SYNGRESS®

BEST DAMN Server Virtualization BOOK PERIOD

Everything You Need to Know About Virtualization and Leading Virtual Machine Products

- Complete Configurations for VMware, Microsoft Virtual Server, and Xen
- How to Install and Configure a Virtual Machine, Gold Builds, and Clones; Administer a Virtual Infrastructure; Program for the Virtual Infrastructure; Modify Virtual Machines; Migrate Physical Machines; and Troubleshoot
- Detailed Instructions on Planning and Implementing a Server Consolidation: Project Requirements, Sample Forms and Templates, and Virtual Migration Strategies

David Rule
Rogier Dittner

The Best Damn Server Virtualization Book Period

Kris Buytaert
Rogier Dittner
Juan R. Garcia
Twan Grotenhuis
David E. Hart
Andy Jones
Kenneth Majors
Al Muller
David Payne
Jeremy Pries
Rami Rosen
David Rule Jr.
Paul Summitt
Matthijs ten Seldam
David E. Williams

This page intentionally left blank

KEY	SERIAL NUMBER
001	HJIRTCV764
002	PO9873D5FG
003	829KM8NJH2
004	PLTY567AWQ
005	CVPLQ6WQ23
006	VBP965T5T5
007	HJJJ863WD3E
008	2987GVTWMK
009	629MP5SDJT
010	IMWQ295T6T

PUBLISHED BY
Syngress Publishing, Inc.
Elsevier, Inc.
30 Corporate Drive
Burlington, MA 01803

The Best Damn Server Virtualization Book Period

Printed in the United States of America
1 2 3 4 5 6 7 8 9 0

ISBN 13: 978-1-59749-217-1

Publisher: Amorette Pedersen
Acquisitions Editor: Andrew Williams
Page Layout and Art: SPi
Cover Designer: Michael Kavish

For information on rights, translations, and bulk sales, contact Matt Pedersen, Commercial Sales Director and Rights, at Syngress Publishing; email m.pedersen@elsevier.com.

This page intentionally left blank

Contributing Authors

Kris Buytaert is Founder and CTO of X-Tend. He is a longtime Linux, Security, and Open Source consultant. He has consulting and development experience with multiple enterprise-level clients and government agencies. In addition to his high-level technical experience, he is also a team leader who likes to deliver his projects on time. He is a contributor to the Linux Documentation Project and author of various technical publications. Kris is a Red Hat Certified Engineer and is currently the maintainer of the openMosix HOWTO Web site. Kris is also a frequent speaker at Linux and OpenSource conferences. He is currently focusing on Linux clustering (both HA and HPC), virtualization, and large infrastructure management.

Rogier Dittner (MCSE NT4, 2000, 2003, MCDBA, MCT, MSF Practitioner) is a consultant at a Microsoft partner offering solutions based on Microsoft technology to customers. As a consultant he supports the sales organization and takes part in planning and designing complex Microsoft-oriented implementations.

Because of his personal interest in Microsoft products and more than 10 years' experience, he has deep technical working knowledge in a broad range of Microsoft products. Within his company he performs the leading role in operations management solutions and training.

He would like thank his wife and children for giving him the time and space to write (Pascalle, bedankt, je bent een schat!)

Juan R. Garcia is a Principal Consultant at Williams & Garcia, LLC. He provides strategic and technical consulting in legacy systems migrations, enterprise architecture, disaster recover planning, and enterprise IT resource consolidation to Williams & Garcia's customers. He specializes in open systems (UNIX/Linux), virtualization technologies (VMware, Xen, and AIX 5L), storage solutions, and RDMBS technologies. Juan's previous positions include Solutions Architect for Bellsouth, Senior Enterprise Architect for John H. Harland Co., and Technical Manager for Sun Professional Services.

Twan Grotenhuis (MCT, MCSE NT4, 2000 and 2003, MCSE+messaging 2000 and 2003, CCNA) is a consultant with Sylis Netherlands. He currently provides strategic and technical consulting to several of the Sylis customers in the Netherlands. His specialties include Microsoft Exchange and ISA architecture, design, implementation, troubleshooting and optimization. Twan has been involved in several Virtual Server 2005 projects where virtualization of physical servers was his main focus.

David E. Hart (MCSE#300790, ASE #220919, VCP #4970) is a senior consultant with Callisma. He currently provides senior-level strategic and technical consulting to all Callisma clients in the south-central region of the U.S. His specialties include virtualization technologies, Microsoft Active Directory design and implementation, emerging technology planning, collaboration architecture and design, content delivery design and implementations, enterprise operating systems troubleshooting and optimization, and desktop architecture design and implementation. David's background spans over 15 years in the industry and includes positions at one of the top five consulting firms as the "South Central Microsoft Practice and VMware Lead" for seven years, Microsoft Practice Lead and Senior Microsoft Consultant at a top three telecommunication company for five years, and Desktop Enterprise Practice Lead for a nationwide consulting firm for two years.

Andy Jones (MCSE+I, MCT, CCIA, CCEA, CCI, CCNA, CCDA, MCIW, Network+, A+,) is the Services Director for MTM Technologies, previously known as Vector ESP. He provides comprehensive solutions focused on Citrix and Microsoft technologies for clients ranging from 50 to 50,000 users, focusing mainly on architecting and deploying Access Infrastructure solutions for enterprise customers. One of Andy's primary focuses is in developing best practices, processes, and methodologies surrounding Access Infrastructure that take into consideration and integrate with virtually every part of a customer's infrastructure.

In addition to field work and business development, Andy regularly instructs Microsoft and Citrix courses. Andy holds a master's degree from Morehead State University.

Kenneth Majors (MCSE, MCSA, Project+, VMware VCP, Citrix CCEA, CCA, IBM X-Series Expert, Avaya ACA) is a consultant for Choice Solutions LLC, a systems integrator headquartered in Overland Park, KS. Choice Solutions provides IT design, project management, and support for enterprise computing systems. Kenneth is a key contributor to defining best practices for deployment and implementation of Microsoft technologies, including Windows Server, Virtual Server, and SharePoint; Citrix Presentation Server; VMware ESX; and development of documentation standards. He develops technology solutions and methodologies focused on improving client business processes. These technology solutions touch every part of a system's life cycle from assessment, blueprint, construction, and deployment on projects to operational management and strategic planning for the business process. Kenneth holds a bachelor's degree from Colorado Technical University. He currently resides in Olathe, KS, with his lovely, supportive wife, Sandy, and near their children, Tabitha and Keith, and their grandsons, Wesley and Austin.

Al Muller is a consultant for Callisma, a wholly owned subsidiary of AT&T. He has been in the IT field since 1995, getting his start as a database administrator in the Navy. In 2002 he began using VMware's GSX Server and within a year was involved in his first virtualization project. Since then, he has been an eager proponent of virtualization technology and has worked on a number of different server consolidation and virtualization projects.

He holds a bachelor's degree in English and plans on writing a series of books on the virtual evolution taking place in data centers worldwide and the tools required to leverage and support a virtual infrastructure.

David Payne is an IT enthusiast with a decade of real-world experience in the data center. David is currently CTO of Xcedex, the only U.S.-based professional services firm solely focused on virtualization solutions. David has been key in developing the virtualization practice for Xcedex Professional Services. Specifically over the last four years, David has been engaged in dozens of virtualization initiatives, providing architecture guidance and hands on services for organizations of all sizes across the United States. His practical approach has taken some of the largest US companies in finance, retail, and manufacturing beyond the marketing spin and into real results

with today's virtualization technologies. David is a VMware Authorized Consultant (VAC) and a VMware Certified Professional (VCP).

Xcedex is a VMware Premier Partner, joining this invitation-only program as one of the first 10 partners in 2004. Xcedex is recognized nationwide for its professionalism, deep knowledge of virtual infrastructure, and experience in real-world implementations. With a laser focus on virtualization consulting, Xcedex has become one of the top go-to service delivery partners for VMware, Dell, and EMC.

Jeremy Pries is a Virtualization Architect at Xcedex. He has an extensive background in computing infrastructure dating back 10 years, with experience ranging from networking and storage to security and Intel based operating systems. Jeremy's current focus is 100% on virtualization technologies, gaining valuable experience on some of the largest ESX implementations. Jeremy's specialty is filling gaps in management tools to speed project timelines and increase accuracy. His expertise has made him one of the most sought after Xcedex architects. Jeremy is a VMware Authorized Consultant (VAC) and a VMware Certified Professional (VCP).

Xcedex is a VMware Premier Partner, joining this invitation-only program as one of the first 10 partners in 2004. Xcedex is recognized nationwide for its professionalism, deep knowledge of virtual infrastructure, and experience in real-world implementations. With a laser focus on virtualization consulting, Xcedex has become one of the top go-to service delivery partners for VMware, Dell, and EMC.

Rami Rosen (B.Sc, Computer Science, Technion—Israel High Institute of Technology) is working as a Linux and Open Solaris kernel programmer accompanying advanced networking and security projects. His background includes positions in Ethernet switching and Avionic operating system start-ups. His specialities include virtualization technologies and kernel networking internals. His articles are occasionally published in the *Linux Journal* and the lwn.net Web site.

David Rule Jr. (VMware VCP, VAC, MCP, Citrix CCEA, CCA) is a Senior Consultant for Choice Solutions LLC, an Overland Park, KS-based systems integrator that provides IT design, project management,

and support for enterprise computing systems. David's primary role is in developing virtualization strategies for Choice Solutions' clients. The strategies include the virtualization of both servers and storage, including the development of best practice and standards documentation. The combination of virtual servers and storage provides systems with enterprise-class stability, performance, and scalability. These technologies are implemented to provide clients with environments that facilitate management of and increase the performance of day-to-day operations while also making them more reliable and cost-effective.

David is currently working toward completing a degree from Park University in Computer Information Systems Management. He resides in Shawnee, KS, with his wife, Kristine, and their two wonderful children, Christian and Collin.

Paul Summitt (MCSE, CCNA, MCP+I, MCP) holds a master's degree in mass communication. Paul has served as a network, an Exchange, and a database administrator, as well as a Web and application developer. Paul has written on virtual reality and Web development and has served as technical editor for several books on Microsoft technologies. Paul lives in Columbia, MO, with his life and writing partner, Mary.

Matthijs ten Seldam (MCSE, CISSP) is a principal consultant with the infrastructure technologies group at Microsoft Consulting Services. His expertise focuses on virtualization, platform management and deployment, security, and networking. One of his specialties is automation of management tasks through various interfaces like WMI and COM using languages like VBScript and C#.

He has developed a technical training on Virtual Server 2005 R2 and delivers this to customers and partners. He currently provides consulting to enterprise customers, delivers technical workshops, and runs early adoption programs of the next generation of virtualization products like Virtual Server 2005 R2 Service Pack 1 and System Center Virtual Machine Manager.

David E. Williams is a principal at Williams & Garcia, LLC, a consulting practice based in Atlanta, GA, specializing in effective enterprise infrastructure

solutions. He specializes in the delivery of advanced solutions for x86 and x64 environments. Because David focuses on cost containment and reduction of complexity, virtualization technologies have played a key role in his recommended solutions and infrastructure designs. David has held several IT leadership positions in various organizations, and his responsibilities have included the operations and strategy of Windows, open systems, mainframe, storage, database, and data center technologies and services. He has also served as a senior architect and an advisory engineer for Fortune 1000 organizations, providing strategic direction on technology infrastructures for new enterprise-level projects.

David studied Music Engineering Technology at the University of Miami, and he holds MCSE+I, MCDBA, VCP, and CCNA certifications. When not obsessed with corporate infrastructures, he spends his time with his wife and three children.

Contents

Chapter 1

An Introduction to Virtualization

Solutions in this chapter:

- **What Is Virtualization?**
- **Why Virtualize?**
- **How Does Virtualization Work?**
- **Types of Virtualization**
- **Common Use Cases for Virtualization**

- ☑ **Summary**
- ☑ **Solutions Fast Track**
- ☑ **Frequently Asked Questions**

Introduction

Virtualization is one of those buzz words that has been gaining immense popularity with IT professionals and executives alike. Promising to reduce the ever-growing infrastructure inside current data center implementations, virtualization technologies have cropped up from dozens of software and hardware companies. But what exactly is it? Is it right for everyone? And how can it benefit your organization?

Virtualization has actually been around more than three decades. Once only accessible by the large, rich, and prosperous enterprise, virtualization technologies are now available in every aspect of computing, including hardware, software, and communications, for a nominal cost. In many cases, the technology is freely available (thanks to open-source initiatives) or included for the price of products such as operating system software or storage hardware.

Well suited for most inline business applications, virtualization technologies have gained in popularity and are in widespread use for all but the most demanding workloads. Understanding the technology and the workloads to be run in a virtualized environment is key to every administrator and systems architect who wishes to deliver the benefits of virtualization to their organization or customers.

This chapter will introduce you to the core concepts of server, storage, and network virtualization as a foundation for learning more about Xen. This chapter will also illustrate the potential benefits of virtualization to any organization.

What Is Virtualization?

So what exactly is virtualization? Today, that question has many answers. Different manufacturers and independent software vendors coined that phrase to categorize their products as tools to help companies establish virtualized infrastructures. Those claims are not false, as long as their products accomplish some of the following key points (which are the objectives of any virtualization technology):

- Add a layer of abstraction between the applications and the hardware
- Enable a reduction in costs and complexity
- Provide the isolation of computer resources for improved reliability and security
- Improve service levels and the quality of service
- Better align IT processes with business goals
- Eliminate redundancy in, and maximize the utilization of, IT infrastructures

While the most common form of virtualization is focused on server hardware platforms, these goals and supporting technologies have also found their way into other critical—and expensive—components of modern data centers, including storage and network infrastructures.

But to answer the question "What is virtualization?" we must first discuss the history and origins of virtualization, as clearly as we understand it.

The History of Virtualization

In its conceived form, virtualization was better known in the 1960s as time sharing. Christopher Strachey, the first Professor of Computation at Oxford University and leader of the Programming Research Group, brought this term to life in his paper *Time Sharing in Large Fast Computers*. Strachey, who was a staunch advocate of maintaining a balance between practical and theoretical work in computing, was referring to what he called multi-programming. This technique would allow one programmer to develop a program on his console while another programmer was debugging his, thus avoiding the usual wait for peripherals. Multi-programming, as well as several other groundbreaking ideas, began to drive innovation, resulting in a series of computers that burst onto the scene. Two are considered part of the evolutionary lineage of virtualization as we currently know it—the Atlas and IBM's M44/44X.

The Atlas Computer

The first of the supercomputers of the early 1960s took advantage of concepts such as time sharing, multi-programming, and shared peripheral control, and was dubbed the Atlas computer. A project run by the Department of Electrical Engineering at Manchester University and funded by Ferranti Limited, the Atlas was the fastest computer of its time. The speed it enjoyed was partially due to a separation of operating system processes in a component called the supervisor and the component responsible for executing user programs. The supervisor managed key resources, such as the computer's processing time, and was passed special instructions, or extracodes, to help it provision and manage the computing environment for the user program's instructions. In essence, this was the birth of the hypervisor, or virtual machine monitor.

In addition, Atlas introduced the concept of virtual memory, called one-level store, and paging techniques for the system memory. This core store was also logically separated from the store used by user programs, although the two were integrated. In many ways, this was the first step towards creating a layer of abstraction that all virtualization technologies have in common.

The M44/44X Project

Determined to maintain its title as the supreme innovator of computers, and motivated by the competitive atmosphere that existed, IBM answered back with the M44/44X Project. Nested at the IBM Thomas J. Watson Research Center in Yorktown, New York, the project created a similar architecture to that of the Atlas computer. This architecture was first to coin the term *virtual machines* and became IBM's contribution to the emerging time-sharing system concepts. The main machine was an IBM 7044 (M44) scientific computer and several simulated 7044 virtual machines, or 44Xs, using both hardware and software, virtual memory, and multi-programming, respectively.

Unlike later implementations of time-sharing systems, M44/44X virtual machines did not implement a complete simulation of the underlying hardware. Instead, it fostered the notion that virtual machines were as efficient as more conventional approaches. To nail that notion, IBM successfully released successors of the M44/44X project that showed this idea was not only true, but could lead to a successful approach to computing.

CP/CMS

A later design, the IBM 7094, was finalized by MIT researchers and IBM engineers and introduced Compatible Time Sharing System (CTSS). The term "compatible" refers to the compatibility with the standard batch processing operating system used on the machine, the Fortran Monitor System (FMS). CTSS not only ran FMS in the main 7094 as the primary facility for the standard batch stream, but also ran an unmodified copy of FMS in each virtual machine in a background facility. The background jobs could access all peripherals, such as tapes, printers, punch card readers, and graphic displays, in the same fashion as the foreground FMS jobs as long as they did not interfere with foreground time-sharing processors or any supporting resources.

MIT continued to value the prospects of time sharing, and developed Project MAC as an effort to develop the next generation of advances in time-sharing technology, pressuring hardware manufacturers to deliver improved platforms for their work. IBM's response was a modified and customized version of its System/360 (S/360) that would include virtual memory and time-sharing concepts not previously released by IBM. This proposal to Project MAC was rejected by MIT, a crushing blow to the team at the Cambridge Scientific Center (CSC), whose only purpose was to support the MIT/IBM relationship through technical guidance and lab activities.

The fallout between the two, however, led to one of the most pivotal points in IBM's history. The CSC team, lead by Norm Rassmussen and Bob Creasy, a defect from Project MAC, to the development of CP/CMS. In the late 1960s, the CSC developed the first successful virtual machine operating system based on fully virtualized hardware, the CP-40. The CP-67 was released as a reimplementation of the CP-40, as was later converted and implemented as the S/360-67 and later as the S/370. The success of this platform won back IBM's credibility at MIT as well as several of IBM's largest customers. It also led to the evolution of the platform and the virtual machine operating systems that ran on them, the most popular being VM/370. The VM/370 was capable of running many virtual machines, with larger virtual memory running on virtual copies of the hardware, all managed by a component called the virtual machine monitor (VMM) running on the real hardware. Each virtual machine was able to run a unique installation of IBM's operating system stably and with great performance.

Other Time-Sharing Projects

IBM's CTSS and CP/CMS efforts were not alone, although they were the most influential in the history of virtualization. As time sharing became widely accepted and recognized as an effective way to make early mainframes more affordable, other companies joined the time-sharing fray. Like IBM, those companies needed plenty of capital to fund the research and hardware investment needed to aggressively pursue time-sharing operating systems as the platform for running their programs and computations. Some other projects that jumped onto the bandwagon included

- **Livermore Time-Sharing System (LTSS)** Developed by the Lawrence Livermore Laboratory in the late 1960s as the operating system for the Control Data CDC 7600 supercomputers. The CDC 7600 running LTSS took over the title of the world's fastest computer, trumping on the Atlas computer, which suffered from a form of trashing due to inefficiencies in its implementation of virtual memory.
- **Cray Time-Sharing System (CTSS)** (This is a different CTSS; not to be confused with IBM's CTSS.) Developed for the early lines of Cray supercomputers

in the early 1970s. The project was engineered by the Los Alamos Scientific Laboratory in conjunction with the Lawrence Livermore Laboratory, and stemmed from the research that Livermore had already done with the successful LTSS operating system. Cray X-MP computers running CTSS were used heavily by the United States Department of Energy for nuclear research.

- **New Livermore Time-Sharing System** (NLTSS) The last iteration of CTSS, this was developed to incorporate recent advances and concepts in computers, such as new communication protocols like TCP/IP and LINCS. However, it was not widely accepted by users of the Cray systems and was discontinued in the late 1980s.

Virtualization Explosion of the 1990s and Early 2000s

While we have discussed a summarized list of early virtualization efforts, the projects that have launched since those days are too numerous to reference in their entirety. Some have failed while others have gone on to be popular and accepted technologies throughout the technical community. Also, while efforts have been pushed in server virtualization, we have also seen attempts to virtualize and simplify the data center, whether through true virtualization as defined by the earlier set of goals or through infrastructure sharing and consolidation.

Many companies, such as Sun, Microsoft, and VMware, have released enterprise-class products that have wide acceptance, due in part to their existing customer base. However, Xen threatens to challenge them all with their approach to virtualization. Being adopted by the Linux community and now being integrated as a built-in feature to most popular distributions, Xen will continue to enjoy a strong and steady increase in market share. Why? We'll discuss that later in the chapter. But first, back to the question… What is virtualization?

Configuring & Implementing…

Evolution of the IBM LPAR—More than Just Mainframe Technology

IBM has had a long history of Logical Partitions, or LPARs, on their mainframe product offerings, from System390 through present-day System z9 offerings. However, IBM has extended the LPAR technology beyond the mainframe, introducing it to its Unix platform with the release of AIX 5L. Beginning with AIX 5L Version 5.1, administrators could use the familiar Hardware Management Console (HMC) or the Integrated Virtualization Manager to create LPARs with virtual hardware resources (dedicated or

Continued

shared). With the latest release, AIX 5L Version 5.3, combined with the newest generation of System p with POWER5 processors, additional mainframe-derived virtualization features, such as micro-partitioning CPU resources for LPARs, became possible.

IBM's LPAR virtualization offerings include some unique virtualization approaches and virtual resource provisioning. A key component of what IBM terms the Advanced POWER Virtualization feature, is the Virtual I/O Server. Virtual I/O servers satisfy part of the VMM, called the POWER Hypervisor, role. Though not responsible for CPU or memory virtualization, the Virtual I/O server handles all I/O operations for all LPARs. When deployed in redundant LPARs of its own, Virtual I/O servers provide a good strategy to improve availability for sets of AIX 5L or Linux client partitions, offering redundant connections to external Ethernet or storage resources.

Among the I/O resources managed by the Virtual I/O servers are

- **Virtual Ethernet** Virtual Ethernet enables inter-partition communication without the need for physical network adapters in each partition. It allows the administrator to define point-to-point connections between partitions. Virtual Ethernet requires a POWER5 system with either IBM AIX 5L Version 5.3 or the appropriate level of Linux and an HMC to define the Virtual Ethernet devices.
- **Virtual Serial Adapter (VSA)** POWER5 systems include Virtual Serial ports that are used for virtual terminal support.
- **Client and Server Virtual SCSI** The POWER5 server uses SCSI as the mechanism for virtual storage devices. This is accomplished using a pair of virtual adapters; a virtual SCSI server adapter and a virtual SCSI client adapter. These adapters are used to transfer SCSI commands between partitions. The SCSI server adapter, or target adapter, is responsible for executing any SCSI command it receives. It is owned by the Virtual I/O server partition. The virtual SCSI client adapter allows the client partition to access standard SCSI devices and LUNs assigned to the client partition. You may configure virtual server SCSI devices for Virtual I/O Server partitions, and virtual client SCSI devices for Linux and AIX partitions.

The Answer: Virtualization Is…

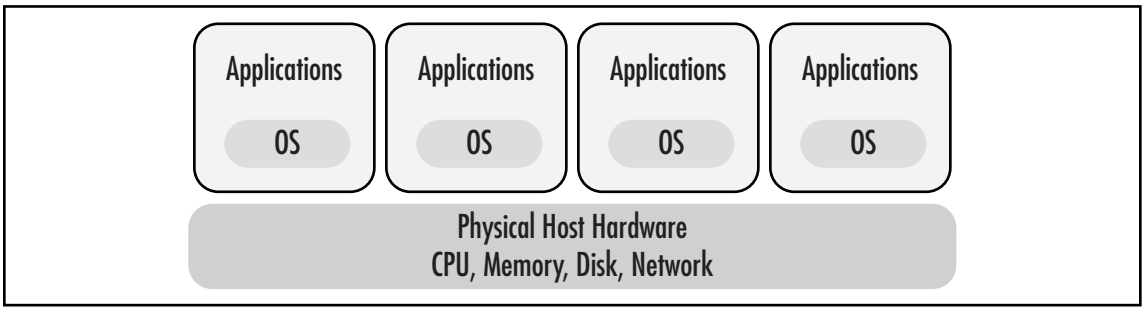
So with all that history behind us, and with so many companies claiming to wear the virtualization hat, how do we define it? In an effort to be as all-encompassing as possible, we can define virtualization as:

> A framework or methodology of dividing the resources of a computer hardware 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.

Just as it did during the late 1960s and early 1970s with IBM's VM/370, modern virtualization allows multiple operating system instances to run concurrently on a single computer, albeit much less expensive than the mainframes of those days. Each OS instance shares the available resources available on the common physical hardware, as illustrated in Figure 1.1. Software, referred to as a virtual machine monitor (VMM), controls use and access to the CPU, memory, storage, and network resources underneath.

Figure 1.1 Virtual Machines Riding on Top of the Physical Hardware



Why Virtualize?

From the mid-1990s until present day, the trend in the data center has been towards a decentralized paradigm, scaling the application and system infrastructure outward in a horizontal fashion. The trend has been commonly referred to as "server sprawl." As more applications and application environments are deployed, the number of servers implemented within the data center grows at exponential rates. Centralized servers were seen as too expensive to purchase and maintain for many companies not already established on such a computing platform. While big-frame, big-iron servers continued to survive, the midrange and entry-level server market bustled with new life and opportunities for all but the most intense use cases. It is important to understand why IT organizations favored decentralization, and why it was seen as necessary to shift from the original paradigm of a centralized computing platform to one of many.

Decentralization versus Centralization

Virtualization is a modified solution between two paradigms—centralized and decentralized systems. Instead of purchasing and maintaining an entire physical computer, and its necessary peripherals for every application, each application can be given its own operating environment, complete with I/O, processing power, and memory, all sharing their underlying physical hardware. This provides the

benefits of decentralization, like security and stability, while making the most of a machine's resources and providing better returns on the investment in technology.

With the popularity of Windows and lighter-weight open systems distributed platforms, the promise that many hoped to achieve included better return on assets and a lower total cost of ownership (TCO). The commoditization of inexpensive hardware and software platforms added additional fuel to the evangelism of that promise, but enterprises quickly realized that the promise had turned into a nightmare due to the horizontal scaling required to provision new server instances.

On the positive side, companies were able to control their fixed asset costs as applications were given their own physical machine, using the abundant commodity hardware options available. Decentralization helped with the ongoing maintenance of each application, since patches and upgrades could be applied without interfering with other running systems. For the same reason, decentralization improves security since a compromised system is isolated from other systems on the network. As IT processes became more refined and established as a governance mechanism in many enterprises, the software development life cycle (SDLC) took advantage of the decentralization of n-tier applications. Serving as a model or process for software development, SDLC imposes a rigid structure on the development of a software product by defining not only development phases (such as requirements gathering, software architecture and design, testing, implementation, and maintenance), but rules that guide the development process through each phase. In many cases, the phases overlap, requiring them to have their own dedicated n-tier configuration.

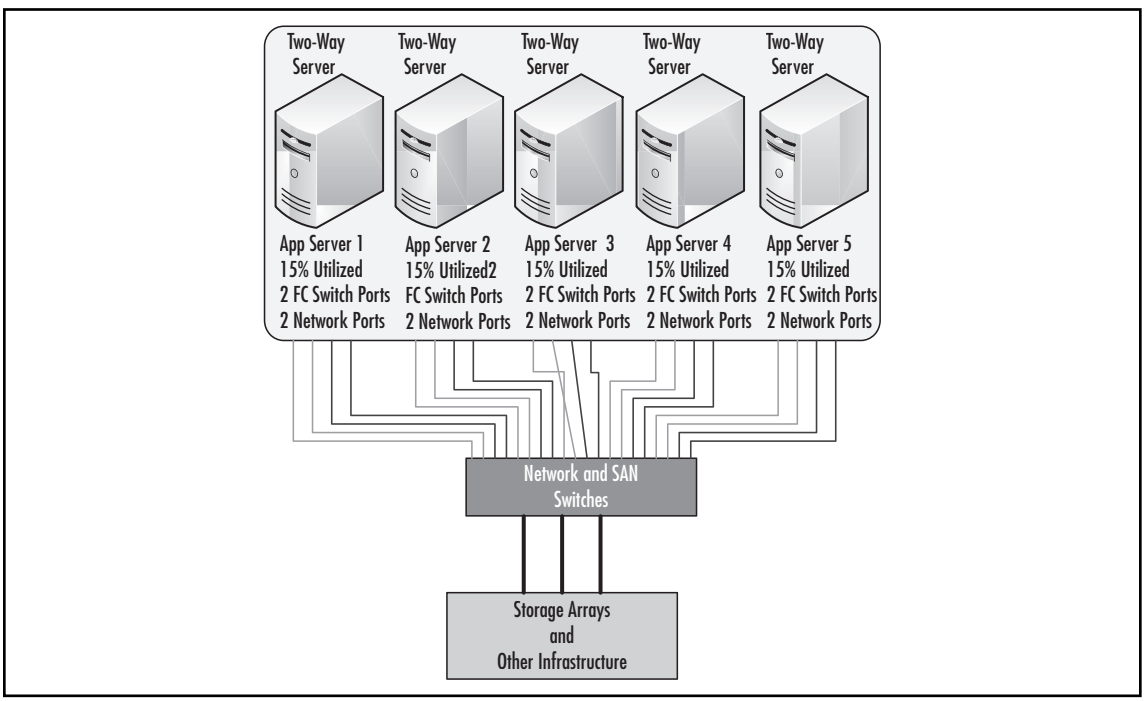
However, the server sprawl intensified, as multiple iterations of the same application were needed to support the SDLC for development, quality assurance, load testing, and finally production environments. Each application's sandbox came at the expense of more power consumption, less physical space, and a greater management effort which, together, account for up to tens (if not hundreds) of thousands of dollars in annual maintenance costs per machine. In addition to this maintenance overhead, decentralization decreased the efficiency of each machine, leaving the average server idle 85 to 90 percent of the time. These inefficiencies further eroded any potential cost or labor savings promised by decentralization.

In Table 1.1, we evaluate three-year costs incurred by Foo Company to create a decentralized configuration comprised of five two-way x86 servers with software licensed per physical CPU, as shown in Figure 1.2. These costs include the purchase of five new two-way servers, ten CPU licenses (two per server) of our application, and soft costs for infrastructure, power, and cooling. Storage is not factored in because we assume that in both the physical and virtual scenarios, the servers would be connected to external storage of the same capacity; hence, storage costs remain the same for both. The Physical Cost represents a three-year cost since most companies depreciate their capital fixed assets for 36 months. Overall, our costs are $74,950.

Table 1.1 A Simple Example of the Cost of Five Two-Way Application Servers

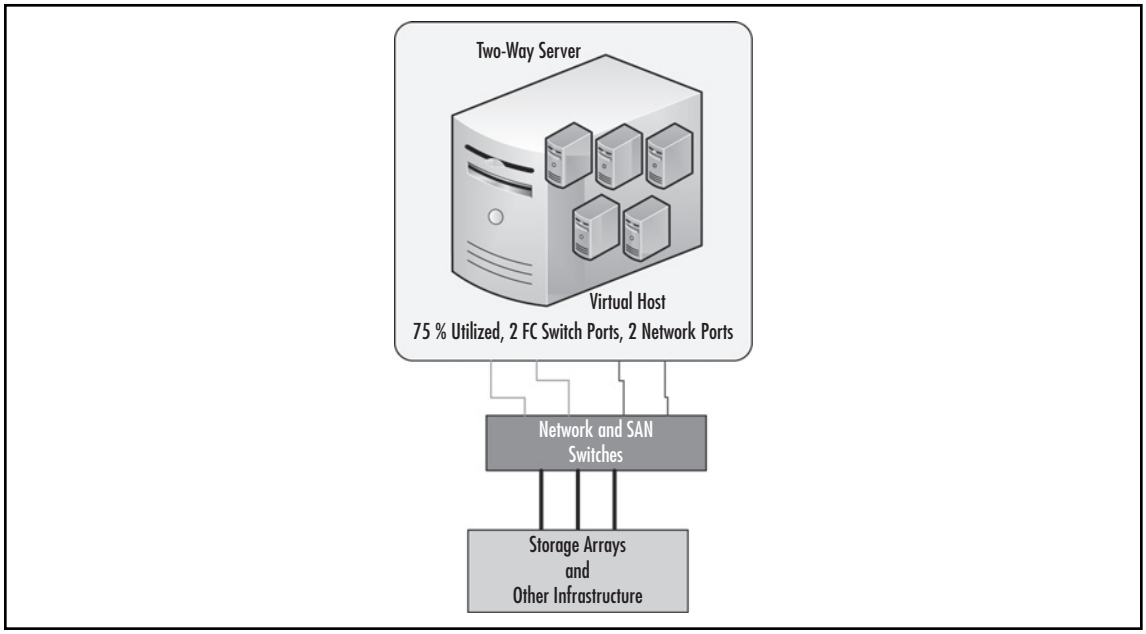
Component	Unit Cost	Physical Cost	Virtual Cost
Server hardware	$7,500.00	$37,500.00	$7,500.00
Software licenses/CPU	$2,000.00	$20,000.00	$4,000.00
Supporting infrastructure	$2,500.00	$12,500.00	$2,500.00
Power per server year	$180.00	$2,700.00	$540.00
Cooling per server year	$150.00	$2,250.00	$450.00
Total three-year costs:		$74,950.00	$16,490.00
Realized savings over three years:	**$58,460.00**		

Figure 1.2 A Decentralized Five-Server Configuration



In contrast, the table also shows a similarly configured centralized setup of five OS/application instances hosted on a single two-way server with sufficient hardware resources for the combined workload, as shown in Figure 1.3. Although savings are realized by the 5:1 reduction in server hardware, that savings is matched by the savings in software cost (5:1 reduction in physical CPUs to license), supporting infrastructure, power, and cooling.

Figure 1.3 A Centralized Five-Server Configuration



WARNING

When building the business case and assessing the financial impact of virtualization, be sure not to over-commit the hosts with a large number of virtual machines. Depending on the workload, physical hosts can manage as many as 20 to 30 virtualization machines, or as little as 4 to 5. Spend time upfront gathering performance information about your current workloads, especially during peak hours, to help properly plan and justify your virtualization strategy.

Assuming that each server would average 15-percent utilization if run on physical hardware, consolidation of the workloads into a centralized virtual is feasible. The hard and soft costs factored into the calculations more closely demonstrate the total cost of ownership in this simple model, labor excluded. It is important to note that *Supporting Infrastructure*, as denoted in the table, includes rack,

cabling, and network/storage connectivity costs. This is often overlooked; however, it is critical to include this in your cost benefit analysis since each Fibre-Channel (FC) switch port consumed could cost as much as $1,500, and each network port as much as $300. As illustrated in the figures, there are ten FC and ten network connections in the decentralized example compared to two FC and two network connections. Port costs alone would save Foo a considerable amount. As the table shows, a savings of almost 80 percent could be realized by implementing the servers with virtualization technologies.

Designing & Planning…

A Virtualized Environment Requires a Reliable, High-Capacity Network

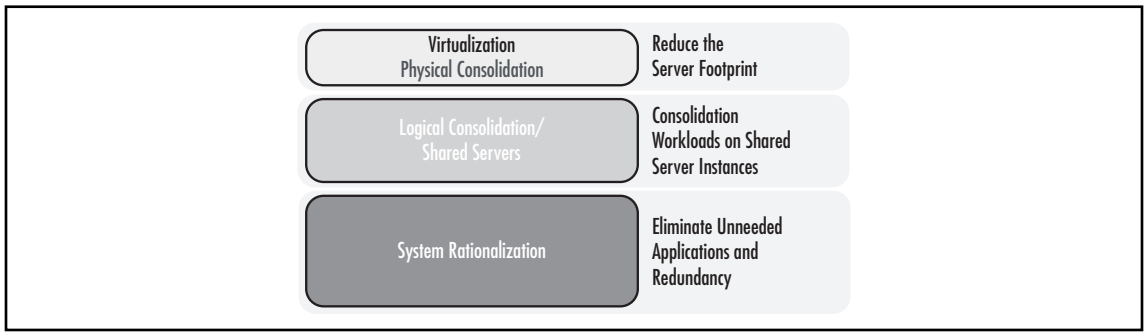
To successfully consolidate server workloads onto a virtualized environment, it is essential that all server subsystems (CPU, memory, network, and disk) can accommodate the additional workload. While most virtualization products require a single network connection to operate, careful attention to, and planning of, the networking infrastructure of a virtual environment can ensure both optimal performance and high availability.

Multiple virtual machines will increase network traffic. With multiple workloads, the network capacity needs to scale to match the requirements of the combined workloads expected on the host. In general, as long as the host's processor is not fully utilized, the consolidated network traffic will be the sum of the traffic generated by each virtual machine.

True Tangible Benefits

Virtualization is a critical part of system optimization efforts. While it could simply be a way to reduce and simplify your server infrastructure, it can also be a tool to transform the way you think about your data center as a whole. Figure 1.4 illustrates the model of system optimization. You will notice that virtualization, or physical consolidation, is the foundation for all other optimization steps, followed by logical consolidation and then an overall rationalization of systems and applications, identifying applications that are unneeded or redundant and can thus be eliminated.

Figure 1.4 Virtualization's Role in System Optimization



In Table 1.2 you will find a sample list of benefits that often help IT organization justify their movement toward a virtual infrastructure. Although each organization's circumstances are different, you only need a few of these points to apply to your situation to build a strong business case for virtualization.

Table 1.2 Benefits of Virtualization

Category	Benefit
Consolidation	Increase server utilization
	Simplify legacy software migration
	Host mixed operating systems per physical platform
	Streamline test and development environments
Reliability	Isolate software faults
	Reallocate existing partitions
	Create dedicated or as-needed failover partitions
Security	Contain digital attacks through fault isolation
	Apply different security settings to each partition

Consolidation

Three drivers have motivated, if not accelerated, the acceptance and adoption of virtualization technologies—consolidation, reliability, and security. The goal behind consolidation is to combine and unify. In the case of virtualization, workloads are combined on fewer physical platforms capable of sustaining their demand for computing resources, such as CPU, memory, and I/O. In modern data centers, many workloads are far from taxing the hardware they run on, resulting in infrastructure waste and lower returns. Through consolidation, virtualization allows you to combine server instances,

or operating systems and their workloads, in a strategic manner and place them on shared hardware with sufficient resource availability to satisfy resource demands. The result is increased utilization. It is often thought that servers shouldn't be forced to run close to their full-capacity levels; however, the opposite is true. In order to maximize that investment, servers should run as close to capacity as possible, without impacting the running workloads or business process relying on their performance. With proper planning and understanding of those workloads, virtualization will help increase server utilization while decreasing the number of physical platforms needed.

Another benefit of consolidation virtualization focuses on legacy system migrations. Server hardware has developed to such levels that they are often incompatible with legacy operating systems and applications. Newer processor technologies, supporting chipsets, and the high-speed buses sought after can often cripple legacy systems, if not render them inoperable without the possibility of full recompilation. Virtualization helps ease and simplify legacy system migrations by providing a common and widely compatible platform upon which legacy system instances can run. This improves the chances that applications can be migrated for older, unsupported, and riskier platforms to newer hardware and supported hardware with minimal impact.

In the past, operating systems were bound to a specific hardware platform. This tied many organizations' hands, forcing them to make large investments in hardware in order to maintain their critical business applications. Due to the commoditization of hardware, though, many of the common operating systems currently available can run on a wide range of server architectures, the most popular of which is the x86 architecture. You can run Windows, Unix, and your choice of Linux distributions on the x86 architecture. Virtualization technologies built on top of x86 architecture can, in turn, host heterogeneous environments. Multiple operating systems, including those previously mentioned, can be consolidated to the same physical hardware, further reducing acquisition and maintenance costs.

Finally, consolidation efforts help streamline development and test environments. Rather than having uncontrolled sprawl throughout your infrastructure as new projects and releases begin or existing applications are maintained, virtualization allows you to consolidate many of those workloads onto substantially fewer physical servers. Given that development and test loads are less demanding by nature than production, consolidation of those environments through virtualization can yield even greater savings than their production counterparts.

Designing & Planning…

More Cores Equal More Guests… Sometimes

When designing the physical platform for your virtualization and consolidation efforts, be sure to take advantage of the current offering of Intel and AMD multi-core processors. Do keep in mind, though, that increasing your core count, and subsequently your total processing power, does not proportionally relate to how many virtual machines you can host. Many factors can contribute to reduced

Continued

guest performance, including memory, bus congestion (especially true for slower Intel front-side bus architectures or NUMA-based four-way Opteron servers), I/O bus congestion, as well as external factors such as the network infrastructure and the SAN.

Carefully plan your hardware design with virtual machine placement in mind. Focus more on the combined workload than the virtual machine count when sizing your physical host servers. Also consider your virtualization product's features that you will use and how it may add overhead and consume resources needed by your virtual machines. Also consider the capability of your platform to scale as resource demands increase—too few memory slots, and you will quickly run out of RAM; too few PCI/PCI-X/PCI-e slots and you will not be able to scale your I/O by adding additional NICs or HBAs.

Finally, consider the level of redundancy and known reliability of the physical server hardware and supporting infrastructure. Remember that when your host fails, a host outage is much more than just one server down; all the virtual machines it was hosting will experience the outage as well.

Always keep in mind the key hardware traits required for any virtualization host:

- Performance
- Flexibility
- Reliability

Reliability

More than ever before, reliability has become a mandate and concern for many IT organizations. It has a direct relationship to system availability, application uptime, and, consequently, revenue generation. Companies are willing to, and often do, invest heavily into their server infrastructure to ensure that their critical line-of-business applications remain online and their business operation goes uninterrupted. By investing in additional hardware and software to account for software faults, infrastructures are fortified to tolerate failures and unplanned downtime with interruption. Doing so, though, has proven to be very costly.

Virtualization technologies are sensitive to this and address this area by providing high isolation between running virtual machines. A system fault in one virtual machine, or partition, will not affect the other partitions running on the same hardware platform. This isolation logically protects and shields virtual machines at the lowest level by causing them to be unaware, and thus not impacted, by conditions outside of their allocations. This layer of abstraction, a key component in virtualization, makes each partition just as if it was running on dedicated hardware.

Such isolation does not impede flexibility, as it would in a purely physical world. Partitions can be reallocated to serve other functions as needed. Imagine a server hosting a client/server application that is only used during the 8A.M. to 5P.M. hours Monday through Friday, another that runs batch processes to close out business operations nightly, and another that is responsible for data maintenance jobs over the weekend. In a purely physical world, they would exist as three dedicated servers that are

highly utilized during their respective hours of operation, but sit idle when not performing their purpose. This accounts for much computing waste and an underutilization of expensive investments. Virtualization addresses this by allowing a single logical or physical partition to be reallocated to each function as needed. On weekdays, it would host the client/server application by day and run the batch processes at night. On the weekends, it would then be reallocated for the data maintenance tasks, only to return to hosting the client/server application the following Monday morning. This flexibility allows IT organizations to utilize "part-time" partitions to run core business processes in the same manner as they would physical servers, but achieve lower costs while maintaining high levels of reliability.

Another area that increases costs is the deployment of standby or failover servers to maintain system availability during times of planned or unplanned outages. While capable of hosting the targeted workloads, such equipment remains idle between those outages, and in some cases, never gets used at all. They are often reduced to expensive paperweights, providing little value to the business while costing it much. Virtualization helps solve this by allowing just-in-time or on-demand provisioning of additional partitions as needed. For example, a partition that has been built (OS and applications) and configured can be put into an inactive (powered-off or suspended) state, ready to be activated when a failure occurs. When needed, the partition becomes active without any concern about hardware procurement, installation, or configuration. Another example is an active/passive cluster. In these clusters, the failover node must be active and online, not inactive. However, the platform hosting the cluster node must be dedicated to that cluster. This has caused many organizations to make a large investment in multiple failover nodes, which sit in their data centers idle, waiting to be used in case of an outage. Using server virtualization, these nodes can be combined onto fewer hardware platforms, as partitions hosting failover nodes are collocated on fewer physical hosts.

Security

The same technology that provides application fault isolation can also provide security fault isolation. Should a particular partition be compromised, it is isolated from the other partitions, stopping the compromise from being extended to them. Solutions can also be implemented that further isolate compromised partitions and OS instances by denying them the very resources they rely on to exist. CPU cycles can be reduced, network and disk I/O access severed, or the system halted altogether. Such tasks would be difficult, if not impossible, to perform if the compromised instance was running directly on a physical host.

When consolidating workloads through virtualization, security configurations can remain specific to the partition rather than the server as a whole. An example of this would be super-user accounts. Applications consolidated to a single operating system running directly on top of a physical server would share various security settings—in particular, root or administrator access would be the same for all. However, when the same workloads are consolidated to virtual partitions, each partition can be configured with different credentials, thus maintaining the isolation of system access with administrative privileges often required to comply with federal or industry regulations.

Simply put, virtualization is an obvious move in just about any company, small or large. Just imagine that your manager calls you into the office and begins to explain his or her concerns about cost containment, data center space diminishing, timelines getting narrower, and corporate mandates doing more with less. It won't take too many attempts to explain how virtualization can help address

all of those concerns. After realizing you had the answer all along, it will make your IT manager's day to learn this technology is the silver bullet that will satisfy the needs of the business while providing superior value in IT operations and infrastructure management and delivery.

NOTE

Most Virtual Machine Monitor (VMM) implementations are capable of interactive sessions with administrators through CLI or Web interfaces. Although secure, a compromised VMM will expose every virtual machine managed by that VMM. So exercise extreme caution when granting access or providing credentials for authentication to the VMM management interface.

How Does Virtualization Work?

While there are various ways to virtualize computing resources using a true VMM, they all have the same goal: to allow operating systems to run independently and in an isolated manner identical to when it is running directly on top of the hardware platform. But how exactly is this accomplished? While hardware virtualization still exists that fully virtualizes and abstracts hardware similar to how the System370 did, such hardware-based virtualization technologies tend to be less flexible and costly. As a result, a slew of software hypervisor and VMMs have cropped up to perform virtualization through software-based mechanisms. They ensure a level of isolation where the low-level, nucleus core of the CPU architecture is brought up closer to the software levels of the architecture to allow each virtual machine to have its own dedicated environment. In fact, the relationship between the CPU architecture and the virtualized operating systems is the key to how virtualization actually works successfully.

OS Relationships with the CPU Architecture

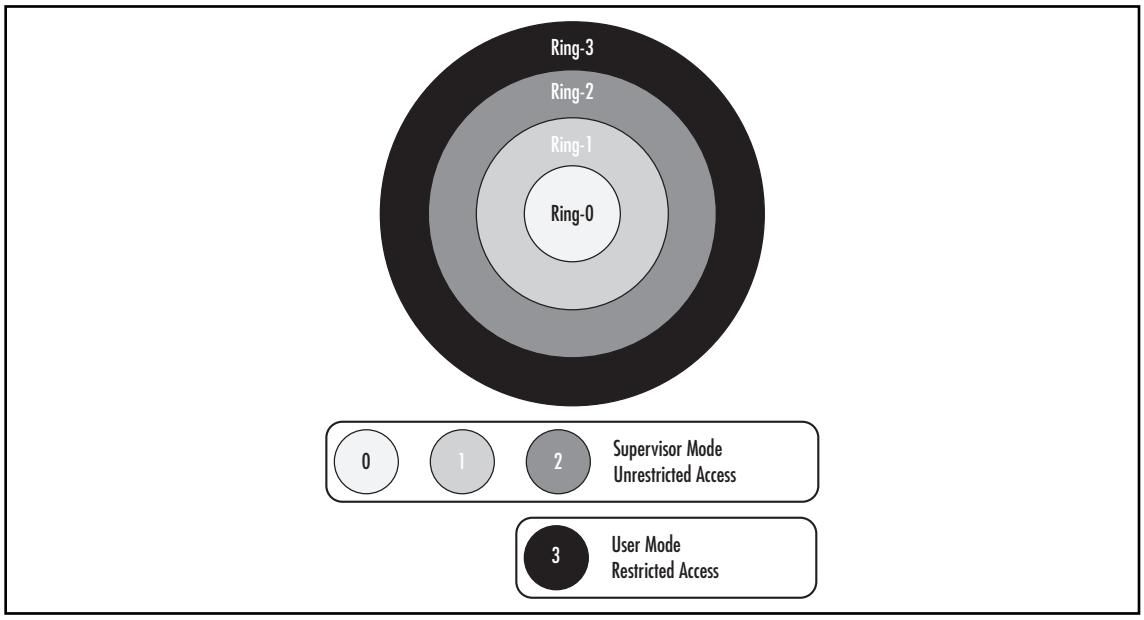
Ideal hardware architectures are those in which the operating system and CPU are designed and built for each other, and are tightly coupled. Proper use of complex system call requires careful coordination between the operating system and CPU. This symbiotic relationship in the OS and CPU architecture provides many advantages in security and stability. One such example was the MULTICS time-sharing system, which was designed for a special CPU architecture, which in turn was designed for it.

What made MULTICS so special in its day was its approach to segregating software operations to eliminate the risk or chance of a compromise or instability in a failed component from impacting other components. It placed formal mechanisms, called *protection rings*, in place to segregate the trusted operating system from the untrusted user programs. MULTICS included eight of these protection rings, a quite elaborate design, allowing different levels of isolation and abstraction from the core nucleus of the unrestricted interaction with the hardware. The hardware platform, designed in tandem by GE and MIT, was engineered specifically for the MULTICS operating system and incorporated hardware "hooks" enhancing the segregation even further. Unfortunately, this design approach proved to be too costly and proprietary for mainstream acceptance.

The most common CPU architecture used in modern computers is the IA-32, or x86-compatible, architecture. Beginning with the 80286 chipset, the x86 family provided two main methods of addressing memory: real mode and protected mode. In the 80386 chipset and later, a third mode was introduced called virtual 8086 mode, or VM86, that allowed for the execution of programs written for real mode but circumvented the real-mode rules without having to raise them into protected mode. Real mode, which is limited to a single megabyte of memory, quickly became obsolete; and virtual mode was locked in at 16-bit operation, becoming obsolete when 32-bit operating systems became widely available for the x86 architecture. Protected mode, the saving grace for x86, provided numerous new features to support multitasking. These included segmenting processes, so they could no longer write outside their address space, along with hardware support for virtual memory and task switching.

In the x86 family, protected mode uses four privilege levels, or rings, numbered 0 to 3. System memory is divided into segments, and each segment is assigned and dedicated to a particular ring. The processor uses the privilege level to determine what can and cannot be done with code or data within a segment. The term "rings" comes from the MULTICS system, where privilege levels were visualized as a set of concentric rings. Ring-0 is considered to be the innermost ring, with total control of the processor. Ring-3, the outermost ring, is provided only with restricted access, as illustrated in Figure 1.5.

Figure 1.5 Privilege Rings of the x86 Architecture



NOTE

The same concept of protection rings exists in modern OS architecture. Windows, Linux, and most Unix variants all use rings, although they have reduced the four-ring structure to a two-layer approach that uses only Rings 0 and 3. Ring-0 is commonly called *Supervisor Mode*, while Ring-3 is known as *User Mode*. Security mechanisms in the hardware enforce restrictions on Ring-3 by limiting code access to segments, paging, and input/output. If a user program running in Ring-3 tries to address memory outside of its segments, a hardware interrupt stops code execution. Some assembly language instructions are not even available for execution outside of Ring-0 due to their low-level nature.

The Virtual Machine Monitor and Ring-0 Presentation

The Supervisor Mode is the execution mode on an x86 processor that enables the execution of all instructions, including privileged instructions such as I/O and memory management operations. It is in Supervisor Mode (Ring 0) where the operating system would normally run. Since Ring-3 is based on Ring-0, any system compromise or instability directly impacts User Mode running in Ring-3. In order to isolate Ring-0 for each virtualized guest, it then becomes necessary to move Ring-0 closer to the guests. By doing so, a Ring-0 failure for one virtualized guest does not impact Ring-0, or consequently Ring-3, of any other guest. The perceived Ring-0 for guests can reside in either Ring-1, -2, or -3 for x86 architectures. Of course, the further the perceived Ring-0 is away from the true Ring-0, the more distant it is from executing direct hardware operations, leading to reduced performance and independence.

Virtualization moves Ring-0 up the privilege rings model by placing the Virtual Machine Monitor, or VMM, in one of the rings, which in turn presents the Ring-0 implementation to the hosted virtual machines. It is upon this presented Ring-0 that guest operating systems run, while the VMM handles the actual interaction with the underlying hardware platform for CPU, memory, and I/O resource access. There are two types of VMMs that address the presentation of Ring-0 as follows:

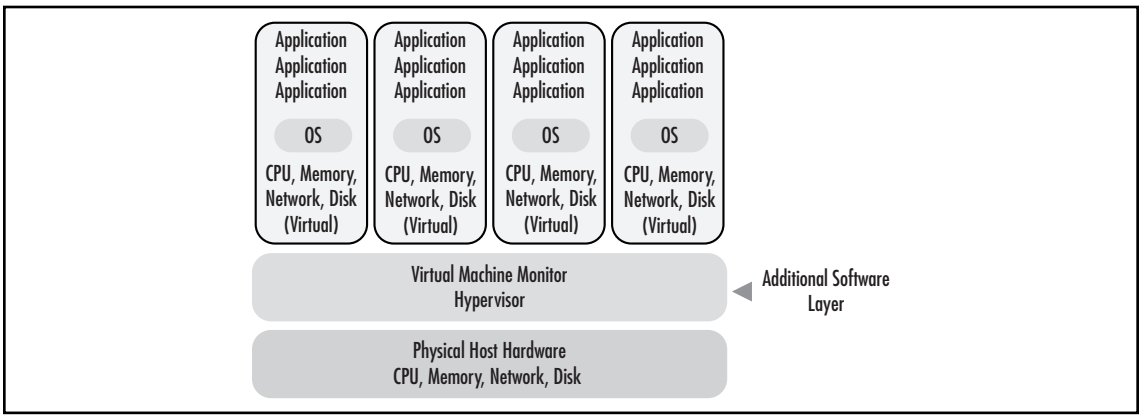
- **Type 1 VMM** Software that runs directly on top of a given hardware platform on the true Ring-0. Guest operating systems then run at a higher level above the hardware, allowing for true isolation of each virtual machine.
- **Type 2 VMM** Software that runs within an operating system, usually in Ring-3. Since there are no additional rings above Ring-3 in the x86 architecture, the presented Ring-0 that the virtual machines run on is as distant from the actual hardware platform as it can be. Although this offers some advantages, it is usually compounded by performance-impeding factors as calls to the hardware must traverse many diverse layers before the operations are returned to the guest operating system.

See Chapter 2 of this book for an overview of several virtualization products and how they accomplish Ring-0 placement.

The VMM Role Explored

To create virtual partitions in a server, a thin software layer called the Virtual Machine Monitor (VMM) runs directly on the physical hardware platform. One or more guest operating systems and application stacks can then be run on top of the VMM. Figure 1.6 expands our original illustration of a virtualized environment presented in Figure 1.1.

Figure 1.6 The OS and Application Stack Managed by the VMM Software Layer



The VMM is the center of server virtualization. It manages hardware resources and arbitrates the requests of the multiple guest operating systems and application stacks. It presents a virtual set of CPU, memory, I/O, and Disk resources to each guest either based on the actual physical hardware or based on a standard and consistent selection of custom hardware. This section further discusses the role of the VMM and design considerations that are used when designing a VMM.

The Popek and Goldberg Requirements

Often referred to as the original reference source for VMM criteria, the Popek and Goldberg Virtualization Requirements define the conditions for a computer architecture to support virtualization. Written in 1974 for the third-generation computer systems of those days, they generalized the conditions that the software that provides the abstraction of a virtual machine, or VMM, must satisfy. These conditions, or properties, are

- **Equivalence** A program running under the VMM should exhibit a predictable behavior that is essentially identical to that demonstrated when running on the underlying hardware platform directly. This is sometimes referred to as *Fidelity*.
- **Resource Control** The VMM must be in complete control of the actual hardware resources virtualized for the guest operating systems at all times. This is sometimes referred to as *Safety*.
- **Efficiency** An overwhelming number of machine instructions must be executed without VMM intervention or, in other words, by the hardware itself. This is sometimes referred to as *Performance*.