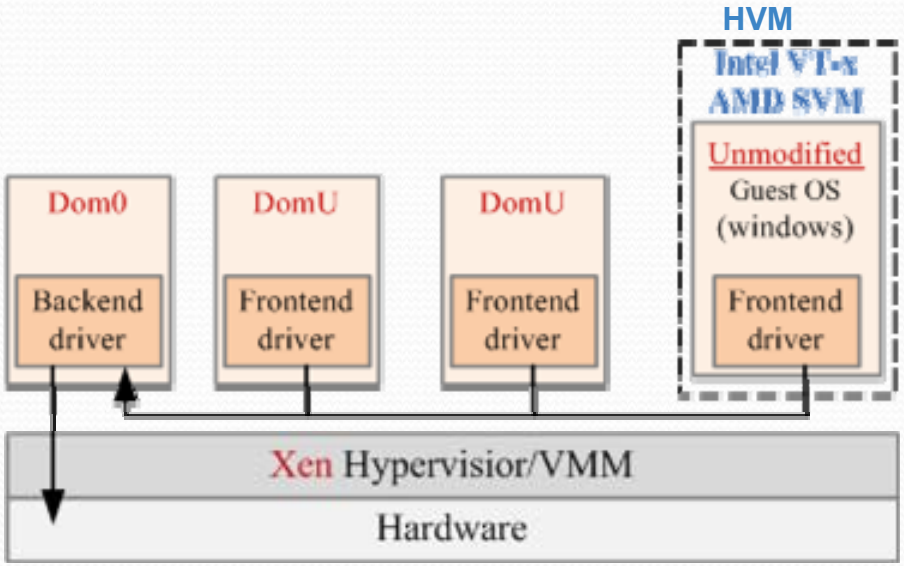
Xen Virtual Machine

- Domain0是一個特權Guset ，負責IO設備驅動程式的工作，並提供與管理員對話的界面去管理Hypervisor。
- 在創建虛擬機時，管理員使用配置程序與Domain0直接對話，並利用Xen的一些工具來創建其它虛擬機（DomainU），這些domainU屬於無特權Guest。
- 在Domain0中，還會載入一個xend行程來管理所有其它DomainU ，並提供這些虛擬機控制台Console(即管理員)的存取 。

Para-Virtualization Xen

- 旁虛擬化Hypervisor或VMM爲所有客戶機要存取到硬體層的介面， 這是一種安全性的考量和設計。
- Domo必需運行於Linux之下執行IO虛擬化，旁虛擬化之Hypervisor或VMM不含IO Devices Drivers，它執行CPU和Memory虛擬化。
- Hardware Virtual Machine (HVM)是運行於Native OS情況，因此使它不覺得是在虛擬層上運作，但是必需執行在CPU有支持Intel VT or AMD-V的硬體環境。例如，在客戶機中使用Windows XP。

Xen Architecture



Dom 0 / Host - 開機後就會啓動的 domain 0

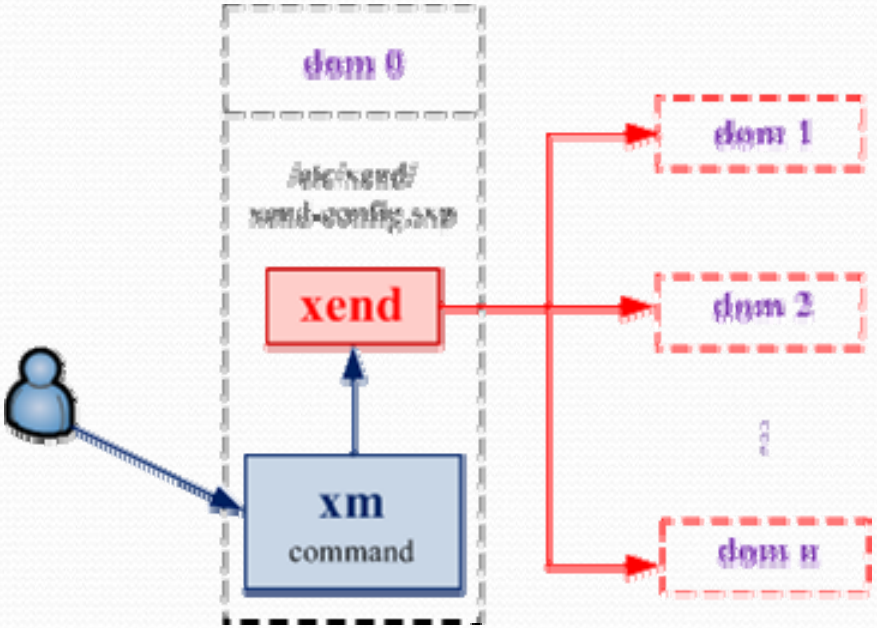
Dom U / Guest - Virtual machine (需透過 dom0 來控制硬體資源)

HVM - Xen可藉由 CPU 指令集(Intel VT-X, AMD SVM)的支援來做到 Full Virtualization

Hardware Virtual Machine (HVM)
AMD SVM (Secure Virtual Machine) or AMD-V (AMD Virtualization)
Intel VT-x (VT: Virtualization Technology)

Fig. 6.6 XEN Architecture (source: NCHC [7])

How to Operate It



xend - Xen 的核心程序
xm - 使用者透過 xen manager 來管理 xend 和控制 VM

Fig. 6.7 XEN Management (source: NCHC [7])

Xen - Direct IO Path not Supported from Various VMs

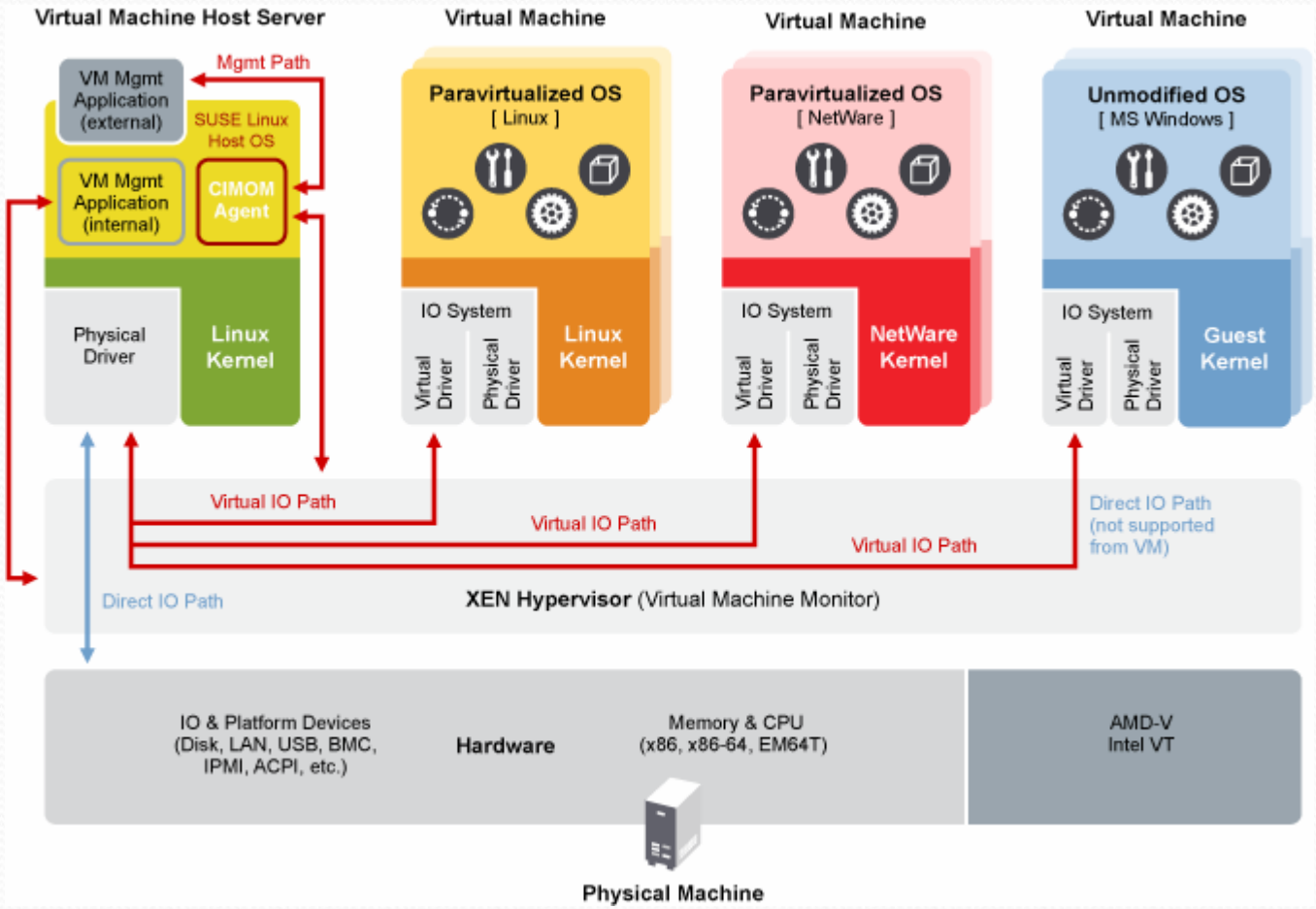


Fig. 6.8 Xen Virtualization Architecture (source: Novell [8])

Xen 3.0 - Making I/O Feel Secure

AGP: Accelerated Graphics Port

ACPI: Advanced Configuration and Power Interface

PCI: Peripheral Connect Interface

SMP: Symmetric Multi-Processor

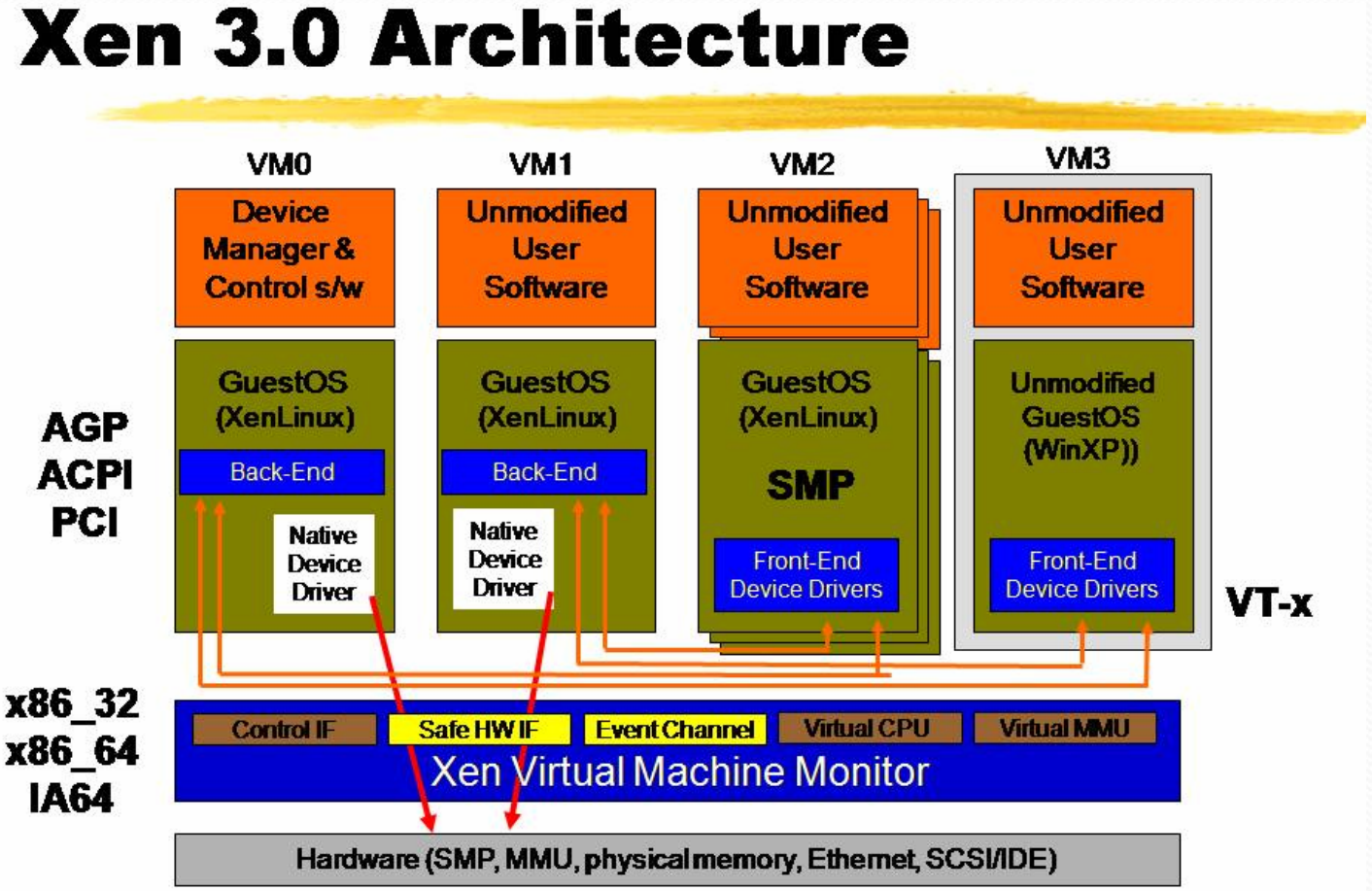


Fig. 6.9 Xen 3.0 Architecture (source: IT 2.0 [9])

RedHat Xen Full Virtualization

PV for full virtualized Linux

QEMU for full virtualized Windows XP

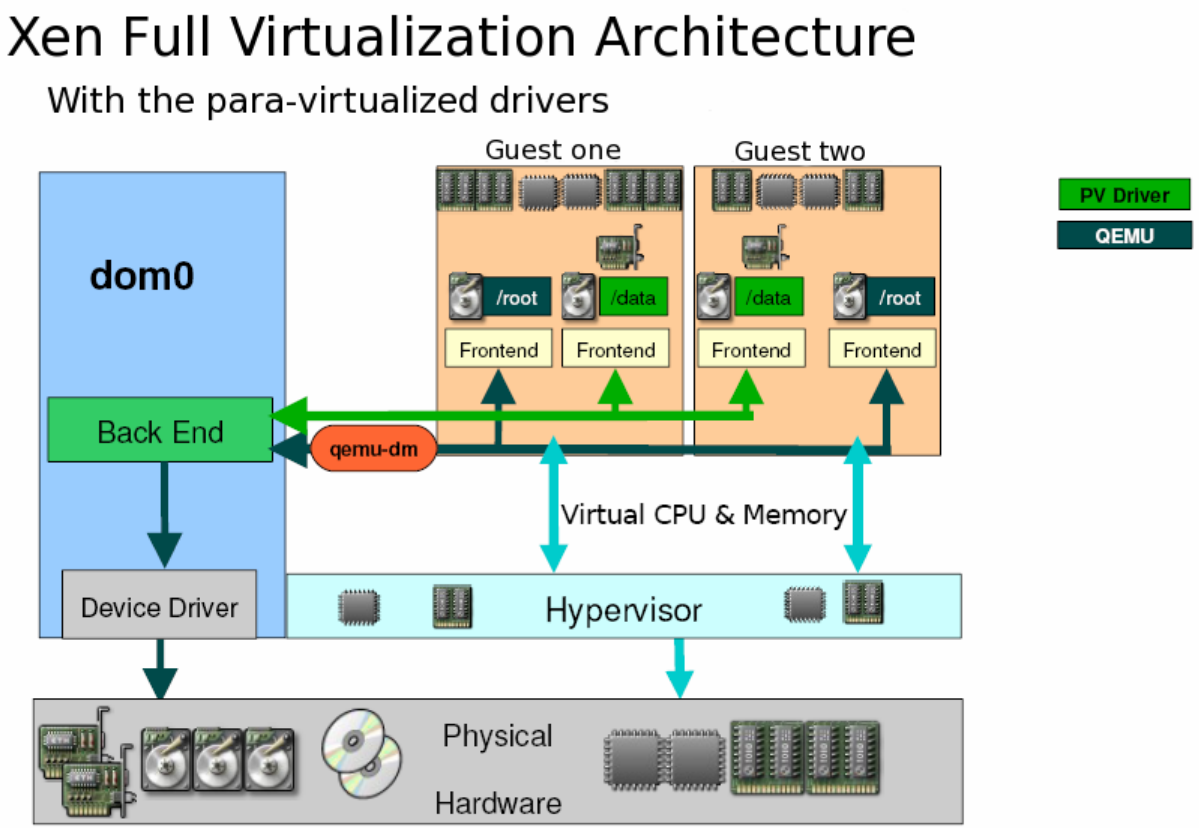


Fig. 6.10 RedHat Xen Full Virtualization (source: RedHat [10])

RedHat Xen Para-Virtualization

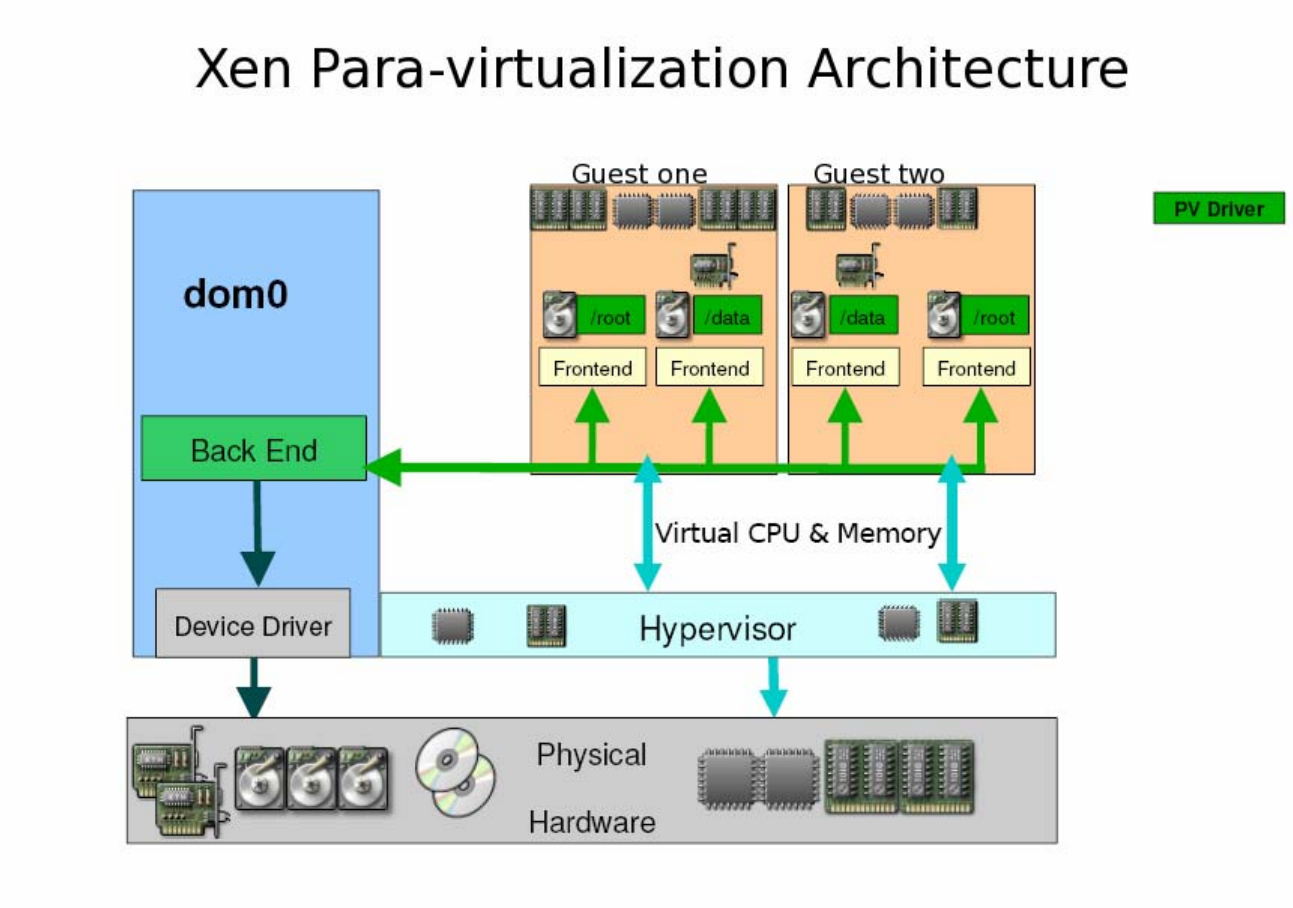


Fig. 6.11 RedHat Xen Para-virtualization (source: RedHat [10])

旁虛擬化的說明

- 旁虛擬化核心使用Xen kernel 或 *kernel-xen* package，旁虛擬化客戶機的核心同時執行及使用主機的函式庫和設備。
- 安裝旁虛擬化，在安全設定(SELinux and file controls)限制下，可以完全存取所有系統的設備。
- 旁虛擬化比全虛擬化快，因爲旁虛擬化可有效達成負載平衡、隨需配置、安全、合併等優勢。

Citrix XenServer Architecture

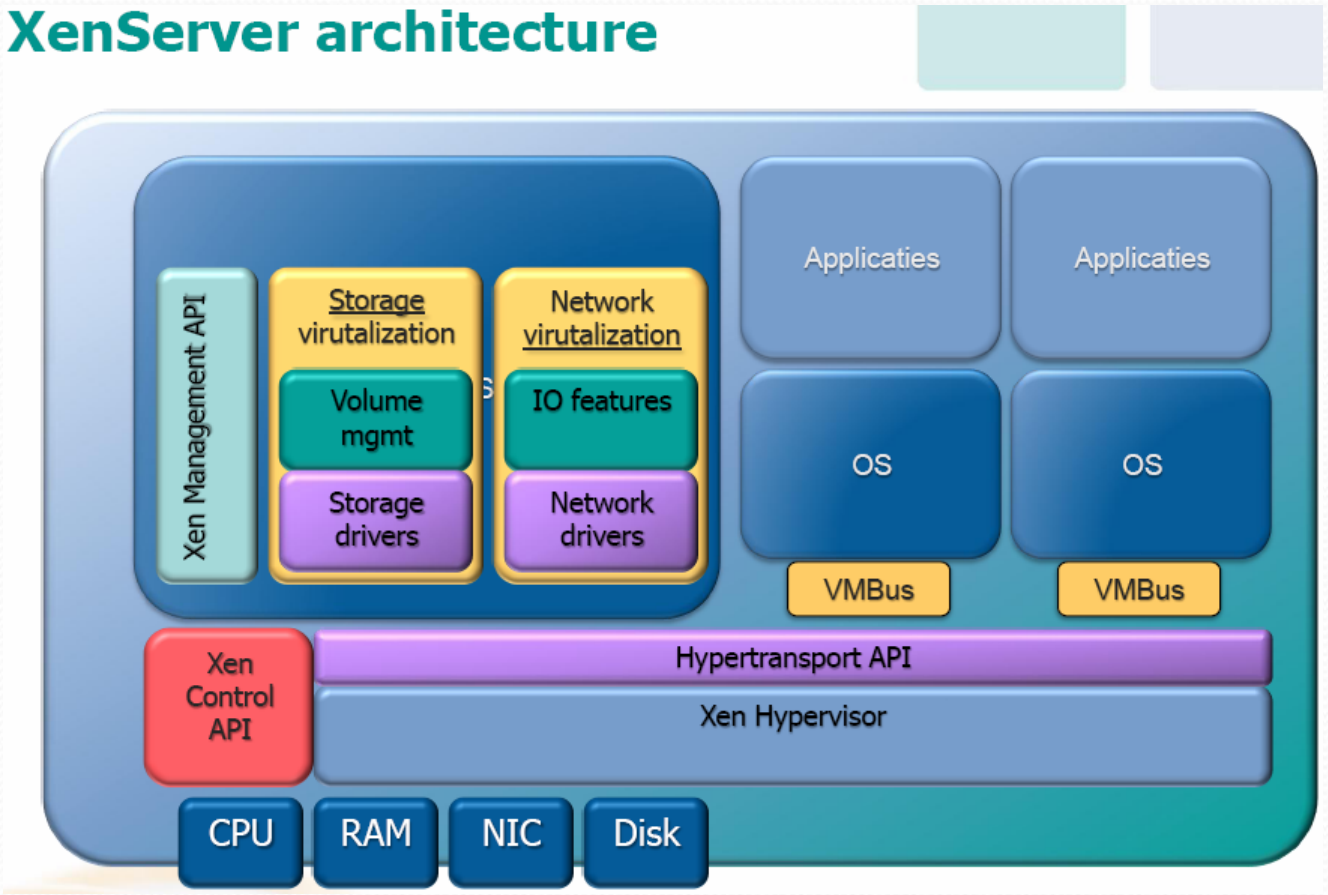


Fig. 6.12 Citrix XenServer Architecture (source: PQR [11])

虛擬化堆疊概念

- 虛擬化堆疊
  - 硬體虛擬化 (Hardware Virtualization)
  - 作業系統虛擬化 (OS Virtualization)
  - 應用虛擬化 (Application Virtualization)
- 例如: Virtualizaton Stack in Citrix XenApp+XenServer

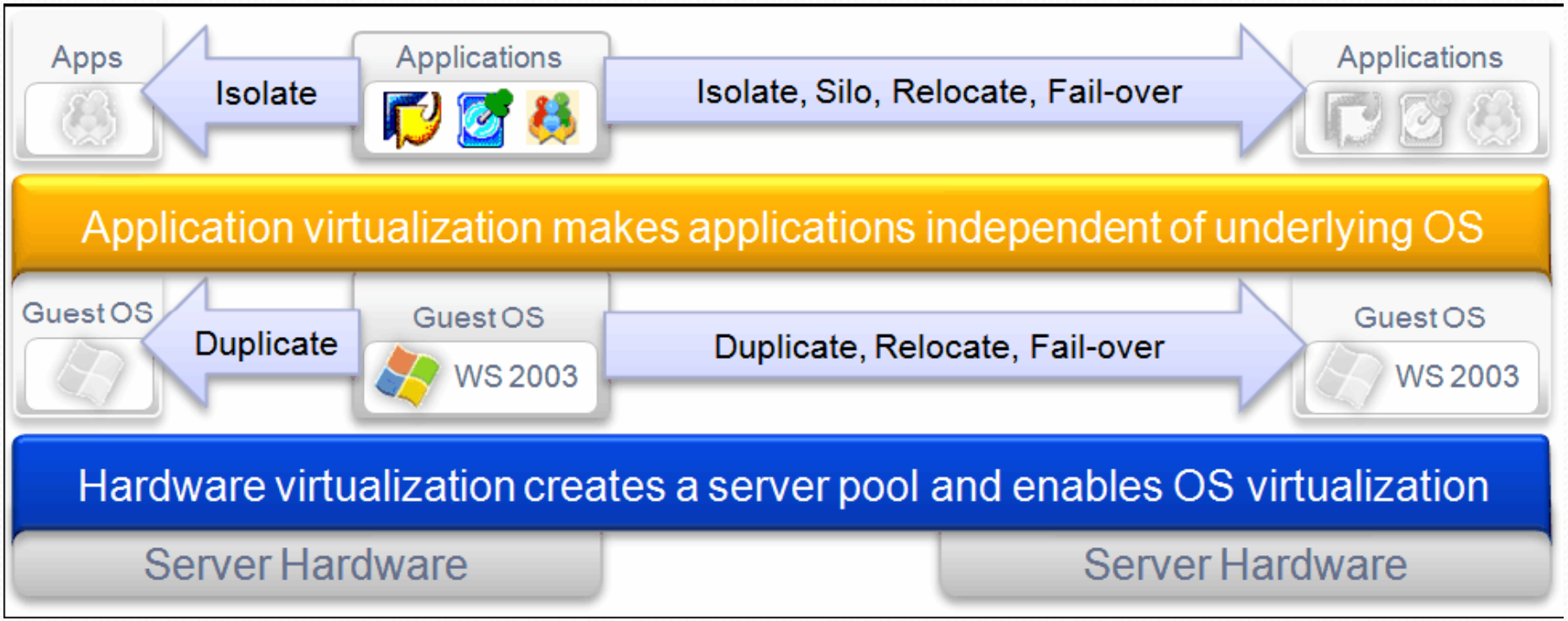
- 硬體虛擬化: 一種技術建立軟體介界模擬底層硬體，使一個實體同時支撐多個客戶機作業系統，也整合多個實體成爲一個資源池硬體，可讓客戶機作業系統在多個實體間移動，以這樣方式改進硬體的效能、相容性、容量。

虛擬化堆疊概念(續)

- 作業系統虛擬化:一種技術隔離作業系統和運行它的硬體，如此作業系統可在不同的硬體設備間移動，並可同時也與其它作業系統在相同設備上做平行運算。

- 應用虛擬化:一種技術隔離應用和在它底層作業系統，如此許多應用和其它相容應用可以進行平行運算，且它們可以自由地在硬體設備和作業系統間移動。

Virtualization Stack



Silo:筒倉　　Fail-over:故障排除

Fig. 6.13 Citrix XenApp+XenServer (source: Citrix [12])

Hyper-V R2

- 微軟於2008年5月推出Hyper-V以來，9月也發表虛擬化管理軟體SCVMM（System Center Virtual Machine Manager）2008，不但可以管理 Hyper-V虛擬化環境，又可管理VMware的環境。

- 微軟爲了強化虛擬市場的競爭，更推出單機免費版的Hyper-V Server 2008。

- 2010年推出Windows Server 2008 R2[13]時，將現階段虛擬機器線上轉移功能的名稱從Quick Migration更名爲Live Migration，將可做到線上不停機轉移的功能。

Hyper-V 各版本功能比較表

Table 6.1 Comparison of various Hyper-V

	Microsoft Hyper-V 2008 Server	Windows Server 2008 Standard	Windows Server 2008 Enterprise	Windows Server 2008 Datacenter
支援多種作業系統 (Windows & Linux ....)	V	V	V	V
本地端圖形介面		V	V	V
高可用性(支援叢集)			V	V
支援快速移轉			V	V
Host OS支援大於32GB記憶體			V	V
Host OS支援大於四顆CPU			V	V
Host OS包含Guest OS授權數量(以Windows OS計算)	每個Guest OS都需額外授權	含一個Host OS及一個Guest OS授權	含一個Host OS及四個Guest OS授權	含一個Host OS授權，無限個Guest OS

Hyper-V R2 (cont.)

- Type 1 提供最佳效能及可用性。而Type 2提供較佳的相容性，Hyper-V採用Type 1的架構，因此可提供虛擬機器較佳的執行效能。但相對的，Hyper-V目前可支援的作業系統便少得多。 [13]

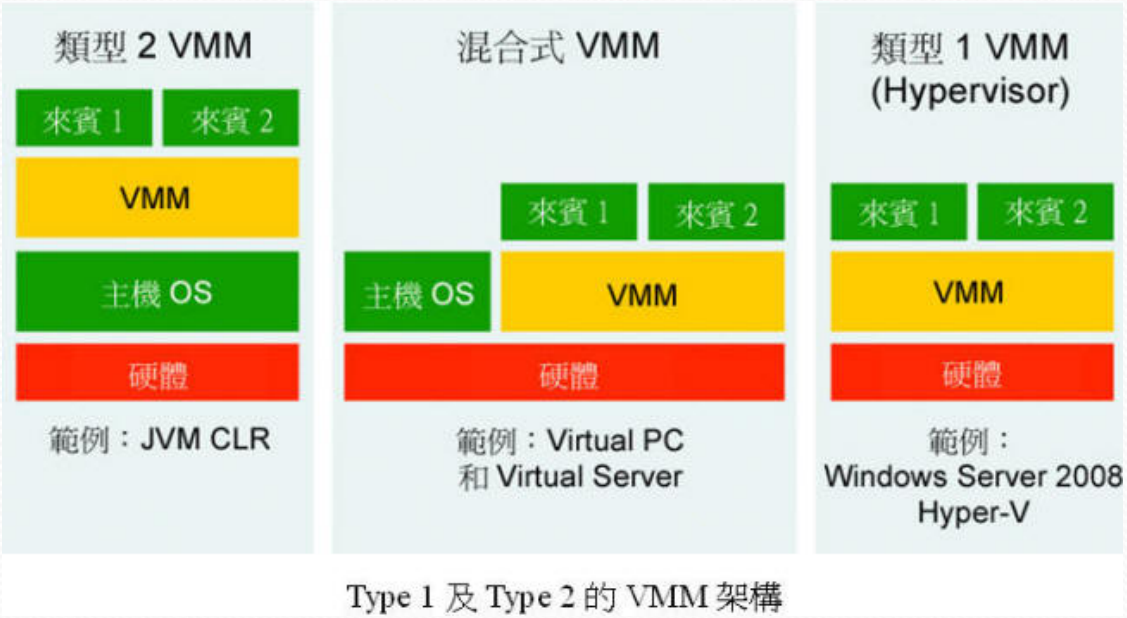


Fig. 6.14 Type1 and type 2 of VM architecture (source: cc, ntu [13])

Hyper-V R2 Requirements

- 使用Windows Server 2008已內建了Hyper-V，只需要角色(Role)啓動的地方去啓動Hyper-V即可。如果使用單獨的Hyper-V Server，只有Server Core的介面可用。如果硬體很特殊的話，甚至必須手動安裝網卡。

- 執行Hyper-V R2必須採用64 bit的CPU，而且CPU也必須支援Inter-VT或AMD-V技術。此外，CPU也必須支援硬體DEP(Data Execution Prevention)功能，它是系統安全與防護，避免執行某些不被允許的程式。例如在記憶體中某些位址只能用來存放資料，而非可以執行的程式。

Hyper-V R2 Requirements

- 除了以上Intel-VT或AMD-V以及支援DEP的限制外，並不是每一個64 bit的CPU都能執行Hyper-V，還可能有其他限制。知道你的硬體是否支援最好的方法是，就是利用SecurAble這套工具去測試。

Hyper-V R2 Features [14]

- **High Availability & Live Migration**
- Cluster Shared Volumes with **I/O Redirection**
- 64 Logical Processor (Core) Support
- Core Parking & Processor Compatibility Mode
- **Hot Add/Remove Storage**
- Second Level Address Translation
  - Leveraging new Virtualisation technology built into next generation of Intel (EPT) / AMD (NPT) chips
- **Boot from VHD**
- Networking Improvements
  - NIC Teaming, Jumbo Frames & TCP Offload
- Virtualised I/O

Hyper-V R2 Architecture

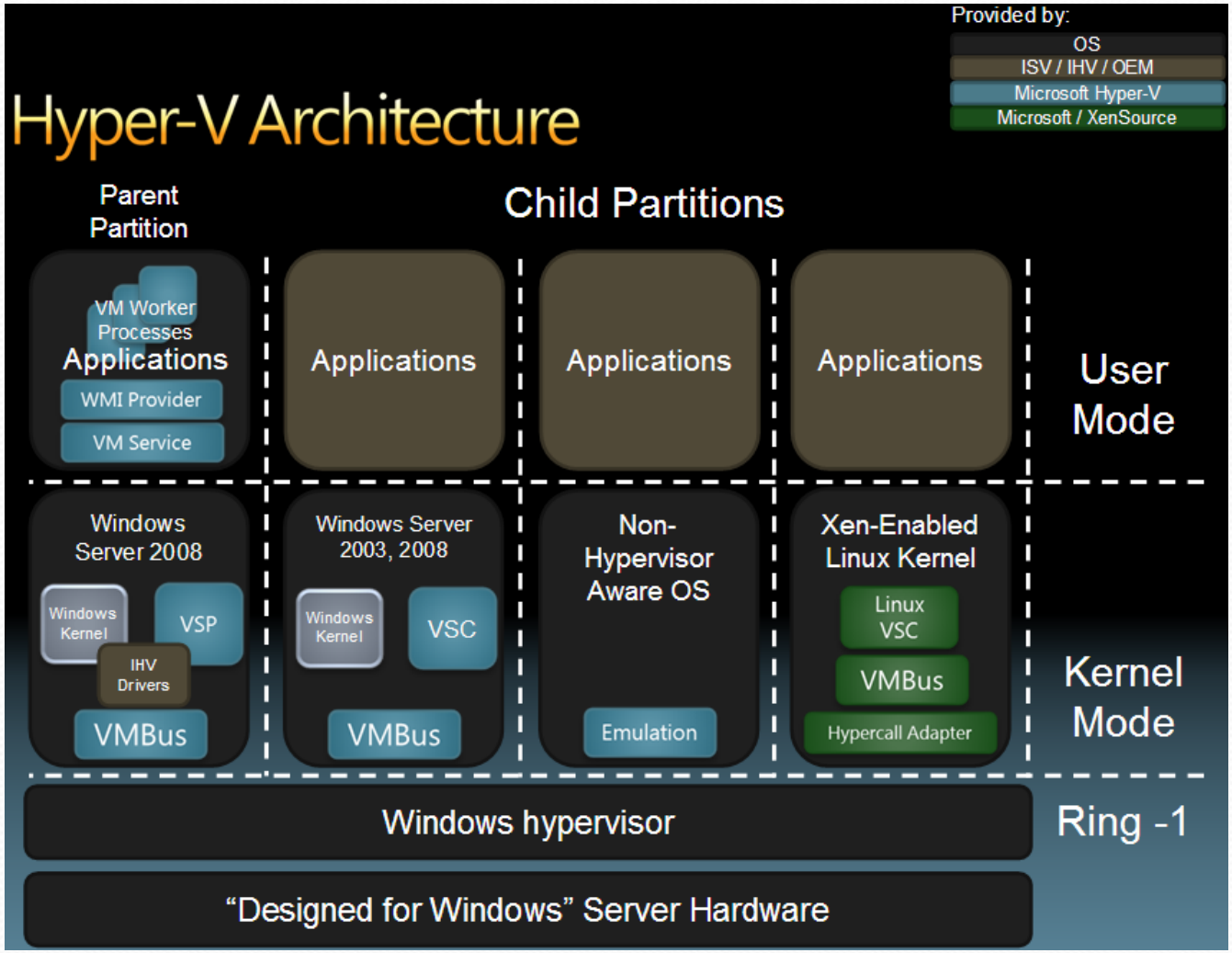


Fig. 6.15 Hyper-V R2 architecture (source: Microsoft Taiwan [14])

Hyper-V R1 vs R2

Table 6.2 Comparison of Hyper-V R1 to R2 [14]

	Microsoft Hyper-V Server 2008	**Microsoft Hyper-V Server R2**
Processor Support	Up to 4 processors up to 24 LPs	Up to 8 processors Up to 64 LPs
Physical Memory Support	Up to 32 GB	Up to 2 TB
Virtual Machine Memory Support	Up to 32 GB total (e.g. 31 1 GB VMs or 5 6 GB VMs)	64 GB of memory per VM
Live Migration	No	Yes
High Availability	No	Yes
Management Options	Free Hyper-V Manager MMC SCVMM	Free Hyper-V Manager MMC SCVMM R2

VMware

- VMware是一個已商業化之全虛擬化技術Hypervisor，可將它視爲位於客戶作業系統和硬體之間的抽象層，該抽象層允許任何其它客戶作業系統運行在主機作業系統之上(Hosted architecture)， VMware也能虛擬I/O設備,增加了高性能設備驅動程式到Hypervisor中(Bare Metal Architecture)。
- 整個虛擬機環境實際上是一個單獨的檔案,這意味整個系統(包括客戶作業系統、虛擬機、虛擬硬體)可以簡單快速地移轉到另一個新的主機上,從而實現熱遷移或負載平衡。

Vitual Machine in VMware [15]

- 虛擬機可允許一台實體主機同時執行多個作業系統。
- 一台強大電腦主機能做30台電腦的事，在佔用面積、耗電量、空調、維護人事成本上，都能有效節約。
- 可在一台實體主機內執行多個虛擬主機，每一台虛擬主機既可以互相用線路連線、又可以獨立運作，互不干涉。
- 可將多台實體主機合併成一台大的資源池(Resource Pool)，統籌管理、分配在上面的虛擬機。

Vitual Machine in VMware [15]

- 只要實體主機的硬體能力許可，便能靈活地建立或移除多台虛擬機，可保持高度彈性。
- 虛擬主機往往在映像檔內直接執行、存取的檔案、作業系統，使得備份、搬移虛擬機較實體主機容易許多。
- 虛擬化可使實體主機擴增、維修、更換不須關閉虛擬機，只要把裡面的虛擬機搬移到別的虛擬機即可。
- 可使用虛擬機測試不穩定的軟體，再利用快照完全還原整個系統(僅需數秒鐘)，而不須在實體主機測試。

Virtualization Fabric

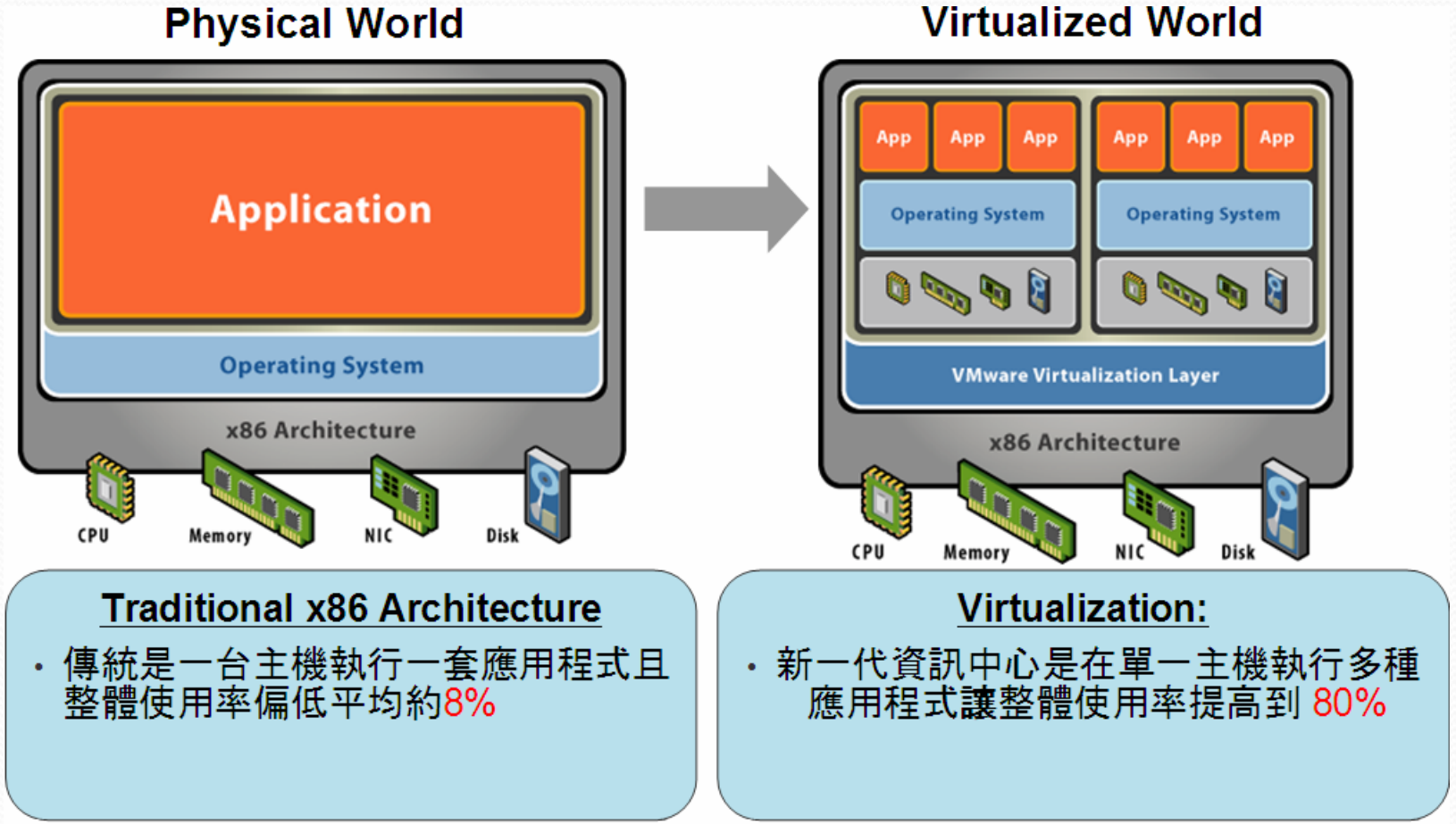


Fig. 6.16 Virtualization fabric (source: VMware [15])

VMware – Server in progress

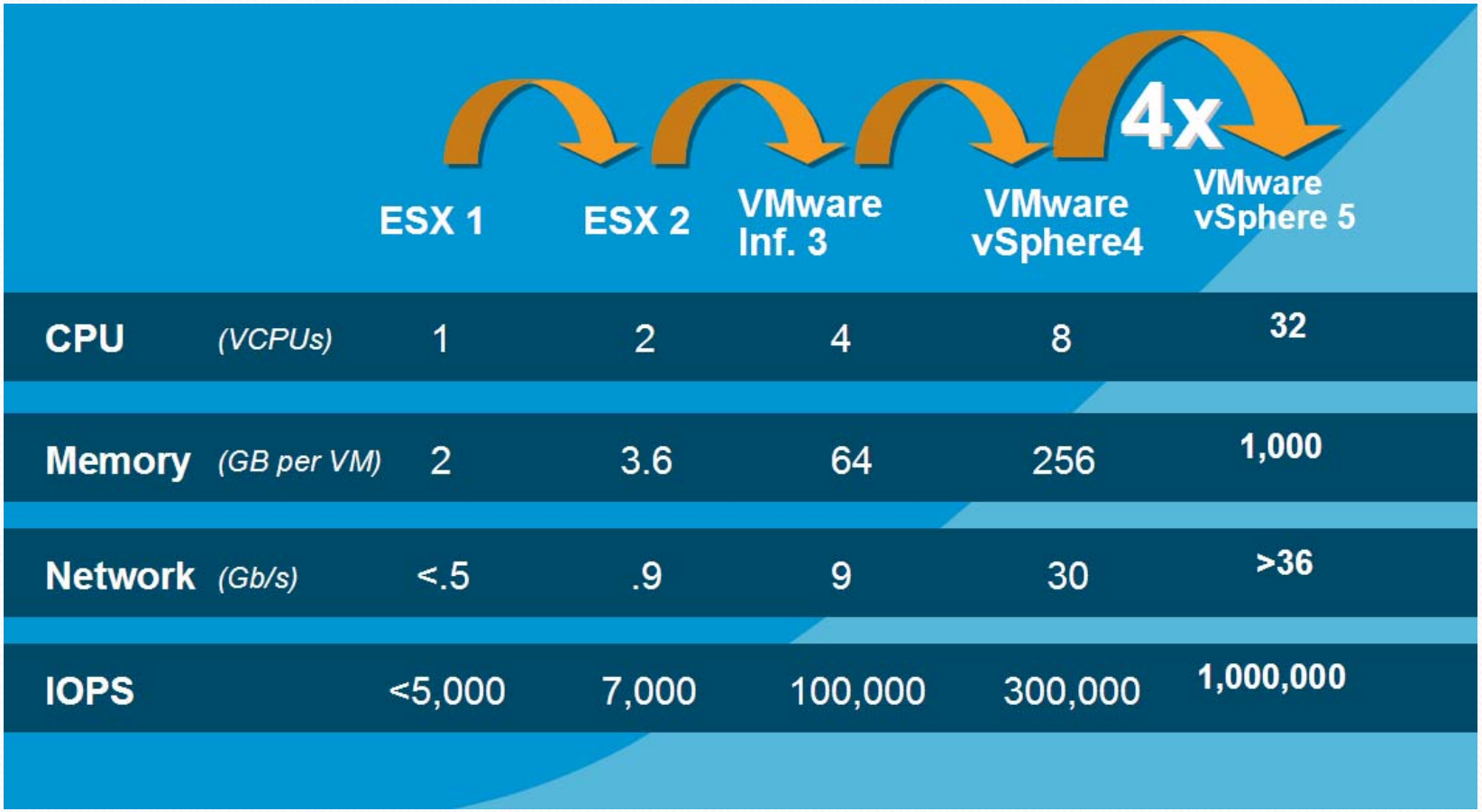


Fig. 6.17 VMware server in progress (source: VMware [15])

VMware -- Cloud Platform

Fig. 6.18 VMware cloud platorm (source: VMware [16])

vSphere Architecture

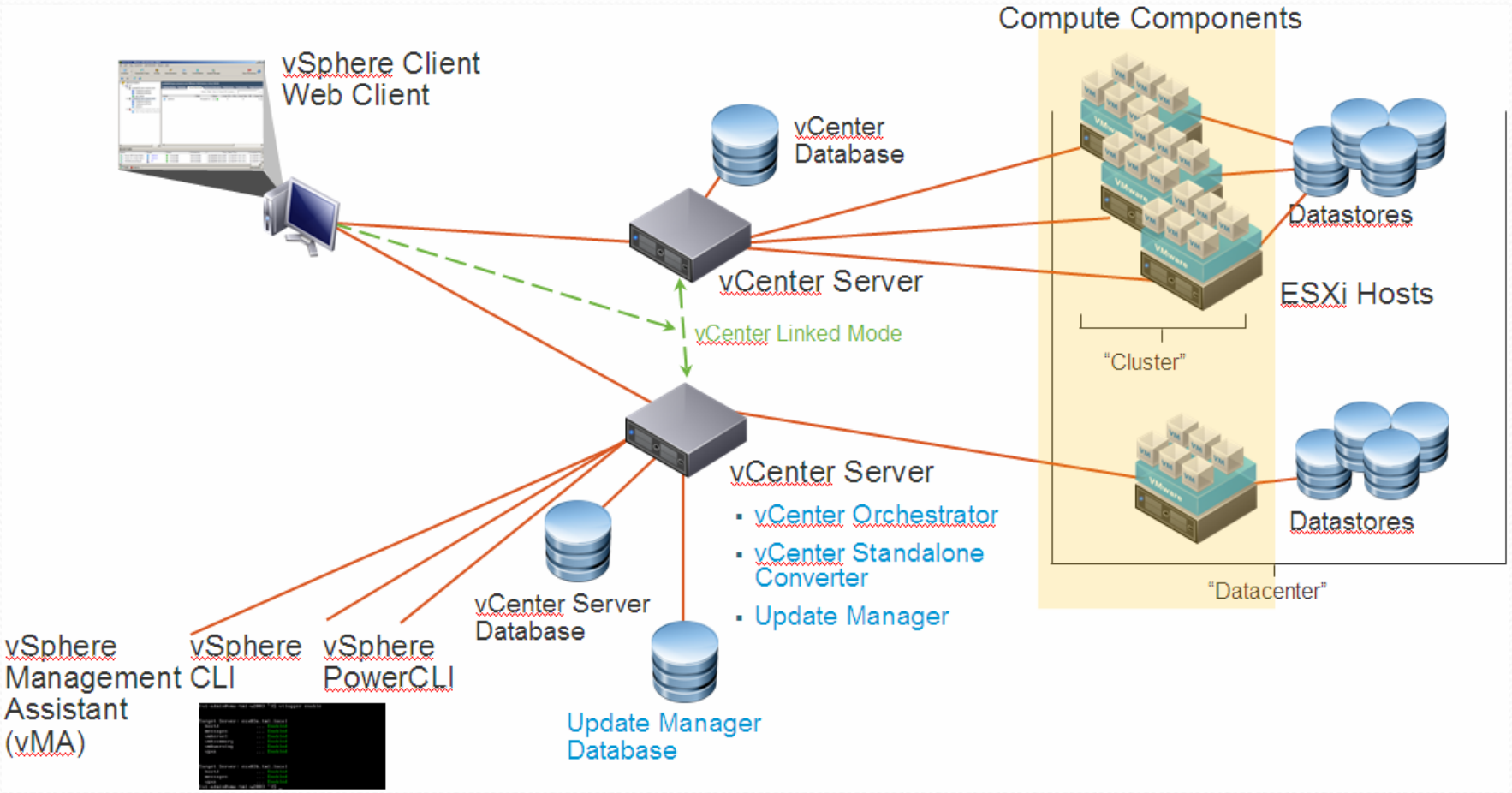


Fig. 6.19 vSphere architecture (source: VMware [15])

OpenStack

- OpenStack[16] 整合NASA Nebula Platform和Rackspace's Cloud Files Platform程式碼，在2010年7月所共同成立IaaS 的開放原始碼軟體專案。在2012年全球已經有超過 150家公司陸續加入參與這個專案，其中還包括 Intel、AMD、Canonical、SUSE Linux、Cisco、Dell、HP等大廠。

- OpenStack[17] 採用Apache License授權方式，使用Python撰寫程式，2012.4.5 釋放Essex版本。

- OpenStack主要是集合現有的開放原始碼專案軟體來提供一個可在大型分散環境中運行的雲端作業系統，同時，整合雲端技術相關的開放原始碼軟體專案也是重要任務之一 。

OpenStack Projects

- OpenStack [17]目前共分爲五個專案在進行:
  - 負責雲端運算的 OpenStack Compute (Nova)
  - 負責儲存的 OpenStack Object Storage (Swift)
  - 負責映像檔管理服務的 Image Service (Glance)
  - 負責辨識的 Open Stack Identity Management (Keystone)
  - 負責使用者介面的 User Interface Dashboard (Horizon)

- Nova 提供供虛擬伺服器，就像 Rackspace Cloud Servers or Amazon EC2。
- Swift提供供物件/區塊儲存，就像Rackspace Cloud Files or Amazon S3。
- Glance提供發現、儲存、檢索Nova所需虛擬機映像檔。

OpenStack vs OpenNebula

- OpenStack 目前主要是安裝在 Ubuntu 10.04 LTS 版本，不過，OpenStack 也可以安裝在 Redhat Enterprise Linux 6.0 上，但是，會有些的限制，在資料庫的部分，也只有支援 PostgreSQL 和 MySQL 兩種資料庫。

- OpenStack 目前有API相容於Amazon EC2和S3，客戶在Amazon Web Services上的任何應用程式可以直接Porting到OpenStack平台上使用。

- OpenNebula[18]是早期歐洲發展出來的VMM，而OpenStack近期由美國所發展出來的VMM。兩者皆是OpenSource。

Cloud Provider Conceptual Architecture

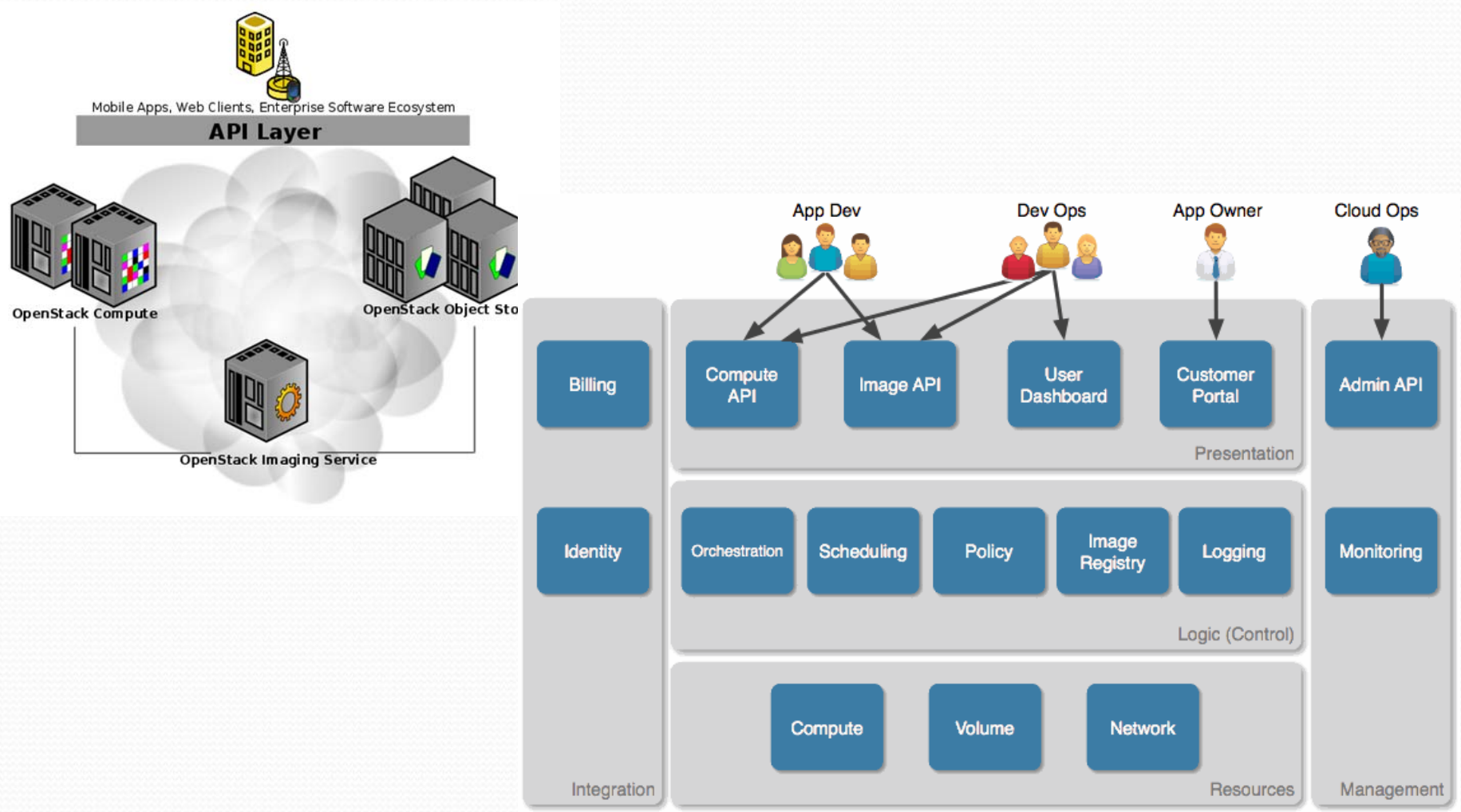


Fig. 6.20 Cloud provider conceptual architecture (source: OpenStack.org [16])

Compute Logical Architecture

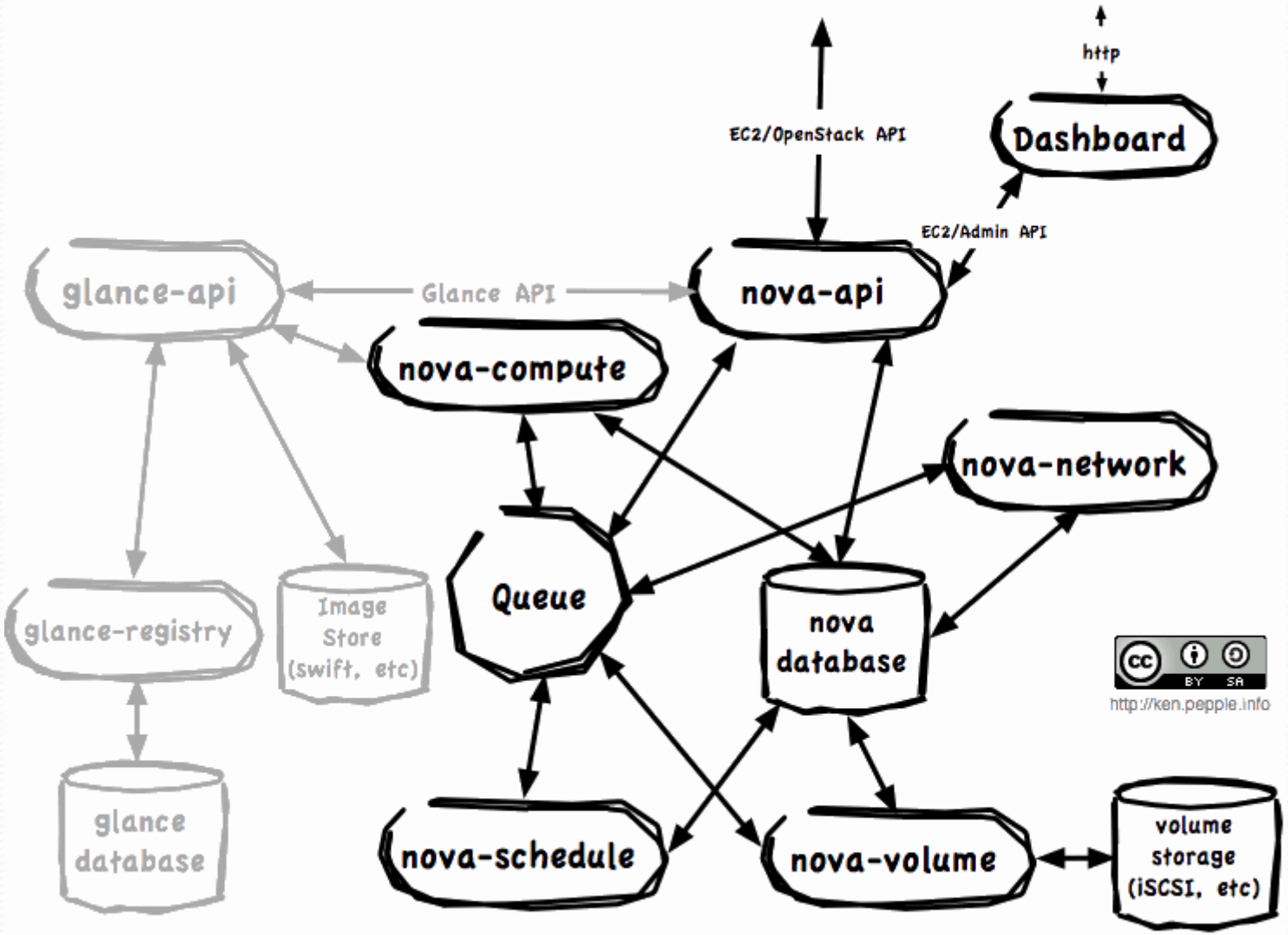


Fig. 6.21 Compute logic architecture (source: OpenStack.org [16])

Nova Conceptual Mapping

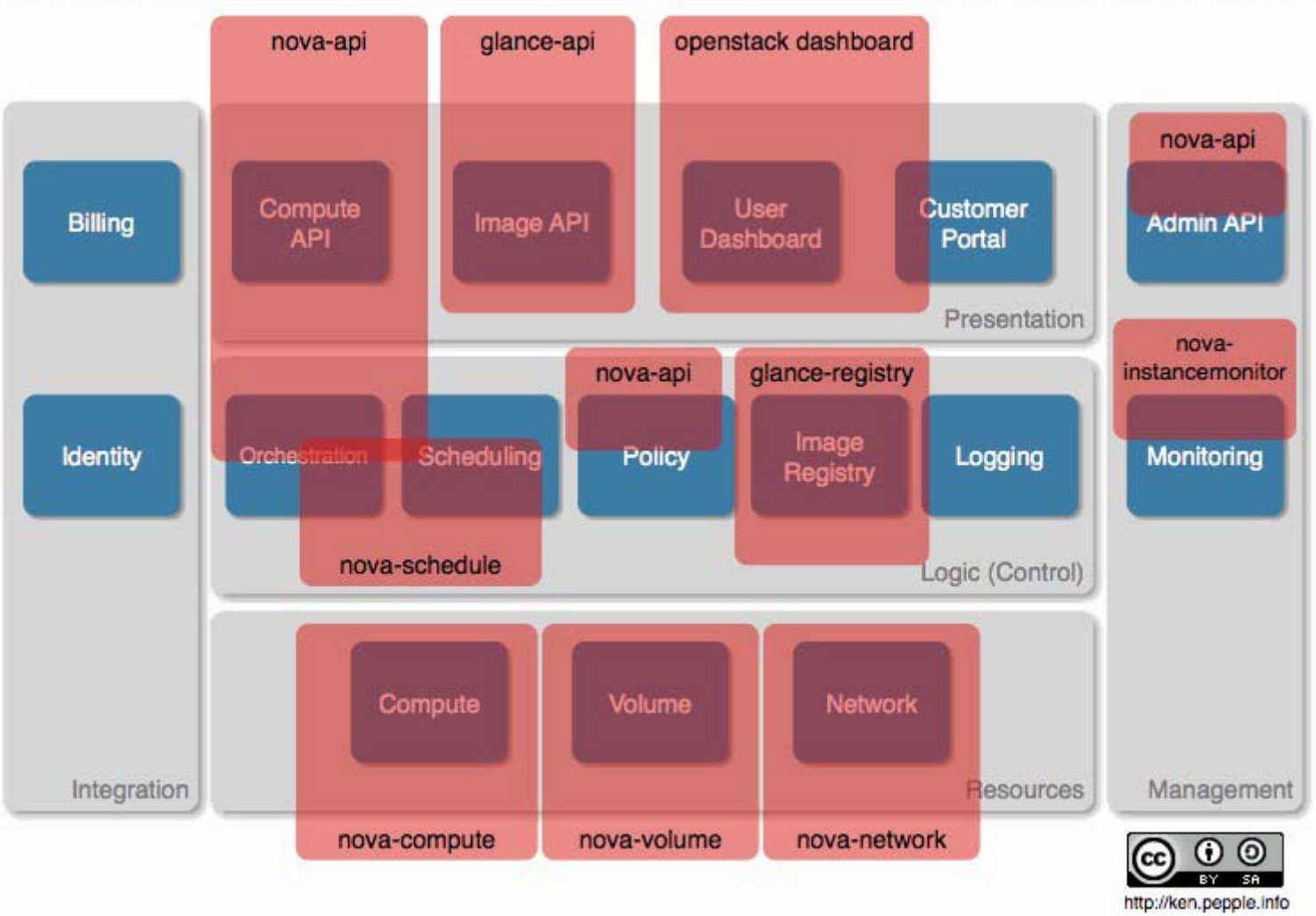


Fig. 6.22 Nova conceptual mapping (source: OpenStack.org [16])

Proxmox Virtual Environment

- **Proxmox Virtual Environment** 是運行虛擬設備、虛擬機器的一個易於使用的開放原始碼虛擬平台 [19]。
- Proxmox VE 是一個開放原始碼專案，由Proxmox Server Solutions 公司開發以及維護 [20]。
- Proxmox VE 1.x 採用 Gnu General Public License version 2授權許可 [19]。
- Proxmox VE 2.x 後改採用GNU Affero General Public License, version 3授權許可 [19]。

Virtualization Platform

- Proxmox VE是一個針對企業所設計的開放原始碼虛擬平台。

- Proxmox VE對於虛擬化你的伺服器有許多優勢，包含以下所列 [19]:
  - 增加硬體使用率 。
  - 節約用電，減少二氧化碳 。
  - 減少實體硬體，以及更少的硬體維護人員。
  - 不要浪費時間在軟體安裝， Proxmox VE在幾分鐘內就可以完成設備安裝。
  - 簡易的備份、還原功能。

Container and Full Virtualization

- Proxmox VE 對於效能以及可用性有所優化。爲了獲得最大的靈活性，使用ISO-installer來做裸機安裝，以下所列的虛擬化技術將可被使用 [19]。
  - **Container Virtualization (OpenVZ)**
  - **Full Virtualization (KVM)**
  - **Para-Virtualization (KVM)**

Bare Metal ISO Installer

- Proxmox VE 裸機安裝完整的作業系統和管理工具只需3到5分鐘 (取決於所使用的硬體) 。
- “裸機”表示您可以從空的伺服器開始安裝，而沒有必要事先安裝一個基本的作業系統。
- 裸機安裝包含以下項目 [19]:
  - 完整的作業系統 (Debian Lenny 64)
  - 使用LVM2的來對硬碟作磁碟分區
  - Proxmox VE 核心支援OpenVZ 和 KVM
  - 備份、還原工具
  - 基礎的網頁管理介面
  - Proxmox使用LVM - 備份和恢復LVM資料

Central Web-based Management

- Proxmox VE 是樸實的，沒有必要再安裝一個獨立的管理工具，一切都可以通過網頁瀏覽器 (IE6/7/8/9，Firefox 2/3/4，還有其他一些瀏覽器也可運行)進行管理。

- 您可以藉由Web介面進入管理工具 https://MasterIPaddress (預設登入帳號: root，密碼則是在安裝過程中設置的密碼)

Central Web-based Management

- 集中化網頁介面管理的特點 [19]:
  - 整合式主控台查看虛擬機
  - 無縫式整合和管理的Proxmox VE叢集
  - AJAX [21] 技術動態更新的資源
  - 確保所有虛擬機的訪問通過SSL加密(https); (注意:工作正在進行中，VNC控制台沒有加密)

Three Cluster Nodes

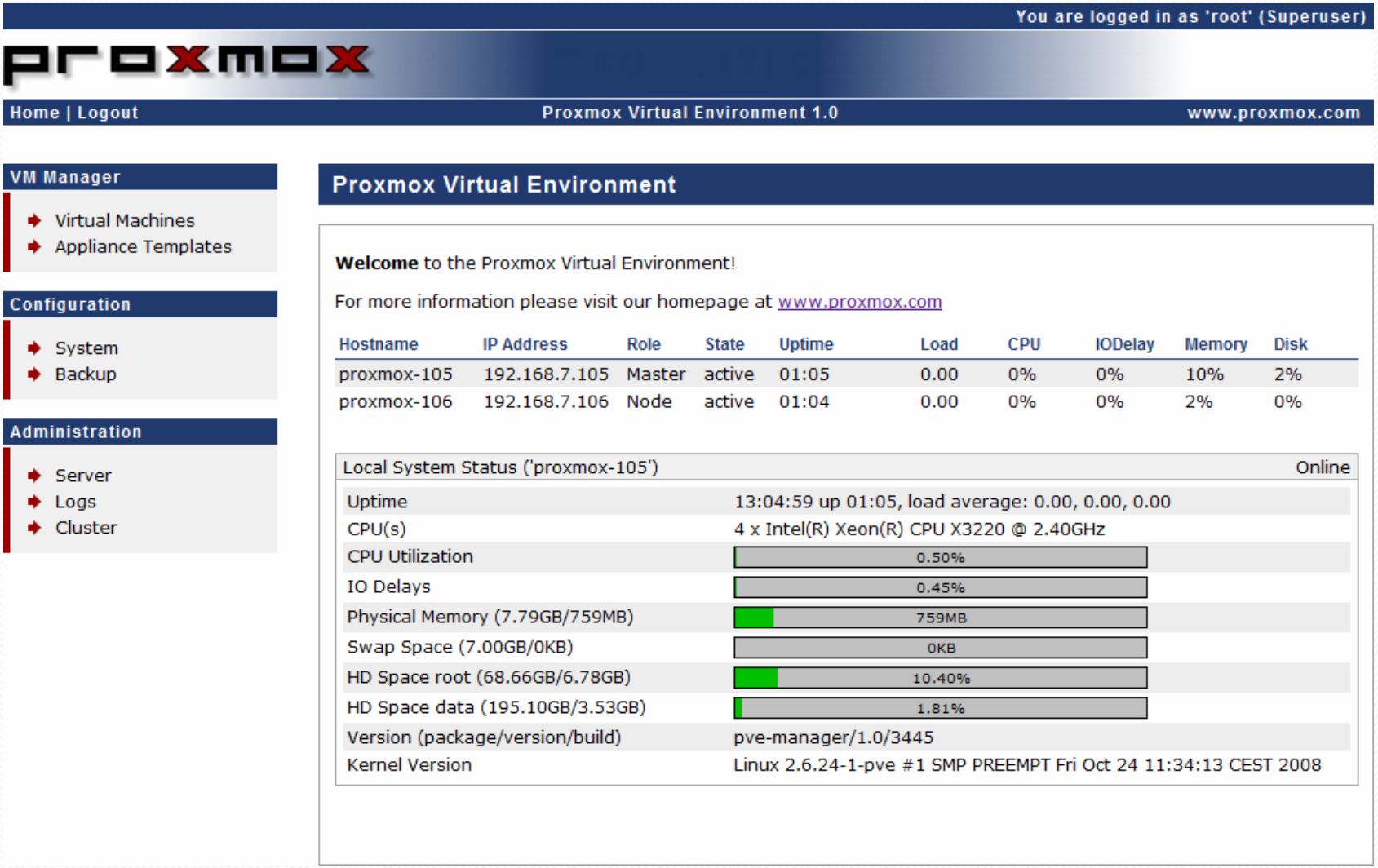


Fig. 6.23 Cluster nodes in Proxmox (source: Proxmox VE WiKi [19])

List of Virtual Machines

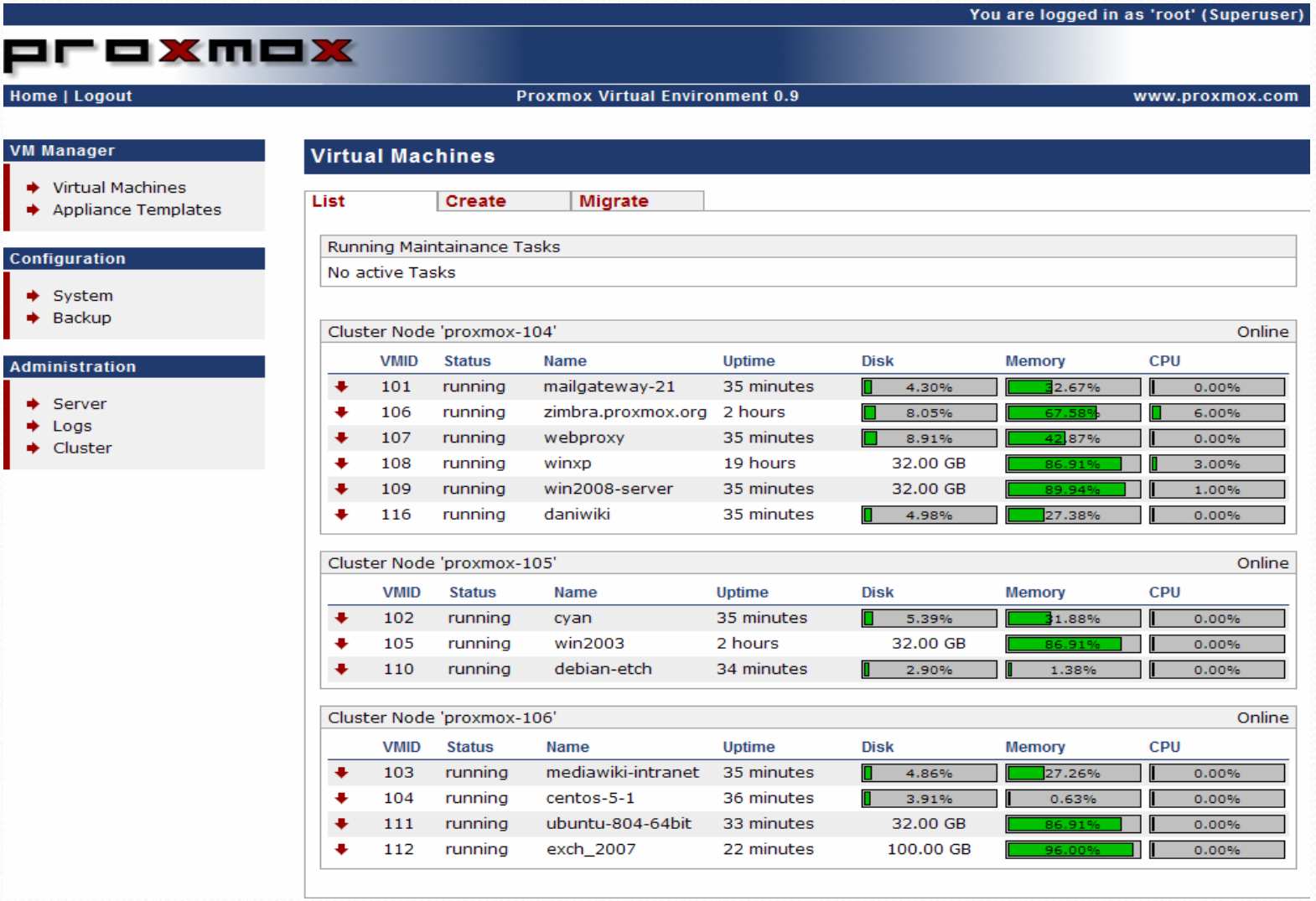


Fig. 6.24 Virtual machine list in Proxmox (source: Proxmox VE WiKi [19])

Create Virtual Machine

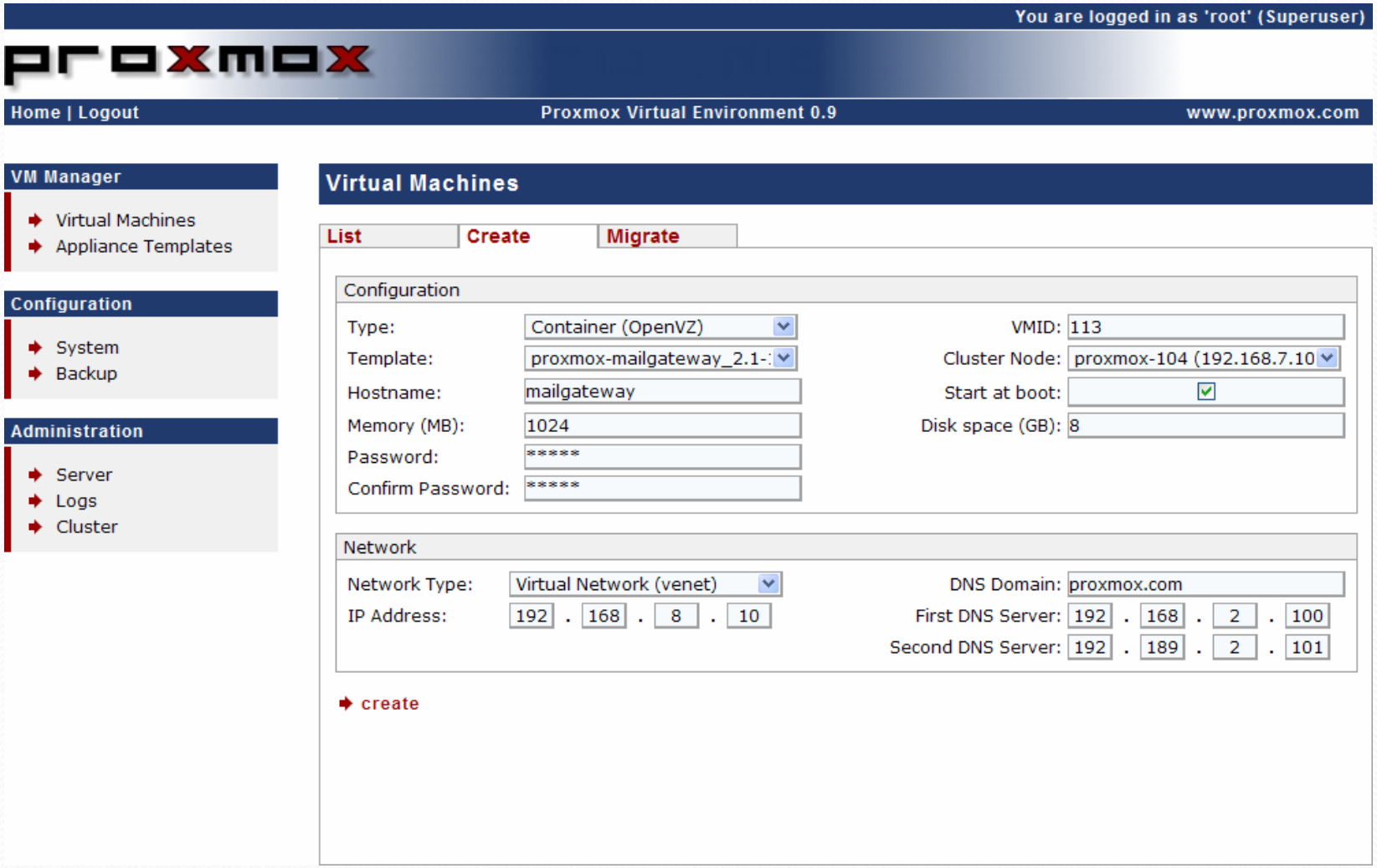


Fig. 6.25 Creating virtual machine in Proxmox (source: Proxmox VE WiKi [19])

Virtual Machine Status

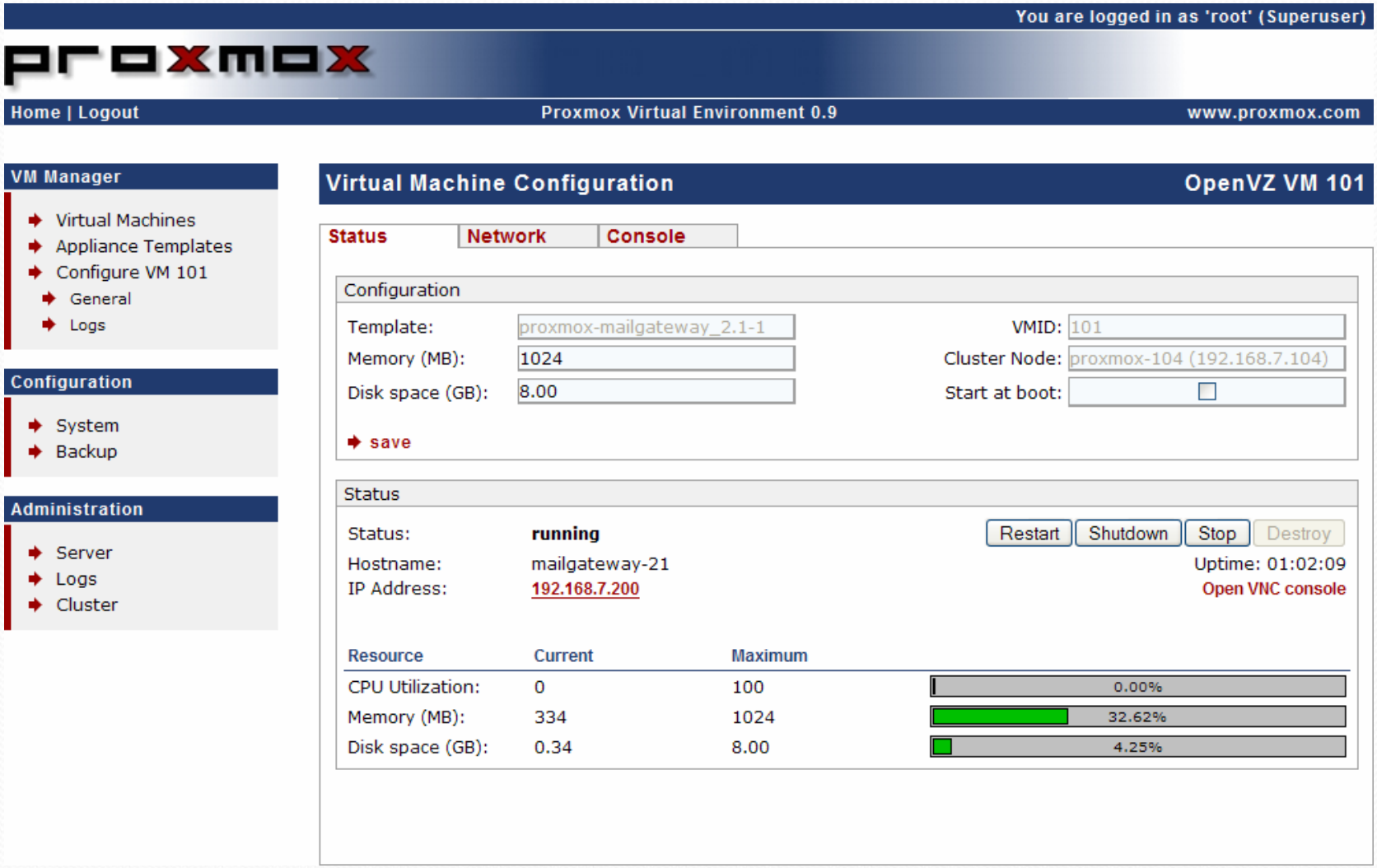


Fig. 6.26 Virtual status in Proxmox (source: Proxmox VE WiKi [19])

Proxmox VE Cluster

- Proxmox VE 叢集允許集中管理多個實體伺服器。Proxmox VE 叢集是由一個Master和若干個Nodes所組成（最小的叢集是一個Master和一個Node所組成）。

- **Proxmox VE Cluster**主要特點:
  - 集中式Web管理 (Centralized web management)
  - 一份帳密就可訪問、管理所有Node (One login and password for accessing all nodes and guests)
  - 主控台查看所有虛擬機器 (Console view to all Virtual Machines)
  - 在實體機器間搬遷虛擬機器 (Migration of Virtual Machines between physical hosts)
  - 同步虛擬設備樣板儲存 (Synchronized Virtual Appliance template store)

7. 虛擬化的優勢及顧慮

- 優勢
  (1) 充分利用現有資源的程度
  (2) 透過縮減實體基礎架構和提升伺服器/管理員比率以降低運轉成本
  (3) 提高硬體和應用程式的可用性，進而提高事務連貫性
  (4) 呈現了運營方面靈活性
  (5) 提升桌面的可管理性和安全性

- 虛擬化技術可實現在雲端計算的架構上，再配合可信計算導入的安全機制，相信未來會有更爲強大、更加普及的虛擬化應用層面會因應而生，從而可能會激發出爲人類服務的資訊技術產生重大的變革。

虛擬化的優勢及顧慮(續)

- 顧慮

  雖然目前虛擬化技術已經邁入廣大的應用範圍，但現在的虛擬化技術仍然還有一些瓶頸，在逐漸成熟階段它還有一些問題需要解決。大家所公認的以下兩點，是虛擬化技術未來繼續發展過程中必需要解決的問題：

  (1) 硬體使用效率: 如何在多虛擬機模式下，充分發揮硬體的極緻能力。

  (2) 安全及可靠性: 安全性是最重要的，然而可靠性也是同等重要。

8. 虛擬化的安全問題

- 因爲虛擬化技術在成本優化，支持跨平台等應用具有相當優勢，但也衍生出虛擬環境的安全問題，然而可信計算可提但供它一種安全保障的基礎設施。

- 研究可信計算機制可爲虛擬化技術提供安全保障服務。透過可信計算提供的可信量測、可信存儲和可信報告機制，將可淨化虛擬機的計算環境，搭建虛擬機之間的可信連接，構建誠實、互相信任的虛擬空間。

虛擬化的安全問題(續)

- 另一方面值得一提的是虛擬化技術對可信計算技術的支持，實際上透過虛擬機監視器所提供的隔離和監控機制，舒解了軟體可信動態量測的理論危機，如此可爲作業系統和軟體應用層建立可信計算環境，提供一套解決方案。因此之故兩套技術可以互相配合、互相支持。

9. 虛擬化的未來展望

- 虛擬化技術現在主要的一個焦點擺在伺服器和作業系統的虛擬化，例如像Linux虛擬化技術爲性能、可移植性和複雜性提供了許多選項 這也意味者你能夠爲你的應用程式和服務項目選擇最合適的虛擬化方法。

- 但隨著近年來Linux桌面技術的快速進展，虛擬化另一個新且重要的領域---個人電腦桌面虛擬化已開始顯現。虛擬化技術在解決桌面系統的安全性、可靠性、體驗性、使用效率和業務整合能力等方面，都會有很大的提升。

虛擬化的未來展望(續)

- 推廣Linux桌面虛擬化，使用虛擬機加Linux桌面的組合產品的優點是既可以兼容windows系統，同時還改善安全性差等缺點，提高用戶體驗，熟悉Linux桌面系統。
- 目前虛擬化技術的高效性主要展現在CPU運算和Memory存取上，在I/O操作和圖形性能方面還需要很大的改進，而且圖形性能是桌面用戶尤其關注的地方。目前KVM主要元件已放入Linux的Kernel 2.6.20版中， RedHat將特別重視桌面技術並尋求解決KVM圖形性能不足的問題。

10. 參考文獻

1. Gerald J. Popek and Robert P. Goldberg, "Formal requirements for virtualizable third generation architectures, " Communications of the ACM, Vol. 17, No. 7, July 1974.
2. VMware Virtualization, 2011. http://www.vmware.com/virtualization/
3. KVM, RedHat-Qumranet, 2011. http://www.redhat.com/f/pdf/rhev/DOC-KVM.pdf
4. Howto: Linux and Windows virtualization with KVM and Qemu, TuxRadar, 2011. http://www.tuxradar.com/content/howto-linux-and-windows-virtualization-kvm-and-qemu
5. Virtualisation with Linux, 2011. http://www.hagmann.sg/2010/03/07/virtualisation-with-linux

參考文獻 (續)

6. Linux A.B.I., 2011. http://linux-abi.sourceforge.net/
7. Introduction to Xen, NCHC Cloud Computing Research Group, 2010. http://www.nchc.org.tw/tw/rd/fsl/
8. Xen Virtualization Architecture, Novell Doc., 2011. http://www.novell.com/documentation/sles11/book_xen/?page=/documentation/sles11/book_xen/data/sec_xen_basics_arch.html
9. A brief architecture overview of VMware ESX, XEN and MS Viridian, IT 2.0, 2011. http://it20.info/2007/06/a-brief-architecture-overview-of-vmware-esx-xen-and-ms-viridian/

參考文獻 (續)

10. Glossary, Red_Hat Xen, Red_Hat Enterprise Linux 5, 2011. http://docs.redhat.com/docs/en-US/Red_Hat_Enterprise_Linux/5/html/Virtualization/glos.html
11. Cirtix XenServer, War of the Virtual Worlds, PQR, 2009. http://www.virtuall.nl/download-document/war-of-the-virtual-worlds-v07.pdf
12. The Benefits of Virtualizing Citrix XenApp with Citrix XenServer, 2008. http://www.kelarpacific.com/resources/Documents/The%20Benefits%20of%20Virtualizing%20Citrix%20XenApp%20with%20Citrix%20XenServer.pdf

參考文獻 (續)

13. Hyper-V伺服器虛擬化簡介, 2010. http://www.cc.ntu.edu.tw/chinese/epaper/0014/20100920_1408.htmhttp://www.cc.ntu.edu.tw/chinese/epaper/0014/20100920_1408.htm
14. 胡士亮, 微軟虛擬化技術簡介,技術架構顧問,微軟技術中心微軟技術中心,Microsoft Taiwan, 2011.
15. 蘇書平, vSphere5 新一代雲端平台系統, VMware Taiwan, 2011.
16. OpenStack, 2012. http://openstack.org/
17. OpenStack, WiKi, 2012. http://en.wikipedia.org/wiki/OpenStack
18. OpenNebula, 2012. http://opennebula.org/

參考文獻 (續)

19. Proxmox VE WiKi, 2012. http://pve.proxmox.com/wiki/Main_Page
20. Proxmox Server Solution GmbH, 2012. http://www.proxmox.com/
21. Asynchronous JavaScript Technology and XML (Ajax) With the Java Platform, ORACLE, 2012. http://www.oracle.com/technetwork/articles/javaee/ajax-135201.html

簡報完畢，謝謝!