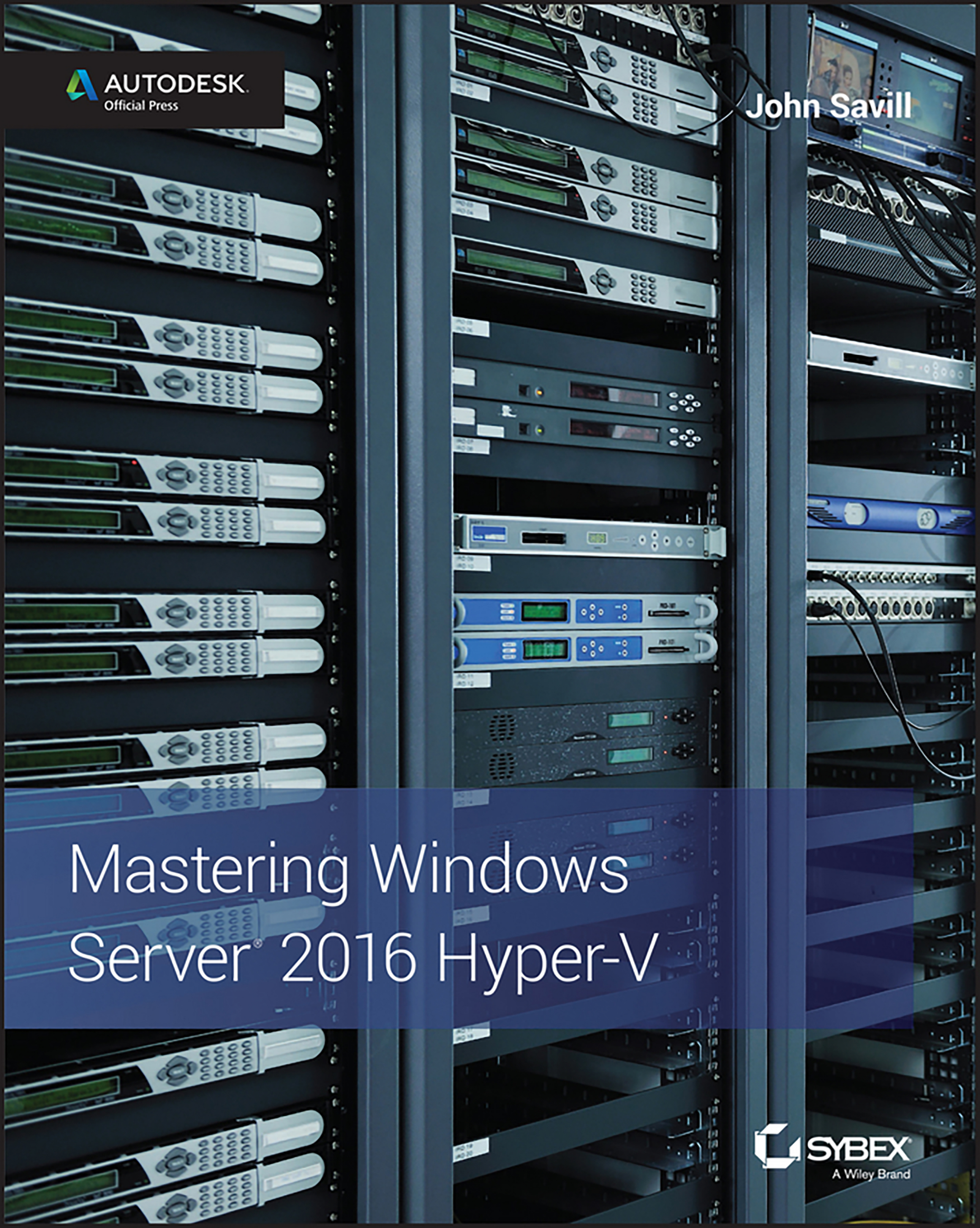


AUTODESK
Official Press

John Savill

Mastering Windows Server® 2016 Hyper-V

SYBEX
A Wiley Brand

Mastering Windows Server® 2016 Hyper-V®

John Savill

SYBEX®
A Wiley Brand

Senior Acquisitions Editor: Kenyon Brown
Development Editor: Gary Schwartz
Production Editor: Christine O'Connor
Copy Editor: Sharon Wilkey
Editorial Manager: Mary Beth Wakefield
Production Manager: Kathleen Wisor
Executive Editor: Jim Minatel
Book Designers: Maureen Forys, Happenstance Type-O-Rama and Judy Fung
Proofreader: Nancy Carrasco
Indexer: Ted Laux
Project Coordinator, Cover: Brent Savage
Cover Designer: Wiley
Cover Image: dotshock/Shutterstock

ISBN: 978-1-119-28618-9
ISBN: 978-1-119-28621-9 (ebk.)
ISBN: 978-1-119-28620-2 (ebk.)
Manufactured in the United States of America

Library of Congress Control Number: 2016959384

10 9 8 7 6 5 4 3 2 1

For my wife, Julie, and my children,
Abby, Ben, and Kevin. My everythings.

Acknowledgments

I could not have written this book without the help and support of many people. First I need to thank my wife, Julie, for putting up with me for being busier than usual the last six months and for picking up the slack as always—and for always supporting the crazy things I want to do. My children, Abby, Ben, and Kevin, always make all the work worthwhile; they can turn the worst, most tiring day into a good one with a smile and a laugh.

Of course, the book wouldn't be possible at all without the Wiley team: senior acquisitions editor Kenyon Brown, development editor Gary Schwartz, production editor Christine O'Connor, copy editor Sharon Wilkey, and proofreader Nancy Carrasco.

Many people have helped me over the years with encouragement and technical knowledge, and this book is the sum of that. The following people helped out on specific aspects of this book, and I want to thank them and give them the credit they deserve for helping make this book as good as possible (if I've missed anyone, I'm truly sorry): Theo Thompson, Mathew John, Jake Oshins, Chris Huybregts, Neil Peterson, Jim Wooldridge, Steven Ekren, Michael Gray, Darren Moss, Claus Joergensen, Dean Wells, Nir Ben Zvi, Andrew Mason, Jian Yan, Simon Gurevich, Shivam Garg, Elden Christensen, Jason Messer, Taylor Brown, Manish Jha, Shon Shah, Pradeep Reddy, Don Stanwyck, Greg Cusanza, Paul Kimbel, Scott Willwerth, and Tim Aranki.

About the Author

John Savill is a technical specialist who focuses on Microsoft core infrastructure technologies including Windows, Hyper-V, System Center, and anything that does something cool. He has been working with Microsoft technologies for 20 years. He is the creator of the highly popular `NTFAQ.com` website and a senior contributing editor for Windows IT Pro magazine. He has written seven previous books covering Windows, Azure, Hyper-V, and advanced Active Directory architecture. When he is not writing books, he writes magazine articles and white papers; creates many technology videos, which are available on his YouTube channel, `www.youtube.com/ntfaqguy`; and presents online and at industry-leading events, including TechEd, Ignite, and Windows Connections. When he was writing this book, he had just completed running his annual series of Master Classes, which include classes on Azure, Hyper-V, PowerShell, and the complete Microsoft stack. John also has online courses on Pluralsight and O'Reilly.

Outside technology, John enjoys cardio and weight training. In fact, while writing this book, he was training for his third full Ironman triathlon in Chattanooga, Tennessee.

John tries to update his blog at `www.savilltech.com/blog` with the latest news of what he is working on and also tweets at `@NTFAQGuy`.

Contents at a Glance

Contents

Introduction

The book you are holding is the result of 20 years of experience in the IT world and over 15 years of virtualization experience that started with VMware and includes Virtual PC and now Hyper-V. My goal for this book is simple: to help you become knowledgeable and effective when it comes to architecting and managing a Hyper-V-based virtual environment. This means not only understanding how Hyper-V works and its capabilities, but also knowing when to leverage other technologies to provide the most complete and optimal solution. That means leveraging System Center and Microsoft Azure, which I also cover because they relate to Hyper-V. I also dive into some key technologies of Windows Server that bring benefits to Hyper-V.

Hyper-V is now a mature and widely adopted virtualization solution. It is one of only two x86 server virtualization solutions in Gartner's leader quadrant. In addition to being used by many of the largest companies in the world, Hyper-V powers Microsoft Azure, which is one of the largest cloud services in the world.

Hyper-V is a role of Windows Server. If you are a Windows administrator, you will find Hyper-V management fairly intuitive, but there are still many key areas that require attention. I have structured this book to cover the key principles of virtualization and the resources that you will manage with Hyper-V before I cover installing and configuring Hyper-V itself and then move on to advanced topics such as high availability, replication, private cloud, and more.

I am a strong believer in learning by doing, and therefore I highly encourage you to try out all of the technologies and principles I cover in this book. You don't need a huge lab environment. For most topics, you could use a single machine with Windows Server installed and 8GB of memory to enable a few virtual machines to run concurrently. Ideally, though, having at least two servers will help with the replication and high-availability concepts. In this book, sometimes you'll see step-by-step instructions to guide you through a process, sometimes I link to an external source that already has a good step-by-step guide, and sometimes I link to videos that I have posted to ensure maximum understanding. With Windows 10, Hyper-V is included in the box, so even without any kind of server, it is possible to explore many of the Hyper-V technologies.

I have created an application that is available for various platforms: Mastering Hyper-V 2016. It provides easy access to the external links, videos, and code samples that I use in this book. As you read each chapter, check out the application to find related content. The application can be downloaded from `www.savilltech.com/mhv`. Using the various platform stores also allows me to update it over time as required. Please get this application, as I will use it to add videos based on reader feedback that are not referenced in the main text and include additional information where required.

Who Should Read This Book

I am making certain assumptions regarding the reader:

- You have basic Windows Server knowledge and can install Windows Server.
- You have basic knowledge of PowerShell.
- You have access to a Hyper-V server to enable a test implementation of the many covered technologies.

This book is intended for anyone who wants to learn Hyper-V. If you have a basic knowledge of virtualization or a competing technology, such as VMware, that will help, but it is not a requirement. I start off with a foundational understanding of each technology and then build on that to cover more-advanced topics and configurations. If you are an architect, a consultant, an administrator, or really anyone who just wants better knowledge of Hyper-V, this book is for you.

At times I go into advanced topics that may seem over your head. In those cases, don't worry. Focus on the preceding elements that you understand, and implement and test them to solidify your understanding. Then, when you feel comfortable, come back to the more advanced topics. They will seem far simpler once your understanding of the foundational principles is solidified.

What's Inside

Here is a glance at what's in each chapter:

Chapter 1: Introduction to Virtualization and Microsoft Solutions This chapter focuses on the core value proposition of virtualization and how the datacenter has evolved. It covers the key changes and capabilities of Hyper-V in addition to the role System Center plays in a Hyper-V environment. I cover the types of cloud services available and how Hyper-V forms the foundation of private cloud solutions.

Chapter 2: Virtual Machine Resource Fundamentals This chapter covers the core resources of a virtual machine, specifically architecture (generation 1 and generation 2 virtual machines), processor, and memory. You will learn about advanced configurations to enable many types of operating system support along with best practices for resource planning.

Chapter 3: Virtual Networking This chapter covers one of the most complicated aspects of virtualization, especially when using the new network virtualization capabilities in Hyper-V. This chapter covers the key networking concepts, how to architect virtual networks, and how to configure them. I also cover networking using System Center Virtual Machine Manager (SCVMM) and how to design and implement network virtualization v2 that is introduced in Windows Server 2016.

Chapter 4: Storage Configurations This chapter covers the storage options for Hyper-V environments, including the VHD and VHDX formats, plus capabilities in Windows Server 2016 that help manage direct attached storage, including Storage Spaces Direct and Storage Replica. You will learn about storage technologies for virtual machines such as iSCSI, Virtual Fibre Channel, and shared VHDX; their relative advantages; as well as the storage migration and resize functions.

Chapter 5: Managing Hyper-V This chapter walks you through the installation of and best practices for managing Hyper-V. The basics of configuring virtual machines, installing operating systems, and using the Hyper-V Integration Services are all covered. Strategies for migrating from other hypervisors, physical servers, and other versions of Hyper-V are explored.

Chapter 6: Maintaining a Hyper-V Environment This chapter focuses on the tasks required to keep Hyper-V healthy after you've installed it, which includes patching, malware protection, backup, and monitoring. Key actions, such as taking checkpoints of virtual machines, setting up service templates, and performance tuning are covered.

Chapter 7: Failover Clustering and Migration Technologies This chapter covers making Hyper-V highly available by using Failover Clustering, and it includes a deep dive into exactly what makes a cluster tick, specifically when running Hyper-V. Key migration technologies such as Live Migration, Shared Nothing Live Migration, and Storage Migration are explored in addition to configurations related to mobility outside a cluster and placement optimization for virtual machines.

Chapter 8: Hyper-V Replica and Cloud Orchestration This chapter shifts from high availability to a requirement of many organizations today: providing disaster-recovery protection in the event of losing an entire site. This chapter looks at the options for disaster recovery, including leveraging Hyper-V Replica, orchestrating failovers with Microsoft Azure in the event of a disaster, and using Azure as the DR target location.

Chapter 9: Implementing the Private Cloud, SCVMM, and Microsoft Azure Stack This chapter shows the many benefits of the Microsoft stack to organizations, beyond just virtualization. This chapter explores the key benefits of a private cloud and describes what a private cloud using Microsoft technologies looks like. Key components and functional areas, including the actual end-user experience and how you can leverage all of System Center for different levels of private cloud capability, are all covered. The Microsoft Azure Stack solution is introduced and its key capabilities explored.

Chapter 10: Containers and Docker This chapter focuses on the new Windows and Hyper-V container technologies available in Windows Server 2016. This chapter dives into the architectural components and management with Docker.

Chapter 11: Remote Desktop Services This chapter shifts the focus to another type of virtualization, virtualizing the end-user experience, which is a critical capability for most organizations. Virtual desktop infrastructure is becoming a bigger component of the user environment. This chapter looks at the types of desktop virtualization available with Remote Desktop Services, with a focus on capabilities that are enabled by Hyper-V, such as advanced graphical capabilities with RemoteFX.

Chapter 12: Microsoft Azure IaaS, Storage, and Networking This chapter explores the capabilities of one of the biggest public cloud services in the world, which is powered by Hyper-V. This chapter covers the fundamentals of Microsoft Azure and how to create virtual machines in Microsoft Azure. The chapter also covers the networking options available both within Microsoft Azure and to connect to your on-premises network. I examine the migration of virtual machines and how to leverage Azure Storage. Ways to provide a seamless management experience are also explored.

Chapter 13: Bringing It All Together with a Best-of-Breed Cloud Solution This chapter brings together all of the technologies and options to help architect a best-of-breed virtualization and cloud solution.

Don't forget to download the companion Windows Store application, Mastering Hyper-V, from www.savilltech.com/mhv.

The Mastering Series

The Mastering series from Sybex provides outstanding instruction for readers with intermediate and advanced skills in the form of top-notch training and development for those already working in their field and clear, serious education for those aspiring to become pros. Every Mastering book includes the following elements:

- Skill-based instruction, with chapters organized around real tasks rather than abstract concepts or subjects
- Self-review test questions, so you can be certain that you're equipped to do the job right

How to Contact the Author

I welcome feedback from you about this book or about books you'd like to see from me in the future. You can reach me by writing to john@savilltech.com. For more information about my work, visit my website at www.savilltech.com and follow me on Twitter at @NTFAQGuy.

Sybex strives to keep you supplied with the latest tools and information that you need for your work. Please check the Sybex website at www.sybex.com/go/masteringhyperv2016, where we'll post additional content and updates that supplement this book should the need arise.

Chapter 1

Introduction to Virtualization and Microsoft Solutions

This chapter lays the foundation for the core fabric concepts and technologies discussed throughout not just this first part of this book, but the entire book. Virtualization has radically changed the layout and operation of the datacenter, and this datacenter evolution and its benefits are explored.

Microsoft's solution for virtualization is its Hyper-V technology, which is a core part of Windows Server, and it is also available in the form of a free, stand-alone hypervisor. The virtualization layer is only part of the solution. Management is just as critical, and in today's world, the public cloud is also a consideration. Thus a seamless management story with compatibility between your on- and off-premises resources provides the model implementation.

In this chapter, you will learn to:

- Articulate the key value propositions of virtualization.
- Understand the differences in functionality between the various versions of Hyper-V.
- Differentiate between the types of cloud services and when each type is best utilized.

The Evolution of the Datacenter

Many books are available that go into a great amount of detail about the history of datacenters, but that is not the goal of the following sections. Instead, I am going to take you through the key changes that I have seen in my 20 years of working in and consulting about datacenter infrastructure. This brief look at the evolution of datacenters will help you understand the challenges of the past, why virtualization has become such a key component of every modern datacenter, and why there is still room for improvement.

One Box, One Operating System

As recent as 10 years ago, datacenters were all architected in a similar way. These huge rooms with very expensive cabling and air conditioning were home to hundreds, if not thousands, of servers. Some of these servers were mainframes, but the majority were regular servers (although today the difference between a mainframe and a powerful regular server is blurring). Although the processor architecture running in these servers may have been different—for example, some were x86 based, some Alpha, some MIPS, some SPARC—each server ran an operating system (OS) such as Windows, Linux, or OpenVMS. Some OSs supported different processor architectures, while others were limited to a specific architecture. Likewise, some processor architectures would dictate which OS had to be used. The servers themselves may have been freestanding, but as technology advanced, servers got smaller and became rack mountable, enabling greater compression of the datacenter.

Understanding x86

Often, the term *x86* is used when talking about processor architecture, but its use has been generalized beyond just the original Intel processors that built on the 8086. *x86* does not refer only to Intel processors, but it is used more generally to refer to 32-bit operating systems running on any processor leveraging x86 instruction sets, including processors from AMD. *x64* represents a 64-bit instruction set extension processor (primarily from Intel and AMD), although you may also see *amd64* to denote 64-bit. What can be confusing is that a 64-bit processor is still technically x86, and it has become more common today simply to use *x86* to identify anything based on x86 architecture, which could be 32-bit or 64-bit from other types of processor architecture. Therefore, if you see *x86* within this book, or in other media, it does not mean 32-bit only.

Even with all this variation in types of server and operating systems, there was something they had in common. Each server ran a single OS, and that OS interacted directly with the hardware in the server and had to use hardware-specific drivers to utilize the available capabilities. In the rest of this book, I focus primarily on x86 Windows; however, many of the challenges and solutions apply to other OSs as well.

Every server comprises a number of resources, including processor, memory, network, and storage (although some modern servers do not have local storage such as blade systems, and instead rely completely on external storage subsystems). The amount of each resource can vary drastically, as shown in the following sections.

Processor

A server can have one or more processors, and it's common to see servers with two, four, or eight processors (although it is certainly possible to have servers with more). Modern processors use a core architecture that allows a single processor to have multiple cores. Each core consists of a discrete central processing unit (CPU) and L1 cache (very fast memory used for temporary storage of information related to computations) able to perform its own computations. Those multiple cores can then share a common L2 cache (bigger but not as fast as L1) and bus interface. This allows a single physical processor to perform multiple parallel computations and actually act like many separate processors. The first multicore processors had two cores (dual-core), and this continues to increase with eight-core (octo-core) processors available and a new "many-core" generation on the horizon, which will have tens of processor cores. It is common to see a physical processor referred to as a *socket*, and each processor core referred to as a *logical processor*. For example, a dual-socket system with quad-core processors would have eight logical processors (four on each physical processor, and there are two processors). In addition to the number of sockets and cores, variations exist in the speed of the processors and the exact instruction sets supported. (It is because of limitations in the continued increase of clock speed that moving to multicore became the best way to improve overall computational performance, especially as modern operating systems are multithreaded and can take advantage of parallel computation.)

Some processors also support *hyperthreading*, which is a means to split certain parts of a processor core into two parallel computational streams to avoid wasted processing. Hyperthreading does not double computational capability, but it generally gives a 10 to 15 percent performance boost. Typically with hyperthreading, this would therefore double the number of logical processors in a system. However, for virtualization, I prefer not to do this doubling, but this does not mean that I turn off hyperthreading. Hyperthreading may sometimes help, but it certainly won't hurt.

Is There a Big and a Little Thread with Hyperthreading?

Hyperthreading enables two streams of execution on a single processor core, and you often hear numbers such as a 15 percent performance improvement. This leads to the belief that there is the main thread on the core and then a little "mini-me" thread that has a smaller capability. This is not true. With hyperthreading, a single core has some components duplicated, enabling two sets of logical state per core. Typically, during a thread of execution, the core is not fully utilized for various reasons, such as when a particular instruction stream uses only specific types of ALU (Arithmetic Logic Unit), leaving others unused, and more commonly when a cache miss occurs that causes the thread execution to stall while data is fetched. With hyperthreading and the two sets of logical state, if one thread is stalled because of a cache miss, the chances are good that the other thread can execute. This, therefore, keeps the core better utilized and improves the overall performance, and this is where the 15 percent performance gain comes from. Notice that both threads are equal and which one does more work just depends on how busy they are kept, the type of computations, the frequency of cache misses, and so on.

Earlier versions of Windows supported different processor architectures, including MIPS, Alpha, PowerPC, and more recently Itanium. However, as of Windows Server 2012, the only supported processor architecture is x86 and specifically only 64-bit from Windows Server 2008 R2 and above. (There are still 32-bit versions of the Windows 8/8.1 client operating system.)

Prior to Windows Server 2008, there were separate versions of the hardware abstraction layer (HAL), depending on whether you had a uniprocessor or multiprocessor system. However, given the negligible performance savings on modern, faster processors that were specific to the uniprocessor HAL on single-processor systems (synchronization code for multiple processors was not present in the uniprocessor HAL), this was removed, enabling a single unified HAL that eases some of the pain caused by moving from uni- to multiprocessor systems.

Memory

The memory resource is generally far simpler, with fewer variations. Some memory supports error-correcting code (ECC), which provides resiliency against the most common types of internal corruption, and memory has different speeds. However, for most environments, the memory consideration is simply how much there is! Generally, the more memory, the better, and with only 64-bit versions of Windows Server, there are no longer considerations around the maximum amount of memory that can be used by an operating system (a 4GB limit exists for 32-bit operating systems).

Storage

Storage falls into one of two buckets: internal or external. If the storage is internal (direct-attached storage, or DAS), the disks are local to the server and attached via a technology such as SCSI, SATA, or SAS. (Even if the storage is in an external storage enclosure but is connected via one of these means, it is still considered direct-attached.) Alternatively, the storage is external, such as storage that is hosted on another server or on a storage area network (SAN) or on network-attached storage (NAS). Various protocols may be used for external storage access that offer either file-level or block-level access to the storage.

File-level access enables the requesting server to access files on the server, but this is offered over a protocol that hides the underlying filesystem and actual blocks of the file on disk. Examples of file-level protocols are Server Message Block (SMB) and Network File System (NFS), typically offered by NAS devices.

Block-level access enables the requesting server to see the blocks on the disk and effectively mount the disk, format the mounted disk with a filesystem, and then directly manipulate blocks on the disk. Block-level access is typically offered by SANs using protocols such as iSCSI (which leverages the TCP/IP network) and Fibre Channel (which requires dedicated hardware and cabling). Typically, block-level protocols have offered higher performance, and the SANs providing the block-level storage offer advanced features, which means that SANs are typically preferred over NAS devices for enterprise storage. However, there is a big price difference between a SAN and potentially the dedicated storage hardware and cabling (referred to as *storage fabric*), and an SMB device that leverages the existing IP network connectivity.

The line between types of storage is also blurring greatly, especially with modern hyperconverged systems that contain both compute and the storage for workloads. Windows Server 2016 includes Storage Spaces Direct (S2D), which enables direct-attached storage in cluster nodes to be aggregated together and utilized as cluster storage. This is commonly referred to as a *VSAN technology* in the industry. When combined with other Windows Server storage features, using direct-attached storage no longer means compromising features and performance.

The hardware for connectivity to storage can vary greatly for both internal storage, such as SCSI controllers, and external storage, such as the host bus adapters (HBAs), which provide the connectivity from a server to a Fibre Channel switch (which then connects to the SAN). Very specific drivers are required for the exact model of storage adapter, and often the driver version must correlate to a firmware version of the storage adapter.

In all components of an environment, protection from a single point of failure is desirable. For internal storage, it is common to group multiple physical disks together into arrays that can provide protection from data loss due to a single disk failure, a redundant array of independent disks (RAID). Windows Server also has other technologies that are covered in later chapters, including Storage Spaces. For external storage, it is possible to group multiple network adapters together into a team for IP-based storage access. For example, SMB, NFS, and iSCSI can be used to provide resiliency from a single network adapter failure, and for non-IP-based storage connectivity, it is common for a host to have at least two storage adapters, which are in turn each connected to a different storage switch (removing single points of failure). Those storage adapters are effectively joined using multipath I/O (MPIO), which provides protection from a single storage adapter or storage switch failure. Both the network and storage resiliency configurations are very specific and can be complex.

Finally, the disks themselves have different characteristics, such as size and speed. The higher availability of SSD storage and its increase in size and reduced cost is making it a realistic component of modern datacenter storage solutions. This is especially true in tiered solutions, which allow a mix of fast and slower disks, with the most used and important data moved to the faster disks. Disk speed is commonly measured in input/output operations per second, or IOPS (pronounced *eye-ops*). The higher the IOPS, the faster the storage.

The storage also contains the actual operating system (which can be local or on a remote SAN using boot-from-SAN capabilities).

Networking

Compute, memory, and storage enable a server to perform work, but in today's environments, that work often relies on work done by other servers. In addition, access to that work from clients and the communication between computers is enabled through the network. To participate in an IP network, each machine has to have at least one IP address, which can be statically or automatically assigned. To enable this IP communication, a server has at least one network adapter, and that network adapter has one or more ports that connect to the network fabric, which is typically Ethernet. As is true when connecting to storage controllers, the operating system requires a driver specific to the network adapter to connect to the network. In high-availability network configurations, multiple network adapters are teamed together, which can be done in many cases through the driver functionality or in Windows Server 2012 using the native Windows NIC Teaming feature. Typical networking speeds in datacenters are 1 gigabit per second (Gbps) and 10Gbps, but faster speeds are available. As with IOPS for storage, the higher the network speed, the more data that you can transfer and the better the network performs.

How Virtualization Has Changed the Way Companies Work and Its Key Values

I spend quite a lot of time talking about resources and how they can vary, and where specific drivers and configurations may be required. This is critical to understand because many benefits of virtualization derive directly from the complexity and variation in all of the resources available to a server. Figure 1.1 shows the Device Manager output from a server. Notice all of the very specific types of network and storage hardware.

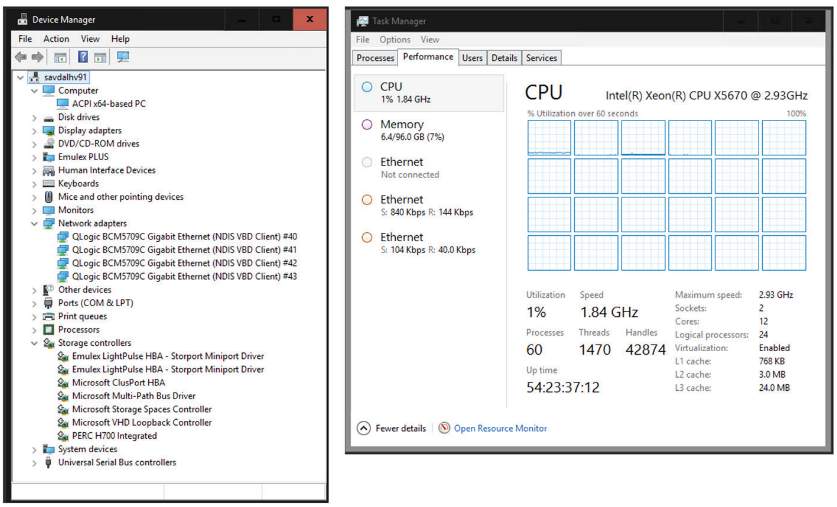
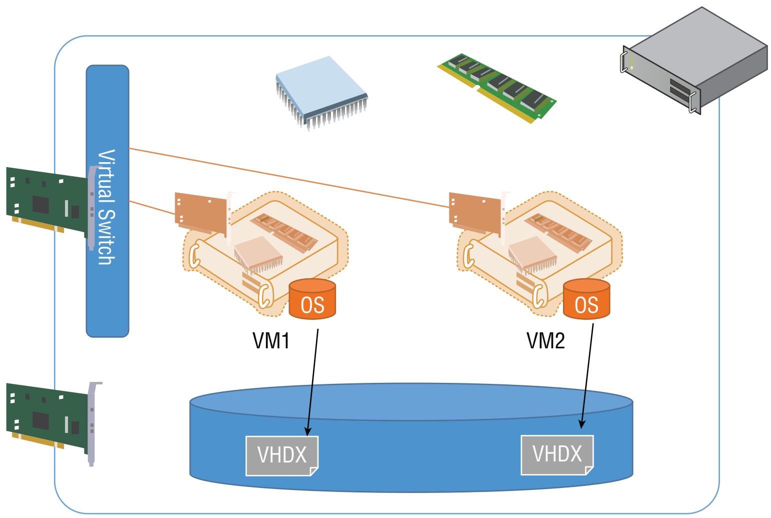


Figure 1.1 The Device Manager view of a typical physical server, with Task Manager showing some of its available resources

All of these resources are specific to the deployed operating system and are not easy to change in normal physical server deployments. If the boot disk from a server is placed in a different server with a different motherboard, network, or storage, there is a strong possibility the server will not boot, and it certainly will lose configuration settings and may not be able to use the hardware in the new server. The same applies to trying to restore a backup of a server to different hardware. This tight bonding between the operating system and the hardware can be a major pain point for organizations when they are considering resiliency from hardware failure but also for their disaster-recovery planning. It's necessary to have near identical hardware in the disaster-recovery location, and organizations start to find themselves locked into specific hardware vendors.

Virtualization abstracts the physical hardware from that of the created virtual machines. At a high level, virtualization allows virtual machines to be created. The virtual machines are assigned specific amounts of resources, such as CPU and memory, in addition to being given access to different networks via virtual switches. They are also assigned storage through virtual hard disks, which are just files on the local filesystem of the virtualization host or on remote storage. Figure 1.2 shows a high-level view of a virtualized environment.

FIGURE 1.2
A high-level view of a virtualization host and resources assigned to virtual machines



Within the virtual machine, an operating system is installed such as Windows Server 2016, Windows Server 2008, Windows 10, or a Linux distribution. No special process is needed to install the operating system into a virtual machine, and it's not even necessary for the operating system to support virtualization. However, most modern operating systems are virtualization-aware today and are considered "enlightened" to be able to understand virtualized hardware directly. The operating system installed in the virtual machine, commonly referred to as the *guest operating system*, does not see the physical hardware of the server but rather a set of virtualized hardware that is completely abstracted from the physical hardware.

Figure 1.3 shows a virtual machine (VM) that is running on the physical server shown in Figure 1.1. Notice the huge difference in what is visible. All of the same capabilities are