Operating Systems: Internals and Design Principles

Chapter 14 Virtual Machines

Ninth Edition
By William Stallings

Virtual Machines (VM)

- Virtualization technology enables a single PC or server to simultaneously run multiple operating systems or multiple sessions of a single OS
- A machine with virtualization software can host numerous applications, including those that run on different operating systems, on a single platform
- The host operating system can support a number of virtual machines, each of which has the characteristics of a particular OS and, in some versions of virtualization, the characteristics of a particular hardware platform
- The solution that enables virtualization is a ***virtual machine monitor (VMM),*** or ***hypervisor***
- This software sits between the hardware and the VMs acting as a resource broker

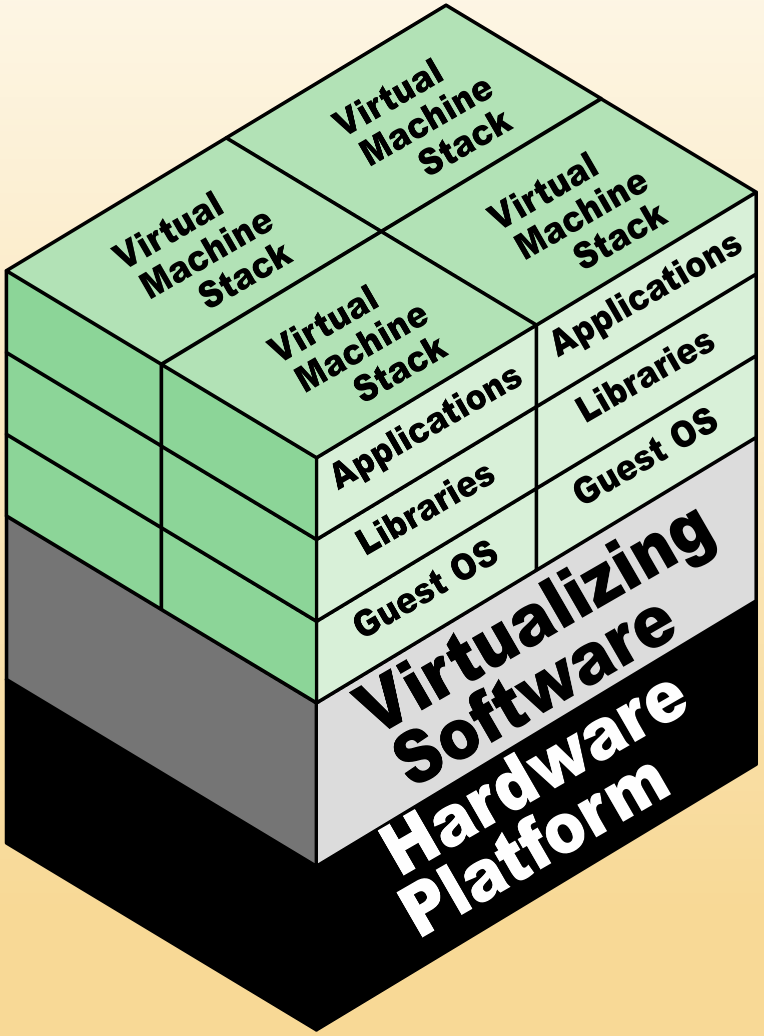


Figure 14.1 Virtual Machine Concept

Key Reasons for Using Virtualization

- We can summarize the key reasons the organizations use virtualization as follows:
- Legacy hardware
  - Applications built for legacy hardware can still be run by virtualizing the legacy hardware, enabling the retirement of the old hardware
- Rapid deployment
  - A new VM may be deployed in a matter of minutes
- Versatility
  - Hardware usage can be optimized by maximizing the number of kinds of applications that a single computer can handle
- Consolidation
  - A large-capacity or high-speed resource can be used more efficiently by sharing the resource among multiple applications simultaneously
- Aggregating
  - Virtualization makes it easy to combine multiple resources in to one virtual resource, such as in the case of storage virtualization
- Dynamics
  - Hardware resources can be easily allocated in a dynamic fashion, enhancing load balancing and fault tolerance
- Ease of management
  - Virtual machines facilitate deployment and testing of software
- Increased availability
  - Virtual machine hosts are clustered together to form pools of compute resources

Hypervisors

A Virtual Machine is a software construct that mimics the characteristics of a physical server

It is configured with some number of processors, some amount of RAM, storage resources, and connectivity through the network ports

Once the VM is created it can be powered on like a physical server, loaded with an operating system and software solutions, and utilized in the manner of a physical server

Unlike a physical server, this virtual server only sees the resources it has been configured with, not all of the resources of the physical host itself

The hypervisor facilitates the translation and I/O from the virtual machine to the physical server devices and back again to the correct virtual machine

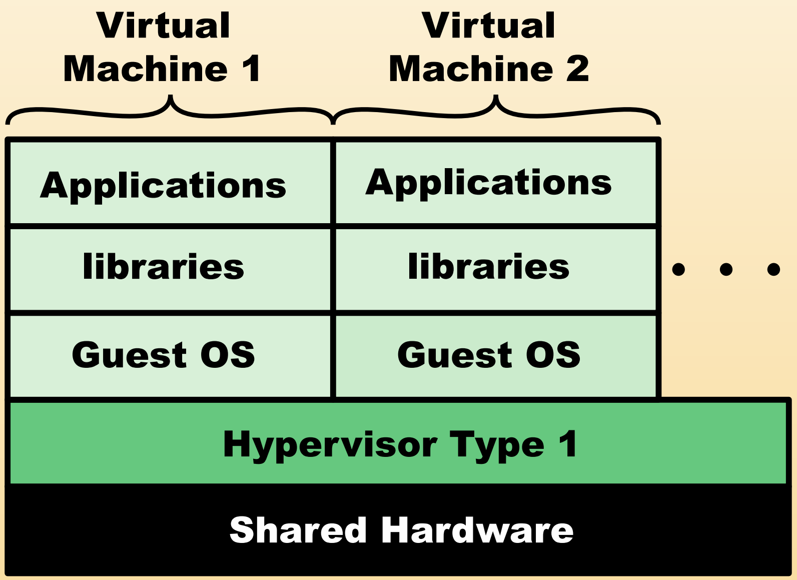
Hypervisors

A VM instance is defined in files:

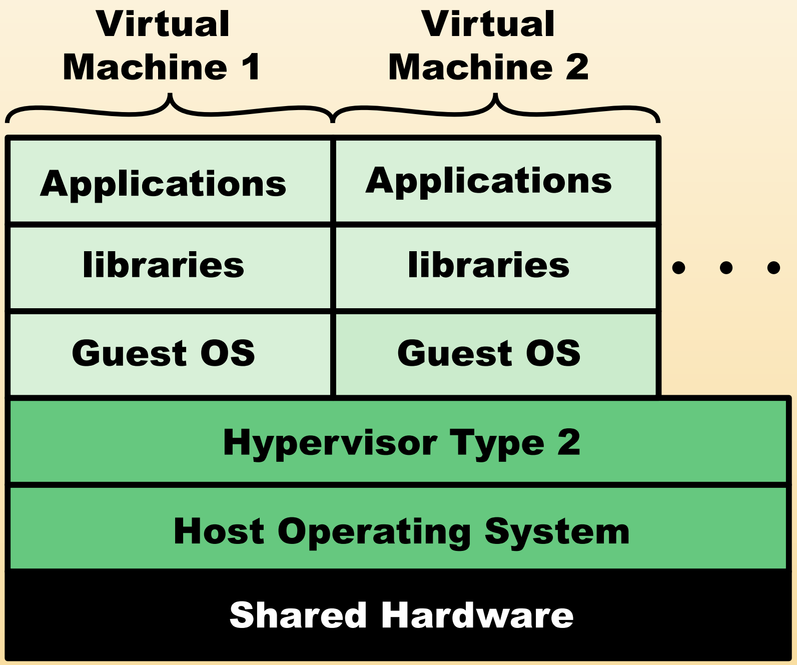
- Configuration file describes the attributes of the virtual machine
- It contains the server definition, how many virtual processors (vCPUs) are allocated to this virtual machine, how much RAM is allocated, which I/O devices the VM has access to, how many network interface cards (NICs) are in the virtual server, and more
- It also describes the storage that the VM can access
- When a virtual machine is powered on, or instantiated, additional files are created for logging, for memory paging, and other functions
- Since VMs are already files, copying them produces not only a backup of the data but also a copy of the entire server, including the operating system, applications, and the hardware configuration itself

Hypervisor Functions

- The principal functions performed by a hypervisor are:
  - Execution management of VMs
  - Devices emulation and access control
  - Execution of privileged operations by hypervisor for guest VMs
  - Management of VMs (also called VM lifecycle management
  - Administration of hypervisor platform and hypervisor software



(a) Type 1 Hypervisor



(b) Type 2 Hypervisor

Figure 14.2 Type 1 and Type 2 Hypervisors

Paravirtualization

- A software assisted virtualization technique that uses specialized APIs to link virtual machines with the hypervisor to optimize their performance
- The operating system in the virtual machine, Linux or Microsoft Windows, has specialized paravirtualization support as part of the kernel, as well as specific paravirtualization drivers that allow the OS and hypervisor to work together more efficiently with the overhead of the hypervisor translations
- Support has been offered as part of many of the general Linux distributions since 2008

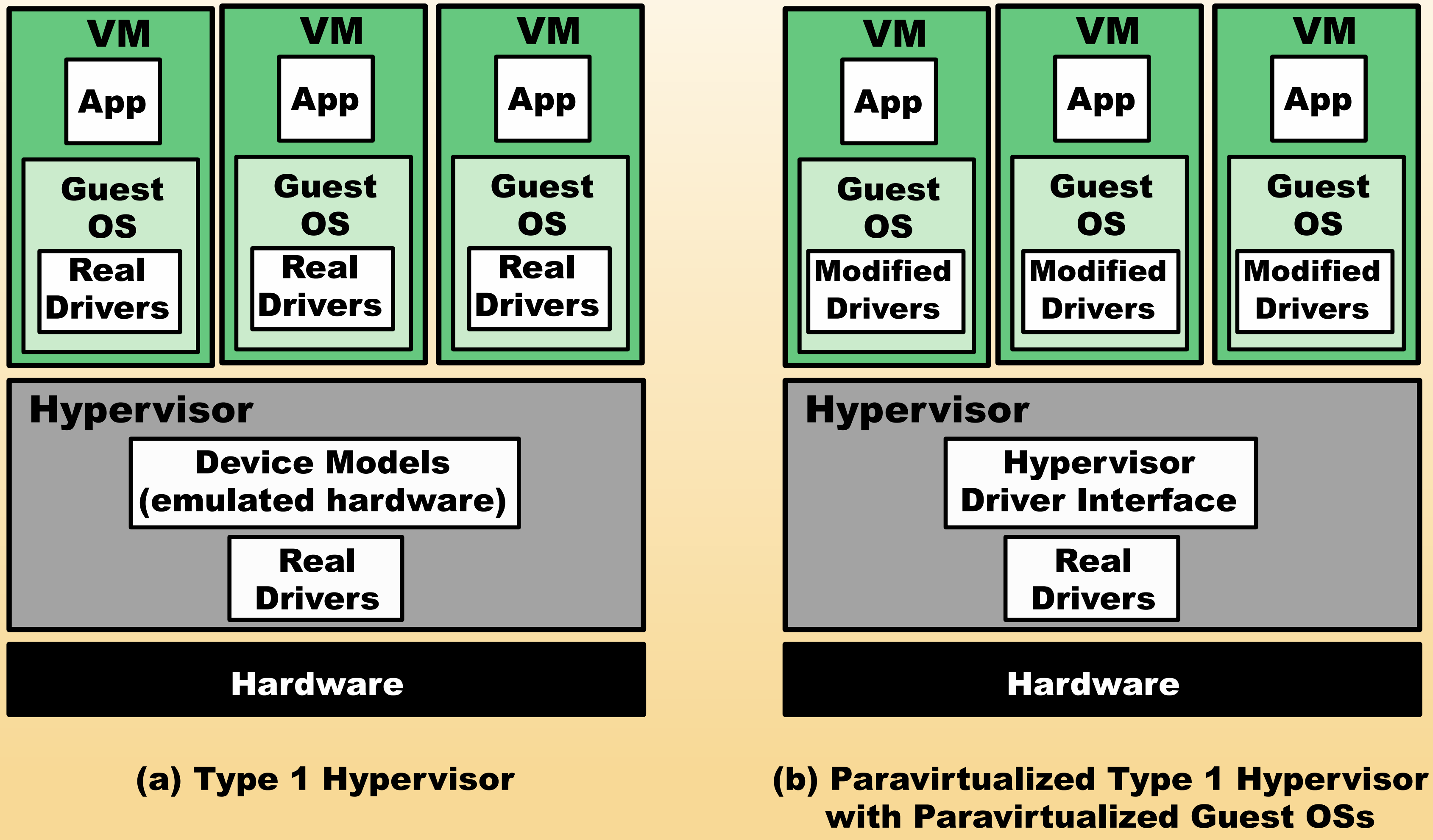


Figure 14.3 Paravirtualization

Hardware-Assisted Virtualization

- Processor manufacturers AMD and Intel added functionality to their processors to enhance performance with hypervisors
- AMD-V and Intel's VT-x designate the hardware assisted virtualization extensions that the hypervisors can take advantage of during processing
- Intel processors offer an extra instruction set called Virtual Machine Extensions (VMX)
- By having some of these instructions as part of the processor, the hypervisors no longer need to maintain these functions as part of the processor

Virtual Appliance

- A virtual appliance is standalone software that can be distributed as a virtual machine image
- It consists of a packaged set of applications and guest OS
- It is independent of hypervisor or processor architecture, and can run on either a type 1 or type 2 hypervisor
- Virtual appliances are becoming a de-facto means of software distribution and have created a need for "the virtual appliance vendor"
- A recent and important development is the ***security virtual appliance*** (SVA)
  - The SVA is a security tool that performs the function of monitoring and protecting the other VMs and is run outside of those VMs in a specially security-hardened VM
  - The SVA obtains its visibility into the state of a VM as well as the network traffic between VMs, and between VMs and the hypervisor, through the *virtual machine introspection* API of the hypervisor
  - Advantages of SVA
    - Not vulnerable to a flaw in the Guest OS
    - Independent of the virtual network configuration and does not have to be reconfigured every time the virtual network configuration changes due to migration of VMs or change in connectivity among VMs resident on the hypervisor host

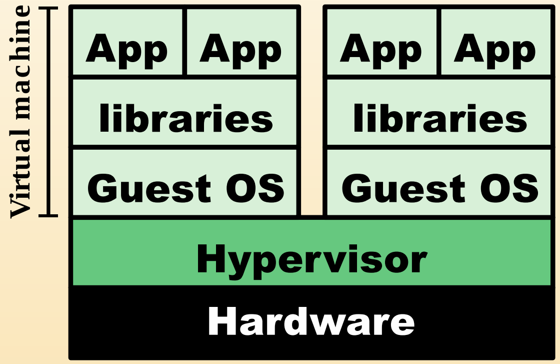
Container Virtualization

Container virtualization is a relatively recent approach to virtualization

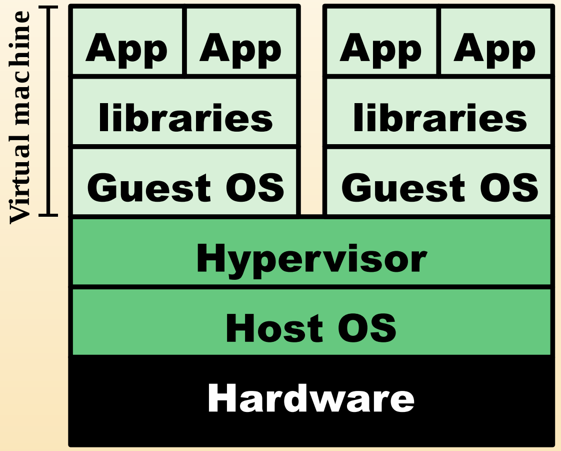
In this approach, software, known as a ***virtualization container,*** runs on top of the host OS kernel and provides an isolated execution environment for applications

Unlike hypervisor-based VMs, containers do not aim to emulate physical servers; instead, all containerized applications on a host share a common OS kernel

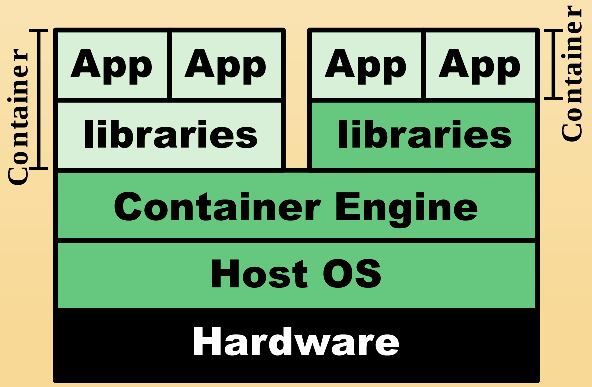
This eliminates the resources needed to run a separate OS for each application and can greatly reduce overhead



(a) Type 1 Hypervisor



(b) Type 2 Hypervisor



(c) Container

Figure 14.4 Comparison of Virtual Machines and Containers

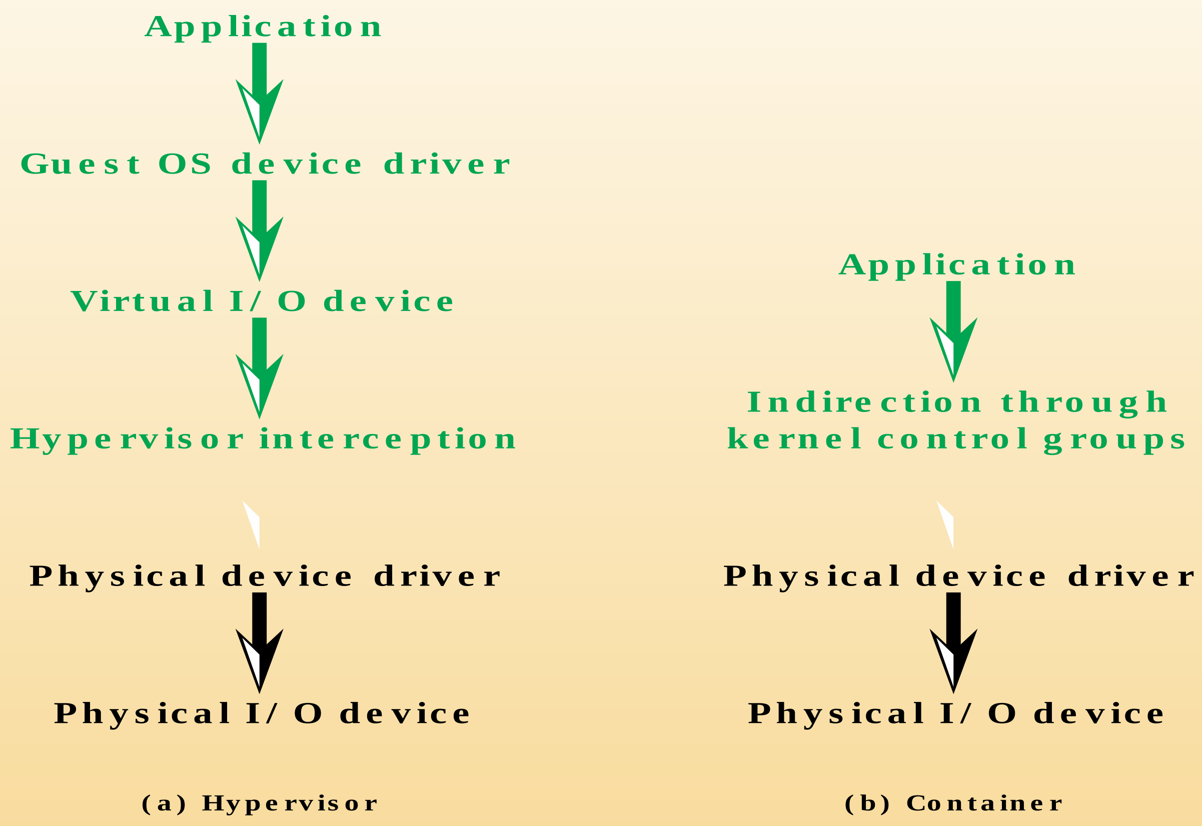


Figure 14.5 Data Flow for I/O Operation via Hypervisor and Container

Virtual containers are feasible due to resource control and process isolations as explained using techniques such as the kernel control group. This approach allows system resources being shared between multiple instances of isolated containers.

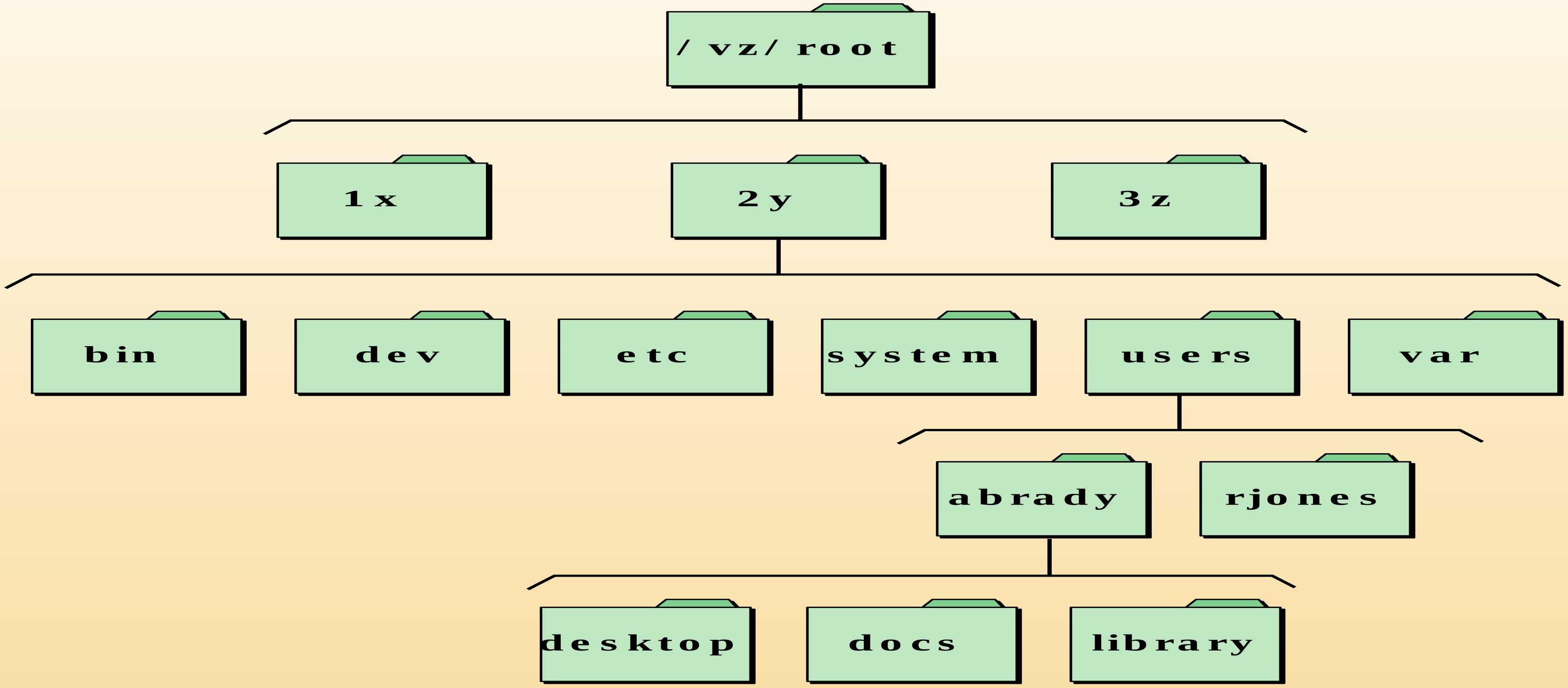


Figure 14.6 OpenVZ File scheme

As part of the isolation of a container, each container must maintain its own isolated file system. The specific features vary from one container product to another, but the essential principals are the same.

As an example, we look at the container file system used in OpenVZ.

Microservices

A concept related to containers is that of microservice.

- NIST SP 800-180 (*NIST Definition of Microservices, Application Containers and System Virtual Machines)* defines a microservice as:

  "a basic element that results from the architectural decomposition of an application's components into loosely coupled patterns consisting of self-contained services that communicate with each other using a standard communication protocol 219 and a set of well-defined APIs, independent of any vendor, product, or technology"

Microservices

Two key advantages of microservices are:

- Microservices implement much smaller deployable units, which then enables the user to push out updates or do features and capabilities much more quickly
  - This coincides with continuous delivery practices, where the goal is to push out small units without having to create a monolithic system
- Mocroservices also support precise scalability
  - Because a microservice is a section of a much larger application, it can easily be replicated to create multiple instances, and spread the load for just that one small piece of the application

Docker

- Provides a simpler and more standardized way to run containers
- Docker container also runs in Linux
- One of the reasons the Docker container is more popular compared to competing containers is its ability to load a container image on a host operating system in a simple and quick manner
- Docker containers are stored in the cloud as images and called upon for execution by users when needed in a simple way
- The principal components of Docker are:
  - Docker image
  - Docker client
  - Docker host
  - Docker engine
  - Docker machine
  - Docker registry
  - Docker hub

Processor Issues

- In a virtual environment there are two main strategies for providing processor resources:
  - Emulate a chip as software and provide access to that resource
    - Examples of this method are QEMU and the Android Emulator in the Android SDK
  - Provide segments of processing time on the physical processors (pCPUs) of the virtualization host to the virtual processors of the virtual machines hosted on the physical server
    - This is how most of the virtualization hypervisors offer processor resources to their guests

Processor Allocation

- When applications are migrated to virtual environments, the number of virtual processors allocated to their virtual machines needs to be determined

The number of processors a server has is one of the more important metrics when sizing a server

Moore's law provides processors that would be four times faster than those on the original physical server

If the consolidation estimate utility cannot be run, there are a number of good practices in place

There are tools available that will monitor resource (processor, memory, network, and storage I/O) usage on the physical server and then make recommendations for the optimum VM sizing

- One basic rule during VM creation is to begin with one vCPU and monitor the application's performance
- Another good practice is not to overallocate the number of vCPUs in a VM

Ring O

- Native operating systems manage hardware by acting as the intermediary between application code requests and the hardware
- One key function of the operating system is to help prevent malicious or accidental system calls from disrupting the applications or the operating system itself
- Protection rings describe level of access or privilege inside of a computer system and many operating systems and processor architectures take advantage of this security model
- The most trusted layer is often called Ring 0 (zero) and is where the operating system kernel works and can interact directly with hardware
- Hypervisors run in Ring 0 controlling hardware access for the virtual machines they host

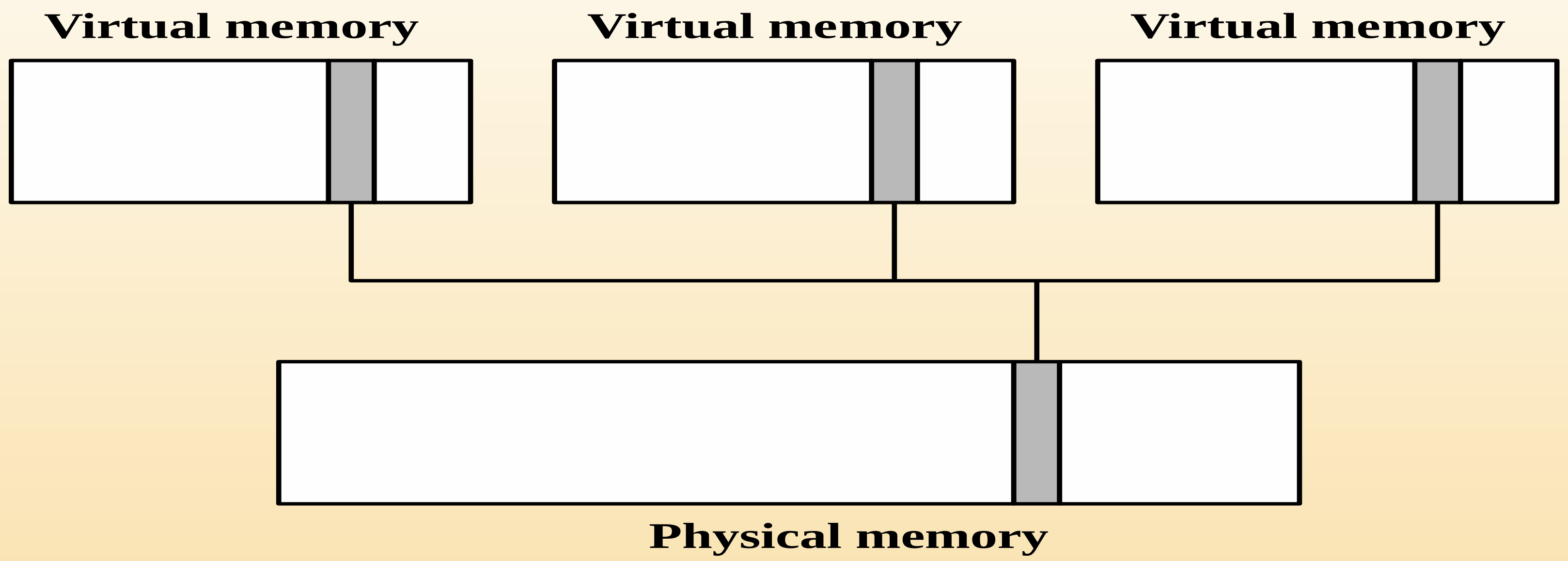


Figure 14.7 Page Sharing

Like the number of vCPUs, the amount of memory allocated to a virtual machine is one of the more crucial configuration choices; in fact, memory resources are usually the first bottleneck that virtual infrastructures reach as they grow. Also, like the virtualization of processors, memory usage in virtual environments is more about the management of the physical resource rather than the creation of a virtual entity. As with a physical server, a virtual machine needs to be configured with enough memory to function efficiently by providing space for the operating system and applications.

Memory Management

- Since hypervisor manages page sharing, the virtual machine operating systems are unaware of what is happening in the physical system
- Ballooning
  - The hypervisor activates a balloon driver that (virtually) inflates and presses the guest operating system to flush pages to disk
  - Once the pages are cleared, the balloon driver deflates and the hypervisor can use the physical memory for other VMs
- Memory overcommit
  - The capability to allocate more memory than physically exists on a host

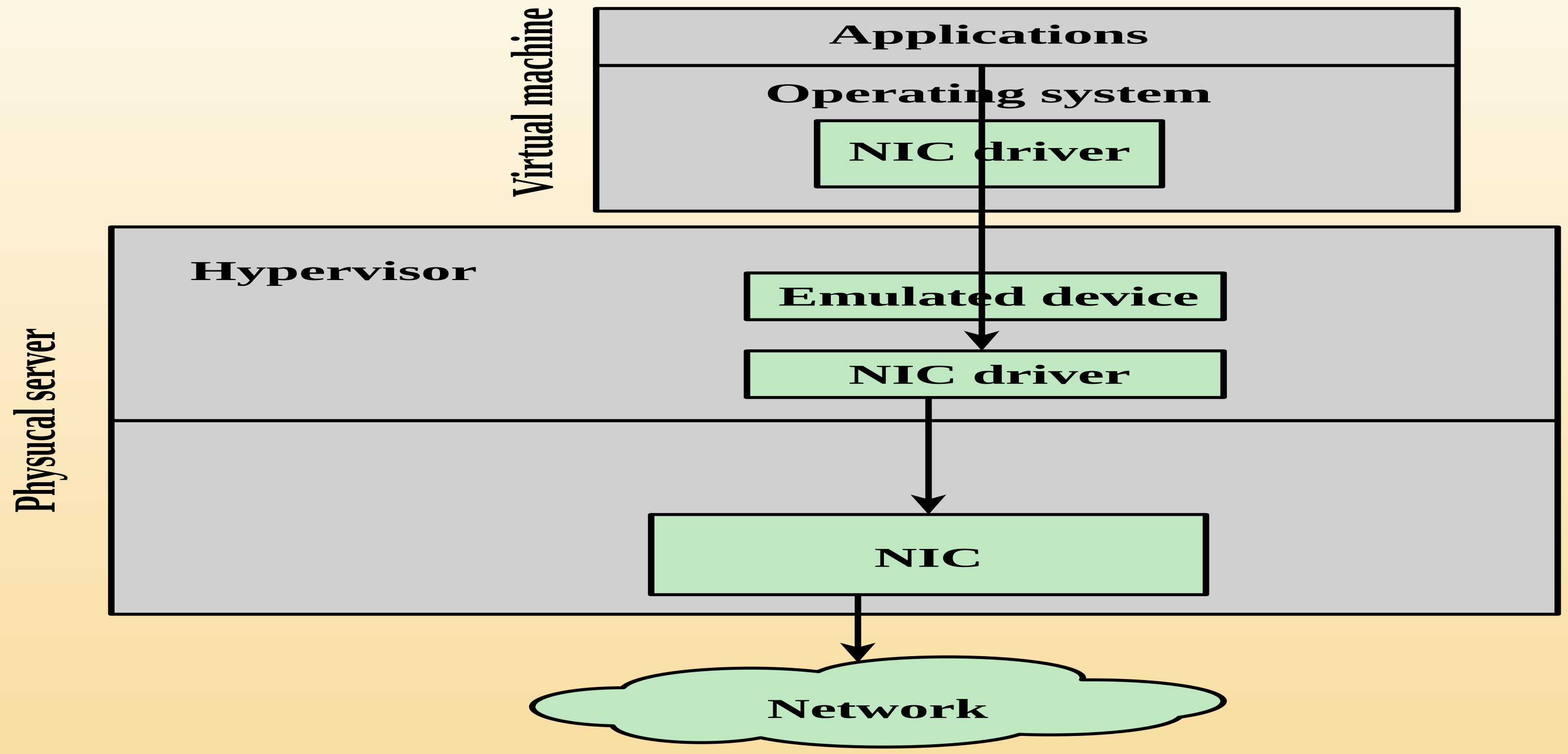


Figure 14.8 I/O in a Virtual Environment

Application performance is often directly linked to the bandwidth that a server has been allocated. Whether it is storage access that has been bottlenecked or constrained traffic to the network, either case will cause an application to be perceived as underperforming. In this way, during the virtualization of workloads, I/O virtualization is a critical item.

I/O Management

- An advantage of virtualizing the workload's I/O path enables hardware independence by abstracting vendor-specific drivers to more generalized versions that run on the hypervisor
- This abstraction enables:
  - Live migration, which is one of virtualization's greatest availability strengths
  - The sharing of aggregate resources, such as network paths
- The memory overcommit capability is another benefit of virtualizing the I/O of a VM
- The trade-off for this is that the hypervisor is managing all the traffic and requires processor overhead
  - This was an issue in the early days of virtualization but now faster multicore processors and sophisticated hypervisors have addressed this concern

Performance Technologies

A faster processor enables the hypervisor to perform its I/O management functions more quickly, and also speeds the rate at which the guest processor processing is done. Explicit hardware changes for virtualization support also improve performance.

I/OAT

I/O Acceleration Technology

Offered by Intel

A physical subsystem that moves memory copies via direct memory access (DMA) from the main processor to this specialized portion of the motherboard

TOE

TCP Offload Engine

Removes the TCP/IP processing from the server processor entirely to the NIC

LRO

Large Receive Offload

Aggregates incoming packets into bundles for more efficient processing

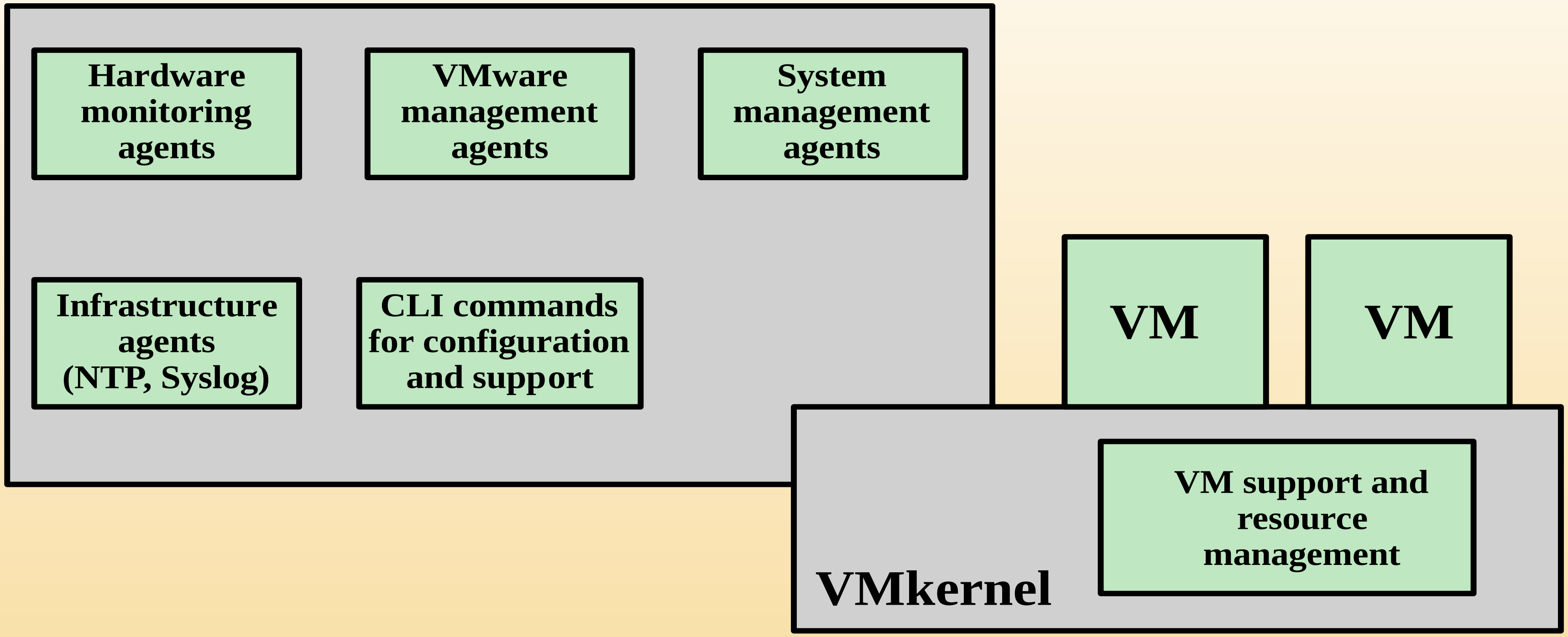
LSO

Large Segment Offload

Allows the hypervisor to aggregate multiple outgoing TCP/IP packets and has the NIC hardware segment them into separate packets

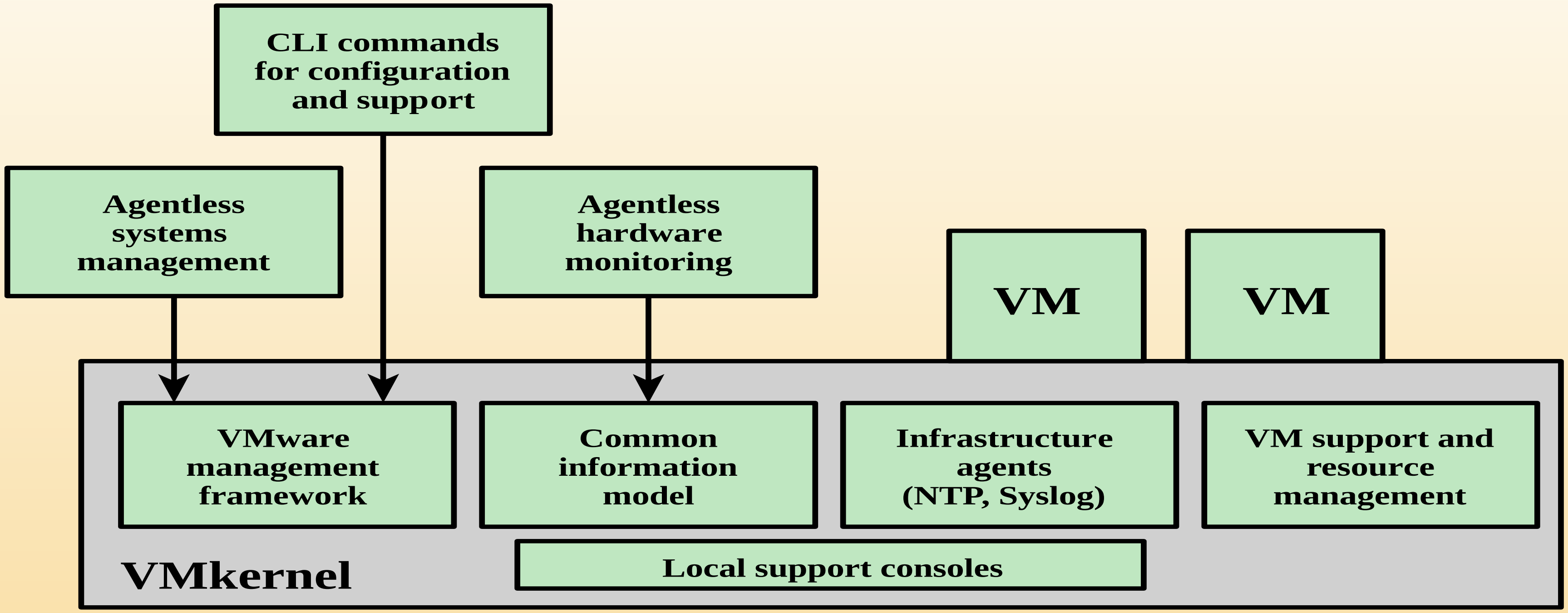
VMware ESXi

- A commercially available hypervisor from VMware that provides users a Type-1, or bare-metal, hypervisor to host virtual machines on their servers
- VMware developed their initial x86-based solutions in the late 1990s and were the first to deliver a commercial product to the marketplace
- This first-to-market timing, coupled with continuous innovations, has kept VMware firmly on top in market share


Hardware monitoring agents
VMware management agents
System management agents
Infrastructure agents (NTP, Syslog)
CLI commands for configuration and support
VM
VM
VM support and resource management
VMkernel

(a) ESX

The virtualization kernel (VMkernel) is the core of the hypervisor and performs all of the virtualization functions. In earlier releases of ESX (Figure 14.9a), the hypervisor was deployed alongside a Linux installation that served as a management layer.


CLI commands for configuration and support
Agentless systems management
Agentless hardware monitoring
VM
VM
VMware management framework
Common information model
Infrastructure agents (NTP, Syslog)
VM support and resource management
VMkernel
Local support consoles

(a) ESXi

Figure 14.9 ESX and ESXi

This new architecture, dubbed ESXi, the "i" for integrated, has all of the management services as part of the VMkernel (Figure 14.9b). This provides a smaller and much more secure package than before. Current versions are in the neighborhood of about 100MB. This small size allows server vendors to deliver hardware with ESXi already available on flash memory in the server.

VMware ESXi Features

Storage VMotion	Permits the relocation of the data files that compose a virtual machine, while that virtual machine is in use
Fault Tolerance	Creates a lockstep copy of a virtual machine on a different host --- if the original host suffers a failure, the virtual machine's connections get shifted to the copy without interrupting users or the application they are using
Site Recovery Manager	Uses various replication technologies to copy selected virtual machines to a secondary site in the case of a data center disaster
Storage and Network I/O Control	Allows an administrator to allocate network bandwidth in a virtual network in a very granular manner
Distributed Resource Scheduler (DRS)	Intelligently places virtual machines on hosts for startup and can automatically balance the workloads via VMotion based on business policies and resource usage

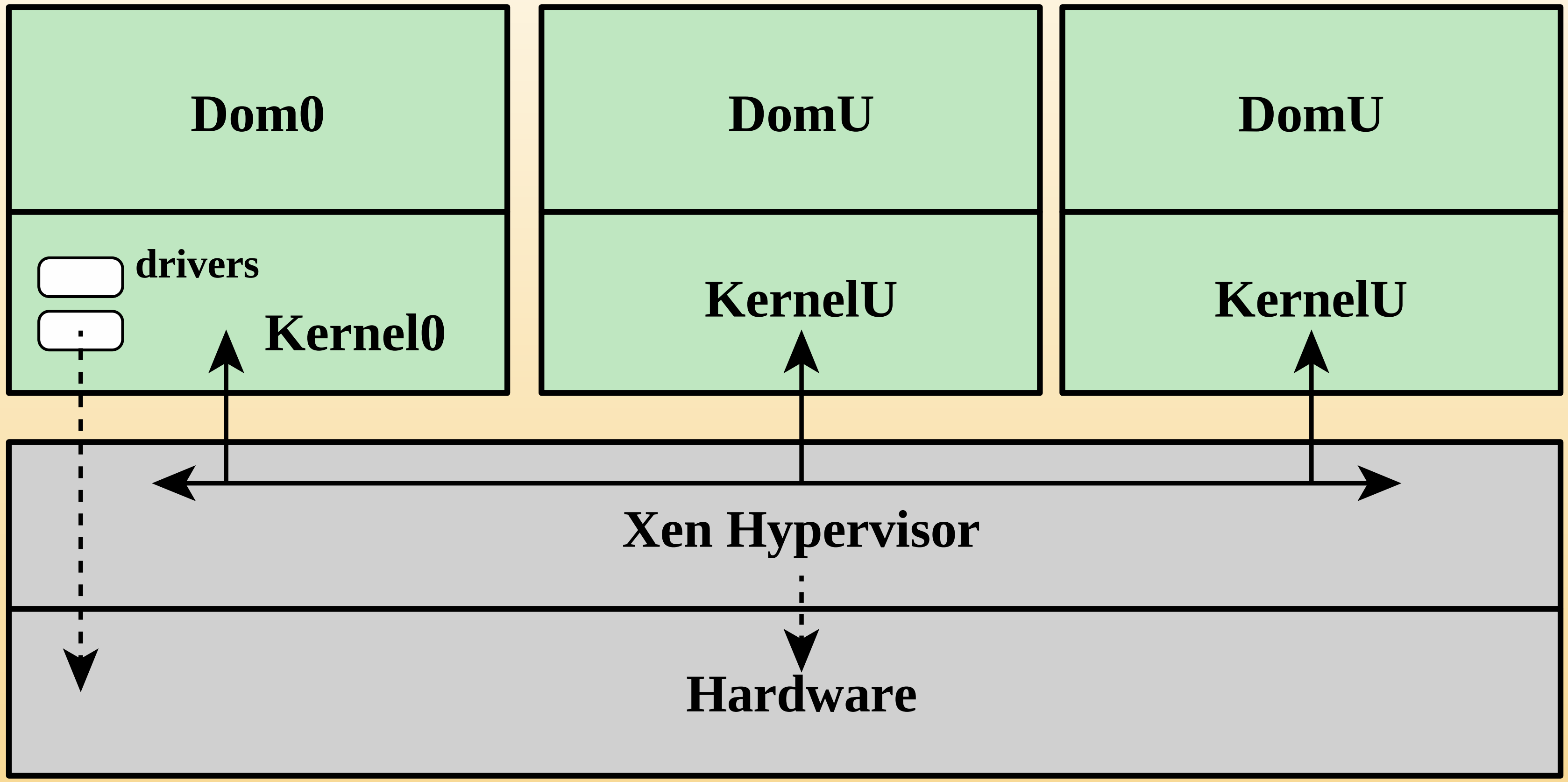


Figure 14.10 Xen

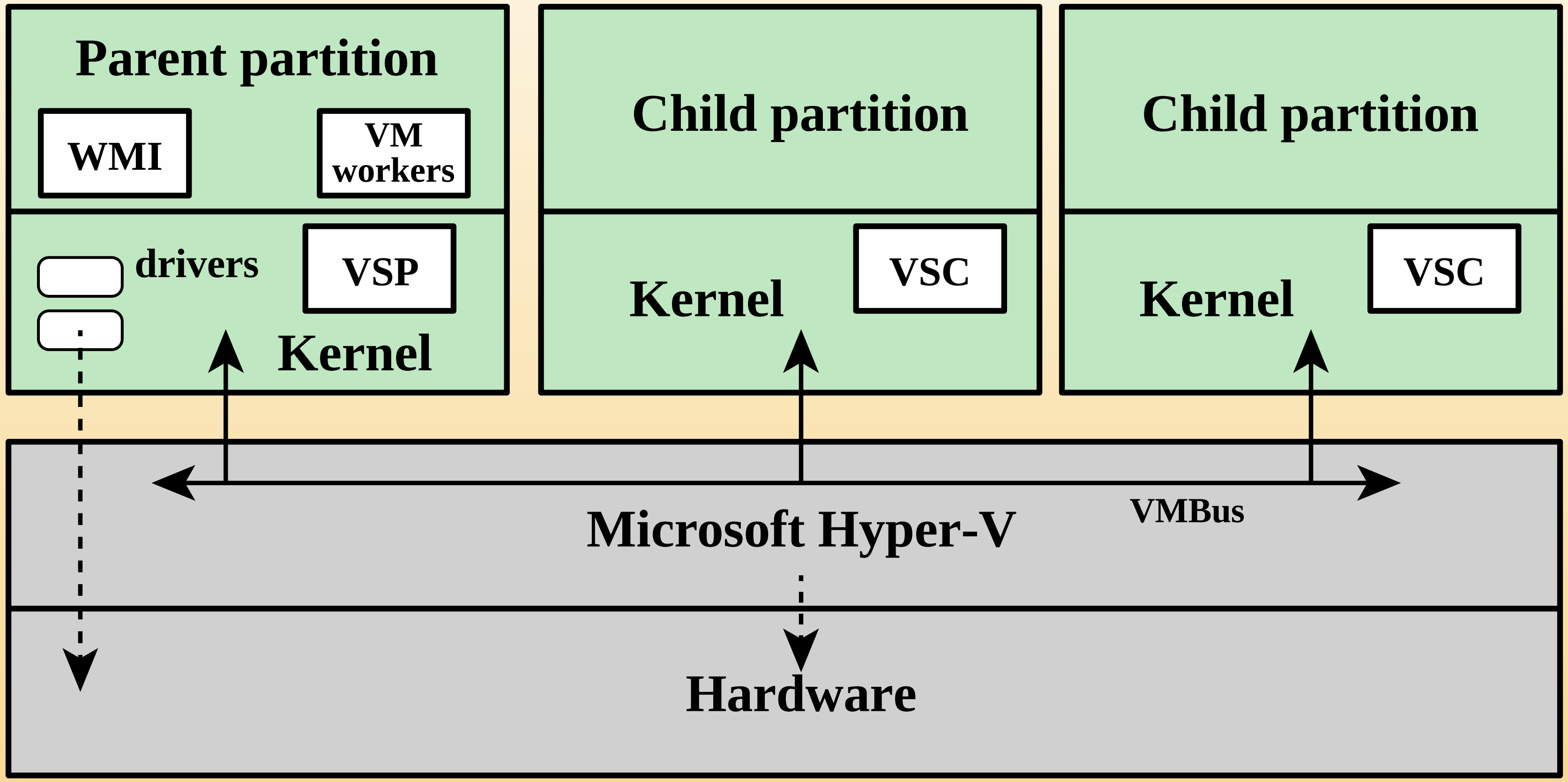


Figure 14.11 Hyper-V

Java VM

- The goal of a Java Virtual Machine (JVM) is to provide a runtime space for a set of Java code to run on any operating system staged on any hardware platform without needing to make code changes to accommodate the different operating systems or hardware
- The JVM can support multiple threads
- Promises “Write Once, Run Anywhere”
- The JVM is described as being an abstract computing machine consisting of:
  - An instruction set
  - A program counter register
  - A stack to hold variables and results
  - A heap for runtime data and garbage collection
  - A method area for code and constants

Linux VServer

- Linux VServer is an open-source, fast, lightweight approach to implementing virtual machines on a Linux server
- Only a single copy of the Linux kernel is involved
- VServer consists of a relatively modest modification to the kernel plus a small set of OS userland tools
- The VServer Linux kernel supports a number of separate virtual servers
- The kernel manages all system resources and tasks, including process scheduling, memory, disk space, and processor time

Architecture

- Each virtual server is isolated from the others using Linux kernel capabilities
- The isolation involves four elements:
  - chroot
    - A UNIX or Linux command to make the root directory (/) become something other than its default for the lifetime of the current process
    - This command provides file system isolation
  - chcontext
    - Linux utility that allocates a new security context and executes commands in that context
    - Each virtual server has its own execution context that provides process isolation
  - chbind
    - Executes a command and locks the resulting process and its children into using a specific IP address
    - System call provides network isolation
  - capablities
    - Refers to a partitioning of the privilege available to a root user
    - Each virtual server can be assigned a limited subset of the root user's privileges which provides root isolation

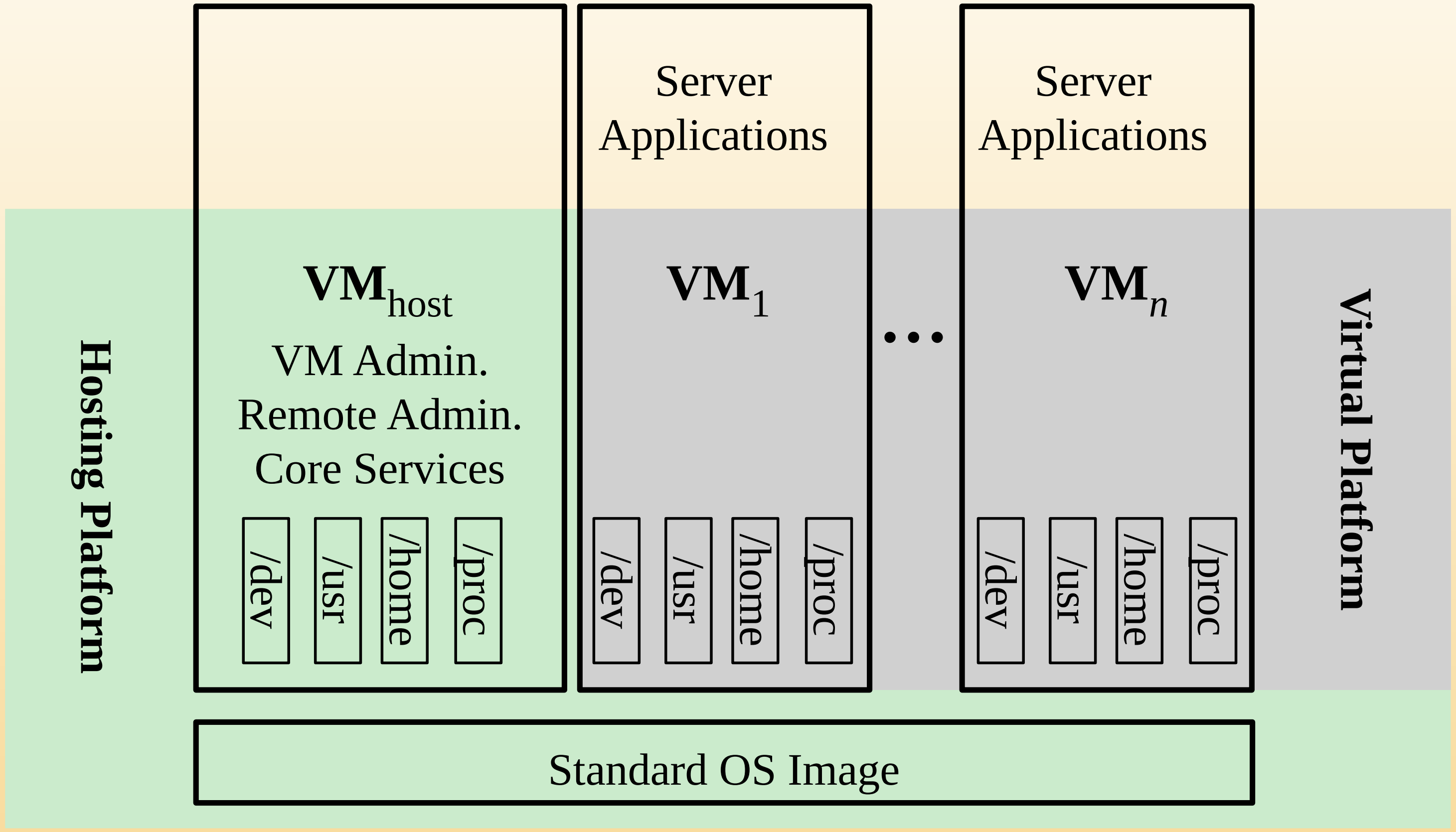
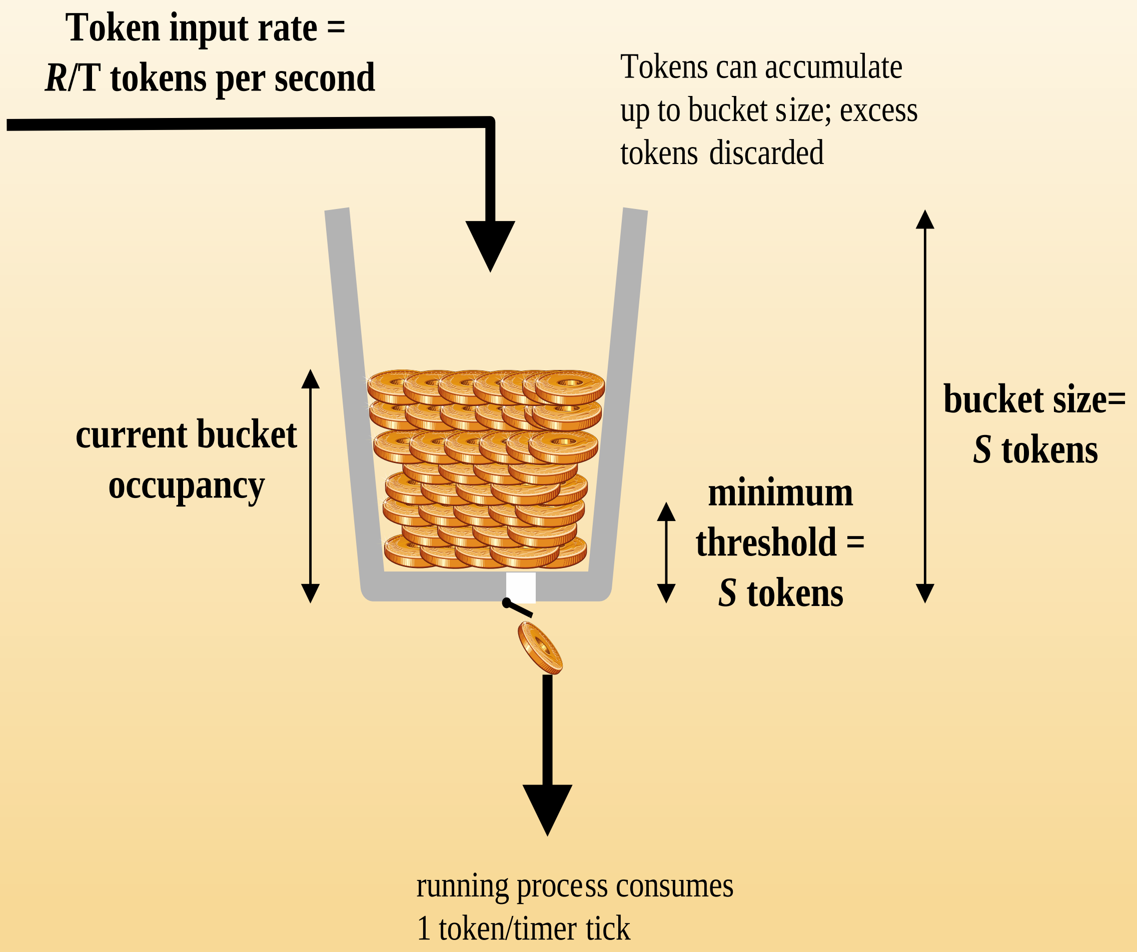


Figure 14.12 shows the general architecture of Linux VServer. VServer provides a shared, virtualized OS image, consisting of a root file system, and a shared set of system libraries and kernel services. Each VM can be booted, shut down, and rebooted independently.

Figure 14.12 Linux VServer Architecture


Token input rate =
R/T tokens per second
Tokens can accumulate up to bucket size; excess tokens discarded
current bucket occupancy
minimum threshold = *S* tokens
bucket size= *S* tokens
running process consumes 1 token/timer tick

The Linux VServer virtual machine facility provides a way of controlling VM use of processor time. VServer overlays a token bucket filter (TBF) on top of the standard Linux schedule. The purpose of the TBF is to determine how much of the processor execution time (single processor, multiprocessor, or multicore) is allocated to each VM.

Figure 14.13 Linux VServer Token Bucket Scheme

Summary

- Virtual machine concepts
- Hypervisors
  - Hypervisors
  - Paravirtualization
  - Hardware-assisted virtualization
  - Virtual appliance
- Processor issues
- Memory management
- I/O management
- VMware ESXi
- Container virtualization
  - Kernel control groups
  - Container concepts
  - Container file system
  - Microservices
  - Docker
- Microsoft Hyper-V and Xen variants
- Java VM
- Linux VServer virtual machine architecture
  - Architecture
  - Process scheduling