Virtualizzazione

1. Perchè utilizzarla?
2. Che cos'è?
3. Quali sono le principali metodologie?

Lo scopo della Virtualizzazione

Lo scopo della virtualizzazione è quello di eseguire contemporaneamente più istanze di sistemi operativi “guest” in un’unica macchina fisica “host”.

I sistemi operativi “**guest**” colloquiano con le risorse messe a disposizione dalla macchina fisica “**host**” attraverso un componente software di livello intermedio generalmente denominata “**hypervisor**” o “virtual machine monitor” VMM.

Lo scopo della Virtualizzazione

Uno sviluppatore di software potrà quindi eseguire la sua applicazione in **diversi ambienti senza dover disporre di più macchine fisiche** o un amministratore di sistemi potrà testare uno scenario complesso che veda interagire più servizi su host diversi, ricreandolo su più **macchine virtuali** (VM) ospitate in una singola macchina fisica.

Ma sono gli utenti finali a trarre i maggiori benefici. Precisamente:

I benefici della Virtualizzazione

- **Aumento dell’affidabilità del sistema**

Sarà infatti possibile **dedicare** una macchina virtuale all’esecuzione di pochi servizi che notoriamente non vanno in conflitto tra di loro. Inoltre l’hypervisor è in grado di **isolare** le macchine guest in esecuzione sullo stesso host affinché eventuali problemi che compromettono il funzionamento di una singola macchina virtuale, non influenzino la stabilità delle altre.

I benefici della Virtualizzazione

- **Consolidamento dei server**

Molte aziende hanno visto crescere vistosamente il numero dei server proprio a causa dell'aumento dei servizi da fornire ai propri utenti. Attraverso la virtualizzazione si possono eseguire più macchine virtuali nella stessa macchina fisica **riducendo il numero dei server di 10 volte o più**.

Infatti è noto che la maggior parte dei server x86 ha un basso utilizzo di CPU e con le attuali tecnologie multiprocessore multicore non è raro spingersi a rapporti di consolidamento superiori.

I benefici della Virtualizzazione

- **Riduzione del Total Cost of Ownership (TCO)**

Il consolidamento ad un numero inferiore di server permette una notevole **riduzione dei costi legati all'energia** utilizzata per alimentare i server e per mantenere la temperatura ambientale adatta alle sale server. Inoltre **si riducono i costi di acquisto e** i canoni **di manutenzione** dei server fisici**.**

I benefici della Virtualizzazione

- **Disaster Recovery**

L’intero sistema operativo “guest” può essere facilmente salvato e ripristinato riducendo notevolmente i tempi di indisponibilità in caso di guasto.

- **Alta disponibilità**

Se è presente una infrastruttura di server fisici con delle caratteristiche hardware tra loro compatibili e questi server condividono una area dati (storage) sulla quale risiedono le macchine virtuali, sarà possibile spostare l’esecuzione di una macchina virtuale su un altro host in caso di failure. Alcuni sistemi prevedono lo spostamento automatico delle macchine virtuali tra i vari host in funzione al carico

I benefici della Virtualizzazione

- **Esecuzione di applicazioni legacy**

È frequente che alcune organizzazioni utilizzino applicazioni sviluppate per sistemi operativi che girano su **hardware ormai obsoleto**, non supportato o addirittura introvabile. Attraverso la virtualizzazione si possono continuare ad utilizzare quelle applicazioni che diversamente dovrebbero essere migrate ad una architettura più attuale affrontando i costi relativi al porting e al debug.

I benefici della Virtualizzazione

- **Sviluppo e Testing**

Possibilità di predisporre ambienti di sviluppo e di testing di varie tipologie in maniera agevole. Isolamento all'interno degli host rispetto all'ambiente di lavoro principale. Orientato sia ai server che ai client.

Virtualizzazione:
1. Perchè utilizzarla?

- **Riduce i costi**
- **Semplifica la gestione di una infrastruttura IT**
- **Consente una migliore affidabilità**
- **Consente una maggiore flessibilità**

Virtualizzazione:
2. Che cos'è?

“virtualization is a framework or **methodology of dividing the resources of a computer into multiple execution environments**, by applying one or more concepts or technologies such as **hardware and software partitioning, time-sharing, partial or complete machine simulation, emulation, quality of service, and many others.**”

Virtualizzazione:
2. Che cos'è?

It is "**a technique for hiding the physical characteristics of computing resources** from the way in which other systems, applications, or end users interact with those resources. This includes making a single physical resource (such as a server, an operating system, an application, or storage device) appear to function as multiple logical resources; or it can include making multiple physical resources (such as storage devices or servers) appear as a single logical resource."

Virtualizzazione: che cos'è

In informatica, la virtualizzazione è un termine generico che si riferisce all'astrazione di risorse di calcolo (computing resources)

- **Platform virtualization** ovvero virtualizzazione di piattaforme hardware (concetto di VM)

- **Resource virtualization** ovvero virtualizzazione di risorse (concetto di qualità del servizio)

The basic concept of platform virtualization, was later extended to the virtualization of specific system resources, such as storage volumes, name spaces, and network resources. **Resource aggregation**, spanning, or concatenation combines individual components into larger resources or resource pools.

Platform Virtualization

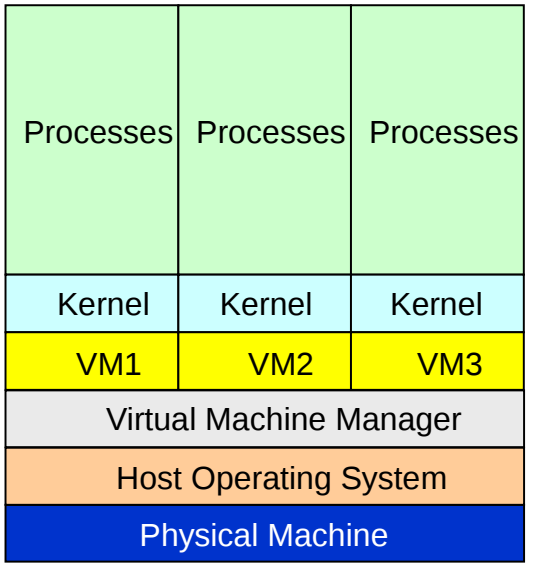
- Emulation or simulation
- Native virtualization or full virtualization
- Hardware enabled virtualization
- Partial virtualization
- Paravirtualization
- Operating system-level virtualization
- Application Virtualization

Resources Virtualization

- RAID
- SAN
- Channel bondings
- VPN/NAT
- Multiprocessor and multi-core
- Cluster and Grid computing
- Partitioning

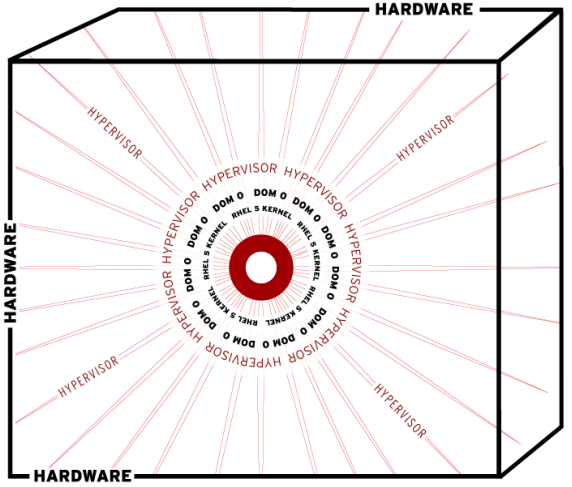
Hypersor (VMM) e Virtual Machine (VM)

- Una **VMM** (Virtual Machine Manager o **Hypevisor**) astrae l'hardware di un singolo calcolatore.
  - **Crea e controlla molti diversi ambienti di esecuzione** (VM).
  - **Ciascuno di questi ambienti può avere un proprio sistema operativo.**
  - **Ciascuno di questi ambienti crede di controllare l'intero sistema hardware.**



Quindi una Virtual Machine (VM) è un'ambiente di esecuzione creato da un'hypervisor (VMM)

Hypervisor



- Provides **protection**, **networking**, **driver coordination**, **and resource management** so that each virtual OS sees itself as running on a bare metal server.
- **Allows you to create, control, monitor, destroy, pause, or migrate** new virtual machines.
- Virtual Machine Monitor, **Scheduling** Virtual CPU, Memory

Hypervisor

Hypervisors (VMM) are currently classified in two types:

- **Type 1** hypervisor is software that **runs** directly **on a given hardware platform** (as an operating system control program). A "guest" operating system thus runs at the second level above the hardware.
  - Recent examples are Xen, VMware's ESX Server, and Sun's Hypervisor.
- **Type 2** hypervisor is software that **runs within an operating system environment**. A "guest" operating system thus runs at the third level above the hardware.
  - Examples include VMware server and Microsoft Virtual Server.

Principali hypervisors

- **Oracle Virtualbox**:
  - http://en.wikipedia.org/wiki/VirtualBox
- **KVM**
  - http://en.wikipedia.org/wiki/Kernel-based_Virtual_Machine
- **QEMU**
  - http://en.wikipedia.org/wiki/QEMU
- **Parallel**
  - http://en.wikipedia.org/wiki/Parallels,_Inc.

Principali hypervisors

- **Xen**:
  - University of Cambridge Computer Laboratory
  - Fully open sourced
  - Set of patches against the linux kernel
- **Vmware ESX** :
  - Closed source
  - Based on Linux 2.4
  - Proprietary drivers
- **Xbox 360**:
  - Closed source, used to assume full backward compatibility with the old Xbox

Tipi di Hypervisor (VMM)

- Type 0

Hardware-based solution

Commonly found in mainframe computers.

Examples: IBM LPARs, Oracle LDOMs

Tipi di Hypervisor (VMM)

- Type 1

Operating-system-like software provides virtualization

Examples: VMware ESX, Joyent SmartOS, Citrix XenServer

General-purpose OSes that provide VMM functions

Examples: Microsoft Windows Server with HyperV, **Red Hat Linux with KVM feature**

Tipi di Hypervisor (VMM)

- Type 2

Applications that run on standard OSes to provide VMM functionality to guest operating systems.

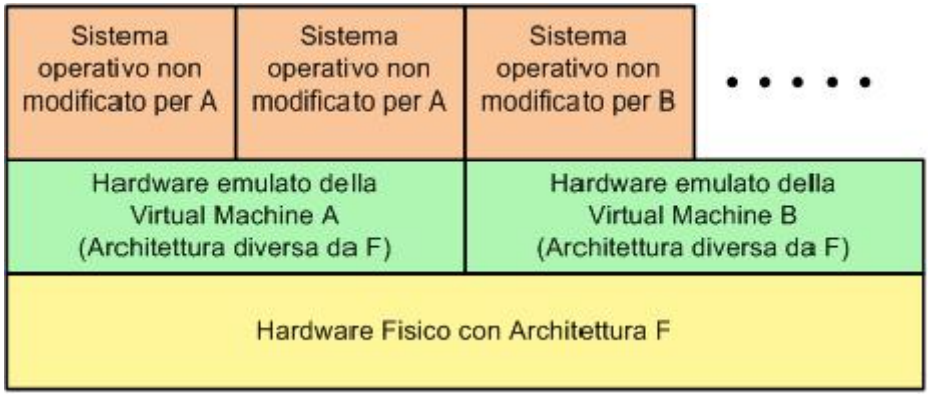
Example: VMware Workstation and Fusion,
Parallels Desktop, Oracle Virtual Box.

Virtualizzazione:
3. Quali sono le principali metodologie?

- Il problema della virtualizzazione è stato affrontato in diversi modi. Tutte le **quattro metodologie di virtualizzazione** attualmente impiegate danno l'illusione di utilizzare un sistema operativo stand alone, non virtualizzato. Esse sono:
  - l'**emulation**,
  - la **full virtualization**,
  - la **paravirtualization**
  - la **operating system level virtualization**.

Emulation

Con l’emulazione l’hypervisor simula l’intero hardware set che permette al sistema operativo guest di essere eseguito senza alcuna modifica. Il software di virtualizzazione **si incarica di presentare al sistema operativo guest un’architettura hardware completa a lui nota, indipendentemente dall’architettura hardware presente sulla macchina host.**



L'emulatore simula una'architettura hardware diversa da quella fisica

I limiti della Emulation

Gli emulatori presentano al sistema operativo guest un'**architettura hardware standard** precludendo quelle che potrebbero essere le funzionalità alle quali siamo abituati, ad esempio quelle implementate in hardware.

Inoltre deve **interfacciare la CPU** (con istruzioni semanticamente equivalenti), **la memoria** (accesso esclusivo e riserva) **e l'I/O** tra sistema host e sistema guest.

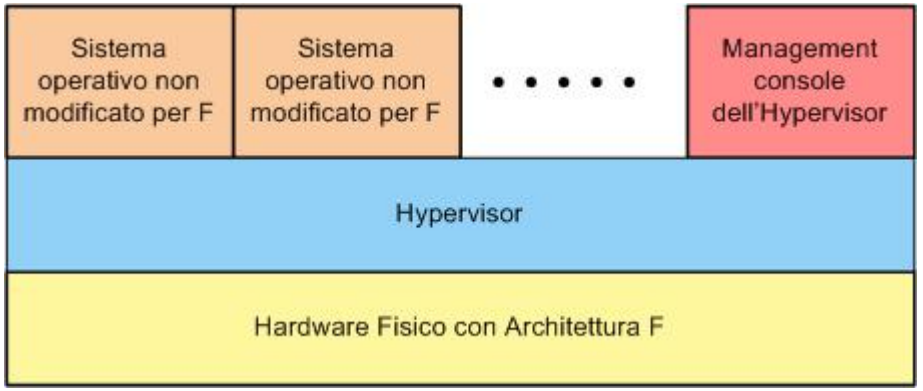
Questo carico di lavoro dell'emulatore rende **difficoltoso** l'uso di questa tecnica di virtualizzazione **quando bisogna emulare sistemi guest che richiedono processori di velocità equivalente al processore dell'host.**

Full Virtualization

La Full (o Native) Virtualization è simile alla Emulation ma **i sistemi operativi guest devono essere compatibili con l'architettura hardware della macchina fisica**. In questo modo molte **istruzioni** possono essere **eseguite direttamente sull'hardware senza bisogno di un software di traduzione garantendo prestazioni superiori** rispetto all'emulazione.

Recentemente Intel e AMD hanno introdotto **VT-x** ed **AMD-v**

Esempi di software che utilizzano la full virtualization sono VMware, Virtual Box, e Xen, limitatamente ai sistemi operativi proprietari non modificabili.



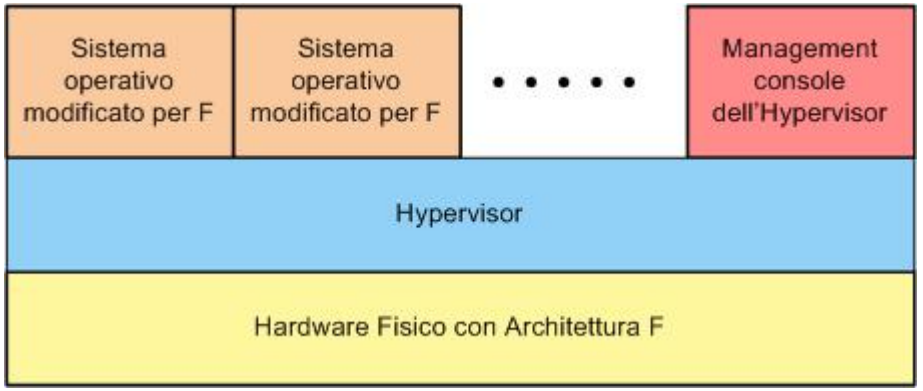
La Full Virtualization presenta al sistema operativo guest la stessa architettura hardware presente sull'host fisico

Paravirtualization

L’hypervisor presenta alle macchine virtuali un versione modificata dell’hardware sottostante, mantenendone tuttavia la medesima architettura (stessa ABI). Il sistema operativo in esecuzione sulle macchine virtuali è invece modificato per **evitare alcune particolari chiamate di sistema**.

Questa tecnica **permette di ottenere un decadimento delle prestazioni minimo rispetto al sistema operativo non virtualizzato, dato che le istruzioni provenienti dalle macchine virtuali vengono eseguite quasi tutte direttamente sul processore senza che intervengano modifiche**.

Esempi di paravirtualizzazione sono Xen e User-mode Linux.

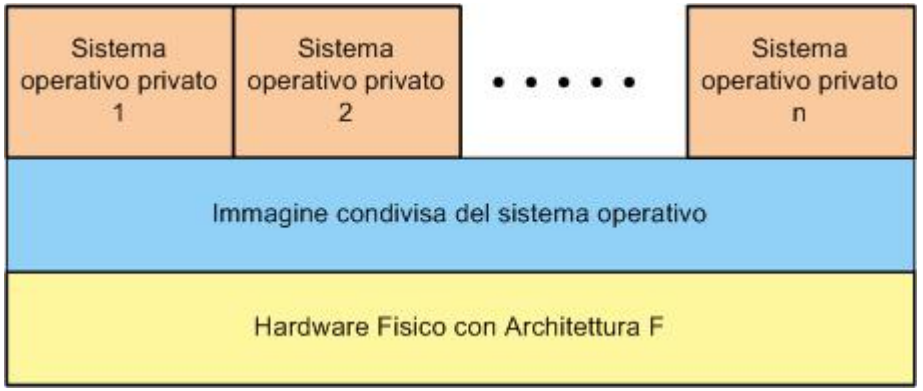


La Paravirtualization è simile alla full virtualization, ma è necessario modificare il sistema operativo guest per ottimizzare l'esecuzione sull'ambiente virtualizzato

Operating System Level Virtualization

Non si utilizza un Hypervisor, ma la virtualizzazione è creata utilizzando **copie del sistema operativo installato sull'host**. I sistemi guest creati saranno a tutti gli effetti istanze del sistema operativo host con un proprio file system, configurazione di rete e applicazioni. Il vantaggio principale di questa tecnica è il miglior utilizzo delle risorse grazie alla condivisione di spazi di memoria. Essendo i sistemi operativi delle macchine guest equivalenti a quello della macchina host, **le istanze guest non richiederanno un kernel privato**, ma utilizzeranno lo stesso con un conseguente minor utilizzo di memoria fisica. Non adatto a sistemi operativi diversi sullo stesso host. Poca stabilità, poco isolamento.

Esempi: Virtuozzo, Linux VServers, Solaris Containers, HPUX 11i Secure Resource Partitions



La Operating System level Virtualization mette a disposizione dei sistemi guest l'immagine del sistema operativo in esecuzione sull'host.

Altri tipi di virtualizzazione

Application Virtualization or Programming-environment virtualization

typically for the purpose allowing application binaries to be portably run on many different computer architectures and operating systems.

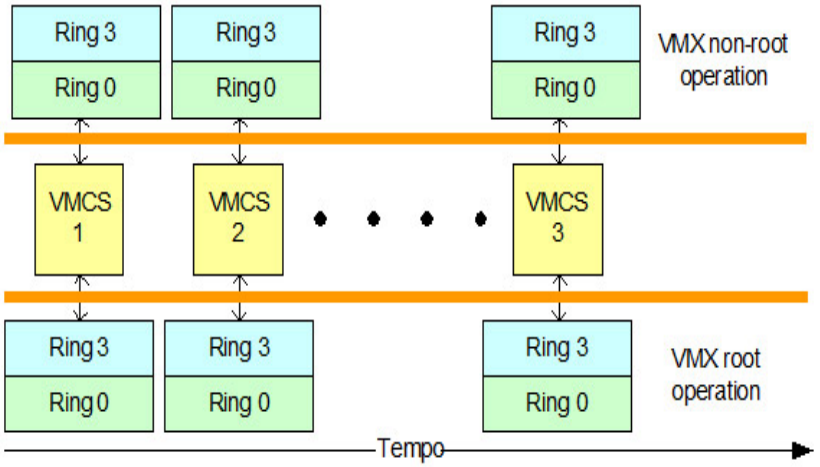
http://en.wikipedia.org/wiki/Comparison_of_Application_Virtual_Machines

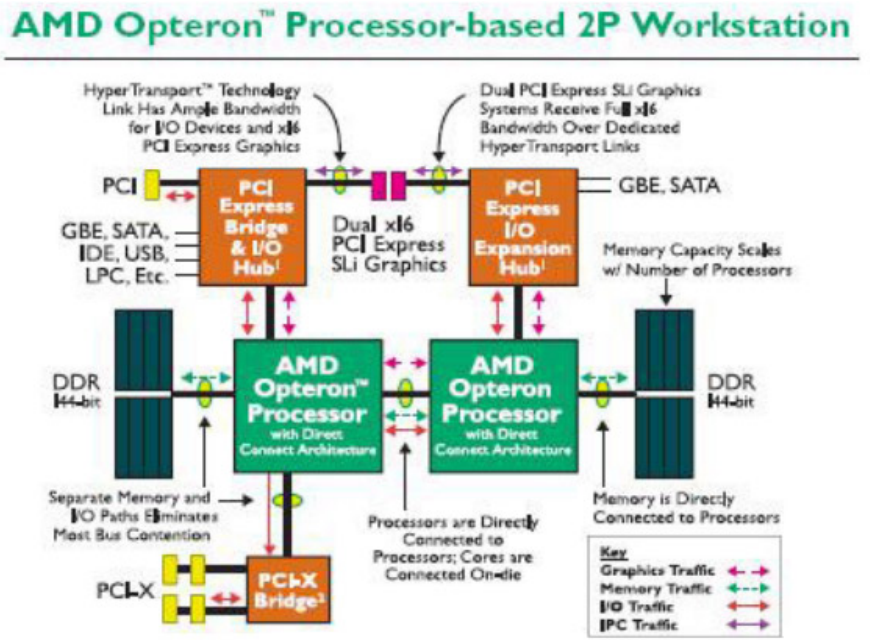
Examples:

- .NET CLR
- JVM
- Script Languages:Python,Ruby,Javascript...

Il supporto hardware alla virtualizzazione

- Le aziende produttrici di hardware stanno investendo molto per trovare soluzioni che riducano l'intervento dell'hypervisor incrementando le performance delle macchine virtuali. Intel e AMD hanno lanciato nel mercato delle CPU che offrono un supporto alla virtualizzazione (Dal 2005).





Il supporto hardware alla virtualizzazione: **Intel**

Intel ha reso disponibile il supporto alla virtualizzazione nelle CPU Xeon e Itanium con le tecnologie Intel **VT-x** e Intel VT-i. Queste tecnologie hanno **lo scopo di limitare, o addirittura eliminare, l'intervento del VMM** liberando cicli di CPU **e di gestire il cambiamento di contesto** tra i sistemi operativi guest, il VMM ed eventualmente la service console.

VMX Root elimina la necessità che il VMM mascheri **il livello di privilegio** in alcune chiamate di sistema (VM entry e VM exit).

VM entry e VM exit sono gestite da una struttura dati chiamata Virtual machine Control Structure (VMCS). La VMCS contiene due aree, la guest-state area e la host-state area.

Il supporto hardware alla virtualizzazione: **AMD**

AMD-V consiste in un insieme di **estensioni hardware all'architettura x86 per ridurre, e in alcuni casi eliminare, l'intervento del VMM**.

La **Direct Connect Architecture** ha lo scopo di aumentare le prestazioni eliminando il collo di bottiglia rappresentato dall'architettura con front side bus (FSB di Intel). Controller della memoria integrato in ciascun processore che connette i core ad un area di memoria dedicata. Un bus con una tecnologia denominata **HyperTransport Link** in uscita da ciascun controller I/O come PCI o PCI-X, un bus con la stessa tecnologia che collega i processori. Questi bus dedicati punto a punto evitano la gestione dell'arbitraggio sull'utilizzo e riducono i tempi di latenza minimi per l'accesso alla memoria.

Sia VMware, sia Xen incrementano le prestazioni nei sistemi multiprocessore che fanno uso dell'architettura Non-Uniform Memory Access (NUMA).

AMD-V Extended Migration consente la migrazione a caldo delle macchine virtuali tra tutti i processori AMD Opteron.

Hypervisor (VMM) di tipo 2

Example: VirtualBox

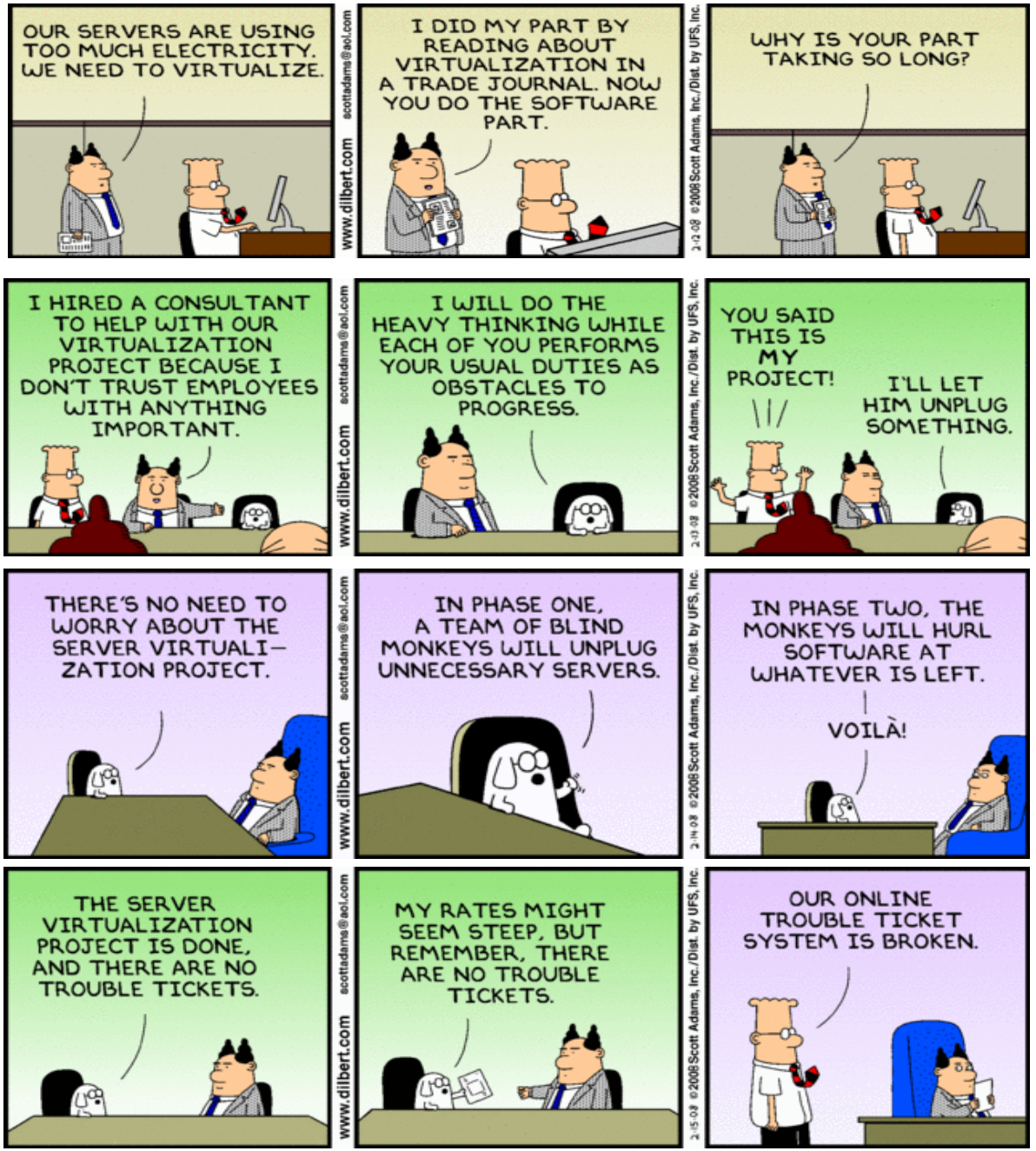
The VMM is just another process running on the host.
The host OS doesn't even know that virtualization is taking place.The VMM may run as a user application.

It might not have the administrative privileges to access the hardware assistance features of modern CPUs. Therefore, type 2 hypervisors can have poorer performance than type 0 or type 1.

No changes are required to the host operating system.

Any student can run VirtualBox to experiment and learn from different guest operating systems.

Intervallo


OUR SERVERS ARE USING TOO MUCH ELECTRICITY. WE NEED TO VIRTUALIZE.
I DID MY PART BY READING ABOUT VIRTUALIZATION IN A TRADE JOURNAL. NOW YOU DO THE SOFTWARE PART.
WHY IS YOUR PART TAKING SO LONG?
I HIRED A CONSULTANT TO HELP WITH OUR VIRTUALIZATION PROJECT BECAUSE I DON'T TRUST EMPLOYEES WITH ANYTHING IMPORTANT.
I WILL DO THE HEAVY THINKING WHILE EACH OF YOU PERFORMS YOUR USUAL DUTIES AS OBSTACLES TO PROGRESS.
YOU SAID THIS IS MY PROJECT!
I'LL LET HIM UNPLUG SOMETHING.
THERE'S NO NEED TO WORRY ABOUT THE SERVER VIRTUALI-ZATION PROJECT.
IN PHASE ONE, A TEAM OF BLIND MONKEYS WILL UNPLUG UNNECESSARY SERVERS.
IN PHASE TWO, THE MONKEYS WILL HURL SOFTWARE AT WHATEVER IS LEFT.
VOILÀ!
THE SERVER VIRTUALIZATION PROJECT IS DONE, AND THERE ARE NO TROUBLE TICKETS.
MY RATES MIGHT SEEM STEEP, BUT REMEMBER, THERE ARE NO TROUBLE TICKETS.
OUR ONLINE TROUBLE TICKET SYSTEM IS BROKEN.
www.dilbert.com
scottadams@aol.com
© 2008 Scott Adams, Inc./Dist. by UFS, Inc.

Componenti della Virtualizzazione

CPU scheduling

- There may be more virtual CPUs than physical CPUs.
  - **Overcommitment**

- The VMM must share the available physical CPUs among the guests.

Componenti della Virtualizzazione

Memory management

- The VMM may overcommit memory among the guests.
- The VMM determines how much real memory each guest can use.
  - Each guest has the illusion that it has all the memory it wants.
- The VMM does its own memory page allocation.
- It works with the host system’s memory management.

Componenti della Virtualizzazione

I/O management

- The VMM can dedicate physical I/O devices to guests.
  - Example: Assign a physical CD ROM drive to a guest.
- The VMM can provide idealized device drivers to guests.
  - The VMM maps guest I/O requests to a device to the actual device driver.

Componenti della Virtualizzazione

Storage management

- The VMM ensures that each guest can only access the disk blocks allocated to it.

Networking

- The VMM provides each guest with at least one IP address.
- The VMM provides routing between the guest and the network.
- The VMM provides network address translation (NAT).

Sistemi distribuiti

- A distributed system is
  - A collection of loosely coupled processors
  - Interconnected by a communications network

- Site: The physical location of a machine.
- Node: The machine or system located at a site.
  - AKA: host
- Server: A node that provides resources to other nodes.
- Client: A node that consumes resources from a server.

Sistemi distribuiti avanzati

- Resource sharing
  - Share and print files at remote sites.
  - Process information in a distributed database.
  - Use remote specialized hardware devices.
- Load sharing
  - Computation speedup.
  - Load balancing.

I vantaggi dei sistemi distribuiti

- Reliability
  - Detect and recover from a site failure.
  - Transfer functions from a failed site.
  - Reintegrate a failed site.
- Communication
  - Message passing.
- Flexibility
  - Replace mainframes with networks of workstations.
  - More cost effective.

Comparison of platform virtualization hypervisors

Name	Guest OS SMP available	Runs arbitrary OS	Supported guest OS drivers	Method of operation	Typical use	Speed relative to host OS	Commercial support available
KVM	Yes[10]	Yes	Yes	AMD-V and Intel-VT-x	Virtualized server isolation, server/desktop consolidation, software development, cloud computing, other purposes	Up to near native [*citation needed*] [5]	Yes[11]
Linux-VServer	Yes	No	Compatible	Operating system-level virtualization	Virtualized server isolation and security, server consolidation, cloud computing	Up to near native [*citation needed*] [6]	Yes
Oracle VM Server for x86	Yes	Yes	Yes	Paravirtualization and hardware virtualization	Server consolidation and security, enterprise and business deployment	Up to near native [*citation needed*]	Yes
Oracle VirtualBox	Yes	Yes	Yes	Virtualization	Business workstation, server consolidation, service continuity, developer, hobbyist	Up to near native [*citation needed*]	Yes (with commercial license)
Virtuozzo	Yes	No	Compatible	Operating system-level virtualization	Server consolidation, service continuity, disaster recovery, service providers	Up to near native [*citation needed*]	Yes
VMware Server	Yes (2-way)	Yes	Yes	Virtualization	Server/desktop consolidation, dev/test	Up to near native [*citation needed*]	Yes
VMware Workstation 6.0	Yes (2-way)	Yes	Yes	Paravirtualization (VMI) and virtualization	Technical professional, advanced dev/test, trainer	Up to near native [*citation needed*]	Yes
VMware Player 6.0	Yes[14]	Yes	Yes	Virtualization	Technical professional, advanced dev/test, trainer, end user on prebuilt machines	Up to near native [*citation needed*]	Yes
Xen	Yes, v4.0.0: up to 128 VCPUs per VM	No, bare hypervisor	Yes	Paravirtualization and porting or hardware virtualization.	Virtualized server isolation, server/desktop consolidation, software development, cloud computing, other purposes. Xen powers most public cloud services and many hosting services, such as Amazon Web Services, Rackspace Hosting and Linode.	Up to native[15]	Yes

Gli hypervisor più utilizzati

- For production environments the most tested hypervisors are KVM and Xen-based hypervisors. **KVM** runs through libvirt, Xen runs best through XenAPI calls. KVM is selected by default in Openstack.
- For deskyop environment the most tested hypervisors are VMWare and **Virtualbox**.

Attività di laboratorio del primo giorno

2 strumenti per la full virtualization:

Virtualbox + KVM-QEMU (virt-manager)

- Utilizzo di Virtualbox
- Utilizzo di KVM-QEMU (virt-manager)

Virtualizzazione e Linux

Project	**Type**	**License**
Bochs	Emulation	LGPL
QEMU	Emulation	LGPL/GPL
VMware	Full virtualization	Proprietary
Oracle Virtualbox	Full virtualization	GPL + Proprietary
Xen	Paravirtualization	GPL
UML	Paravirtualization	GPL
Linux-VServer	Operating system-level virtualization	GPL
OpenVZ	Operating system-level virtualization	GPL

Qemu (emulation)

QEMU supports two modes of operation. The first is the **Full System Emulation mode** which emulates a full personal computer (PC) system with processor and peripherals. This mode emulates a number of processor architectures, such as x86, x86_64, ARM, SPARC, PowerPC, and MIPS, with reasonable speed using dynamic translation. Using this mode, you can emulate the Windows operating systems and Linux on Linux, Solaris, and FreeBSD. Many other operating system combinations are also supported.

QEMU also supports a second mode called **User Mode Emulation**. In this mode, which can only be hosted on Linux, a binary for a different architecture can be launched. This allows, for example, a binary compiled for the MIPS architecture to be executed on Linux running on x86. Other architectures supported in this mode include ARM, SPARC, and PowerPC, though more are under development.

Vmware (full virtualization)

VMware is a **commercial solution for full virtualization**. A hypervisor sits between the guest operating systems and the bare hardware as an abstraction layer. This abstraction layer allows any operating system to run on the hardware without knowledge of any other guest operating system.

VMware also virtualizes the available I/O hardware and places drivers for high-performance devices into the hypervisor.

The entire virtualized environment is kept as a file, meaning that a full system (including guest operating system, VM, and virtual hardware) can be easily and quickly migrated to a new host for load balancing.

Oracle Virtualbox (full virtualization)

VirtualBox is a "virtualization" product, we refer to "full virtualization", that is, the particular kind of virtualization that **allows an unmodified operating system with all of its installed software to run** in a special environment, on top of your existing operating system.

Guest code is not allowed to run directly on the host. Instead, every single machine instruction is translated ("emulated"). While emulators theoretically allow running code written for one type of hardware on completely different hardware (say, running 64-bit code on 32-bit hardware), they are typically quite slow. Virtualizers such as VirtualBox, on the other hand, can achieve near-native performance for the guest code, but can only run guest code that was written for the same target hardware (such as 32-bit Linux on a 32-bit Windows host).

VirtualBox is also different from so-called "paravirtualization" solutions such as Xen, which require that the guest operating system be modified.

Xen (paravirtualization)

Xen is a free open source solution for operating system-level paravirtualization from XenSource. Recall that in paravirtualization the hypervisor and the operating system collaborate on the virtualization, requiring operating system changes but resulting in near native performance.

As Xen requires collaboration (modifications to the guest operating system), only those operating systems that are patched can be virtualized over Xen. From the perspective of Linux, which is itself open source, this is a reasonable compromise because the result is better performance than full virtualization. But from the perspective of wide support (such as supporting other non-open source operating systems), it's a clear disadvantage.

It is possible to run Windows as a guest on Xen, but only on systems running the Intel Vanderpool or AMD Pacifica. Other operating systems that support Xen include Minix, Plan 9, NetBSD, FreeBSD, and OpenSolaris.

KVM (Kernel Virtual Module)

- Incorporation of the KVM into the Linux kernel (2.6.20).
- KVM (for Kernel-based Virtual Machine) is a **full virtualization solution** that is unique in that it turns a Linux kernel into a hypervisor using a kernel module.
- The loadable KVM module in the kernel exposes the virtualized hardware through the /dev/kvm character device including virtualization extensions (Intel VT or AMD-V).
- The guest operating system interfaces to the KVM module using a modified QEMU process for PC hardware emulation.

Linux KVM (Kernel Virtual Module)

The KVM module introduces a new execution mode into the kernel. Where vanilla kernels support kernel mode and user mode, the KVM introduces a **guest mode**. The guest mode is used to execute all non-I/O guest code, where normal user mode supports I/O for guests.

The introduction of the KVM is an interesting evolution of Linux, as it represents the first virtualization technology that is part of the mainline Linux kernel. When run on hardware that supports virtualization, Linux (32-and 64-bit) and Windows (32-bit) guests are supported.

Why use KVM?

More efficient use of existing hardware

- What was once a single Linux machine can now host multiple virtual machines

Low/No Cost solution virtualization

- KVM is OpenSource and Free
- Virt Manager is OpenSource and Free
- oVirt is OpenSource and Free
- Base requirements are included in most modern distributions

KVM and QEMU: Type 1 or Type 2 hypervisor

KVM needs QEMU to provide full hypervisor functionality. By itself, KVM is more of a virtualization infrastructure provider. QEMU by itself is a Type-2 hypervisor. It intercepts the instructions meant for Virtual CPU and uses the host operating system to get those instructions executed on the physical CPU. When **QEMU uses KVM for hardware acceleration**, the combination becomes a Type-1 hypervisor.

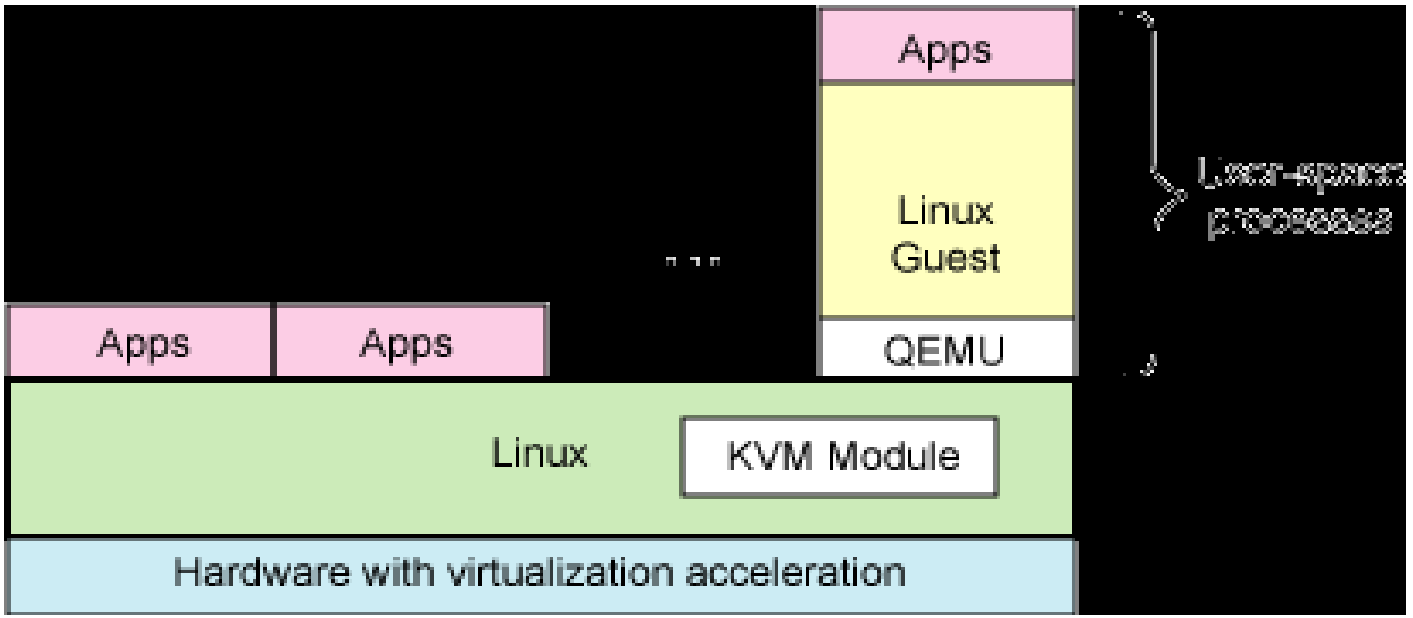
KVM and QEMU – the x86 dependency

Since KVM is really a driver for the physical CPU capabilities, it is very tightly associated with the CPU architecture (the x86 architecture). This means that **the benefits of hardware acceleration will be available only if the Virtual Machine CPU also uses the same architecture (x86)**.

If a VM needs to run Power PC CPU but the hypervisor server has an Intel CPU, then KVM will not work. You must use QEMU as the Virt Type and live with the performance overhead.

KVM-QEMU

Based on the discussion above, it is quite clear that QEMU plays a very critical role in Linux based Open Source virtualization solutions. For all practical applications, **QEMU needs KVM's performance boost. However, it is clear that KVM by itself cannot provide the complete virtualization solution. It needs QEMU.**



Due concetti importanti: Overcommiting e Thin Provisioning

Overcommitting *

KVM hypervisor supports overcommitting of system resources. Overcommitting means allocating more virtualized **CPUs or memory** than the available resources on the system. Memory overcommitting allows hosts to utilize memory and virtual memory to increase guest densities.

Thin provisioning

Thin provisioning allows the allocation of flexible storage and optimizes the available space for every guest. It gives the appearance that there is more physical storage on the guest than is actually available. This is not the same as overcommitting as this only pertains to **storage** and not CPUs or memory allocations. However, like overcommitting, the same warning applies.

* Overcommitting involves possible risks to system stability.

Strumenti: libvirt

- È una API C costruita sulle capacità di virtualizzazione di Linux che **supporta differenti hypervisors** (KVM, Xen, VMWare)

- Offre quindi una **interfaccia “Hypervisor agnostic”** per costruire strumenti di amministrazione e di monitoring (con compilazioni per molti linguaggi)

- Permette di connettersi a hypervisor remoti (libvirtd) e quindi controllarli

Il futuro della virtualizzazione: miglioramenti continui

- **Multiple Hypervisor Support**
  - (Xen, KVM, ...)
  - Libvirt & virsh provide technology-agnostic management
- **Even better deployment**
  - Cobbler – next gen. Installation server
- **More manageable**
  - oVirt