Parallel & Distributed Computing (WQD7008)

Week 7

Virtual Machines and Virtualization of Data Centers and Clusters

2019/2020 Semester 1

Dr. Hamid Tahaei

DATA-CENTER DESIGN AND INTERCONNECTION NETWORKS

Warehouse-Scale Data Center Design

- The cloud is built on massive datacenters.
- A data center can be as large as a shopping mall (11 times the size of a football field) under one roof.
- Such a data center can house 400,000 to 1 million servers.
- A small data center could have 1,000 servers. The larger the data center, the lower the operational cost.
- The approximate monthly cost to operate a huge 400-server data center is estimated by network cost $13/Mbps; storage cost $0.4/GB; and administration costs. These unit costs are greater than those of a 1,000-server data center.The network cost to operate a small data center is about seven times greater and the storage cost is 5.7 times greater.
- Microsoft has about 100 data centers, large or small, which are distributed around the globe.

DATA-CENTER DESIGN AND INTERCONNECTION NETWORKS

Data center Cooling System

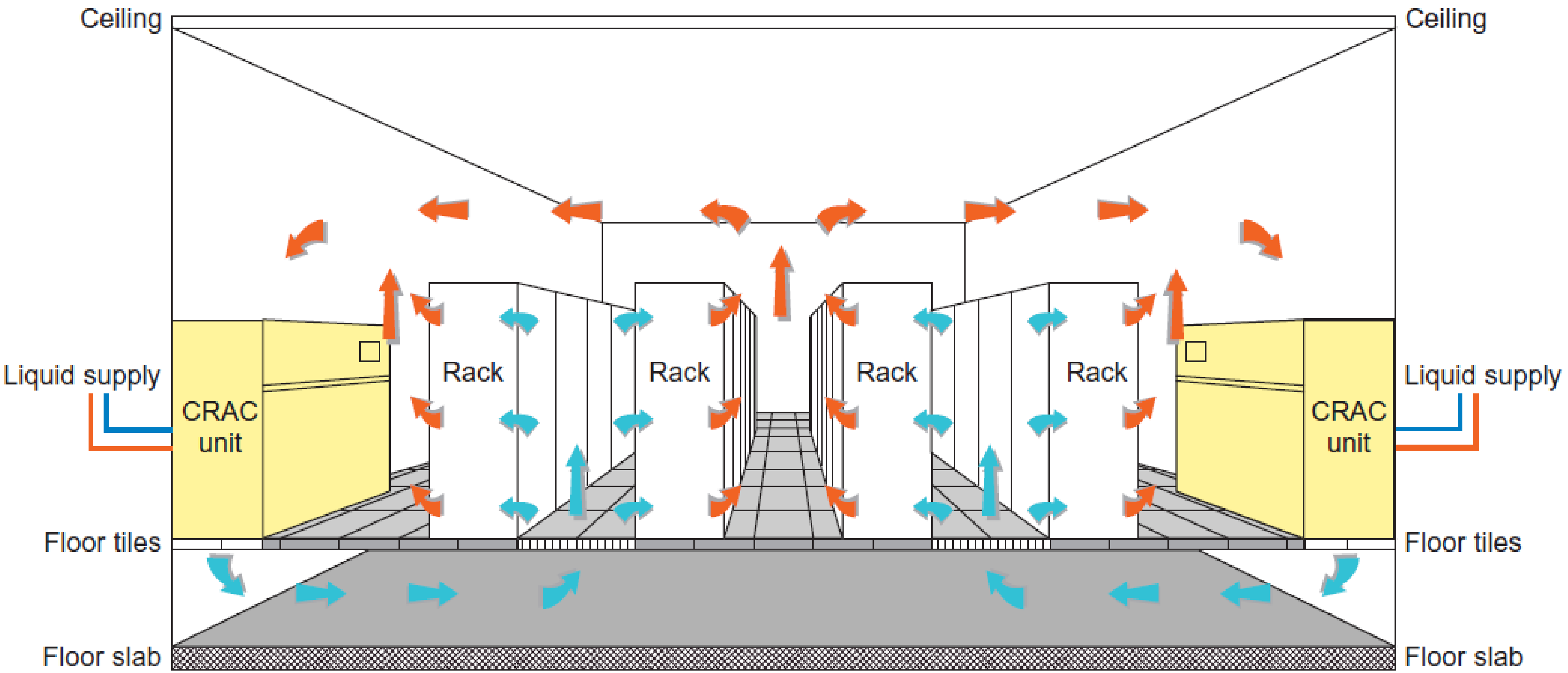


FIGURE 4.9

The cooling system in a raised-floor data center with hot-cold air circulation supporting water heat exchange facilities.

(Courtesy of DLB Associates, D. Dyer [22])

CRAC (computer room air conditioning)

DATA-CENTER DESIGN AND INTERCONNECTION NETWORKS

Data Center Interconnection Networks

- A critical core design of a data center is the interconnection network among all servers in the datacenter cluster. This network design must meet five special requirements:
  - low latency, high bandwidth, low cost, message-passing interface (MPI) communication support, and fault tolerance.
- Application Traffic Support
  - point-to-point
  - Collective MPI communications
- Network Expandability
  - The fat-tree and crossbar networks can be implemented with low-cost Ethernet switches.
- Fault Tolerance and Graceful Degradation
  - Provide some mechanism to tolerate link or switch failures
  - multiple paths should be established between any two server nodes in a data center.
  - Fault tolerance of servers is achieved by replicating data and computing among redundant servers. Similar redundancy technology should apply to the network structure. Both software and hardware network redundancy apply to cope with potential failures.

DATA-CENTER DESIGN AND INTERCONNECTION NETWORKS

Data Center Interconnection Networks

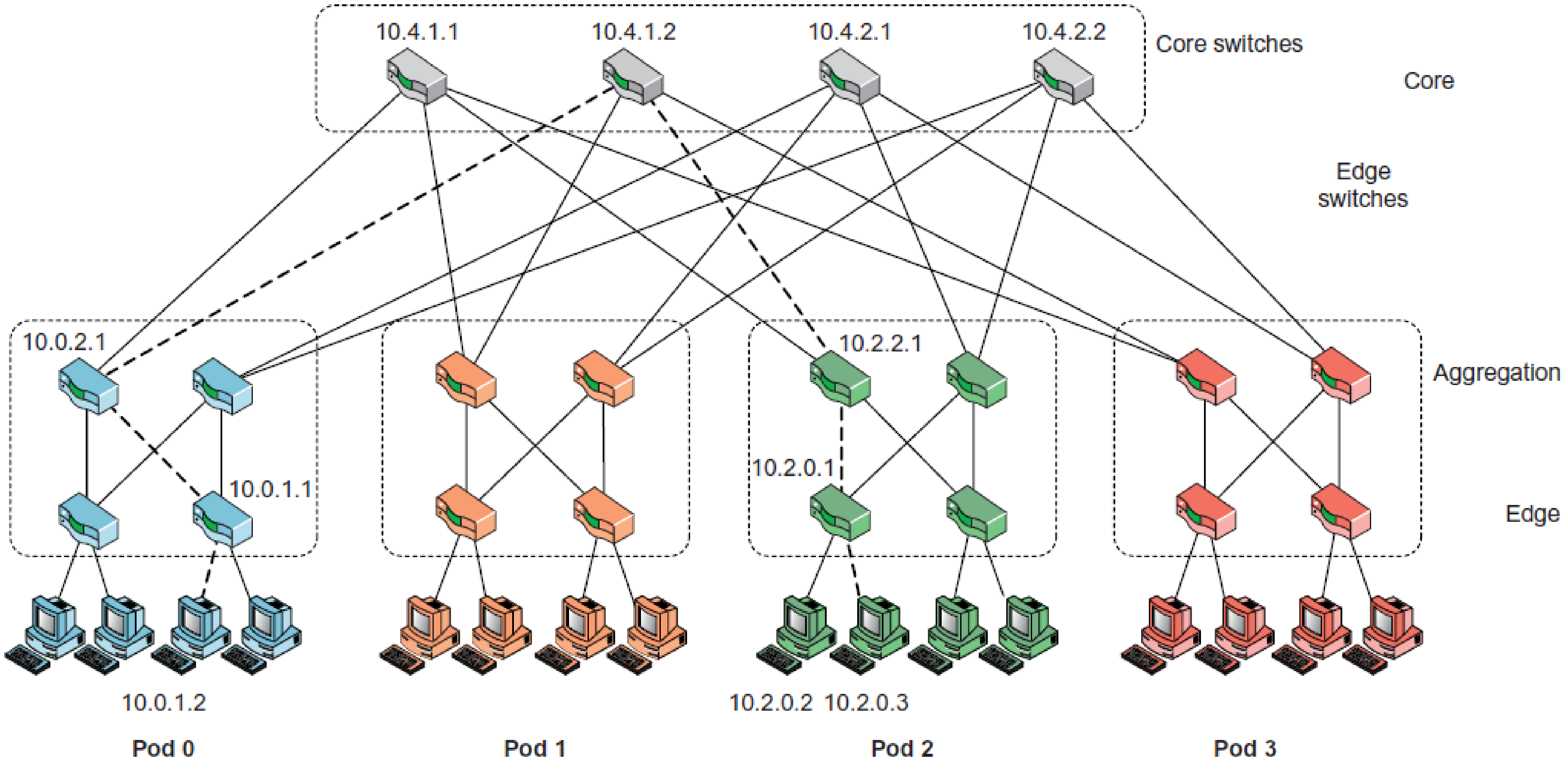


FIGURE 4.10

A fat-tree interconnection topology for scalable data-center construction.

(*Courtesy of M. Al-Fares, et al. [2]*)

DATA-CENTER DESIGN AND INTERCONNECTION NETWORKS

Modular Data Center in Shipping Containers

- A modern data center is structured as a shipyard of server clusters housed in truck-towed containers.

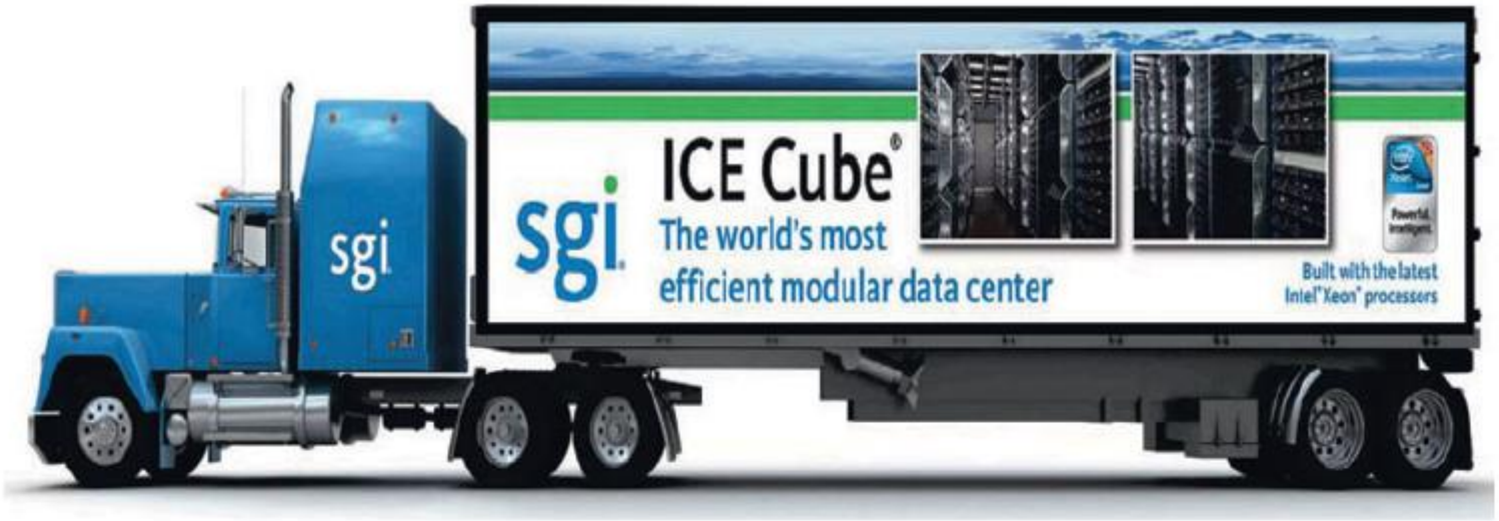


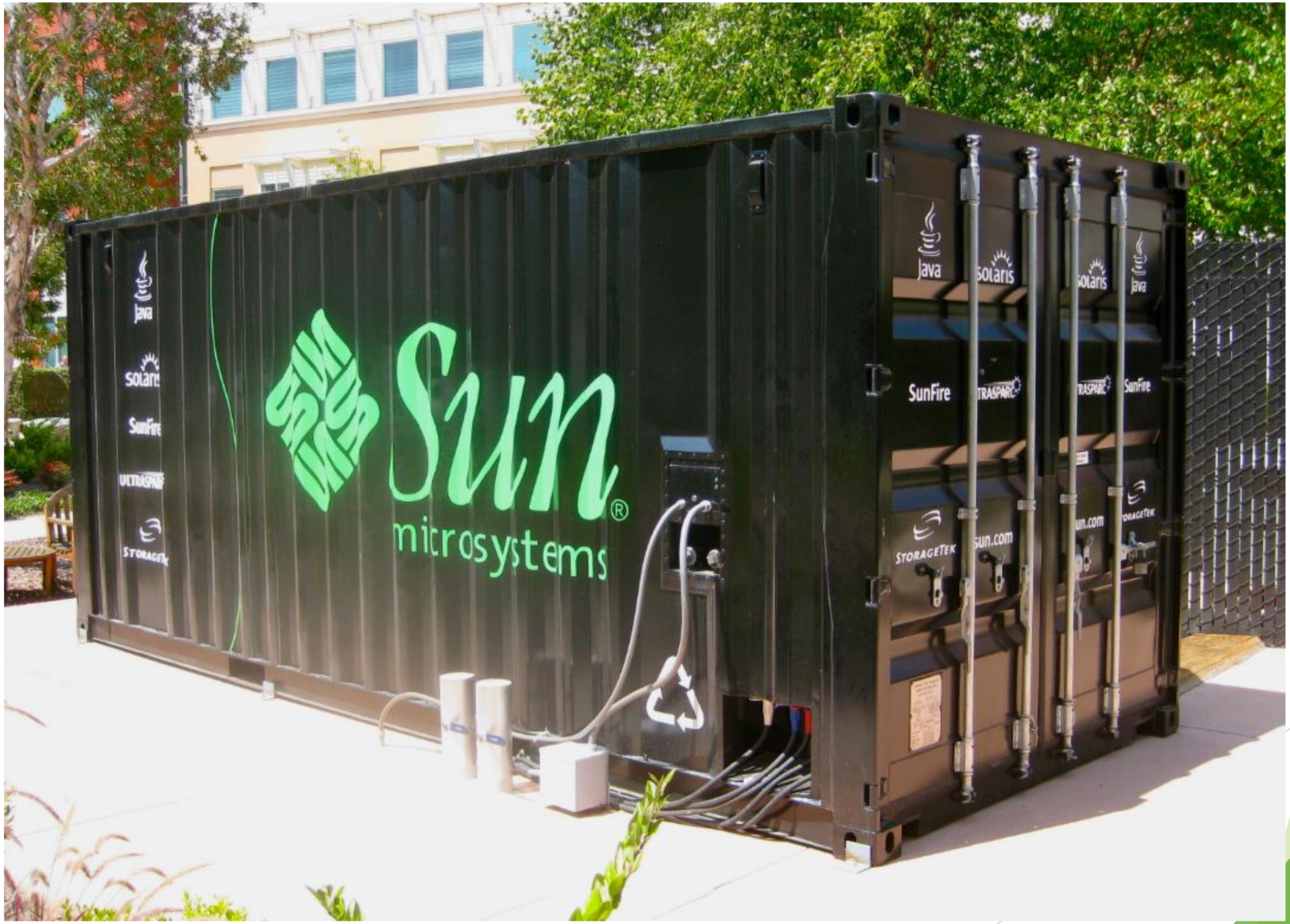
FIGURE 4.11

A modular data center built in a truck-towed ICE Cube container, that can be cooled by chilled air circulation with cold-water heat exchanges.

(Courtesy of SGI, Inc., http://www.sgi.com/icecube)

DATA-CENTER DESIGN AND INTERCONNECTION NETWORKS

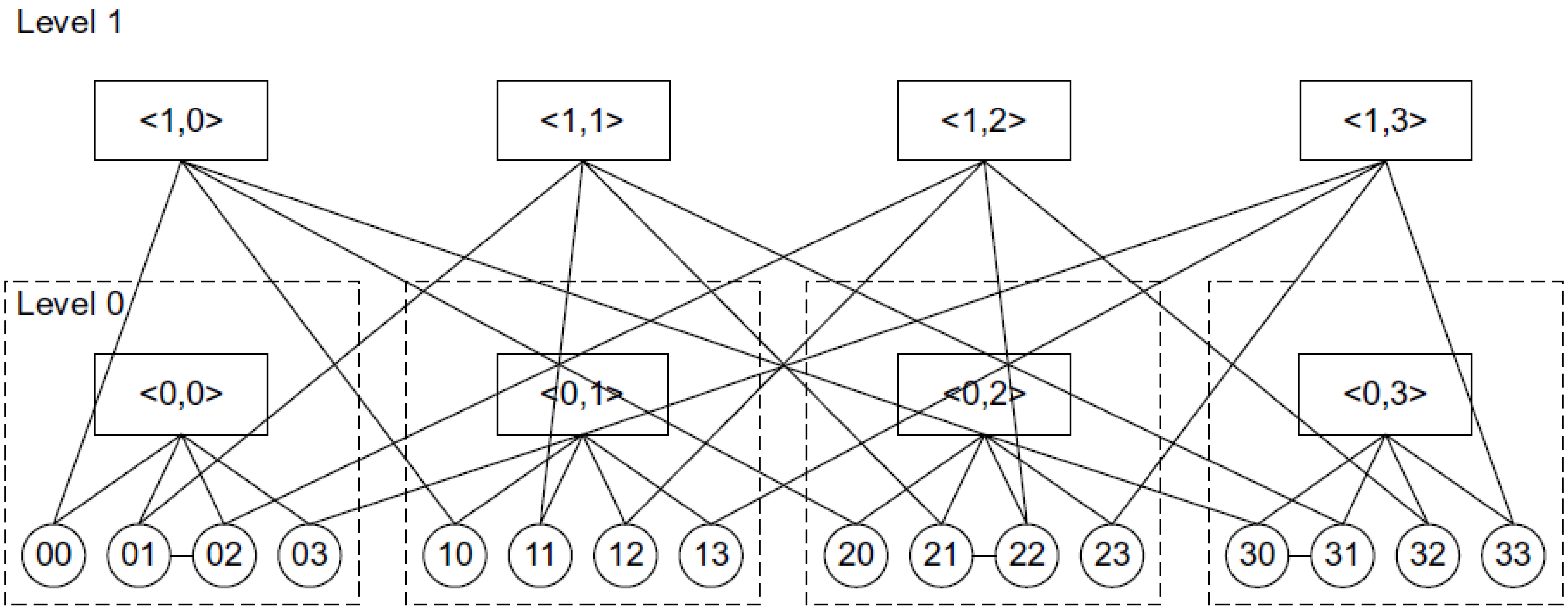
Modular Data Center in Shipping Containers



DATA-CENTER DESIGN AND INTERCONNECTION NETWORKS

Modular Data Center in Shipping Containers - Interconnection of Modular Data Centers

- A BCube is used as a server-centric network design for modular Data Centers.
- The BCube provides a layered structure. The bottom layer contains all the server nodes and they form Level 0. Level 1 switches form the top layer of $BCube_0$. BCube is a recursively constructed structure. The $BCube_0$ consists of n servers connecting to an n-port switch. $BCube_k$ (k ≥ 1) is structured from n $BCube_{k-1}$ with n^k n-port switches. The connection rule is that the i-th server in the j-th $BCube_0$ connects to the j-th port of the i-th Level 1 switch.



DATA-CENTER DESIGN AND INTERCONNECTION NETWORKS

Data-Center Management Issues

Basic requirements for managing the resources of a data center:

- **Making common users happy** The data center should be designed to provide quality service to the majority of users for at least 30 years.
- **Controlled information flow** Information flow should be streamlined. Sustained services and high availability (HA) are the primary goals.
- **Multiuser manageability** The system must be managed to support all functions of a data center, including traffic flow, database updating, and server maintenance.
- **Scalability to prepare for database growth** The system should allow growth as workload increases. The storage, processing, I/O, power, and cooling subsystems should be scalable.

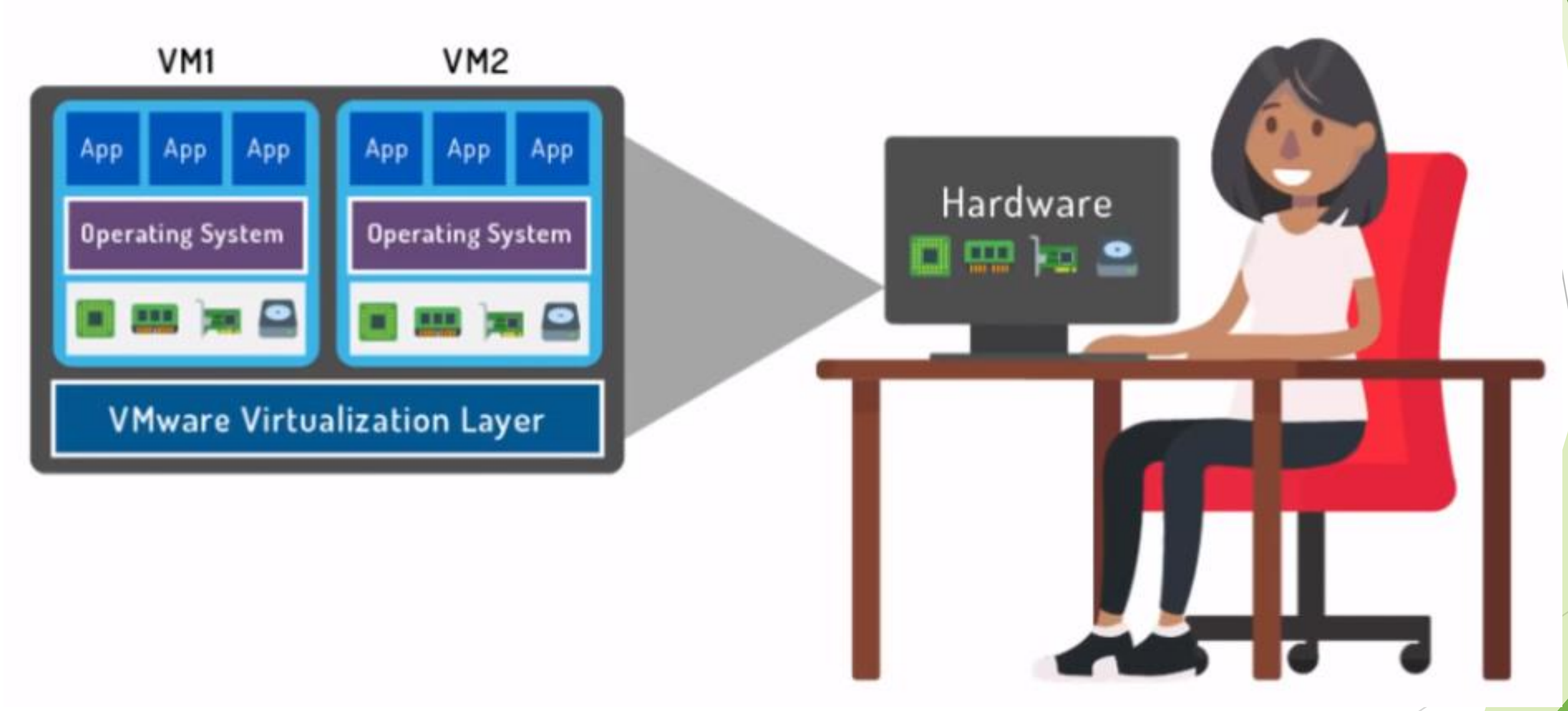
DATA-CENTER DESIGN AND INTERCONNECTION NETWORKS

Data-Center Management Issues

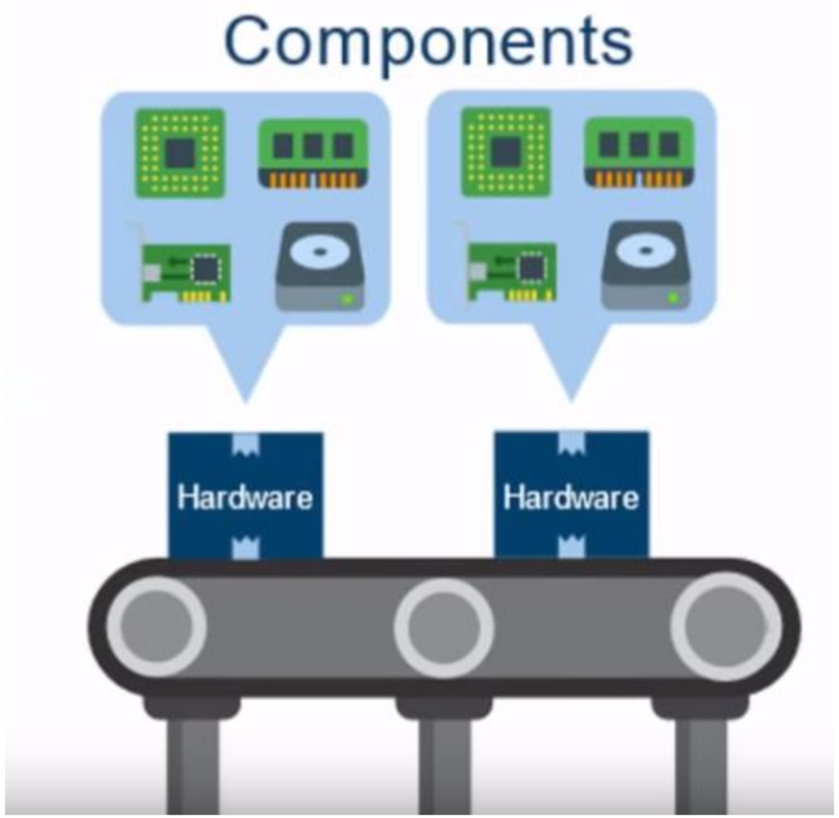
Basic requirements for managing the resources of a data center:

- **Reliability in virtualized infrastructure** Failover, fault tolerance, and VM live migration should be integrated to enable recovery of critical applications from failures or disasters.
- **Low cost to both users and providers** The cost to users and providers of the cloud system built over the data centers should be reduced, including all operational costs.
- **Security enforcement and data protection** Data privacy and security defense mechanisms must be deployed to protect the data center against network attacks and system interrupts and to maintain data integrity from user abuses or network attacks.
- **Green information technology** Saving power consumption and upgrading energy efficiency are in high demand when designing and operating current and future data centers.

What is virtualization



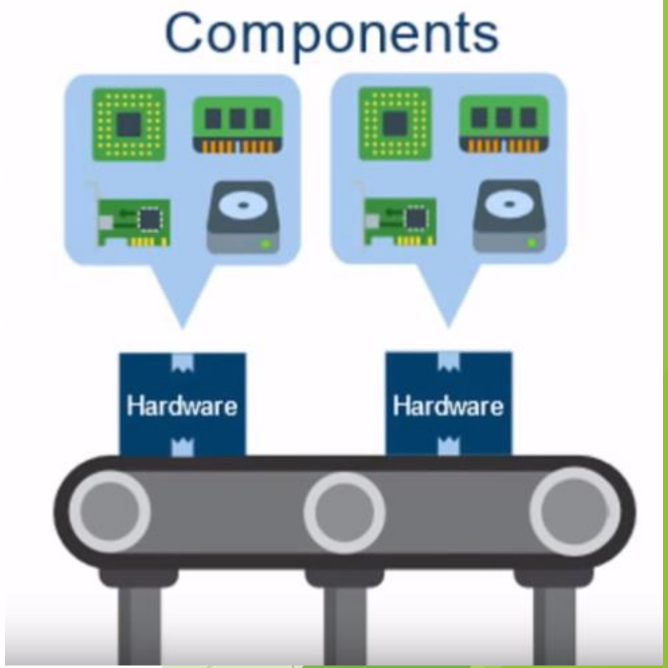
Tons of processing power!



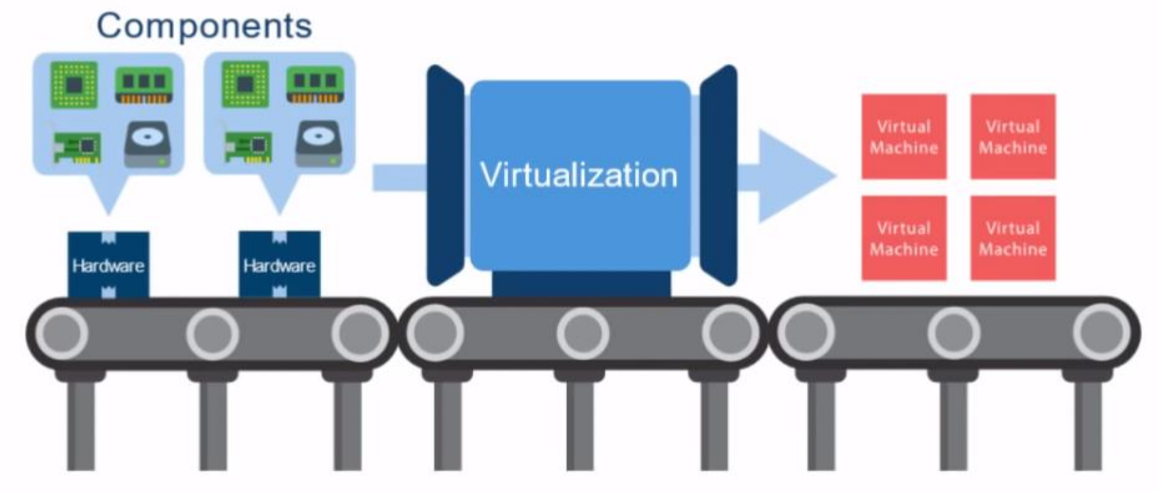
Tons of processing power!

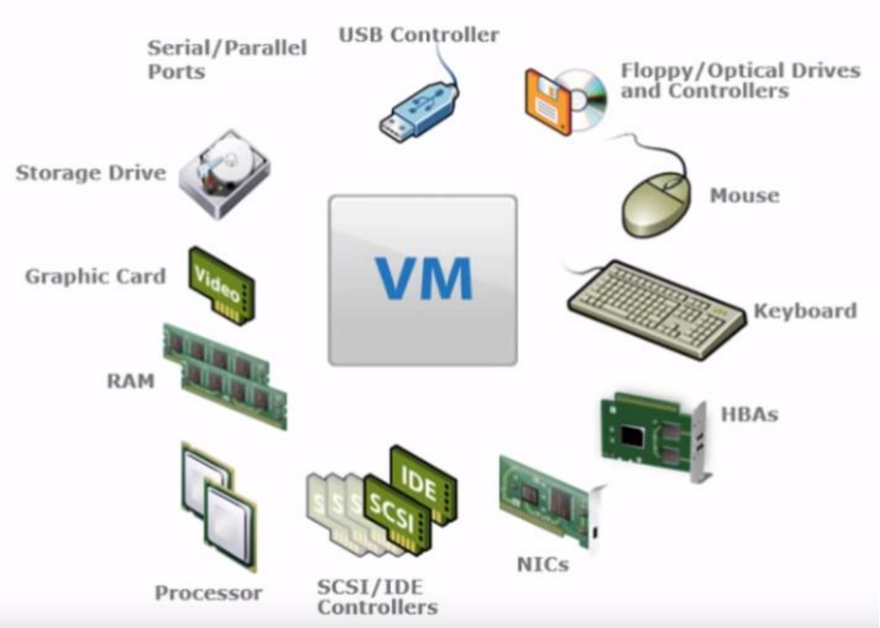
- Is the computing power being used efficiently?

*The hardware and processing power is under utilized, and the electricity is wasted



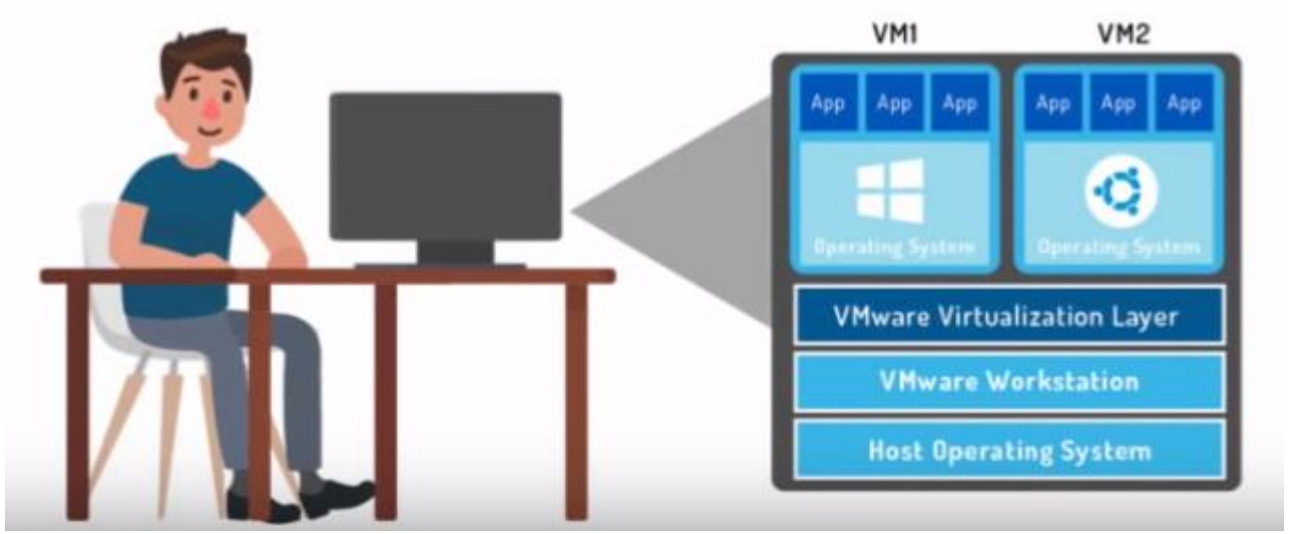
Tons of processing power!





Benefit

- Manageability – the ability to move, copy and isolate VMs
- Sustainability – energy saving by the way of less hardware and electricity
- Availability – the ability to snapshot, clone, and run redundant VMs
- Security – isolations of VMs an applications



Hypervisor

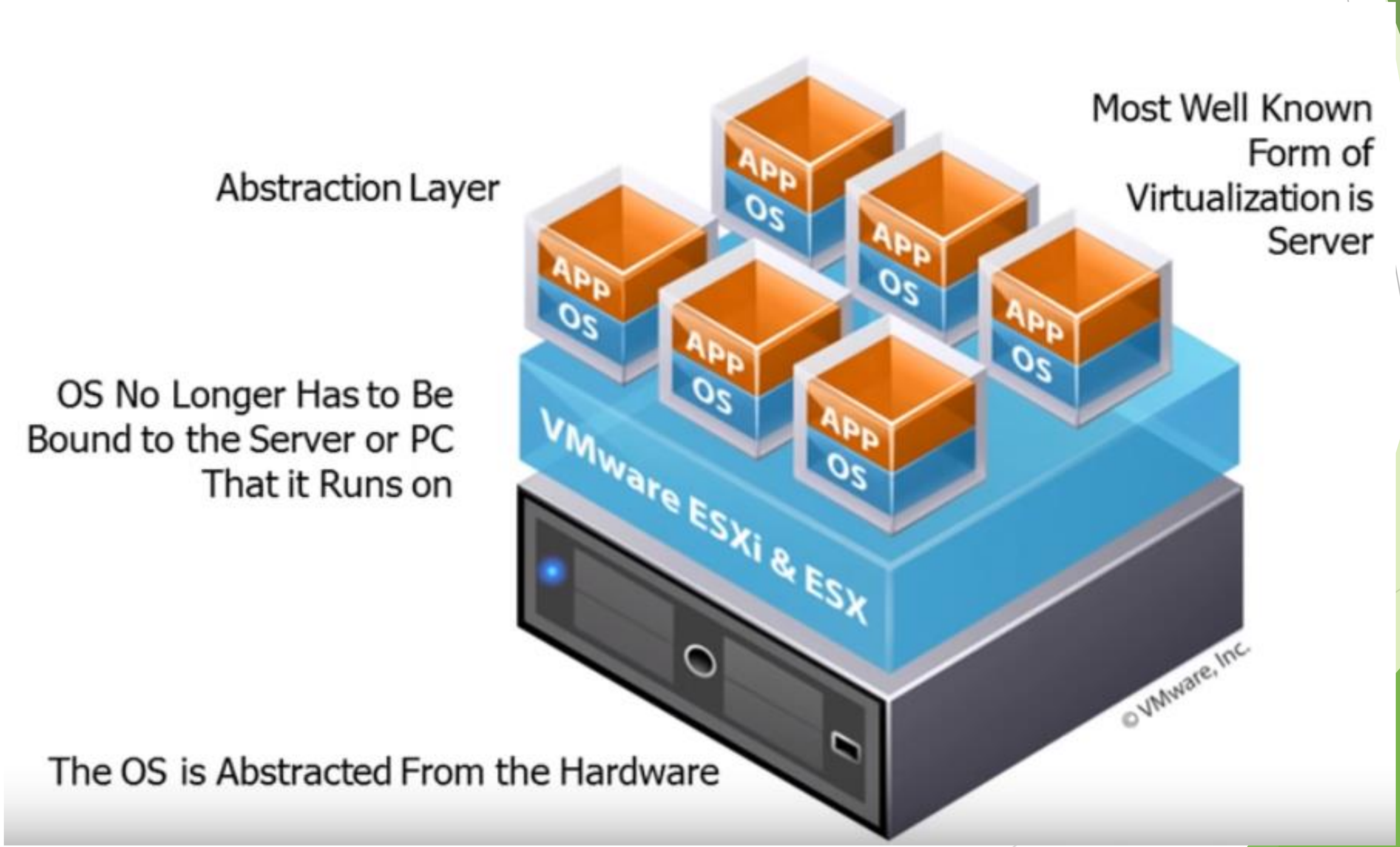
Examples:

- Vmware ESX/ESXi
- Hyper-V
- Vmware workstation
- Fusion
- Virtual Server
- XenServer

Type1 vs. Type2 hypervisors

- Type1
  - Loaded directly on the hardware
  - ESX/ESXi
  - Hyper-V
  - XenServer
- Type2
  - Loaded on an OS running on the hardware
  - Workstation
  - Virtual server
  - Fusion

Hypervisor



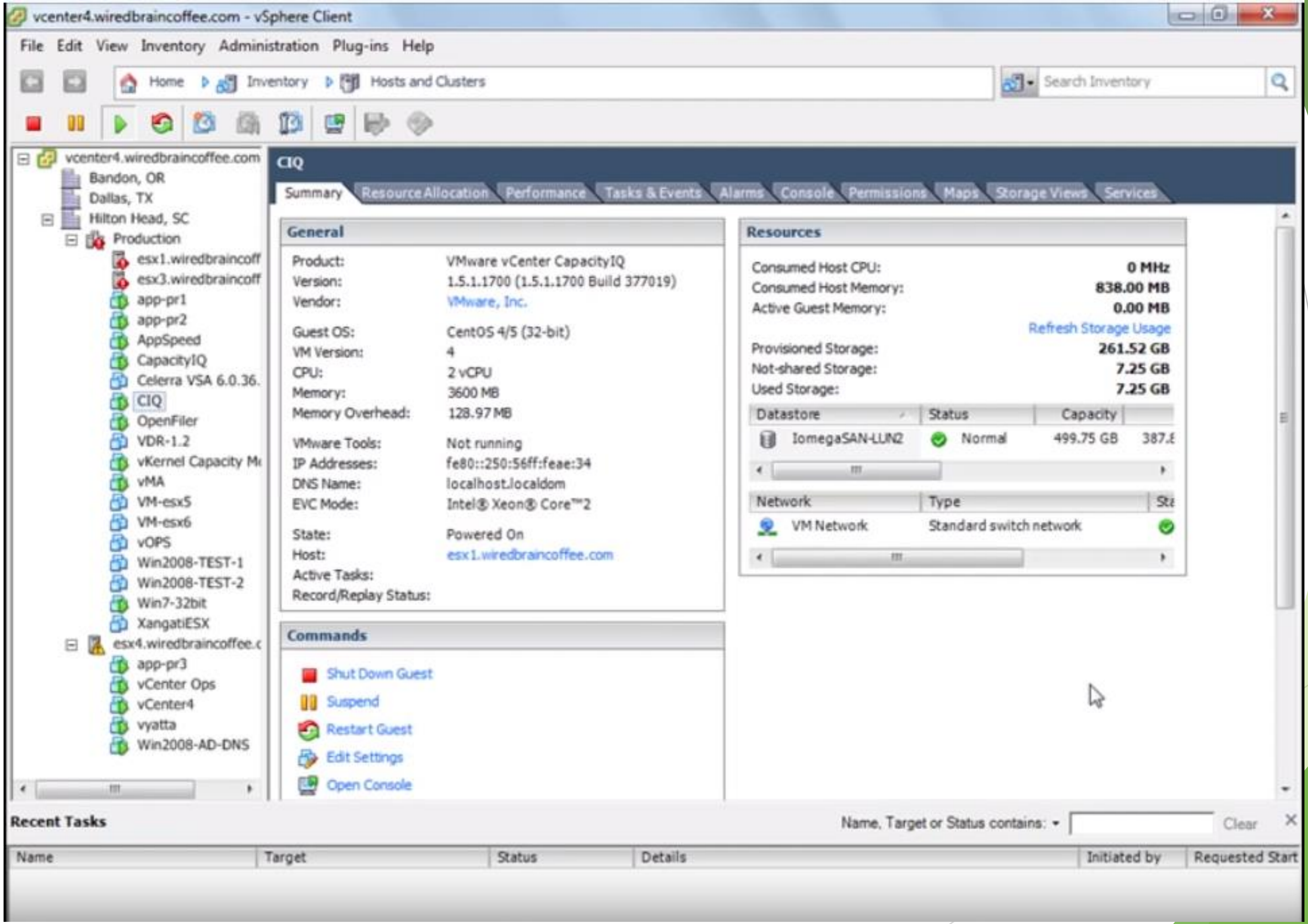
Abstraction Layer

Most Well Known
Form of
Virtualization is
Server

OS No Longer Has to Be
Bound to the Server or PC
That it Runs on

The OS is Abstracted From the Hardware

Hypervisor



General

Product: VMware vCenter CapacityIQ
Version: 1.5.1.1700 (1.5.1.1700 Build 377019)
Vendor: VMware, Inc.

Guest OS: CentOS 4/5 (32-bit)
VM Version: 4
CPU: 2 vCPU
Memory: 3600 MB
Memory Overhead: 128.97 MB

VMware Tools: Not running
IP Addresses: fe80::250:56ff:feae:34
DNS Name: localhost.localdom
EVC Mode: Intel® Xeon® Core™2

State: Powered On
Host: esx1.wiredbraincoffee.com
Active Tasks:
Record/Replay Status:

Resources

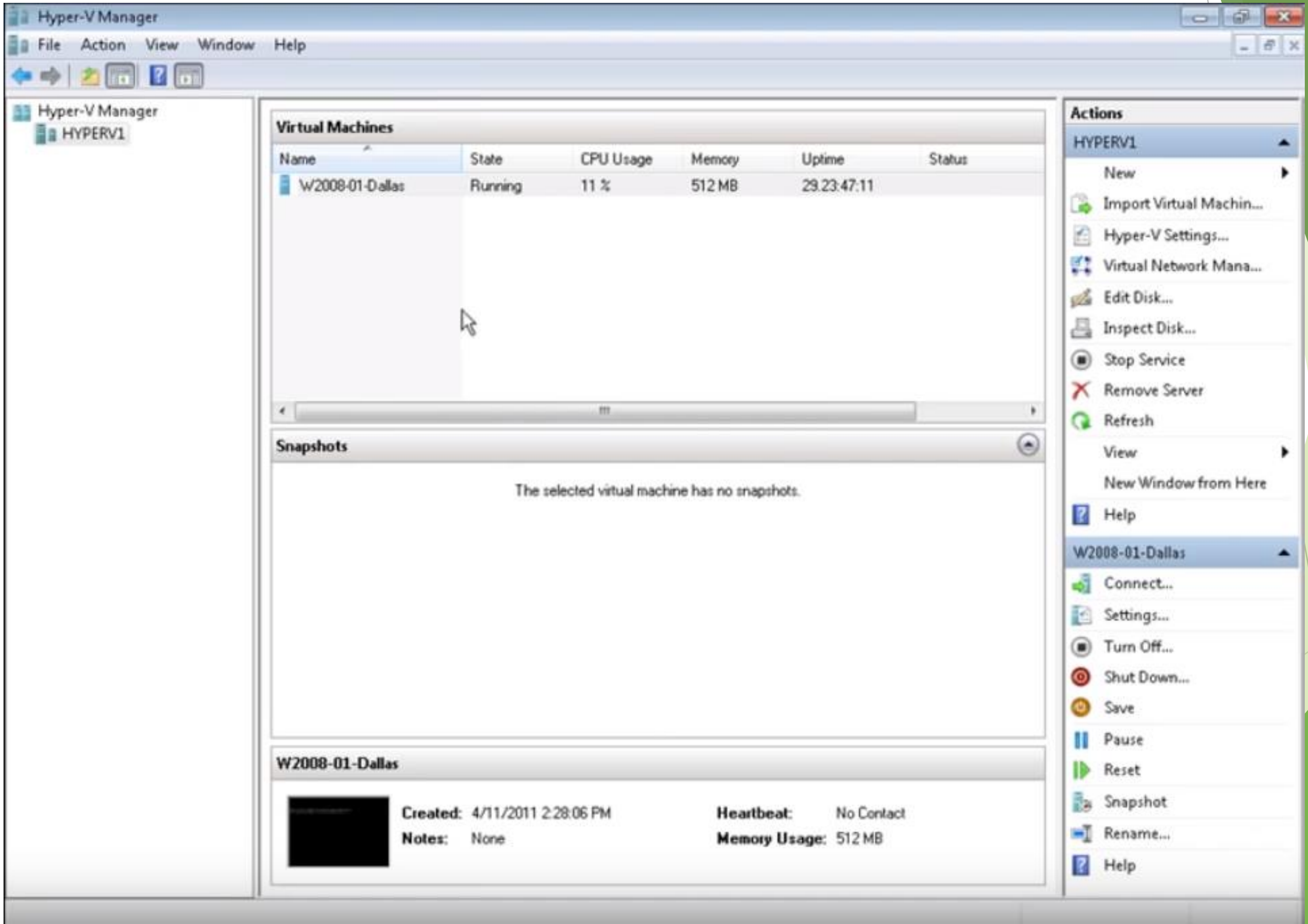
Consumed Host CPU: 0 MHz
Consumed Host Memory: 838.00 MB
Active Guest Memory: 0.00 MB
Refresh Storage Usage
Provisioned Storage: 261.52 GB
Not-shared Storage: 7.25 GB
Used Storage: 7.25 GB

Datastore	Status	Capacity	
IomegaSAN-LUN2	Normal	499.75 GB	387.8

Network	Type	Sta
VM Network	Standard switch network	✓

Name	Target	Status	Details	Initiated by	Requested Start

Hypervisor



Virtual Machines

Name	State	CPU Usage	Memory	Uptime	Status
W2008-01-Dallas	Running	11 %	512 MB	29.23:47:11	

Snapshots

The selected virtual machine has no snapshots.

W2008-01-Dallas

Created: 4/11/2011 2:28:06 PM
Notes: None

Heartbeat: No Contact
Memory Usage: 512 MB

Other type of virtualization

Desktop

I/O

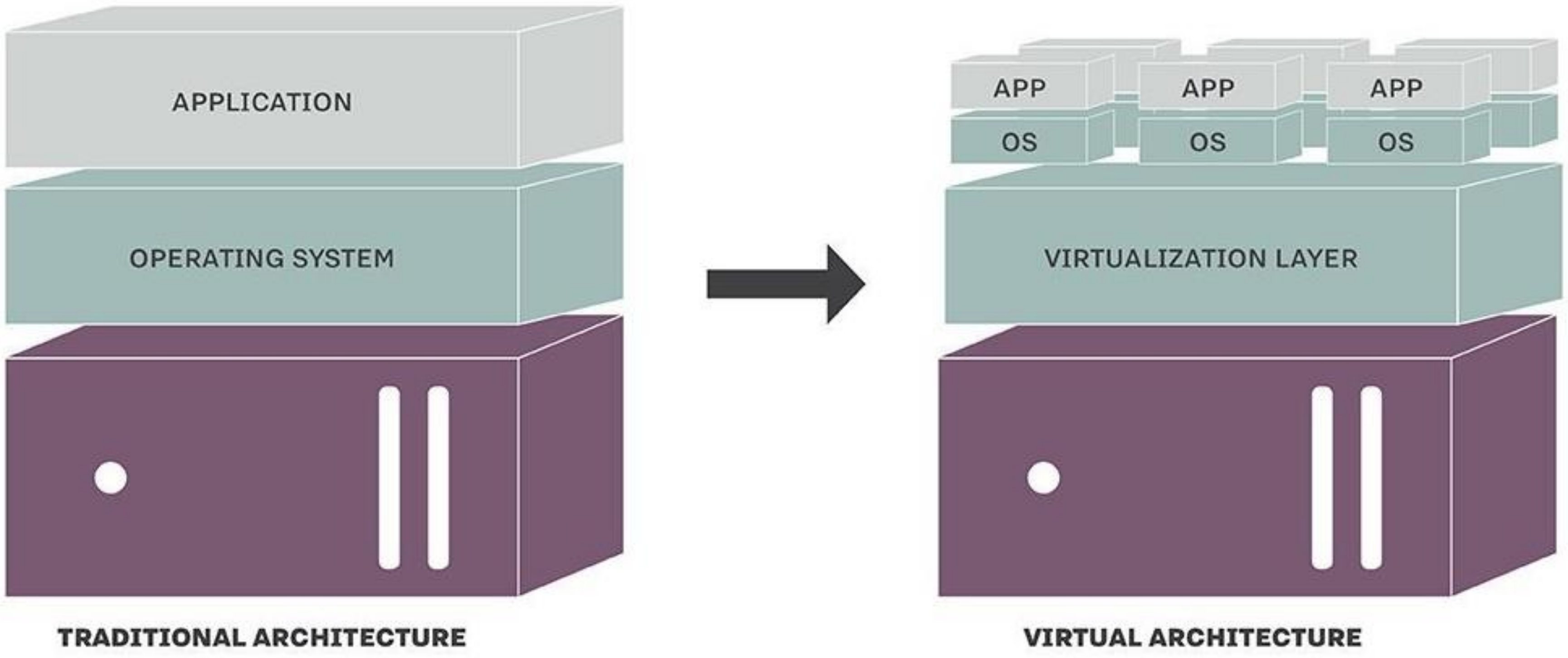
Network

Storage

Application

Virtualization

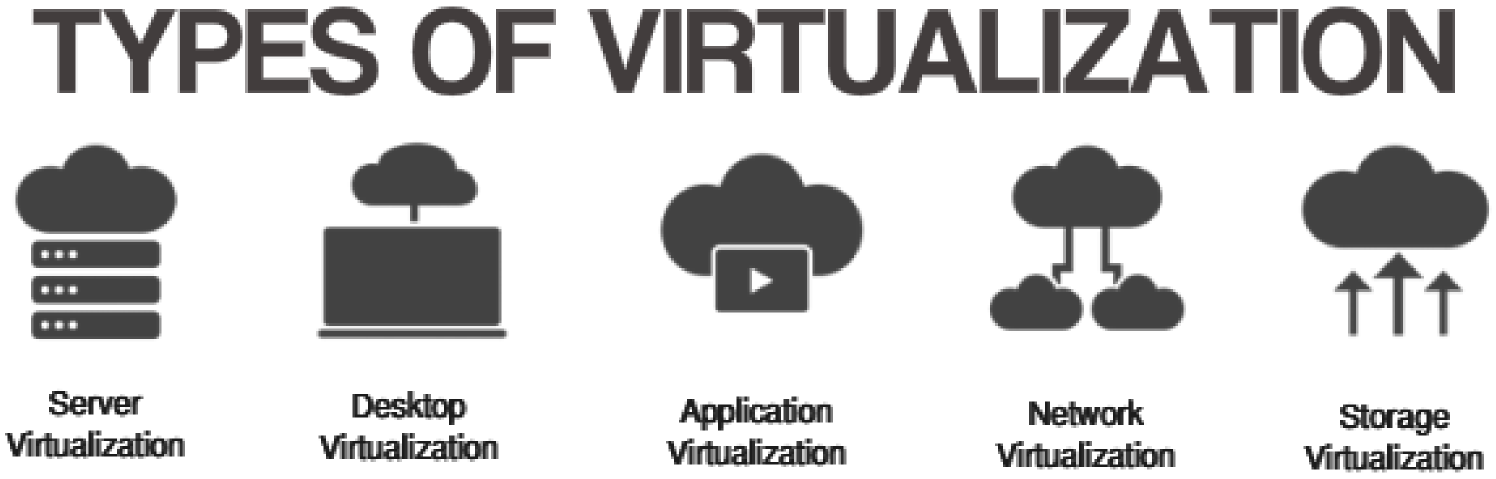
- Virtualization is the creation of a virtual rather than actual version of something, such as an operation system, a server , a storage device or network resources.
- Virtualization is a computer architecture technology by which multiple virtual machines (VMs) are multiplexed in the same hardware machine.



TRADITIONAL AND VIRTUAL ARCHITECTURE

Virtualization

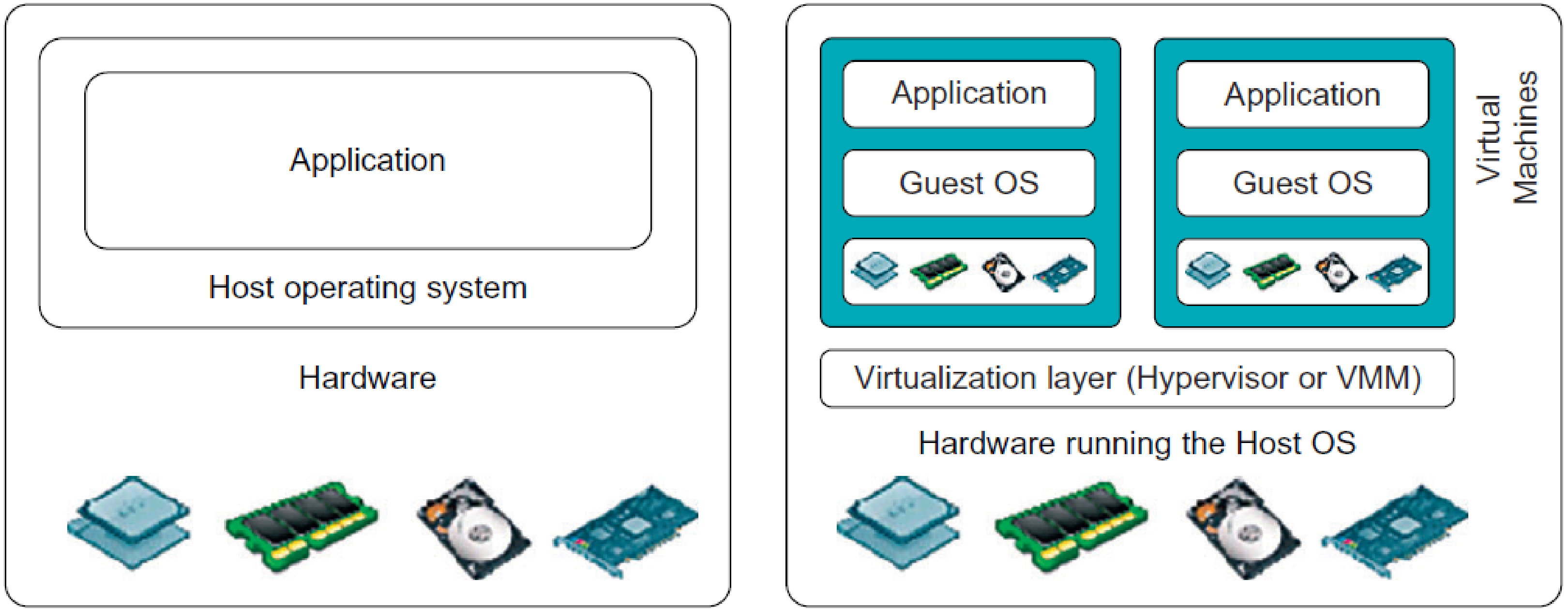
- The idea of VMs can be dated back to the 1960s.
- The purpose of a VM is to enhance resource sharing by many users and improve computer performance in terms of resource utilization and application flexibility.
- Hardware resources (CPU, memory, I/O devices, etc.) or software resources (operating system and software libraries) can be virtualized in various functional layers.
- This virtualization technology has been revitalized as the demand for distributed and cloud computing increased sharply in recent years.
- The idea is to separate the hardware from the software to yield better system efficiency.



IMPLEMENTATION LEVELS OF VIRTUALIZATION

Levels of Virtualization Implementation

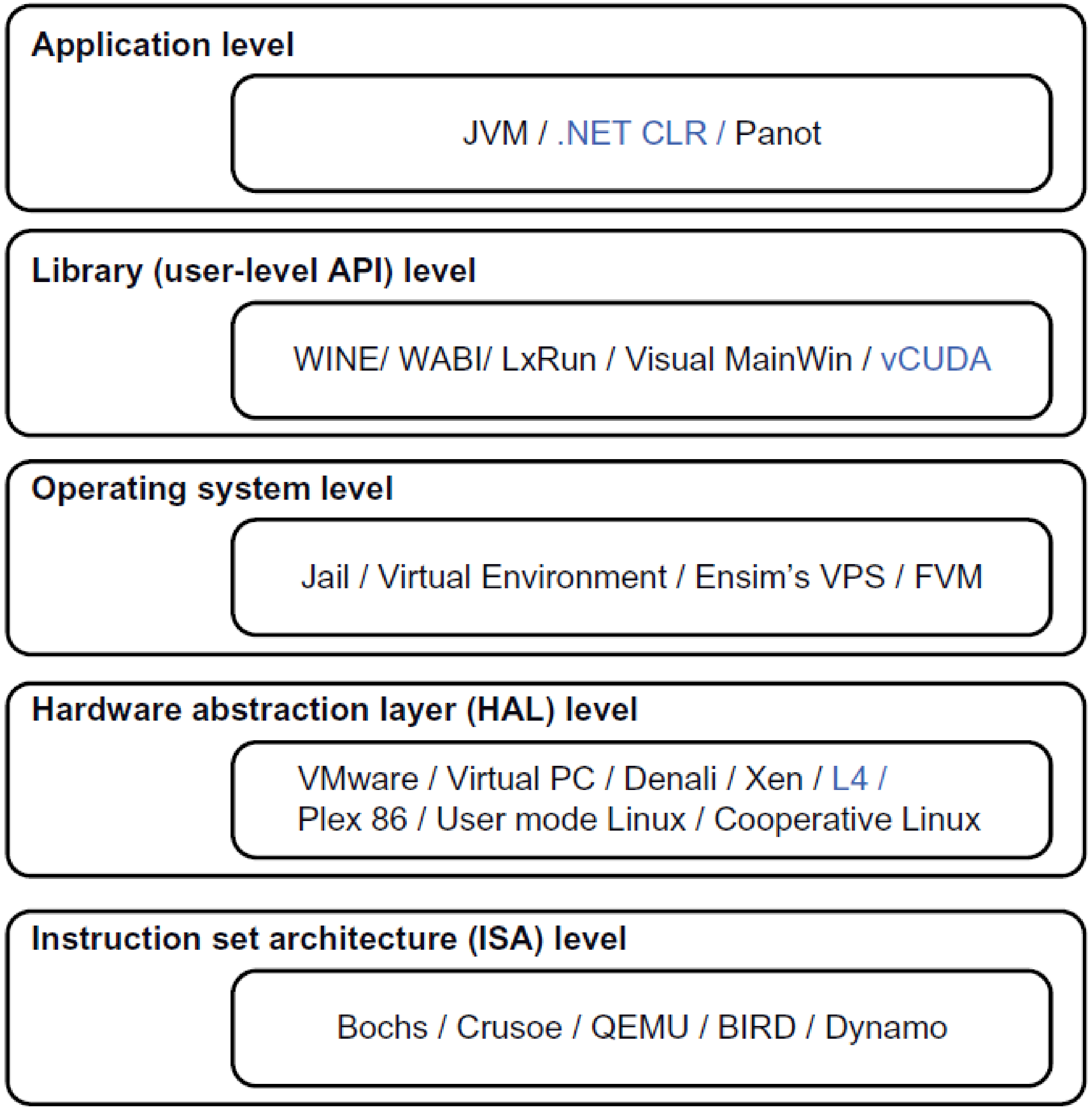
- A traditional computer runs with a host operating system specially tailored for its hardware architecture.
- After virtualization, different user applications managed by their own operating systems (guest OS) can run on the same hardware, independent of the host OS. This is often done by adding additional software, called a virtualization layer.
- This virtualization layer is known as hypervisor or virtual machine monitor (VMM)



(a) Traditional computer

(b) After virtualization

IMPLEMENTATION LEVELS OF VIRTUALIZATION



Application level

JVM / .NET CLR / Panot

Library (user-level API) level

WINE/ WABI/ LxRun / Visual MainWin / vCUDA

Operating system level

Jail / Virtual Environment / Ensim's VPS / FVM

Hardware abstraction layer (HAL) level

VMware / Virtual PC / Denali / Xen / L4 /
Plex 86 / User mode Linux / Cooperative Linux

Instruction set architecture (ISA) level

Bochs / Crusoe / QEMU / BIRD / Dynamo

IMPLEMENTATION LEVELS OF VIRTUALIZATION

VMM Design Requirements and Providers

- Hardware-level virtualization inserts a layer between real hardware and traditional operating systems.
- This layer is commonly called the Virtual Machine Monitor (VMM) and it manages the hardware resources of a computing system.
- Each time programs access the hardware the VMM captures the process. In this sense, the VMM acts as a traditional OS. One hardware component, such as the CPU, can be virtualized as several virtual copies. Therefore, several traditional operating systems which are the same or different can sit on the same set of hardware simultaneously.

Three requirements for a VMM.

- First, a VMM should provide an environment for programs which is essentially identical to the original machine.
- Second, programs run in this environment should show, at worst, only minor decreases in speed.
- Third, a VMM should be in complete control of the system resources. Any program run under a VMM should exhibit a function identical to that which it runs on the original machine directly.

IMPLEMENTATION LEVELS OF VIRTUALIZATION

VMM Design Requirements and Providers

- Complete control of resources by a VMM includes the following aspects:
  - (1) The VMM is responsible for allocating hardware resources for programs;
  - (2) it is not possible for a program to access any resource not explicitly allocated to it;
  - 3) it is possible under certain circumstances for a VMM to regain control of resources already allocated. Not all processors satisfy these requirements for a VMM. A VMM is tightly related to the architectures of processors. It is difficult to implement a VMM for some types of processors, such as the x86.

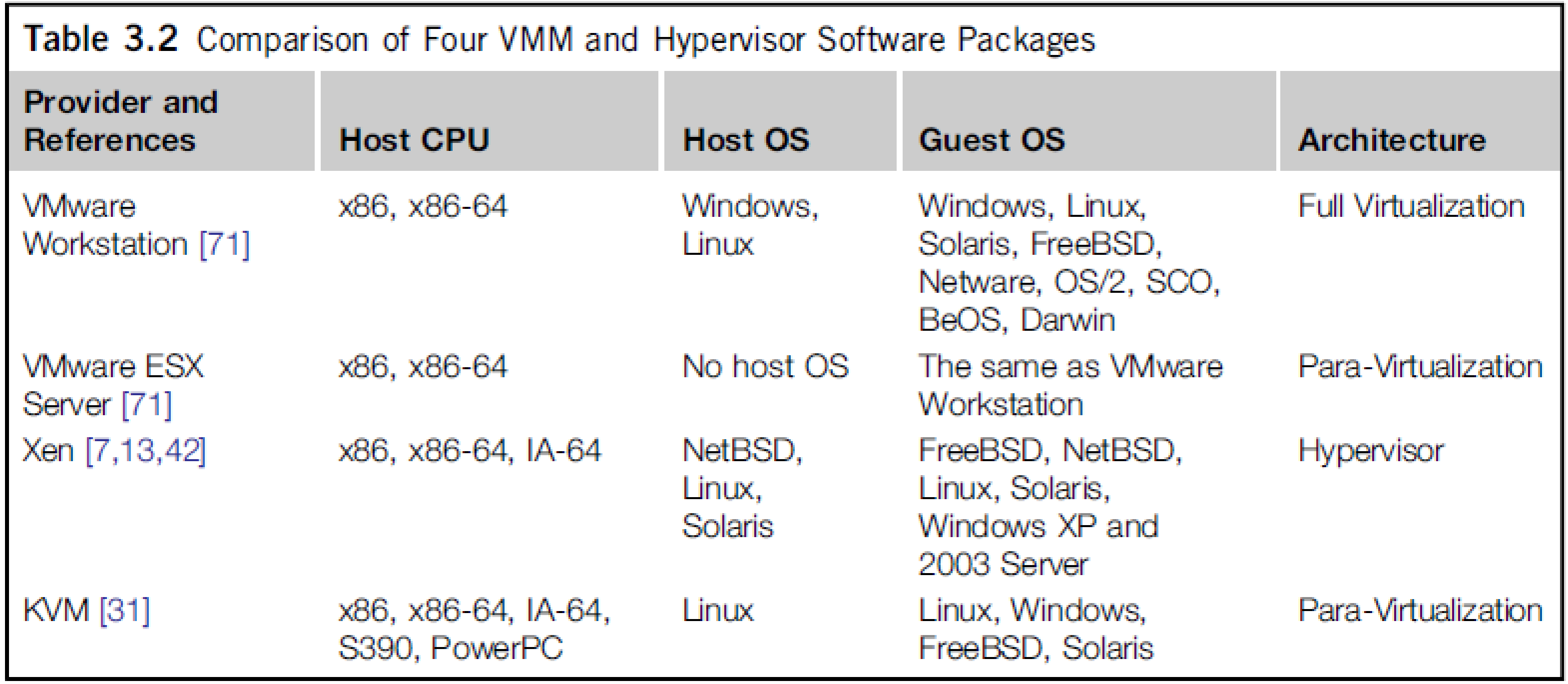


Table 3.2 Comparison of Four VMM and Hypervisor Software Packages

Provider and References	Host CPU	Host OS	Guest OS	Architecture
VMware Workstation [71]	x86, x86-64	Windows, Linux	Windows, Linux, Solaris, FreeBSD, Netware, OS/2, SCO, BeOS, Darwin	Full Virtualization
VMware ESX Server [71]	x86, x86-64	No host OS	The same as VMware Workstation	Para-Virtualization
Xen [7,13,42]	x86, x86-64, IA-64	NetBSD, Linux, Solaris	FreeBSD, NetBSD, Linux, Solaris, Windows XP and 2003 Server	Hypervisor
KVM [31]	x86, x86-64, IA-64, S390, PowerPC	Linux	Linux, Windows, FreeBSD, Solaris	Para-Virtualization

Full virtualization	Para virtualization
Guest operating systems are unaware of each other	unlike full virtualization, guest servers are aware of one another
Provide support for unmodified guest operating system.	Hypervisor does not need large amounts of processing power to manage guest OS
Hypervisor directly interact with the hardware such as CPU, disks.	The entire system work as a cohesive unit
Hypervisor allow to run multiple OS simultaneously on host computer.	
Each guest server run on its own operating system	
Few implementations: Oracle's VirtualBox, VMware server, Microsoft Virtual PC	
Advantages	**Advantages**
This type of virtualization provides best isolation and security for Virtual machine.	As a guest OS can directly communicate with hypervisor
Truly isolated multiple guest OS can run simultaneously on same hardware.	This is efficient virtualization
It's only option that requires no hardware assist or OS assist to virtualize sensitive and privileged instructions.	Allow users to make use of new or modified device drivers
Limitations	**Limitations**
full virtualization is usually bit slower, because of all emulation. hypervisor contain the device driver and it might be difficult for new device drivers to be installer by users.	Para virtualization requires the guest OS to be modified in order to interact with para virtualization interfaces. It requires significant support and maintainability issues in production environment.

VIRTUALIZATION STRUCTURES/TOOLS AND MECHANISMS

- In general, there are three typical classes of VM architecture.
  - hypervisor architecture
  - Paravirtualization
  - host-based virtualization
- Before virtualization, the operating system manages the hardware. After virtualization, a virtualization layer is inserted between the hardware and the operating system. In such a case, the virtualization layer is responsible for converting portions of the real hardware into virtual hardware. Therefore, different operating systems such as Linux and Windows can run on the same physical machine, simultaneously. Depending on the position of the virtualization layer, there are several classes of VM architectures, namely the hypervisor architecture, paravirtualization, and host-based virtualization. The hypervisor is also known as the VMM (Virtual Machine Monitor).

VIRTUALIZATION STRUCTURES/TOOLS AND MECHANISMS

Hypervisor Architecture.

- The hypervisor supports hardware-level virtualization on bare-metal devices like CPU, memory, disk and network interfaces. The hypervisor software sits directly between the physical hardware and the OS. This virtualization layer is referred to as either the VMM or the hypervisor.
- The hypervisor provides hyper-calls for the guest OSes and applications. A hypervisor can assume a micro-kernel architecture like Microsoft Hyper-V. It can also assume a monolithic hypervisor architecture like VMware ESX for server virtualization.
- A micro-kernel hypervisor includes only the basic and unchanging functions (such as physical memory management and processor scheduling). The device drivers and other changeable components are outside the hypervisor. A monolithic hypervisor implements all the aforementioned functions, including those of the device drivers.
- Therefore, the size of the hypervisor code of a micro-kernel hypervisor is smaller than that of a monolithic hypervisor. Essentially, a hypervisor must be able to convert physical devices into virtual resources dedicated for the deployed VM to use.

VIRTUALIZATION STRUCTURES/TOOLS AND MECHANISMS

Binary Translation with Full Virtualization.

- Depending on implementation technologies, hardware virtualization can be classified into two categories:
  - **Full virtualization** does not need to modify the host OS. It relies on binary translation to trap and virtualize the execution of certain sensitive, non-virtualizable instructions. The guest OSes and their applications consist of noncritical and critical instructions.
  - **host-based system** both a host OS and a guest OS are used. A virtualization software layer is built between the host OS and guest OS.

VIRTUALIZATION STRUCTURES/TOOLS AND MECHANISMS

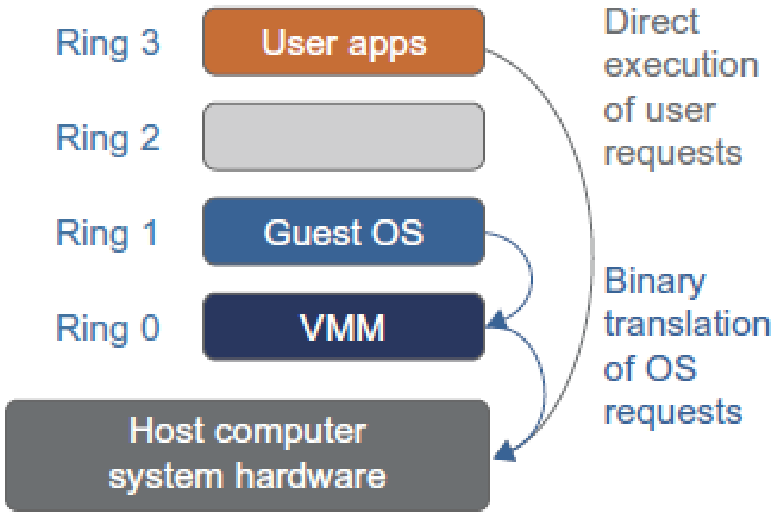
Full Virtualization.

- With full virtualization, noncritical instructions run directly on the hardware while critical instructions are discovered and replaced with traps into the VMM to be emulated by software. **Both the hypervisor and VMM** approaches are considered full virtualization.
- Why are only critical instructions trapped into the VMM? This is because binary translation can incur a large performance overhead. Noncritical instructions do not control hardware or threaten the security of the system, but critical instructions do. Therefore, running noncritical instructions on hardware not only can promote efficiency, but also can ensure system security.
- This approach was implemented by VMware and many other software companies.

VIRTUALIZATION STRUCTURES/TOOLS AND MECHANISMS

Host-Based Virtualization

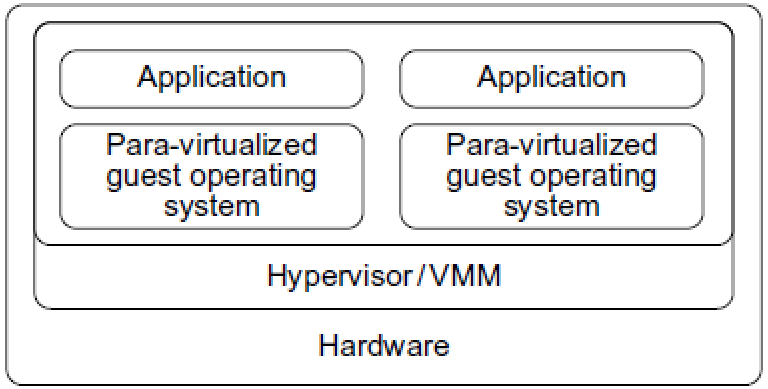
- An alternate VM architecture is to install a virtualization layer on top of the host OS. This host OS is still responsible for managing the hardware.
- User can install this VM architecture without modifying the host OS.
- Compared to the hypervisor/VMM architecture, the performance of the host-based architecture might also be low. When an application requests hardware access, it involves four layers of mapping, which downgrades performance significantly. When the Internet Security and Acceleration (ISA) of a guest OS is different from the ISA of the underlying hardware, binary translation must be adopted. Although the host-based architecture has flexibility, the performance is too low to be useful in practice.
- Examples:
  - VMware Workstation, Server, Player and Fusion
  - Oracle VM VirtualBox
  - Microsoft Virtual PC
  - Parallels Desktop


Ring 3
User apps
Direct execution of user requests
Ring 2
Ring 1
Guest OS
Ring 0
VMM
Binary translation of OS requests
Host computer system hardware

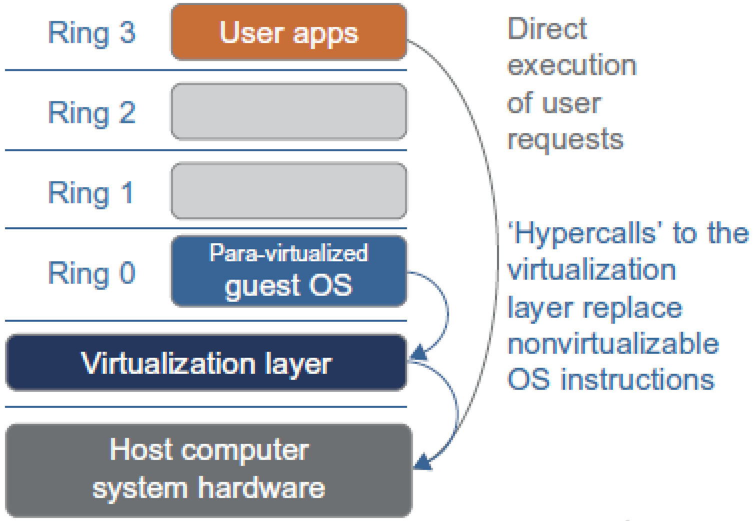
VIRTUALIZATION STRUCTURES/TOOLS AND MECHANISMS

Para-Virtualization with Compiler Support.

- Para-virtualization needs to modify the guest OS.
- Performance degradation is a critical issue of a virtualized system.
- The traditional x86 processor offers four instruction execution rings: Rings 0, 1, 2 and 3. The lower the ring number, the higher the privilege of instruction being executed. The OS is responsible for managing the hardware and the privileged instructions to execute at Ring 0, while user-level applications run at Ring 3. The best example of para-virtualization is kernel-based VM (KVM).



Application
Application
Para-virtualized guest operating system
Para-virtualized guest operating system
Hypervisor/VMM
Hardware



Ring 3
User apps
Ring 2
Ring 1
Ring 0
Para-virtualized guest OS
Virtualization layer
Host computer system hardware
Direct execution of user requests
'Hypercalls' to the virtualization layer replace nonvirtualizable OS instructions

VIRTUALIZATION STRUCTURES/TOOLS AND MECHANISMS

Para-Virtualization with Compiler Support.

- Although para-virtualization reduces overhead, it incurs other problems. First, its compatibility and portability may be in doubt, because it must support the unmodified OS as well. Second, the cost of maintaining para-virtualized OSes is high, because they could require deep OS kernel modifications.
- Finally, the performance advantage of para-virtualization varies greatly due to workload variations. Compared with full virtualization, para-virtualization is relatively easy and more practical. The main problem in full virtualization is its low performance in binary translation. Speeding up binary translation is difficult. Therefore, many virtualization products employ the para-virtualization architecture. The popular Xen, KVM and VMware ESX are good examples.
- KVM is a hardware-assisted para-virtualization tool, which improves performance and supports unmodified guest OSes such as Windows, Linux, Solaris and other Unix variants.
- This is a Linux para-virtualization system—part of the Linux version 2.6.20 kernel. The existing Linux kernel carries out memory management and scheduling activities. KVM does the rest, which makes it simpler than the hypervisor that controls the entire machine.
- Unlike the full virtualization architecture that intercepts and emulates privileged and sensitive instructions at run time, para-virtualization handles these instructions at compile time. The guest OS kernel is modified to replace the privileged and sensitive instructions with hypercalls to the hypervisor or VMM. Xen is one example of such para-virtualization architecture.
- The privileged instructions are implemented by hypercalls to the hypervisor. After replacing the instructions with hypercalls, the modified guest OS emulates the behavior of the original guest OS. On a Unix system, a system call involves an interrupt or service routine. The hypercalls apply a dedicated service routine in Xen.

VIRTUALIZATION OF CPU, MEMORY, AND I/O DEVICES

Hardware Support for Virtualization.

- Modern operating systems and processors permit multiple processes to run simultaneously.
- If there is no protection mechanism in a processor, all instructions from different processes will access the hardware directly and cause a system crash.
- Therefore, all processors have at least two modes, user mode and supervisor mode, to ensure controlled access of critical hardware.
- Instructions running in supervisor mode are called privileged instructions. Other instructions are unprivileged instructions.
- In a virtualized environment, it is more difficult to make OSes and applications run correctly because there are more layers in the machine stack.

VIRTUALIZATION OF CPU, MEMORY, AND I/O DEVICES

CPU Virtualization.

- VM is a duplicate of an existing computer system in which a majority of the VM instructions are executed on the host processor in native mode. Thus, unprivileged instructions of VMs run directly on the host machine for higher efficiency. Other critical instructions should be handled carefully for correctness and stability. The critical instructions are divided into three categories: privileged instructions, **control sensitive instructions, and behavior-sensitive instructions**.
- Privileged instructions execute in a privileged mode and will be trapped if executed outside this mode.
- Control-sensitive instructions attempt to change the configuration of resources used.
- Behavior-sensitive instructions have different behaviors depending on the configuration of resources, including the load and store operations over the virtual memory.
- A CPU architecture is virtualizable if it supports the ability to run the VM's privileged and unprivileged instructions in the CPU's user mode while the VMM runs in supervisor mode. When the privileged instructions including control- and behavior-sensitive instructions of a VM are executed, they are trapped in the VMM.
- In this case, the VMM acts as a unified mediator for hardware access from different VMs to guarantee the correctness and stability of the whole system.

VIRTUALIZATION OF CPU, MEMORY, AND I/O DEVICES

CPU Virtualization - Hardware-Assisted CPU Virtualization.

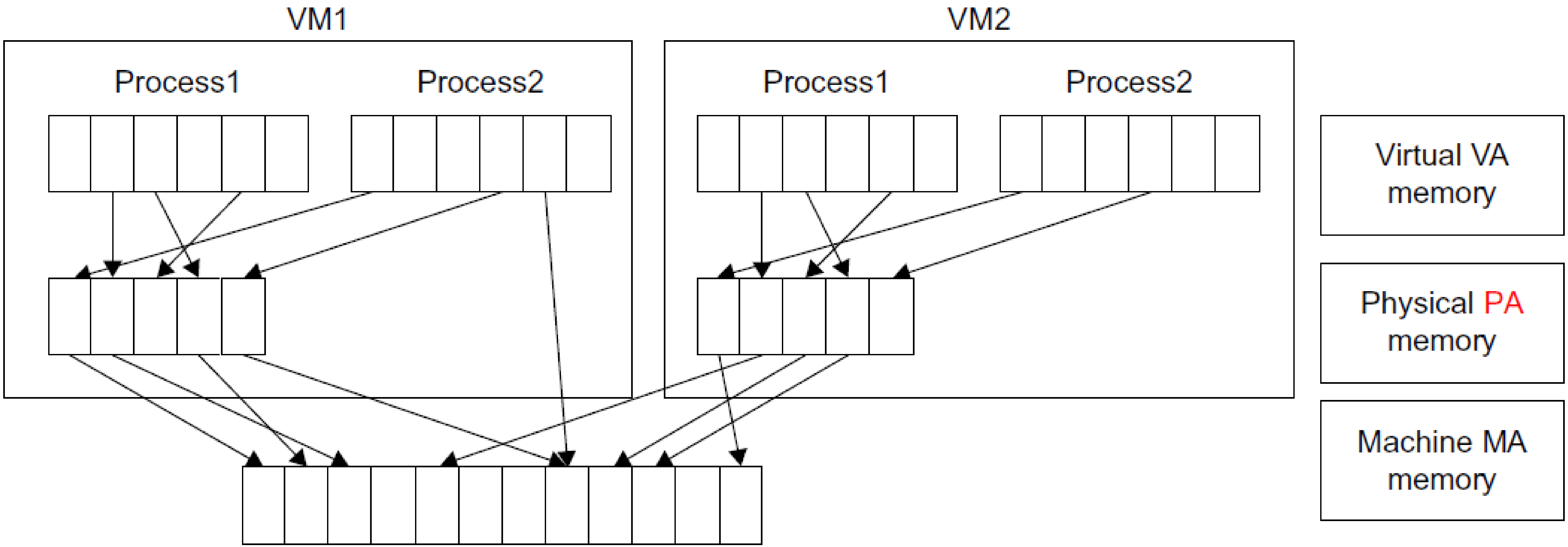
- This technique attempts to simplify virtualization because full or paravirtualization is complicated. Intel and AMD add an additional mode called privilege mode level (some people call it Ring-1) to x86 processors. Therefore, operating systems can still run at Ring 0 and the hypervisor can run at Ring -1. All the privileged and sensitive instructions are trapped in the hypervisor automatically. This technique removes the difficulty of implementing binary translation of full virtualization. It also lets the operating system run in VMs without modification.
- Generally, hardware-assisted virtualization should have high efficiency.
- However, since the transition from the hypervisor to the guest OS incurs high overhead switches between processor modes, it sometimes cannot outperform binary translation. Hence, virtualization systems such as VMware now use a hybrid approach, in which a few tasks are offloaded to the hardware but the rest is still done in software. In addition, para-virtualization and hardware-assisted virtualization can be combined to improve the performance further.

VIRTUALIZATION OF CPU, MEMORY, AND I/O DEVICES

Memory Virtualization.

- Virtual memory virtualization is similar to the virtual memory support provided by modern operating systems. In a traditional execution environment, the operating system maintains mappings of virtual memory to machine memory using page tables, which is a one-stage mapping from virtual memory to machine memory. All modern x86 CPUs include a memory management unit (MMU) and a translation lookaside buffer (TLB) to optimize virtual memory performance. However, in a virtual execution environment, virtual memory virtualization involves sharing the physical system memory in RAM and dynamically allocating it to the physical memory of the VMs.

- That means a two-stage mapping process should be maintained by the guest OS and the VMM, respectively: virtual memory to physical memory and physical memory to machine memory. Furthermore, MMU virtualization should be supported, which is transparent to the guest OS. The guest OS continues to control the mapping of virtual addresses to the physical memory addresses of VMs. But the guest OS cannot directly access the actual machine memory. The VMM is responsible for mapping the guest physical memory to the actual machine memory.


VM1
Process1
Process2
VM2
Process1
Process2
Virtual VA memory
Physical PA memory
Machine MA memory

VIRTUALIZATION OF CPU, MEMORY, AND I/O DEVICES

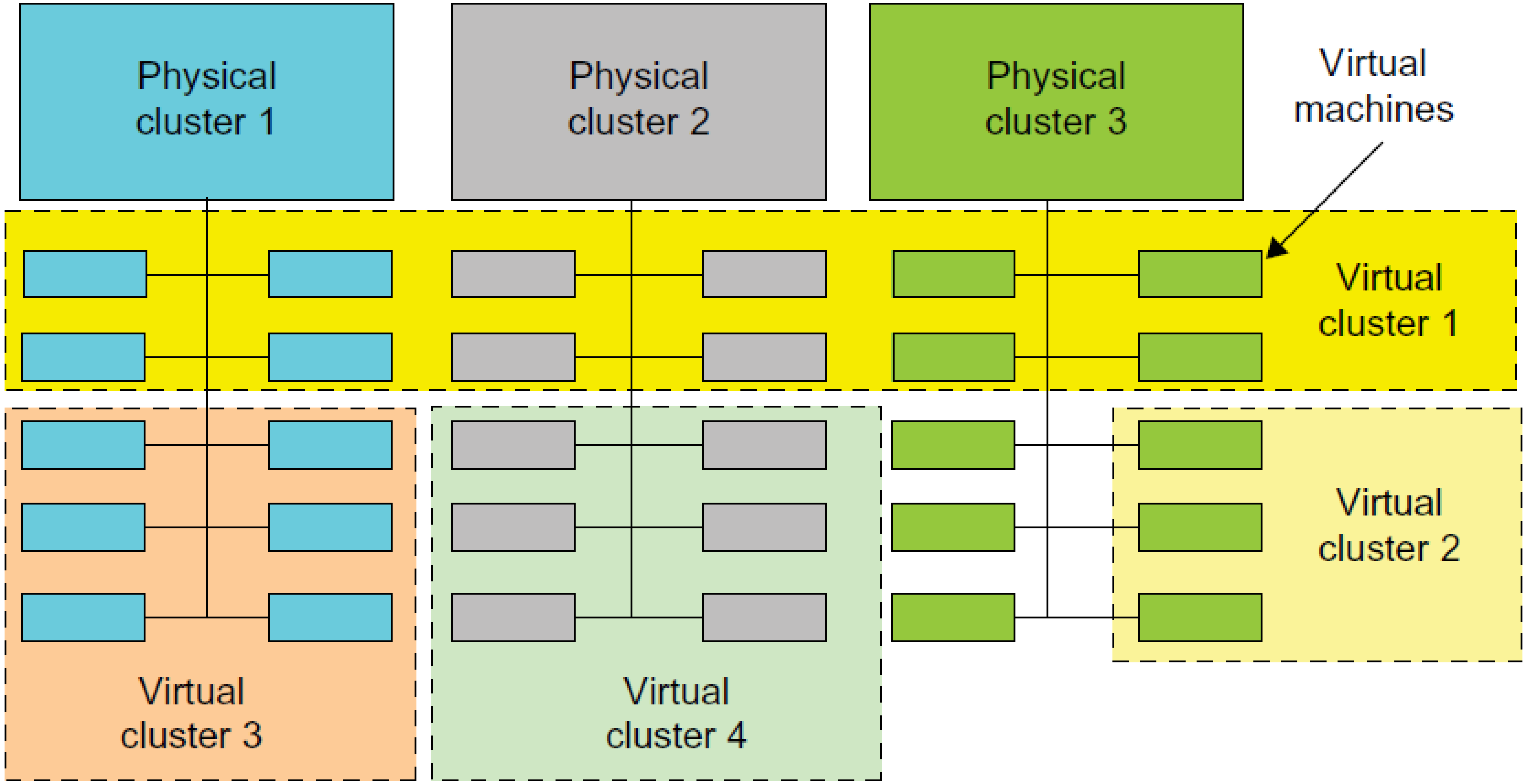
I/O Virtualization.

- I/O virtualization involves managing the routing of I/O requests between virtual devices and the shared physical hardware. Three ways to implement I/O virtualization:
- **Full device emulation**: All the functions of a device or bus infrastructure, such as device enumeration, identification, interrupts, and DMA, are replicated in software. This software is located in the VMM and acts as a virtual device. The I/O access requests of the guest OS are trapped in the VMM which interacts with the I/O devices.
- **Para-virtualization:** is typically used in Xen. It is also known as the split driver model consisting of a frontend driver and a backend driver. The frontend driver is running in Domain U and the backend driver is running in Domain 0. They interact with each other via a block of shared memory. The frontend driver manages the I/O requests of the guest OSes and the backend driver is responsible for managing the real I/O devices and multiplexing the I/O data of different VMs. Although para-I/O-virtualization achieves better device performance than full device emulation, it comes with a higher CPU overhead.
- **Direct I/O:** lets the VM access devices directly. It can achieve close-to-native performance without high CPU costs. However, current direct I/O virtualization implementations focus on networking for mainframes.

VIRTUAL CLUSTERS AND RESOURCE MANAGEMENT

Physical versus Virtual Clusters.

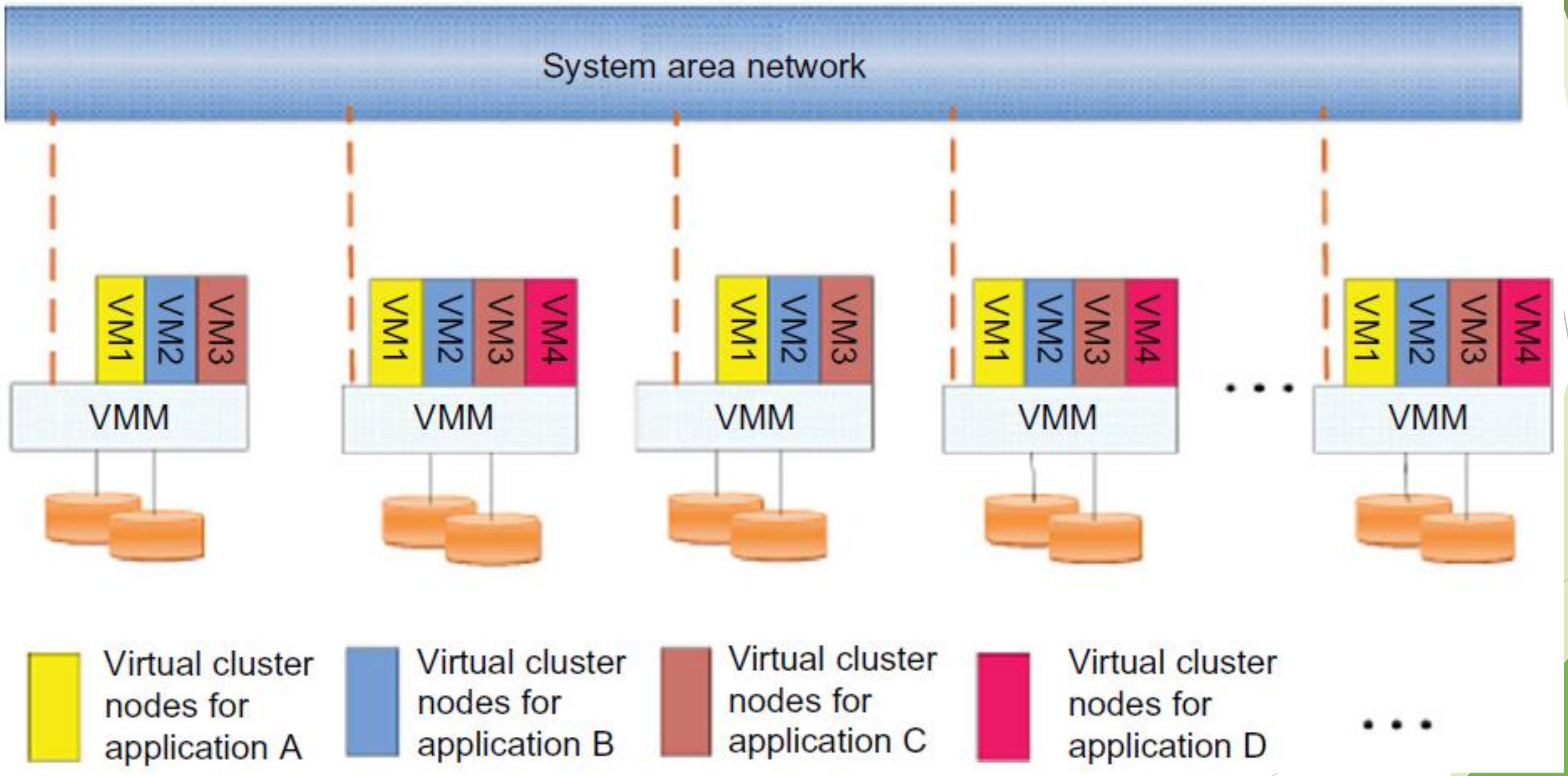
- Virtual clusters are built with VMs installed at distributed servers from one or more physical clusters. The VMs in a virtual cluster are interconnected logically by a virtual network across several physical networks.



VIRTUAL CLUSTERS AND RESOURCE MANAGEMENT

Physical versus Virtual Clusters.

The concept of a virtual cluster based on application partitioning or customization:


System area network
VM1
VM2
VM3
VMM
VM1
VM2
VM3
VM4
VMM
VM1
VM2
VM3
VMM
VM1
VM2
VM3
VM4
VMM
. . .
VM1
VM2
VM3
VM4
VMM
Virtual cluster nodes for application A
Virtual cluster nodes for application B
Virtual cluster nodes for application C
Virtual cluster nodes for application D
. . .

VIRTUAL CLUSTERS AND RESOURCE MANAGEMENT

Physical versus Virtual Clusters.

The provisioning of VMs to a virtual cluster is done dynamically to have the following interesting properties:

- The virtual cluster nodes can be either physical or virtual machines. Multiple VMs running with different OSes can be deployed on the same physical node.
- A VM runs with a guest OS, which is often different from the host OS, that manages the resources in the physical machine, where the VM is implemented.
- The purpose of using VMs is to consolidate multiple functionalities on the same server. This will greatly enhance server utilization and application flexibility.
- VMs can be colonized (replicated) in multiple servers for the purpose of promoting distributed parallelism, fault tolerance, and disaster recovery.
- The size (number of nodes) of a virtual cluster can grow or shrink dynamically, similar to the way an overlay network varies in size in a peer-to-peer (P2P) network.
- The failure of any physical nodes may disable some VMs installed on the failing nodes. But the failure of VMs will not pull down the host system.

VIRTUAL CLUSTERS AND RESOURCE MANAGEMENT

Live VM Migration Steps and Performance Effects.

There are four ways to manage a virtual cluster:

- A guest-based manager, by which the cluster manager resides on a guest system. In this case, multiple VMs form a virtual cluster. For example, openMosix is an open source Linux cluster running different guest systems on top of the Xen hypervisor. Another example is Sun’s cluster Oasis, an experimental Solaris cluster of VMs supported by a VMware VMM.
- A cluster manager on the host systems. The host-based manager supervises the guest systems and can restart the guest system on another physical machine. A good example is the VMware HA system that can restart a guest system after failure.

These two cluster management systems are either guest-only or host-only, but they do not mix.

- An independent cluster manager on both the host and guest systems. This will make infrastructure management more complex.
- An integrated cluster on the guest and host systems. This means the manager must be designed to distinguish between virtualized resources and physical resources. Various cluster management schemes can be greatly enhanced when VM life migration is enabled with minimal overhead.

VIRTUAL CLUSTERS AND RESOURCE MANAGEMENT

Live VM Migration Steps and Performance Effects.

- VMs can be live-migrated from one physical machine to another;
- in case of failure, one VM can be replaced by another VM. Virtual clusters can be applied in **computational grids, cloud platforms, and high-performance computing (HPC) systems**. The major attraction of this scenario is that virtual clustering provides dynamic resources that can be quickly put together upon user demand or after a node failure.
- In particular, virtual clustering plays a key role in cloud computing. When a VM runs a live service, it is necessary to make a trade-off to ensure that the migration occurs in a manner that minimizes all three metrics.
- The motivation is to design a live VM migration scheme with negligible downtime, the lowest network bandwidth consumption possible, and a reasonable total migration time.
- Furthermore, we should ensure that the migration will not disrupt other active services residing in the same host through resource contention (e.g., CPU, network bandwidth).

VIRTUAL CLUSTERS AND RESOURCE MANAGEMENT

Live VM Migration Steps and Performance Effects.

A VM can be in one of the following four states:

- An inactive state is defined by the virtualization platform, under which the VM is not enabled.
- An active state refers to a VM that has been instantiated at the virtualization platform to perform a real task.
- A paused state corresponds to a VM that has been instantiated but disabled to process a task or paused in a waiting state.
- A VM enters the suspended state if its machine file and virtual resources are stored back to the disk.

VIRTUAL CLUSTERS AND RESOURCE MANAGEMENT

Live migration of a VM consists of the following six steps:

- Steps 0 and 1: Start migration. This step makes preparations for the migration, including determining the migrating VM and the destination host. Although users could manually make a VM migrate to an appointed host, in most circumstances, the migration is automatically started by strategies such as load balancing and server consolidation.
- Steps 2: Transfer memory. Since the whole execution state of the VM is stored in memory, sending the VM's memory to the destination node ensures continuity of the service provided by the VM. All of the memory data is transferred in the first round, and then the migration controller recopies the memory data which is changed in the last round. These steps keep iterating until the dirty portion of the memory is small enough to handle the final copy. Although precopying memory is performed iteratively, the execution of programs is not obviously interrupted.
- Step 3: Suspend the VM and copy the last portion of the data. The migrating VM's execution is suspended when the last round's memory data is transferred. Other nonmemory data such as CPU and network states should be sent as well. During this step, the VM is stopped and its applications will no longer run. This "service unavailable" time is called the "downtime" of migration, which should be as short as possible so that it can be negligible to users.
- Steps 4 and 5: Commit and activate the new host. After all the needed data is copied, on the destination host, the VM reloads the states and recovers the execution of programs in it, and the service provided by this VM continues. Then the network connection is redirected to the new VM and the dependency to the source host is cleared. The whole migration process finishes by removing the original VM from the source host.

VIRTUALIZATION FOR DATA-CENTER AUTOMATION

Server Consolidation in Data Centers

- Data centers → a large number of heterogeneous workloads can run on servers at various times.
- heterogeneous workloads can be roughly divided into two categories:
  - chatty workloads: may burst at some point and return to a silent state at some other point. A web video service is an example of this, whereby a lot of people use it at night and few people use it during the day.
  - Noninteractive workloads: do not require people's efforts to make progress after they are submitted. High-performance computing is a typical example of this.
- At various stages, the requirements for resources of these workloads are dramatically different.
- However, to guarantee that a workload will always be able to cope with all demand levels, the workload is statically allocated enough resources so that peak demand is satisfied.
- Therefore, it is common that most servers in data centers are underutilized.
- A large amount of hardware, space, power, and management cost of these servers is wasted.
- Server consolidation is an approach to improve the low utility ratio of hardware resources by reducing the number of physical servers. Among several server consolidation techniques such as centralized and physical consolidation, virtualization-based server consolidation is the most powerful.

VIRTUALIZATION FOR DATA-CENTER AUTOMATION

Server Consolidation in Data Centers

- Data centers need to optimize their resource management. Yet these techniques are performed with the granularity of a full server machine, which makes resource management far from well optimized. Server virtualization enables smaller resource allocation than a physical machine.
- In general, the use of VMs increases resource management complexity. This causes a challenge in terms of how to improve resource utilization as well as guarantee QoS in data centers. In detail, server virtualization has the following side effects:
- Consolidation enhances hardware utilization. Many underutilized servers are consolidated into fewer servers to enhance resource utilization. Consolidation also facilitates backup services and disaster recovery.
- This approach enables more agile provisioning and deployment of resources. In a virtual environment, the images of the guest OSes and their applications are readily cloned and reused.
- The total cost of ownership is reduced. In this sense, server virtualization causes deferred purchases of new servers, a smaller data-center footprint, lower maintenance costs, and lower power, cooling, and cabling requirements.
- This approach improves availability and business continuity. The crash of a guest OS has no effect on the host OS or any other guest OS. It becomes easier to transfer a VM from one server to another, because virtual servers are unaware of the underlying hardware.

VIRTUALIZATION FOR DATA-CENTER AUTOMATION

To automate datacenter operations, several factor should be considered:

- Resource scheduling,
- Architectural support,
- Power management,
- Automatic or autonomic resource management,
- Performance of analytical models
- Dynamic CPU allocation