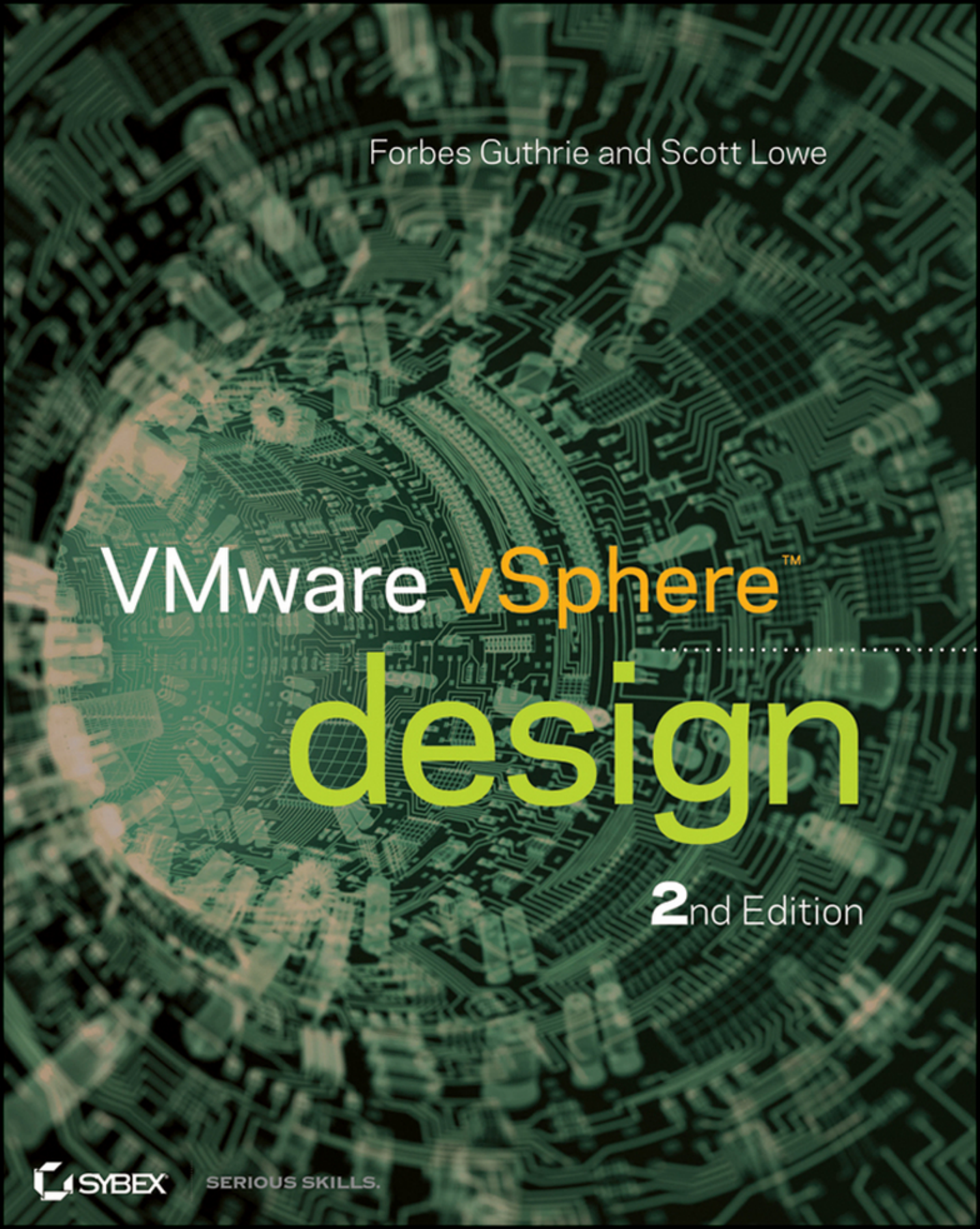
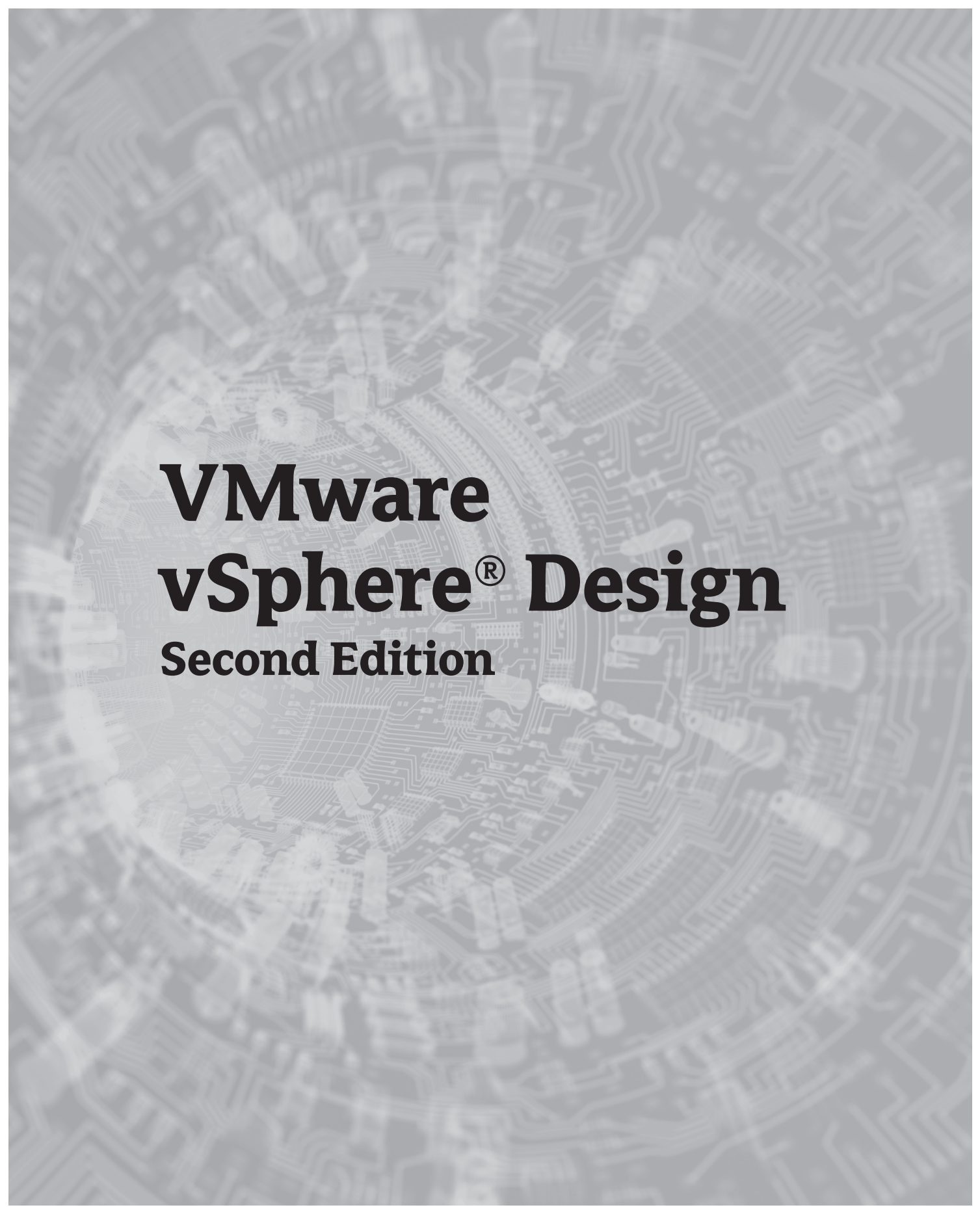

Forbes Guthrie and Scott Lowe
VMware vSphere™
design
2nd Edition
SYBEX
SERIOUS SKILLS.



VMware vSphere® Design

Second Edition

VMware vSphere® Design

Second Edition

Forbes Guthrie

Scott Lowe

SYBEX®
A Wiley Brand

Acquisitions Editor: Mariann Barsolo
Development Editor: Lisa Bishop
Technical Editor: Jason Boche
Production Editor: Eric Charbonneau
Copy Editor: Tiffany Taylor
Editorial Manager: Pete Gaughan
Production Manager: Tim Tate
Vice President and Executive Group Publisher: Richard Swadley
Vice President and Publisher: Neil Edde
Book Designers: Maureen Forys and Judy Fung
Proofreader: Nancy Bell
Indexer: Ted Laux
Project Coordinator, Cover: Katherine Crocker
Cover Designer: Ryan Sneed
Cover Image: © Konstantin Inozemtsev/iStockphoto

Published simultaneously in Canada
ISBN: 978-1-118-40791-2
ISBN: 978-1-118-53823-4 (ebk.)
ISBN: 978-1-118-49394-6 (ebk.)
ISBN: 978-1-118-53819-7 (ebk.)

Library of Congress Control Number: 2012951520

10 9 8 7 6 5 4 3 2 1

Dear Reader,

Thank you for choosing *VMware vSphere Design*, *Second Edition*. This book is part of a family of premium-quality Sybex books, all of which are written by outstanding authors who combine practical experience with a gift for teaching.

Sybex was founded in 1976. More than 30 years later, we're still committed to producing consistently exceptional books. With each of our titles, we're working hard to set a new standard for the industry. From the paper we print on to the authors we work with, our goal is to bring you the best books available.

I hope you see all that reflected in these pages. I'd be very interested to hear your comments and get your feedback on how we're doing. Feel free to let me know what you think about this or any other Sybex book by sending me an email at `nedde@wiley.com`. If you think you've found a technical error in this book, please visit `http://sybex.custhelp.com`. Customer feedback is critical to our efforts at Sybex.

Best regards,

Neil Edde
Vice President and Publisher
Sybex, an Imprint of Wiley

To my beautiful wife Tarn. You are my blessing, my inspiration, my happiness, my courage, my pride, my today, and my tomorrow. Ever yours.
—Forbes Guthrie

First and foremost, I dedicate this work to my Lord and Savior, who goes with me and will never forsake me (Deuteronomy 31:6, NIV). I also dedicate this book to my family—Crystal, Sean, Cameron, and Tim. Thank you for your love and your support; it means the world to me.
—Scott Lowe

A dedication to my family: To my parents, Ron and Carol, for instilling a strong work ethic and sense of family, and supporting my education throughout the years. I wouldn't be where I am today without you. And to my wife Lauren, who has exemplified the virtue of patience and continues to be the most loving and supporting person I know. I love you hunny bunny.
—Kendrick Coleman

Acknowledgments

When accepting the challenge to revise this book from our previous incarnation, I certainly underestimated the voluminous nature of vSphere 5 and the multitude of new and improved features. Indubitable credit is due to VMware for delivering such progress. But that's not the point I'm trying to make. I've spent many evenings and weekends laboring over this task. Second time around, it has only been possible to devote so much time to it with the help and support of my wife, Tarn. Without her encouragement, this book would never have had my initial contributions and never been updated this year. Her *mateship* continues unbounded, and I'm forever grateful for the partnership we have. I would never have gotten through it once more. Thank you. Again.

I would like to acknowledge the book's other primary coauthor, Scott Lowe. He has stepped up to the plate with this edition, and his deep knowledge, experience, and style has enhanced the book considerably. Kendrick Coleman has been a fantastic addition to the team. He worked hard to provide wonderful insight into his disparate design topic, in an area that now complements the rest of the book so well. Thanks!

I continue to be amazed at the number of publishing-house staff involved in a single project. First and foremost I would like to thank Mariann Barsolo, the acquisitions editor, for her project steerage and the encouragement she gave to all the authors. We were all incredibly blessed to have Jason Boche (the Virtualization Evangelist) as our technical editor once more to check the subject matter and make suggestions so that every area was covered appropriately and was technically correct.

I grew up and was educated in Scotland, and I've lived and worked across the UK and in several English-speaking nations including New Zealand, Australia, and subsequently Canada. My interpretation of country-specific English lexicon, mixed with unique colloquialisms, makes for a frankly weird concoction of vernacular English. The Sybex editors' ability to decipher and translate this into something representing a sane American English dialect was undoubtedly no easy task. (However, I still maintain that the Queen's English is the only true authority, and virtualization should really be spelt with an *s*.) Lisa Bishop as the development editor probably bore the brunt of this and was always central to the smooth passage of the editing process. The Sybex team of Pete Gaughan, Connor O'Brien, and Jenni Housh kept a close guard on standards and made sure things were ticking along. The production team, headed up by Eric Charbonneau as the production editor, with Tiffany Taylor as the copy editor tidying the grammar into something respectable. Proofreader Nancy Bell's rapier-like eye for detail helped spot all the little mistakes that the rest of us managed to miss along the way, and Ted Laux had the unenviable but crucial task of indexing the text—thanks, guys.

From a technical perspective, the vast collection of resources from the VMware community, the bloggers, the book writers, the podcasters, the VMworld presenters, the instructors, and the forum members all helped immensely. My knowledge and understanding of the vSphere product line is directly attributable to all of you. There are unfortunately too many people who deserve rich thanks, but for fear of this turning into an Oscar speech, I can only say a huge *thank you*. You all know who you are. Here is a big virtual pat on the back from me.

Finally, I'd like to thank the wonderful baristas of South Granville and Fairview in Vancouver for their delicious highly caffeinated beverages and working refuge. Caffè Artigiano, Waves Coffee House, and in particular Starbucks have provided decidedly agreeable cups of joe to fuel me this time around.

—Forbes Guthrie

As with any book, many people deserve credit for the book you're now reading. First and foremost, I'd like to thank Forbes for the opportunity to collaborate on this revised edition. Forbes's outstanding work and unwavering commitment to getting this book done (and done in the highest possible quality) is a testament to his character, and I sincerely appreciate the chance to work with him once again. We've collaborated a couple of times: first for the original *VMware vSphere Design*, and again when he was a contributing author for *Mastering VMware vSphere 5*. In all instances, his work has been exemplary. Thanks for everything, Forbes—it has been a blast working with you once again.

My thanks also go to Kendrick Coleman, who was willing to jump in and contribute his technical expertise. The addition of his vCloud Director material helps fill an important gap in providing design guidance for an important part of many vSphere environments, and I appreciate his participation in this project.

Of course, there are so many people who need to be called out that it would be impossible to list all of them. I'd like to echo Forbes in saying thanks to all the great bloggers and other VMware community participants for their selfless contributions. It is fantastic to have such a great community from which to draw support and encouragement.

I'd also like to thank the team at Sybex: Mariann Barsolo, Pete Gaughan, Lisa Bishop, Eric Charbonneau, Tiffany Taylor, Nancy Bell, Ted Laux, Neil Edde, and the rest of the Sybex/Wiley team who worked so hard to bring this book to print. As with the previous books I've done with Sybex, it's been a pleasure, and I'm looking forward to more books in the future.

My thanks once again go to our technical editor, Jason Boche, for his efforts on this book. Jason, thank you for your honest feedback; I do believe this book is better as a result of your input.

My thanks also go to my Chinese exchange student, Tim, for bringing so much humor and laughter into our house during the writing of this book.

Last, but most certainly not least, I'd like to thank my family for putting up with me as I raced to meet deadlines while trying to balance work and home life. There is no doubt that without the support of my wife and my family, I would not have been able to complete this project. It's for you that I work so hard—thank you for your support.

—Scott Lowe

I've been approached many times to help author a publication, but I have always turned down the chance for one reason or another. When I was offered the opportunity to publish a chapter alongside Forbes and Scott, there was no way I could ever turn it down. Many thanks to Forbes and Scott for allowing me to tag along and publish a chapter in one of the best technical books ever to hit the shelf. It's an honor to share the stage with these gentlemen.

The VMware community at large also deserves a great deal of gratitude for everything you read in this book. One person can't be held responsible for dictating best practices; we rely on a community to decide. Many bloggers deserve credit for the knowledge they've shared and that has been transferred to this book. Without your real-world experience in the field, we wouldn't have the vast amount of information that's available to us all. You all rock.

Thank you to the Sybex/Wiley team for giving me this opportunity. I'm very grateful and appreciate all the work that goes on behind the scenes that makes publications such as this a success.

—Kendrick Coleman

About the Authors

Forbes Guthrie is an infrastructure architect who specializes in virtualization. He has worked in a variety of technical roles for over 14 years and achieved several industry certifications including VMware Certified Professional–Datacenter Virtualization (VCP2/3/4/5-DV) and VMware Certified Advanced Professional 5–Datacenter Design (VCAP5-DCD). His experience spans many different industries, and he has worked in Europe, Asia-Pacific, and North America. He holds a bachelor's degree in mathematics and business analysis and is a former Captain in the British Army.

Forbes was the lead author of this title's venerable first edition, co-authored by Scott Lowe and Maish Saidel-Keesing. He contributed to Scott's acclaimed *Mastering VMware vSphere 5* book. Forbes has also spoken at VMware's own VMworld conference on the subject of design and vSphere 5.

Forbes' blog, `www.vReference.com`, is well regarded in the virtualization field and is aggregated on VMware's Planet V12n website. He is probably best known for his collection of free reference cards, long revered by those studying for their VMware qualifications. Forbes has been awarded the luminary designation of vExpert by VMware for his contribution to the virtualization community for the last three years in a row. His passion and knowledge have also been rewarded with the peer-reviewed top virtualization bloggers listing for the last four years running.

Scott Lowe is an author, a blogger, and a consultant focusing on virtualization, networking, storage, and other enterprise technologies. Scott is currently a technical architect at VMware, focusing on virtual networking; previously he worked as a technologist at EMC Corporation.

Scott's technical expertise extends into several areas. He holds industry certifications from Cisco, EMC, Microsoft, NetApp, and others. He's also one of the few people who have achieved the status of VMware Certified Design Expert (VCDX); Scott is VCDX #39. For his leadership and contributions in support of the VMware community, Scott is a four-time VMware vExpert award recipient (2009, 2010, 2011, and 2012).

Scott has published numerous articles on virtualization and VMware with a number of different online magazines, and he has been a featured speaker at numerous virtualization conferences as well as VMworld. Scott has spoken at four consecutive VMworld conferences (2009, 2010, 2011, and 2012). In addition to contributing to the first edition of this book, he has three other published books: *Mastering VMware vSphere 4*, *VMware vSphere 4 Administration Instant Reference* (with Jase McCarty and Matthew Johnson), and the best-selling *Mastering VMware vSphere 5*, all by Sybex.

Scott is perhaps best known for his acclaimed blog at `http://blog.scottlowe.org`, where he regularly posts technical articles on a wide variety of topics. Scott's weblog is one of the oldest virtualization-centric weblogs that is still active; he's been blogging since early 2005.

Scott lives in the Denver, Colorado, area with his wife Crystal, his two youngest sons (Sean and Cameron), and his Chinese exchange student (Tim).

Kendrick Coleman is an infrastructure architect focused on enterprise datacenter technologies. In his daily role, he is responsible for being a design and integration expert on many VMware products. Kendrick holds the following VMware certifications: VCP3/4/5-DV, VCAP4/5-DCD, VMware Certified Advanced Professional 4–Datacenter Administration (VCAP4-DCA), and VCP5-Cloud as well as being a Cisco Certified Network Associate (CCNA).

Kendrick has also been recognized as a VMware vExpert (2010, 2011, and 2012) for his contributions to the VMware community.

Kendrick's blog, `www.kendrickcoleman.com`, is known for having various articles focused on vSphere network design, free VMware tools, step-by-step tutorials, and vCloud Director. Year after year, it's ranked as an influential virtualization blog. Kendrick has spoken at three consecutive VMworld conferences (2010, 2011, and 2012) and continues to travel the country to speak at VMUGs and other trade shows.

Contents at a Glance

Contents

Introduction

This book has always stood out as a particularly interesting project for us. A multitude of vSphere textbooks are available, explaining every facet of configuring ESXi and vCenter. If you want to know how to do something in vSphere, you're literally spoiled for choice. However, in our minds, few resources properly encompass the design process. They exist for very specific features; but not many cover the entire design of a vSphere implementation in sufficient depth.

This revised, updated, and largely rewritten second edition of *VMware vSphere Design* has been thoroughly overhauled to encompass all the great new changes that have been introduced in vSphere up to and including version 5.1. We've been blown away by the sheer volume of improvements and additions to this product. Every area of vSphere design has been affected deeply, and the revamped book reflects this.

vSphere is the leading industry standard for hypervisors. It's simply the best enterprise solution available today. It has become this popular largely because of its wide range of features, efficiency, and flexibility. But for it to perform effectively in your datacenter, you must have a suitable architecture in place. This book is written to help you achieve that.

In addition to the changing landscape of vSphere in the datacenter, the book now incorporates another key tenet of VMware's datacenter portfolio: vCloud Director, its private/public cloud integration piece. This emerging technology is now deeply intertwined in the future of vSphere and becoming an essential skill for anyone currently involved or interested in vSphere design.

Above all, this is a technical book about a very complex subject. It's not concerned with the minutiae of every command-line tool, but rather with the underlying concepts. As vSphere has evolved from the early ESX days, it has grown in size to the point that every detail can't be covered in a single tome. But we sincerely believe this book fulfills its intended purpose better than anything else available. We'll dive into some areas not traditionally covered in such depth.

To that end, this book isn't a how-to manual with endless bullet-point instructions, but one that aims to make you think a little. It's for those of us who plan, design, implement, and optimize vSphere solutions. We hope it will challenge some of your preconceptions regarding the norm or what you consider best practice. Just because you designed a particular configuration one way in the past, doesn't mean it's a best fit for the next rollout. Here we try to question that prescriptive bias. Usually, that choice exists because different situations call for different answers. If there was one best solution for every case, then frankly no one would consider it a design choice at all.

This book isn't just for consultants who week by week deliver architectural solutions (although we hope you guys are here for the ride, too); it's for anyone who runs vSphere in their environment. It should make you question why things are set up the way they are, and encourage you to examine how to improve your environment even further.

There are constant advances in hardware, and vSphere is an ever-evolving tool, so it's always worth considering your existing deployments. Even if the hardware and software remain static in your environment, you can bet that new VMs will continue to appear. Nothing stands still for long, so your design should also be constantly growing to embrace those changes.

Each design decision has its own impact, and often these have a domino effect on many other elements. vSphere involves many disparate skills, such as guest OSes, server hardware, storage, and networking; and that's before you begin to consider the actual hypervisor. One of the hardest parts of a creating a viable design is that normally, no individual choice can be made in isolation. Although this book is naturally split into chapters, sections, and subsections, it's only when the design is considered as a complete solution that it can truly succeed.

The book employs several techniques to understand how you can approach design: the critical requirements and constraints; the impacts, benefits, and drawbacks of each choice; the dependencies on and relationships between each decision; and ultimately how to decipher what is best for you.

Who Should Read This Book

This book focuses on the design aspects of vSphere. It isn't primarily intended to teach you how to complete certain vSphere tasks, but rather to make you think about the *why* behind your different architectural decisions. We expect this book will be most useful for the following readers:

- Infrastructure architects designing new vSphere environments
- Engineers and architects charged with maintaining existing vSphere deployments, who wish to further optimize their setup
- Anyone who appreciates the basics of vSphere but wants to learn more by understanding in depth why things are the way they are
- Long-time experts who are always searching for that extra nugget of hidden information

Ways to Read the Book

There are several ways to approach this book. Clearly, you can read it from cover to cover, and we certainly encourage anyone wanting the fullest understanding of vSphere design to do so. Alternatively, if you need to brush up your knowledge on one key area, you can read each chapter in isolation. Or, if you need a specific answer to a key design decision, you should be able to jump in and use this as a reference book. *VMware vSphere Design* has been written so each section stands on its own, if that is all you need from it, but it should also be a jolly good read if you want to sit down and immerse yourself.

Other Resources Available

We're often asked for good sources of vSphere information, for those seeking *absolute knowledge*. Fortunately, there is a plethora of good places to look. The first stop for anyone (beyond this book, obviously) is VMware's own library of technical product documentation, which you can find at `www.vmware.com/support/pubs`. Along with the standard PDFs, the site also offers a wide variety of whitepapers, best practices, case studies, and knowledge-based articles.

Sybex has a number of excellent vSphere-focused books, such as *Mastering VMware vSphere 5*, a VCP5 Study Guide, a *vSphere PowerCLI* reference, and the *vSphere 5 Administration Instant Reference*, among others. A strong community of VMware users share knowledge through a number of different channels. The VMware forums at `http://communities.vmware.com/community/vmtn` are an excellent source of information and support for specific queries. There are a good number of vSphere-oriented blogs, the best of which tend to be aggregated on the popular Planet V12n site at `www.vmware.com/vmtn/planet/v12n`. Finally, if you want something a little closer to home, user groups are available in many places (see `http://vmware.com/vmug`), where you have the chance to meet other VMware users face to face to discuss and learn more about vSphere.

What You Need

To get started with *VMware vSphere Design*, you should have a basic understanding of virtualization, vSphere itself, and the associated VMware products. Both networking and storage concepts are discussed, because they're integral to any vSphere architecture, so a basic knowledge of them is assumed. The more hands-on experience you have with vSphere, the more you're likely to get out of this book. However, you don't need to be an expert beforehand.

No specific hardware or software is required while following this book, as long as you've seen the product before. But a lab is always useful to test some of many concepts we discuss. A simple nested VM lab run on a single platform should be sufficient to practice and explore most of the book's content.

What's Inside

Here is a glance at each chapter:

Chapter 1: An Introduction to Designing VMware Environments We begin by introducing you to the design process for vSphere delivery. This chapter explains how to understand the basic requirements and how to assess and then design a successful, valid implementation.

Chapter 2: The ESXi Hypervisor This chapter explains the fundamental design choices around vSphere's ESXi hypervisor. The chapter looks into the architecture behind ESXi and examines the methods and considerations when deploying it across different organizations. We also provide design advice for its subsequent configuration and management.

Chapter 3: The Management Layer In this chapter, we look at many of the software management pieces and how best to use them in different design configurations.

Chapter 4: Server Hardware This chapter provides an in-depth examination of the components that make up a server and how each one affects the performance of vSphere. You need to consider many factors when selecting server hardware, and we look at them, including scaling-up versus scaling-out approaches. We also debate the merits of blade and rack servers.

Chapter 5: Designing Your Network This chapter covers the complex decisions you need to make to ensure that network traffic provides sufficient throughput, redundancy, and security. We look how different vSphere components can affect those designs, and we provide some example configurations.

Chapter 6: Storage In this chapter, we analyze the different factors that influence a complete virtualization storage strategy, comparing availability, performance, and capacity. We contrast different storage protocols and explain how to configure multipathing in different setups. Finally, we examine vSphere 5's new storage features and how they can enhance your design.

Chapter 7: Virtual Machines In this chapter, we describe each VM component in turn, to help you understand how VMs should be designed to make the most efficient solution for you. We look at how to optimize the OS and the applications within VMs and then explain different methods of efficiently replicating the VM design though the use of clones and templates. Additionally, we look at some techniques to protect those VMs with clustering solutions, and how Infrastructure Navigator can identify the interrelationships between VMs.

Chapter 8: Datacenter Design This chapter examines in detail each element of a vSphere inventory's hierarchy. It looks at the importance of clusters in the design and how to successfully implement the resource-management and redundancy features of a cluster. We discuss resource pools, DRS, DPM, the new version of HA, and FT, and what interdependencies exist when they're used in combination.

Chapter 9: Designing with Security in Mind Chapter 9 highlights some of the areas that security-conscious environments can use to ensure that vSphere is suitably strengthened. It explains the different security measures included in the hypervisor, examines the management tools, and discusses how best to tighten that security as required.

Chapter 10: Monitoring and Capacity Planning This chapter explains the concepts of monitoring and capacity planning. Monitoring relates to the present or recent past, whereas capacity planning looks to the future. The chapter also examines some of the common tools used for both and how to involve them in your design.

Chapter 11: Bringing a vSphere Design Together In this chapter, we return to the overall design strategy by looking at a specific example through a design for a fictitious company. We discuss several of the decisions made during the design, examine the justifications behind those decisions, and consider alternative choices that could have been made.

Chapter 12: vCloud Design This chapter highlights the topics involved in architecting a successful vCloud Director implementation. We examine the role vCloud Director plays in a cloud infrastructure and dive into designing individual vCloud Director components, such as Provider vDCs and Organization vDCs.

How to Contact the Authors

We welcome feedback from you about this book or about books you'd like to see from us in the future. You can reach Forbes Guthrie by writing to `forbesguthrie@vReference.com`, on Twitter at `@forbesguthrie`, or by visiting his blog at `www.vReference.com`. You can reach Scott Lowe at `scott.lowe@scottlowe.org`, on Twitter at `@Scott_Lowe`, or by visiting his blog at `http://blog.scottlowe.org`. You can reach Kendrick Coleman at `kendrickcoleman@gmail.com`, on Twitter at `@KendrickColeman`, or by visiting his website, `www.kendrickcoleman.com`.

Sybex strives to keep you supplied with the latest tools and information you need for your work. Please check the book's website at `www.sybex.com/go/vspheredesign2e`, where we'll post additional content and updates that supplement this book should the need arise.

Chapter 1

An Introduction to Designing VMware Environments

Designing VMware vSphere environments can be a complex topic, one that means many different things to many different people. In this chapter, we'll provide an introduction to designing VMware vSphere implementations. This introduction will give a preview of some of the more detailed discussions that take place in later chapters and will provide a framework for how the other chapters fit into the overall process.

This chapter will cover the following topics:

- The importance of functional requirements in VMware vSphere design
- The what, who, and how questions involved in VMware vSphere design and why they're important
- An overview of the VMware vSphere design process

What Is Design?

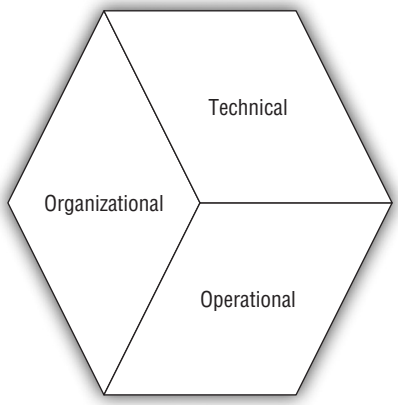
When we talk about "designing your VMware vSphere environment," what exactly does that mean? In the context of VMware vSphere, what is design? What does design entail? These are excellent questions — questions that we intend to answer in this chapter and the coming chapters throughout this book.

In our definition, *design* is the process of determining the way in which the different elements that make up a VMware vSphere environment should be assembled and configured to create a virtual infrastructure that is strong yet flexible. Design also includes the process of determining how this virtual infrastructure will integrate with existing infrastructure as well as how the virtual infrastructure will be operated after the implementation is complete.

That's a reasonable definition; but for someone who is new to VMware vSphere design, does this really describe what design is? Does it help you understand the nature of design, or what makes up a design?

In looking at a VMware vSphere design, we can say that it has three key facets: the technical or structural facet, the organizational facet, and the operational facet. Figure 1.1 shows how these three facets are all part of the larger entity that we refer to as *design*.

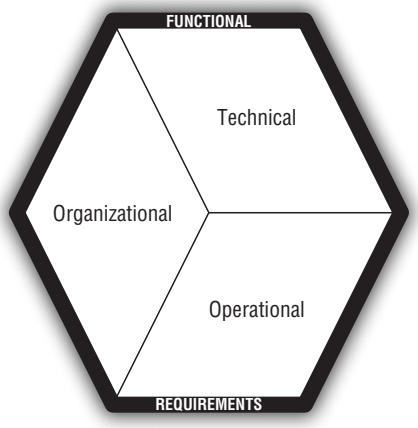
FIGURE 1.1
The different parts of VMware vSphere design are merely facets of a larger entity.



These three facets serve to organize the design in a way that is logical to us, grouping together information, decisions, criteria, constraints, and standards. We'll explore these facets in more detail later in this chapter in the section titled "The Facets of vSphere Design."

When defined or described this way, VMware vSphere design seems simple. But as you'll see in this book — or perhaps as you've already seen, depending on your experience — it can be complex. Even in the most complex of designs, however, a single unifying element brings the different facets together. What is this single unifying element, as illustrated in Figure 1.2? It's the functional requirements of the design.

FIGURE 1.2
The functional requirements unify the different facets of the design.



Functional requirements are incredibly important. In fact, we can't stress enough the key role that functional requirements play in VMware vSphere design (or any IT design task, for that matter). Functional requirements are important because they answer the question "What *things* should this design *do*?"

It's important to remember that companies implement VMware vSphere for a reason, not just for the sake of having vSphere installed. As much as VMware would love for that to be the case, it's not. In every instance, there's a driving factor, a force, a purpose behind the implementation. There's a reason the company or organization is implementing VMware vSphere. That reason, naturally, varies from customer to customer and organization to organization.

Here are some example reasons taken from our own experience in the virtualization industry:

Consolidation The company or organization has too many physical servers and needs to reduce that number. The need to reduce the number of physical servers can be driven by any number of reasons, including a need to reduce data-center space usage, a need to cut power and cooling costs, or an attempt to reduce hardware refresh costs.

New Application Rollout The company or organization is deploying a new application or a new service in its data center, and it has chosen to use virtualization as the vehicle to accomplish that deployment. This may be a deployment of a new version of an application; for example, a company currently using Exchange 2007 may decide to roll out Exchange 2010 in a virtualized environment on VMware vSphere. As another example, a company deploying SAP may choose to do so on VMware vSphere. The reasons for choosing to deploy on a virtualized environment are too numerous to list here, but they can include increased utilization, simplified deployment, and better support for a disaster recovery/business continuity (DR/BC) solution.

Disaster Recovery/Business Continuity (DR/BC) The company or organization is in the midst of developing or enhancing its DR/BC solution and has chosen to use virtualization as a key component of that solution. Perhaps the company is using array-based replication and wishes to use VMware vSphere and VMware Site Recovery Manager (SRM) to provide a more automated DR/BC solution. The choice to use virtualization as a component of a DR/BC solution is almost always a financial one; the company or organization wishes to reduce the amount of downtime (thus minimizing losses due to downtime) or reduce the cost of implementing the solution.

Virtual Desktop Infrastructure The company or organization wishes to deploy a virtual desktop infrastructure (VDI) in order to gain desktop mobility, a better remote-access solution, increased security, or reduced desktop-management costs. Whatever the motivation, the reason for the VMware vSphere environment is to support that VDI deployment.

As you can see, the reasons for adopting virtualization are as varied as the companies and organizations. There is no one reason a company will adopt virtualization, but there will be a reason. There are often multiple reasons. These reasons become the basis for the functional requirements of the design. The reasons are the *things* the design must *do*. Functional requirements formalize the reasons why the company or organization is adopting VMware vSphere and turn them into actionable items that you'll use to drive all the other decisions in the design.

Think about some of the examples we just provided. Does the organization plan to virtualize a new rollout of Microsoft Exchange Server 2010? If so, then the VMware vSphere design had better accommodate that functional requirement. The design must specifically accommodate Microsoft Exchange Server 2010 and its configuration needs, supportability requirements, and resource constraints. If you fail to properly account for the fact that Microsoft Exchange Server 2010 will run in this virtualized environment, then you've failed to consider one of the design's

functional requirements — and, in all likelihood, the implementation will be a failure. The design will fail to *do* the *thing* the company needs it to do: run Microsoft Exchange Server 2010.

With this in mind, you can look back at Figure 1.2 and better understand how the functional requirements both surround and unify the facets of VMware vSphere design. Continuing in our example of an organization that is deploying Exchange Server 2010 in a virtualized environment, the functional requirements that derive from that reason affect a number of different areas:

- The server hardware selected needs to be capable of running the virtual machines configured with enough resources to run Microsoft Exchange Server 2010.
- The virtual machines that run Exchange will, most likely, need to be configured with more RAM, more virtual CPUs (vCPUs), and more available disk space.
- The configuration of Exchange Server 2010 will affect cluster configurations like the use of vSphere High Availability (HA), vSphere Distributed Resource Scheduler (DRS), and vSphere Fault Tolerance (FT).
- The cluster configuration, such as the ability (or inability) to use vSphere FT, will in turn affect the networking configuration of the VMware ESXi hosts in the environment.
- Operational procedures need to be included in the design as a result of the use (or lack of use) of features like vSphere HA, vSphere DRS, and vSphere FT.

The list can go on and on, but at this point you should get the idea. The functional requirements affect almost every decision point in every facet of the design; as a result, they lie at the core of creating a VMware vSphere design. Any design that doesn't directly address the organization's functional requirements is a poor design, and the implementation won't be a success. Any consultant or VMware vSphere architect who attempts to design a vSphere environment without knowledge of the functional requirements will fail. After all, the functional requirements are the targets the design is aiming to hit; how can the design hit those targets if the targets aren't known and understood?

Interestingly, although the functional requirements directly affect the decision points — things like what servers to use, the form factor of the servers, the number and type of network interface cards (NICs), and so on — these decision points also affect the functional requirements. An inherent interdependency exists between the functional requirements and the decisions, as shown in Figure 1.3.

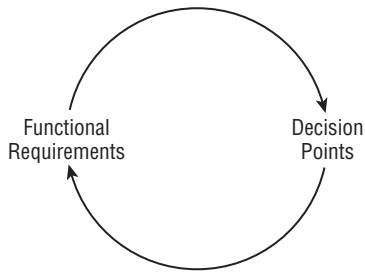


FIGURE 1.3
Functional requirements and design decision points are interdependent.

NOTE Although we've been focusing primarily on requirements in this discussion, formal VMware design documentation typically refers to four primary factors that drive your design: requirements, risks, constraints, and assumptions. We've already discussed *requirements*; these represent the specific features or functions the design must provide or satisfy. *Constraints* are decision points — such as the type of server you'll use, the type of storage you'll use, or the way in which you'll connect to an existing network — that have already been made and can't be changed. *Risks* represent specific areas where the design may not satisfy the requirements or the constraints. For example, if the design has a certain storage-capacity requirement but also has a constraint that prevents the design from meeting that capacity requirement, this is a risk. Finally, *assumptions* are requirements or constraints inserted into the design by the vSphere architect in order to fill in missing information. For example, in order to plan for future growth, certain growth requirements need to be known. If these requirements aren't known, the architect can use an assumption to fill in the blanks when creating the design.

Keep these four factors in mind as you continue to review the process of vSphere design.

As a result of this interdependency, you'll find that creating a design is often an iterative process. Based on the functional requirements, you make a decision. Then, based on that decision, you ensure that the decision is capable of supporting the functional requirements. If so, you proceed with other decision points. If not, you revise the decision point based on the functional requirements. This iterative process again underscores the importance of the functional requirements in the creation of the design.

At the beginning of this section, we told you that design is the process of determining the way in which the different elements that make up a VMware vSphere environment should be assembled and configured to create a virtual infrastructure that is strong yet flexible. When we factor in the key role that functional requirements play in unifying the technical, organizational, and operational facets of a design, perhaps a better definition is that design is the process of determining the way in which the different elements that make up a VMware vSphere environment should be assembled and configured in order to satisfy the functional requirements. Or, in simpler terms, design is making the VMware vSphere environment *do* the *things* it needs to do.

Now that you have a better understanding of what VMware vSphere design is and why it's important, in the next section we'll take a closer look at the facets of design.

The Facets of vSphere Design

As we described in the previous section and illustrated in Figure 1.1, your design must address three facets, or the design is incomplete. These three facets — technical, organizational, and operational — are unified by the functional requirements; but within each facet, a wide variety of decision points must be specified in the design.

The best way to understand how these facets differ from each other is to look at the types of decisions that fall in each facet. This is graphically depicted in Figure 1.4.

In each facet of the design, you'll make decisions based on the functional requirements, followed by an iterative review (as illustrated in Figure 1.3) to ensure that the functional requirements are still met based on the decision. In this section of this chapter, we'll take a deeper and more detailed look at these facets, examining some of the decision points that are involved.

We'll start with the technical facet.

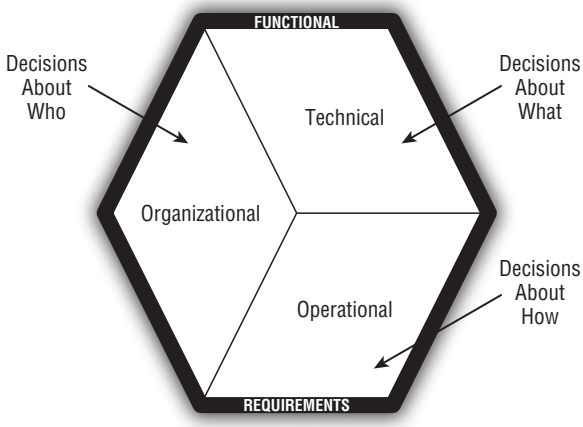


FIGURE 1.4
Each facet of the design primarily addresses a different type of decision, such as who, what, or how.

The Technical Facet

The *technical facet* is the facet that IT people most closely identify with design. It involves the pieces and parts of technology that make up the final environment: things like what servers to use, what quantity of random access memory (RAM) the servers will have, what configuration the storage array will use for its datastores, and what the networking configuration will look like. You may also see the technical facet referred to as the *physical design*, although it incorporates certain logical aspects as well. These are all decisions about what will or won't be included in the design, and these decisions fall into the technical facet, as illustrated in Figure 1.5.

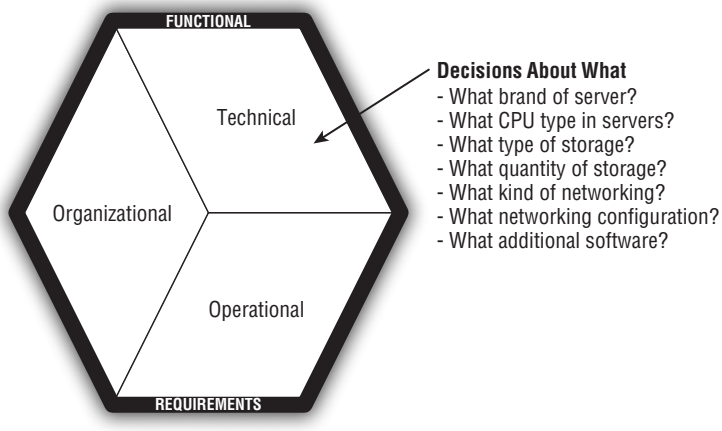


FIGURE 1.5
The technical facet includes the "what" decisions that are familiar to many IT professionals.

It's important to be sure the technical facet is as complete as possible, so the design should include — at the very least — decisions in the following technical areas:

- The number and type of servers in the environment
- The number, type, and speed of the CPUs in the servers
- The amount of RAM in the servers
- The type of connectivity to the shared storage
- The type or configuration of the shared storage
- The number of physical NIC ports available
- The manufacturer and model of the NICs in the servers
- The exact configuration of the virtual switches (vSwitches) and distributed vSwitches in the environment
- The amount of power required by the equipment
- The amount of cooling required by the equipment
- The amount of rack space or floor space required by the equipment

This is, of course, just a small list to get you started thinking about the detail you should provide when crafting a design for a VMware vSphere environment. Subsequent chapters examine each of these areas in much more detail. For example, VMware vSphere networking is covered in detail in Chapter 5, "Designing Your Network"; Chapter 6, "Storage," discusses shared storage in more depth.

A complete and thorough design addresses more than just the technical facet, though. Your design should also address the organizational facet.

The Organizational Facet

Although the technical facet is important, equally as important is the *organizational facet*. It's concerned with questions centered on "who," as you can see in Figure 1.6.

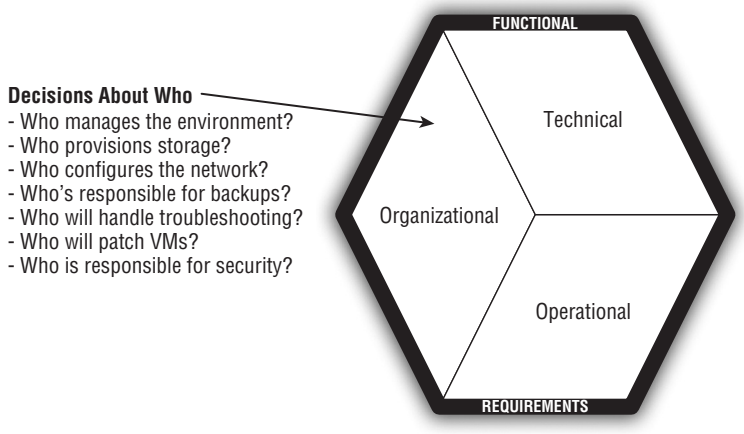


FIGURE 1.6
The "who"-focused decisions of a VMware vSphere design fall into the organizational facet.

You might initially think that these "who"-focused questions aren't important or aren't your responsibility. Aren't these the sort of decisions that should be made by the customer? In a certain respect, yes — these decisions are driven by the functional requirements every bit as much as the "what" questions in the technical facet. As you'll see later in this chapter, in the section "The Process of Design," gathering the functional requirements from the customer or organization (if it's your own organization) means gathering information about who is responsible for the various tasks within a virtualized infrastructure.

The other thing to consider regarding these "who"-focused questions, though, is the fact that the customer or company may not know or understand who will be responsible for certain aspects of the design. For organizations that are new to virtualization, the convergence of server administrators, network administrators, storage administrators, and security administrators often means they're confused and don't understand who can or should be responsible for the different areas of the new VMware vSphere environment. By embedding the answers to these questions in your design, you can help the customer (or your own organization) better understand how these responsibilities should be divided and who should be responsible for each area.

The final facet of VMware vSphere design addresses an equally important but often overlooked type of question: how should the environment be operated? This is the operational facet, and we discuss it in the next section.

The Operational Facet

The *operational facet* of a VMware vSphere design answers questions focused on "how," such as those illustrated in Figure 1.7.

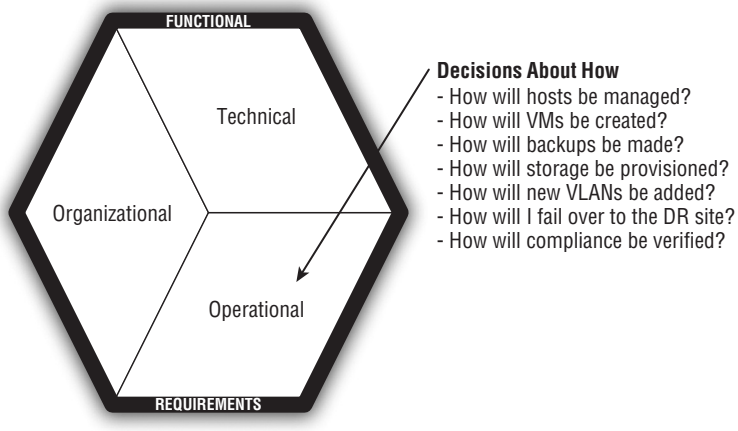


Figure 1.7
Decisions about how you'll operate a VMware vSphere environment fall into the operational facet.

As with the organizational facet, you might ask, "Why would I need to define operational procedures in a VMware vSphere design? Shouldn't these sorts of operations be tasks the organization already knows how to do?"

In the event of an organization that is new to virtualization, the answer to that question is no. Virtualization changes the way the data center operates — and the customer or company implementing virtualization for the first time must have these operational decisions spelled out in the design.

Even for organizations that are familiar with virtualization, the operational facet is still a critical part of a complete and comprehensive VMware vSphere design. It's possible, even likely, that the "what" decisions made in the technical facet of this design are different than the "what" decisions made in the technical facet of earlier designs. Server models change. Storage vendors introduce new products with new functionality — and new operational requirements. Networking vendors change the way their products work over time. All of these factors add up to a need to include operational information in every design to ensure that the organization implementing the design has the information it needs to operate the environment.

As an example, consider an organization that adopted virtualization a couple of years ago. It deployed VMware Infrastructure 3 (VI3) on HP ProLiant rack-mounted servers attached to a NetApp storage array via Fibre Channel (FC). Now, the company is implementing VMware vSphere 5.1 on Cisco Unified Computing Systems (UCS) attached to a Dell Compellent storage array via FC over Ethernet (FCoE). Do you think the operational procedures from the last implementation will be the same as the operational procedures for this new implementation? No, of course not. Just as technology changes over time, operations change over time as well. This is why the operational aspect is important not only for new VMware vSphere users but also for existing users.

Grouping design decisions into these three facets — technical, organizational, and operational — helps us, as architects, ensure that our design is complete and that it addresses all the aspects of creating a vSphere environment that satisfies the functional requirements. What these facets don't do, though, is help us to determine the specific way in which to apply the functional requirements. What are the guiding qualities or principles that help us make the decisions within each of these facets? That's the topic of the next section.

The Principles of Design

In this section, we'll discuss the principles of design — the guiding ideas or thoughts that shape and influence the decisions we make within each facet (technical, organizational, and operational) in order to satisfy the functional requirements.

At a very high level, there are five basic principles of design:

- Availability
- Manageability
- Performance
- Recoverability
- Security

We'll use the acronym AMPRS to refer to these design principles. These principles are, for the most part, pretty self-explanatory, but we'll look at each of them in a bit more detail next.

Availability

As you make decisions that will create a vSphere design, one principle to consider is availability. Availability encompasses a number of areas, including uptime and downtime, reliability, redundancy, and resiliency. Note that some aspects of performance are also on occasion associated with availability; for example, in the context of a service-level agreement (SLA), availability

might also encompass application response times or application latency. We'll have a separate discussion on performance later in this chapter, but be aware of the close relationship between these two principles. With regard to the facets of design, the principle of availability typically has the greatest effect on decisions in the technical facet.

In some cases, the functional requirements explicitly call out availability; for example, a functional requirement may state that the vSphere design must provide 99% availability. In this case, availability is explicitly noted by the functional requirements and therefore must be incorporated into the design. In other cases, the functional requirements may not explicitly state availability demands. In these situations, the architect has to include an appropriate level of availability in the design as the various design decisions are made. The functional requirements may only state that 10 gigabit (Gb) Ethernet is required, in which case a single 10 Gb Ethernet switch will satisfy the functional requirements. However, a single 10 Gb Ethernet switch presents a single point of failure (SPoF). Is that an appropriate level of availability for this design, when availability has not been explicitly called out?

In situations like this, an architect uses an *assumption*. An assumption provides justification for the vSphere designer's decisions within the broader framework of functional requirements and design constraints (see the sidebar in the "What Is Design?" section). When availability isn't explicitly stated, the architect can provide an assumption that the environment will be made as available as possible within the financial limitations of the project.

Manageability

Architects must also consider manageability. The principle of manageability most directly affects decisions within the operational facet because this principle involves the ongoing management or maintenance of the environment. Manageability encompasses a number of related ideas:

- Compatibility (can it be managed as part of the design?)
- Usability (how easy is it to manage?)
- Interoperability (does it integrate with other management structures in the environment?)
- Scalability (how well will it work as the environment grows?)

Performance

Performance is often called out in the functional requirements, and it can affect decisions in the technical and operational facets. For the technical facet, it can affect all manner of design decisions, from the type of servers to the kind of network switches to the storage solution that is selected. With regard to the operational facet, it's most frequently seen as an SLA that defines performance metrics, such as response times, transactions per second, and maximum number of users supported. (As we mentioned earlier, keep in mind that in some cases certain performance metrics are also associated with availability.)

Naturally, even if no performance requirement is explicitly stated, architects designing a vSphere environment should consider performance when making design decisions.

Recoverability

The principle (or quality) of recoverability includes such concepts as mean time to recover (or repair, abbreviated MTTR), maintainability, and DR/BC. Naturally, this makes recoverability related to availability; but whereas availability is more focused on preventing an outage,

recoverability targets how quickly (and with how much effort) the environment can be restored/recover from a failure.

Organizations use a couple different metrics to measure or gauge recoverability. These metrics are recovery point objective (RPO; a measure of how much data loss can be sustained in the event of a major failure or disaster) and recovery time objective (RTO; how quickly an environment can be restored to working operation). Typically, your functional requirements will include RPO/RTO metrics that must be met by the design.

Security

Every facet of the design — technical, organizational, and operational — is affected by security, and every design decision should consider security.

These five principles of design guide and shape all the design decisions. There are multiple ways to fulfill the functional requirements, but as a vSphere architect you must evaluate each option against these five principles. Does the option positively or negatively impact the availability of the design? What about the design's manageability? Or performance? What about recoverability and security? These principles provide guidance and direction on the best way to satisfy the functional requirements for a particular design.

Before we wrap up this chapter and start a more detailed look at VMware ESXi in Chapter 2, "The ESXi Hypervisor," we want to discuss one more area: the process of VMware vSphere design. It's the focus of the next section.

The Process of Design

Now that we've discussed the *facets* of design and the *principles* of design, it's time to discuss the *process* of design. In this section, we'll cover how you go about creating a VMware vSphere design, some of the tasks involved, and some of the tools you can use to complete those tasks.

We'll start with what is, as we've said before, one of the most important areas: functional requirements.

Gathering and Defining Functional Requirements

Functional requirements form the basis, the driver, for almost everything in the design. Most other decisions in the design are based on or affected by the functional requirements, so it's incredibly important to be as thorough and complete as possible during the process of gathering and defining them.

In many situations, some of the functional requirements are provided to you. For example, if an organization is adopting VMware vSphere in order to support a consolidation initiative, the business might clearly specify a functional requirement in a statement like this: "The virtualization environment must support the consolidation of 250 physical server workloads." No additional effort is required on your part to define this requirement. (But additional effort is clearly required to implement that functional requirement in the design.)

It's uncommon, in our experience, to have situations where all the functional requirements are provided. In these cases, you'll have to gather information to define the functional requirements. There are two primary ways to gather the information necessary to define the design's functional requirements:

- Reviewing documentation
- Performing interviews

NOTE You may see VMware define this type of approach to design — performing interviews to gather information — as the *consultative approach*.

Reviewing Documentation

In some cases, the customer or organization implementing VMware vSphere has documentation that outlines the functional requirements. Remembering that virtualization is implemented in order to accomplish a goal (to "do something"), documentation is often created that outlines what the organization is attempting to achieve. For example, perhaps the organization is implementing virtualization as part of a desktop virtualization initiative. In that case, some of the functional requirements of the VMware vSphere environment can be derived directly from the documentation prepared for the desktop virtualization project. If the desktop virtualization documentation specifies that the VMware vSphere environment will automatically restart desktop VMs in the event of a host failure, that should immediately sound a mental alarm — your vSphere environment will need to use vSphere HA in order to meet that functional requirement. And because you'll use vSphere HA, you'll also need to use clusters, which means you'll require redundant management connections, which affects the networking design … and so on.

In another example, suppose the organization is migrating into a new data center and has compiled a list of all the applications to be migrated. You can use that documentation to understand the applications' needs and determine the functional requirements necessary to support those needs. Perhaps the application documentation indicates that the I/O profile is primarily writes instead of reads and that the application needs to sustain a specific number of transactions per second (TPS). That information translates into storage requirements that dictate the RAID level, array type, storage protocol, and capacity in I/O operations per second (IOPS).

Although reviewing documentation can be helpful, it's unlikely that you'll find all the information you need in a company's documentation. If the organization is like a lot of others, documentation is sparse and incomplete. In these instances, you'll need to gather information by going straight to the source: the people in the organization.

Performing Interviews

Interviewing individuals in the organization or company that is implementing VMware vSphere is the second major way to gather the information necessary to understand the functional requirements.

Generally speaking, unless you've already gotten the information you need from somewhere else (and even then, you might want to conduct interviews to ensure that you haven't missed something), you'll want to interview the following people in the organization:

- Desktop support staff
- Server administrators
- Network administrators
- Storage administrators
- Security managers
- Compliance/legal staff

- Application owners
- Business leaders
- Project managers or project sponsors/owners
- Executive/managerial sponsors
- Architects

Not all designs or situations require you to speak with all these individuals, so be selective but thorough.

These individuals can provide you with information about the applications currently supported in the environment, the requirements of the applications, SLAs that are in place, dependencies between different applications or services in the data center, plans for future trends or projects, compliance or regulatory requirements, business-level requirements, financial objectives, operational aspects and workflow, and other facts that can be used to derive the functional requirements for the design.

Assessing the Environment

After you've gathered the information necessary to determine the design's functional requirements, it's then necessary to assess the current environment. Assessing the environment fills a couple of purposes:

- The results of the assessment can, in some instances, verify or clarify the information provided during the information-gathering process of defining the functional requirements. People are just people and are known to make mistakes or accidentally omit information. By performing an assessment of the environment, you can verify that the applications you were told were present are, in fact, present.
- The assessment provides useful information necessary to complete the technical facet of the design. An assessment reveals the current types and configurations of the servers in the environment, the current network configurations, and the current storage configurations. All this information is crucial to putting together a new structure that will properly interoperate with the existing structure. If the organization is currently using iSCSI, then you know that implementing FC might create interoperability issues. Having this knowledge through an assessment of the current environment helps you tailor the technical facet of the design appropriately.

You can use a number of different tools or methods to assess the environment. If the organization already has a robust management system in place, it may have the inventory, configuration, and performance information you need. If not, you'll have to start digging through the environment, gathering information from such sources as these:

- Active Directory
- LDAP directories
- Network-management tools
- Enterprise-wide logging solutions

- IP addressing documentation
- Network equipment configurations
- Server performance data
- Server configuration data

You can imagine that in anything larger than most small environments, assessing the existing environment manually like this can be time-consuming and potentially error-prone. Fortunately, VMware and other vendors have released assessment tools that help gather this information in an automated fashion, to save you time and help you avoid missing critical data. Even the virtualization community has stepped up, providing scripts and other tools that gather information about existing physical and/or virtual environments.

Examples of some tools that have been created by vendors and community members include the following:

- VMware Capacity Planner
- Various community-supplied health-check scripts
- NetIQ PlateSpin Recon (formerly Novell PlateSpin PowerRecon)
- CiRBA

We'll discuss some of these tools in Chapter 10, "Monitoring and Capacity Planning." Because part of capacity planning involves previrtualization assessment, these tools are also useful in assessing an organization's existing environment in preparation for completing a design.

At this point, you're armed with some functional requirements, the information necessary to define other functional requirements, and knowledge of the existing environment. In some instances, before you're ready to assemble the design, you may first need to perform a gap analysis.

Performing a Gap Analysis

Not every vSphere design is a *greenfield implementation*, meaning that not every design is a brand-new implementation in an environment where vSphere wasn't being used previously. In environments where an organization had already adopted VMware vSphere and is now upgrading or expanding the environment, or migrating to a new environment, you may need to perform a gap analysis.

A *gap analysis* is the process of comparing an environment's current state to its desired future state in order to determine what is needed to achieve that future state. Perhaps the current environment doesn't provide the scalability that is needed. A gap analysis can determine what portions of the design may limit the design's growth and point the way toward removing (or, at the very least, increasing sufficiently) those limits.

Once you have determined your functional requirements, gained knowledge about the existing environment, and performed a gap analysis (where necessary), you're ready to start assembling the design.

Assembling the Design

Assembling the design is the iterative process we described earlier and depicted in Figure 1.3. While assembling the design, you make decisions within each of the three facets. Those decisions, focused on the what/who/how theme, are based on the functional requirements you've been given or have defined and are guided by the five design principles (AMPRS). Each decision has a cascading effect, forcing a series of what we call *downstream* decisions, as shown in Figure 1.8.

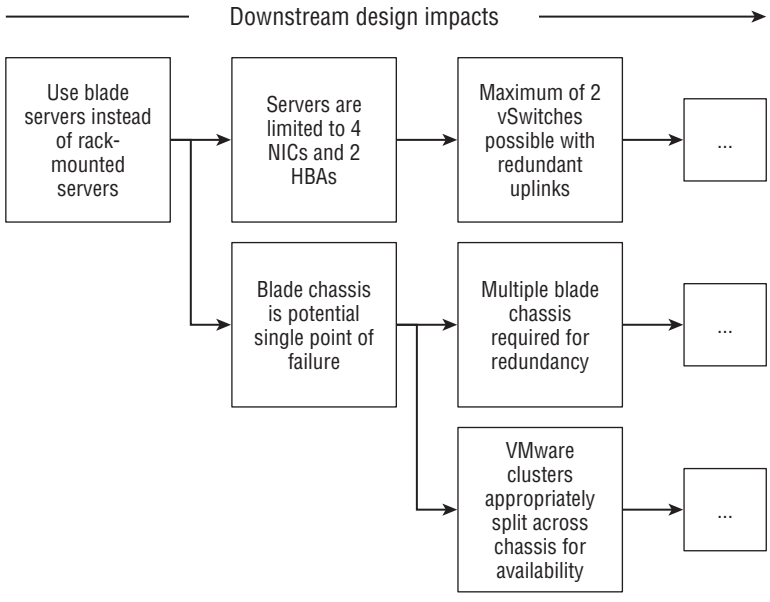


FIGURE 1.8
Each decision forces other decisions in a complex decision tree. The results of each decision must be compared against the functional requirements.

When you make a decision in the design, you then need to examine the result of that decision — and all the downstream decisions resulting from that decision — to ensure that you're still meeting all the functional requirements. If so, you continue; if not, you need to change that decision or violate one or more of the functional requirements.

VIOLATING FUNCTIONAL REQUIREMENTS MAY BE NECESSARY

Sometimes, organizations have an unrealistic view of the functional requirements. In situations like this, it may be necessary to violate a functional requirement. As long as you can show why the functional requirement is violated and can provide a potential remediation for the violation, it's up to the organization to determine whether that functional requirement really is a requirement or the design can be accepted and implemented as described.

It's here, while assembling the design, that you define standards and best practices for the VMware vSphere environment:

Standards A good design defines standards for host names, networking configuration, storage layouts, resource allocation, virtual machine configurations, and so on. Standards are important because standardization reduces complexity. When complexity is reduced, operations are simplified, and costs are generally reduced. Without standards, environments become too complex to operate efficiently — and inefficiency is the bane of a VMware vSphere environment.

Best Practices A good design both defines and enforces best practices, where those best practices align with the organization's needs and functional requirements. The term *best practices* doesn't just mean the recommendations made by vendors for configuring their products; it also means the operational processes that the organization should follow. For example, a best practice states that all Windows-based VMs should have their file systems properly aligned in the virtual disk. Yes, that's a recommendation made by VMware, but it's also an operational recommendation made by a good design to ensure the efficient and stable operation of the environment. A good design includes other best practices that are specific to this particular implementation, such as the structure of new VMs, or the configuration for new datastores.

Don't Accept Best Practices Blindly

When it comes to vendor best practices, don't accept them blindly. Examine those best practices and try to understand the reasons behind them. If the storage vendor says it's a best practice to use a particular multipathing policy, take a deeper look to understand why that multipathing policy is recommended. If a server vendor defines certain BIOS settings as best practices, try to figure out why those settings are used. Doing so will give you a deeper, more complete understanding of the environment and make you better prepared to provide implementation-specific best practices.

Assembling the design is a time-consuming and detailed process. Most, if not all, of the remaining chapters in this book focus on specific technical areas and the decisions you must make when building your VMware vSphere design.

Documenting the Design

When the design has been assembled — that is, after you've made all the decisions that need to be made in each facet of the design, and you've compared the results of those decisions (and all the downstream decisions) against the functional requirements to ensure that the requirements are still being met — then you're ready to document the design.

This portion of the design process may happen concurrently with the assembly of the design and is, for the most part, straightforward. You should ensure that your documentation addresses each of the three facets of the design. In many instances, IT professionals tend to forget about the organizational and operational facets, but these are just as important as the technical facet and deserve equal treatment in your final design documentation.

In particular, be sure the design documentation includes at least the following documents:

- A description of the functional requirements that were provided to you or defined by you
- A comprehensive description of the technical facet decisions (the "what" decisions) made in the design

- A complete review of the organizational facet decisions (the "who" decisions) made in the design
- A thorough review of the operational facet decisions (the "how" decisions) found in the design
- Build documents — sometimes referred to as *blueprints* — that describe the specifics involved in building or constructing the design specified in your documentation
- Test plans describing how you can verify that the design satisfies the functional requirements
- A high-level architectural design document that ties together all these elements and tells the story of the design

Performing the Implementation

After the design is complete and has been accepted by the organization, you may also have the opportunity to perform the implementation. Doing so affords you the opportunity to use your own documentation to build the environment.

If you aren't performing the implementation, then someone else will have to build your design. This is why it's important to be thorough and complete with the design documentation — when someone else comes along to build your design, that person won't have access to the thought processes inside your head. Be sure to provide as much information as possible in the design documentation so the build process is simple and straightforward.

Summary

Throughout this book, we'll draw on our experience and knowledge to help you understand the complexities of VMware vSphere design. In this chapter, we've discussed the three key facets of design — the technical facet, the organizational facet, and the operational facet — and the importance of each. We've shared with you the five guiding principles of design, known as AMPRS: availability, manageability, performance, recoverability, and security. We've also discussed the process of design and some of the tasks involved in creating a design. In coming chapters, we'll take a more detailed look at the different areas of VMware vSphere design, starting in Chapter 2 with an examination of VMware ESXi.

Chapter 2

The ESXi Hypervisor

Underpinning any design based on vSphere technologies is the hypervisor. ESXi host software drives vSphere deployments and makes guest virtualization possible. VMware's hypervisors have evolved rapidly over the years, and the basis of the enterprise offering has gone through a recent transition phase. It has emerged stronger on the other side with the new slim-line ESXi, more efficient, more capable, and more flexible than before.

This chapter looks closely at ESXi: what makes it tick and how to get it out across your organization. We'll dive deeply into its internal structure to understand the components that a design revolves around. There is an abundance of deployment choices, and these speak to the platform's growth and maturity. We'll compare the options, looking at the advantages of each and which may be more appropriate in different circumstances. The enterprise can now push ESXi out facilely and manage it consistently with effective policy management. So, we'll discuss the necessary configuration that is required in the design and how to effectively manage the hosts. The heart of any good vSphere architecture is a solid ESXi design.

In this chapter, you'll learn to

- Understand the evolution of the ESXi hypervisor from its ESX roots
- Understand all the components that make up the ESXi image and what makes ESXi unique as an operating system
- Deploy ESXi using the multitude of options, and the impact those options have on the resulting build
- Upgrade ESXi from previous versions, and how this affects the host configuration
- Migrate ESX hosts to ESXi
- Configure ESXi hosts and achieve the planned design
- Manage the resulting ESXi deployment

Evolution of the vSphere Hypervisor

vSphere 5 hosts run a solitary, unified hypervisor operating system. The *hypervisor* is the software that virtualizes the host's hardware and makes it available to multiple guest OSes. vSphere hosts are what is known as *type 1* hypervisors, or *bare-metal* hypervisors. Unlike a *type 2* (or *hosted*) hypervisor, which runs atop a standard OS such as Windows or Linux, a type 1 hypervisor runs natively on the physical hardware without the need for an intermediary OS. This gives the hypervisor direct access to the hardware, reducing the performance overhead of running on

an intermediary OS, and without the security and stability issues that adding another layer to the stack brings.

VMware vSphere Hypervisor

The term *VMware vSphere hypervisor* is actually a marketing term to specifically refer to the stand-alone version of ESXi that can be downloaded for free from the VMware website. It's a restricted version of the hypervisor that can't be managed by a vCenter instance. Remote connections via APIs are read-only, which limits third-party software such as backup and cloning tools, and there is a hard limit of 32 GB of physical memory for the host. It's offered as a direct response to other free hypervisors on the market and is a useful springboard for many administrators just discovering virtualization.

When we refer to the vSphere hypervisor or ESXi hypervisor in this book, we don't explicitly mean this gratis release of the software. Rather, we're alluding to the more generic term *hypervisor* with its many vSphere-licensed levels.

VMware released its GSX (type 2) and ESX (type 1) products in 2001. VMware GSX was renamed VMware Server in 2006, and support ended in 2011; it followed a lineage close to that of VMware's other hosted products such as Workstation, Player, and Fusion. The ESX enterprise hypervisor steadily evolved over the years; the latest versions, 4.0 and 4.1, were released in 2009 and 2010, respectively.

When ESX 3.5 was released in December 2007, VMware also made the first public release of ESXi 3.5. This marked the first significant fork in the ESX model, and since then VMware has released both ESX and ESXi alongside each other until the arrival of vSphere 5. Ever since the initial release of ESXi, VMware made no secret of the fact that it planned to replace ESX with the newer ESXi product. However, with the announcement of version 4.1, the company proclaimed that it would be the last of the ESX line, and all subsequent vSphere hypervisors would be ESXi based. vSphere 5.0 marked the first ESXi-only release and the death knell of ESX.

Since the introduction of the competing vSphere host with a remarkably similar moniker, ESX began being referred to as *ESX classic* to help distinguish it from its ESXi brethren. They had disparate management designs, but ESX and ESXi had far more in common than they had differences because their hypervisors were based on the same underlying code.

Both products, with the exception of the free stand-alone ESXi version, were priced and licensed identically. ESX classic and ESXi hosts can still coexist in the same cluster at the same time and share resources among VMs. Unless you work hands-on with the hosts themselves, you may not notice the difference in the client when connecting to vCenter. VM administrators, storage or network teams, and IT managers may understandably be oblivious to the difference. However, if you design or manage vSphere hosts, then you'll be interested in the differences that ESXi brought.

VMware openly stated for some time that ESXi was the destiny of the company's bare-metal virtualization line. The reality is that there is no future in ESX; the sooner a business moves to ESXi, the less it will waste on developing processes around and supporting an end-of-life product.

ESX classic consisted of three main elements that ran on the physical hardware, providing the virtualized environment for the guest OSes. Analogous versions of two of these, the

VMkernel and the Virtual Machine Monitor (VMM), are still found in ESXi. The third, the Service Console, is the key differentiator because it's no longer found in ESXi.

The Service Console, also known as the Console Operating System (COS) or VMnix, was the command-line interface and management console of ESX hosts. It was a modified Red Hat Enterprise Linux build and allowed user-privileged access to the VMkernel. It didn't have any direct access to the physical server and its hardware components, although additional hardware drivers could be installed within it. It also enabled scripts to be run and infrastructure, hardware, and third-party agents to run.

The ESXi Concept

The removal of the Service Console fundamentally changed what was possible with VMware's hypervisor. ESXi has a smaller and less demanding footprint, which means the hypervisor consumes fewer host resources to perform essentially the same functions. ESXi also uses significantly less disk space, whether local disk or boot-from-SAN space. The Service Console of ESX hosts effectively ran as a single vCPU guest, which meant all of its processes ran serially. With ESXi, those functions were moved to the VMkernel, meaning there is greater scope for those processes to run in parallel using much more sophisticated scheduling.

One unequivocal advantage of ESXi's reduced code base is the greater level of security it brings. An ESX install came in at around 2 GB of installed files, whereas ESXi currently is near 125 MB. It's easy to see that so much less code means less to keep secure with a smaller attack vector. The Service Console provided additional software that had to be secured, which ESXi avoids.

With fewer patches to apply, ESXi reduces the frequency of host-server reboots and lessens the administrative burden of regular patching. Any large enterprise with a sizable collection of ESX hosts will be only too familiar with the seemingly never-ending cycle of host patching. ESXi patches come as a single relatively small file, as opposed to ESX patches, which could be very large. Patching is also easier to manage if hosts are spread across several remote sites, particularly where slow WAN links cause issues with vCenter Update Manager's (VUM's) ability to push out these large packages. Another advantage with ESXi's patches is that they come as a single firmware-like image that updates everything. Compare this to ESX patches, which came in multiple updates, potentially with dependencies on previous patches, and required multiple reboots.

An ESXi host is also more reliable than an ESX classic host. It effectively has less code to go wrong and fewer processes running over and above the VMs. The ability of the Service Console to run third-party agents was a double-edged sword, because although it allowed you to add extra functionality, some of the available agents caused stability issues on hosts. The inability of ESXi hosts to run unmanaged code means this is no longer a concern. Additionally, the dual-image nature of ESXi means there is always a standby bootable copy of the OS to roll back to, should you have any problems with an update.

ESXi brings with it the possibility of running hosts in a practically Stateless mode, meaning host servers are more comparable to hardware appliances. The deployment techniques available for ESXi are similar to those for ESX, but the install is considerably easier. You're prompted for very little information, and the install time is incredibly short. You don't need to understand the nuances of a POSIX filesystem and how best to carve up ESX's partitions. Even rebooting an ESXi server takes considerably less time than rebooting an equivalent ESX classic host.

The simplification of host management, with no need to understand a Service Console when configuring and troubleshooting, means a lower entry bar for staff to install and maintain

new vSphere environments. The simple Direct Console User Interface (DCUI) screen is more comparable to a BIOS setup screen and far less intimidating to staff unfamiliar with Linux. If a problem exists in a remote office and there are no remote access cards or a keyboard, video and mouse (KVM) switch, then it's more feasible that an onsite employee might be able to assist in restarting a management daemon.

ESXi has so many advantages that it's clearly the better option for VMware and the company's customers moving forward. Despite the many ESX classic installations that still exist awaiting migration to ESXi, clearly the smart option—the only option—is to deploy ESXi. Thanks to vSphere's inherent abstraction, migrating VM workloads is relatively trivial and pain-free.

ESXi Design

The ESXi hypervisor shares many common elements with its older big brother ESX classic, but the main differentiator is that the Linux-based Service Console was stripped out. ESXi retains VMkernel and VMM components similar to ESX but has additional features built into the VMkernel; a new, much smaller management console; and other user-mode processes to replace the old Service Console OS functionality.

ESXi was redesigned this way to allow VMware users to scale out through a hypervisor that is more akin to a hardware appliance. The vision was a base OS that is capable of auto-configuring, receiving its settings remotely, and running from memory without disks. But it's also an OS that's flexible enough to be installed on hard disks along with a locally saved state and user-defined settings for smaller, ready-to-use installations that don't require additional infrastructure.

Removing the Service Console obviously had an impact. A number of services and agents that were normally installed had to be rethought. The familiar command-line interface with its access to management, troubleshooting, and configuration tools is replaced in ESXi. And the Linux-styled third-party agents for backups, hardware monitoring, and the like must be provisioned in different ways.

ESXi Components

The ESXi OS is built on three layers. It achieves the same VM environment as ESX classic, but it has some significant architectural differences:

VMkernel The VMkernel sits at the foundation of ESXi and is built specifically for ESXi. It's a 64-bit microkernel POSIX-styled OS, designed by VMware not to be a general-purpose OS but one specifically tuned to operate as a hypervisor. The VMkernel manages the hardware of the physical server. It coordinates all of the CPU's resource scheduling and the memory allocation. It controls the disk and network I/O stacks and handles the device drivers for all the approved hardware compatibility list (HCL) compliant hardware.

VMkernel Extensions In additional to the VMkernel, there are a number of special kernel modules and drivers. These extensions let the OS interact with the hardware via device drivers, support different filesystems, and allow additional system calls.

Worlds VMware calls its schedulable user spaces *worlds*. These worlds allow for memory protection and sharing as well as CPU scheduling, and define the basis of privilege separation. There are three types of worlds:

System Worlds System worlds are special kernel-mode worlds that can run processes with system privileges. For example, processes such as idle and helper run as system worlds.

VMM Worlds The VMM worlds are user-space abstractions that let each guest OS see its own x86 virtualized hardware. Each VM runs in its own scheduled VMM world. It presents the hardware including the BIOS to each VM, allocating the necessary vCPUs, RAM, disks, vNICs, and so on. It also determines the monitor mode for each VM, depending on the physical CPUs in the server and the guest OS selected, choosing between Full Virtualization (binary translation), or Hardware Assisted Virtualization. (The paravirtualization monitor, also known as Virtual Machine Interface [VMI], was retired with vSphere 5.0.)

User Worlds User worlds are any processes that don't need to make calls with the privileges afforded to the system worlds. They can make system calls to the VMkernel to interact with VMs or the system.

Important—and something that differentiates ESXi from many common OSes—is the fact that because the entire OS is loaded into memory when it boots up, these user-space worlds are never swapped to disk. However, these worlds can be controlled via resource pools much the same as VMs. They have CPU and memory reservations, shares, and limits. This presents an advantage over the way ESX classic worked, where the Service Console ran as one world and a single Service Console agent using excessive memory could affect other processes. This is why Service Console memory was often increased to the maximum amount, to try to prevent the hostd process from being swamped. With ESXi, these processes are better protected.

ESXi Agents

The VMkernel runs several noteworthy agents and daemons:

hostd hostd (which is stewarded by the service `mgmt-vmware`) is the primary management daemon. It's used to interface with the VMkernel and is the connection for direct vSphere Clients and remote API calls. hostd also includes the default SNMP agent, which is disabled by default.

vpxa vpxa is the agent that allows a vCenter instance to connect to the VMkernel. When you first connect a host ESXi server to vCenter, it initiates the vpxa service, which acts as the intermediary to the VMkernel's functions under the vpxuser account.

syslog The syslog daemon can forward logs to a remote syslog server.

ntpd ntpd provides the time service to keep the host synchronized to remote time servers. This is important, because several inter-server services such as high availability (HA) and directory-based authentication rely on accurate timekeeping. You can also set the VMs to synchronize their guest OS to the local host.

DCUI The DCUI is the BIOS-style yellow interface that opens on the server's console screen. It lets you set basic configuration, permit access, and restart management agents.

sfcbd The sfcbd daemon is the Common Information Model (CIM) broker provides agentless access to hardware monitoring via an externally accessible API. This reduces reliance on third-party hardware agents and SNMP.

ESXShell The ESXi Shell (previously known as Technical Support Mode [TSM]) runs as the ESXShell daemon and provides a very slim-line command-line tool. It provides local console access to the ESXCLI toolkit (`esxcli`), several other CLI tools such as `vmkfstools`, the `esxcfg-*` commands that are yet to be replicated in ESXCLI, and the real-time process monitor `esxtop`.

The last three processes are examined in more depth later in the chapter to show the effect these tools can have on the host's management.

ESXi System Image

Before we explain the installation and deployment options of an ESXi design, it's important to understand the structure of the ESXi image. The *system image* is a bootable image that is loaded into memory and runs the hypervisor. The ESXi installer uses this same system image to copy the files onto a bootable device for future boots (into the `/bootbank` partition). On its first boot, the image auto-discovers the hardware and runs an autoconfiguration based on the type of installation used. The system image can boot from CD, PXE, local disk, USB storage (USB flash key or SD card), FC, FCoE or iSCSI SAN (boot from SAN).

In addition to a handful of files used to bootstrap ESXi, an ESXi system image includes two main sets of files:

VMkernel Executives The VMkernel executives are the compressed files that make up the VMkernel. You can recognize them by their `.gz` file extension. These files don't show up in the filesystem after ESXi is loaded.

Archive Files The archives are the files that make up the visible filesystem. They're VMware Installation Bundle (VIB) files, usually with a `.vgz`, `.v0`*n*, or `.tgz` file extension.

As each VIB is extracted, it's overlaid onto the filesystem. As archives are consecutively laid down, only the latest changes are visible. If an archive is removed, then the previous branch is visible in the filesystem.

The last archive to be overlaid is called the *state archive* (`state.tgz`). This archive contains all the configuration settings, such as the `/etc` files. The state archive doesn't exist on the ISO image, because there is no nondefault configuration at that stage, but is created on initialization. To save excessive wear on the boot disk, which may be flash based, the state archive file is only updated to disk when a configuration change occurs. This is commonly every 10 minutes but is limited to no more than 6 backups every hour. This means some very recent changes may not survive a host crash (an update to disk does occur before reboots, though). This state archive tardisk forms the basis of the backup and restore routines.

Because the image is loaded into memory, it doesn't rely on its boot device when it's running. This means the boot device can fail or become disconnected, and the OS will continue to run. Changes that are made to the filesystem but that don't modify the system image are lost after a reboot. All added components and configuration changes must update the image in order to be persistent or must be reloaded each time.

ESXi allows only authorized code, and these modules have to be digitally signed by VMware. This restriction on VMkernel extensions helps to secure the environment, preserve system resources, and maintain a tight codebase.

Vendor-Specific Images

The main server vendors produce their own customized system images. These ISO images are enhanced versions of the regular VMware system images, with hardware-specific additions. They can include new or improved hardware drivers, extra CIM plug-ins to provide better hardware monitoring, and added support information.

ESXi Customized Images

You can customize your own ESXi images. This can be useful, because the VMware provided system image can become outdated. VMware releases patches but usually doesn't release a new image until a less frequent update is provided. By creating a customized image, it's possible to slipstream in several patches. Additionally, the standard system image may not include all the drivers or hardware CIM providers, or those included may be obsolete and require refreshing. It's possible that for newly released hardware, the standard image might not even install without the addition of special drivers. Extra plug-in software such as the HA agent, which is normally pushed from vCenter, or authorized third-party solutions can also be fed into the image.

To allow the creation of customized ESXi images, VMware released a new tool with vSphere 5 called *Image Builder*. Image Builder is a set of PowerCLI commandlets that package together installation bundles into customized ISO images and can fill a special image distribution container called a software depot. Image Builder uses three distinct components to create the customized ESXi images:

VIBs VIBs are installation bundles for ESXi components. Image Builder uses VIBs that have a `.vib` file extension. A VIB is an archive file, similar to a tarball, which contains the payload (for example, the software, plug-in, or driver), a descriptor file, and a signature file. The descriptor file contains several important details including any dependencies on other VIBs (meaning they must be installed first), known conflicts with other VIBs, whether it specifically replaces previously available VIBs, whether the VIBs require a host reboot or maintenance mode, and its *acceptance level*. The signature file is the digital certificate is used to verify that acceptance level.

The Image Builder tool uses four acceptance levels: VMwareCertified, VMwareAccepted, PartnerSupported, and CommunitySupported. Each VIB is set to an acceptance level; when Image Builder creates a customized image, that image can't take on an acceptance level higher than the lowest VIB level. This guarantees that an image is only trusted up the level of its least-accepted component.

VIB Author

VMware provides a tool called *VIB Author* that makes it much easier to create community supported-level VIBs. This can found at `http://labs.vmware.com/flings/vib-author`.

Image Profile An image profile defines the packages in an ESXi image created by the Image Builder tool. Image profiles can be created and VIBs added manually until the desired image is forged. A more straightforward approach is to clone an existing image profile and merely

tweak it with the required VIB additions and deletions necessary. You can create and maintain multiple image profiles for different hardware or different configurations: for example, different server models/manufacturers, with and without VMware Tools, third-party plug-ins, HA agents, and so on. Starting with vSphere 5.1, the Fault Domain Manager (FDM) agent is automatically fed into the image.

In addition to the acceptance level of each VIB, the image profile has its own acceptance level. All VIBs must be at least the same acceptance level as the image profile, if not higher, or the VIBs can't be added. VIB dependencies and conflicts are also checked for compliance when they're added to the image profiles.

Software Depot In addition to the pre-customized images that are sometimes available, server vendors and OEM hardware manufacturers can supply their own VIB files to extend special support for their hardware, including drivers and CIM plug-ins.

VIBs by themselves can be distributed, but these have to be installed via a command-line tool or the graphical VUM. Vendors instead usually choose to distribute their wares via a *software depot* that groups all their VIBs into one package and allows the software to be used directly by Image Builder. A software depot is simply a VIB or a collection of VIBs with some additional special metadata contained in a Zip file. Software depots are a better format for distributing packages because they allow the greatest versatility of installment options. Software depots that contain VIBs but not the base ESXi VIB are sometimes referred to as *software bundles* to distinguish that they're just meant as an addition to a core image.

A software depot can be made available for use in two ways: offline depots or online depots. An offline software depot is just a Zip archive file, much like a software bundle that also contains the ESXi image. When working with Image Builder you need a software depot, and a local Zip file is the easiest way to get started. You can also use an online depot, which may have the same contents as the file-based version but is centrally hosted on a web server. Most mainstream vendors host their own online depots that can be accessed over the Internet. Using an online depot has the advantage that whenever you create a new image, you pull down the latest version of the vendor's tools automatically. But you're at the mercy of the vendor to stay current with the other components. You can also use the online depots in VUM, subscribing to the vendor's site and allowing VUM to notify you when it finds newer software versions.

VMware provides its own software depots, both as offline Zip files and a web-based repository. If there is no custom depot for the server hardware for which you're creating the image, then you can start with VMware's base image. VMware provides two images: a standard one, and one that excludes the VMware Tools. The standard image is around twice the size of the slimmer sans-tools version. How the image is used will usually dictate which version to grab. As you'll see, the full version is good for ISO-style installs and the thinner one for the auto-deploy method of deployment. If you use the smaller ISO, you should set up a central shared locker for VMware Tools and configure each host to point to it.

Software depots are always required as an input resource for Image Builder to create images. Those created images (the output) can be either ISO files for regular installs/upgrades or new transformed software depots for use by VUM, CLI installs, or an Auto Deploy server.

Image Builder Process

Image Builder is a collection of PowerShell commandlets that exists in PowerCLI to create custom images. These commandlets use VIBs, image profiles, and software depots, and generate ESXi images as either ISO files for interactive and scripted installs or as Auto Deploy ready bundles.

The basic process to create a custom image with Image Builder is as follows (the commands provided are just pointers; additional switches and parameters are required):

1. Import/connect to the software depots to Image Builder (`Add-EsxSoftwareDepot`).
2. Create an image profile specifying the base VIBs, or clone an existing profile (`New-EsxImageProfile`).
3. Add/remove VIBs as required (`Add-EsxSoftwarePackage` or `Remove-EsxSoftwarePackage`).
4. Export the image profile into a packaged ISO or depot file image (`New-EsxImageProfile`).

Alternative Option to Create Customized Images

Other third-party options exist for creating customized images. In addition to getting a customized image from a server vendor, one such option is the ESXi-Customizer tool (`http://v-front de/p/esxi-customizer.html`). This is a simple script that presents a GUI tool to add VIBs and software bundles to an ISO image. It's freely available but obviously not supported by VMware. If you just want to squirt a RAID controller VIB or two into an ISO file that the server can successfully boot from, then this may provide a convenient and fast solution without having to get your hands dirty with PowerCLI commands.

It's worth noting that creating your own custom images isn't always a requirement. Large enterprises and service providers can benefit from the ability to customize images if they have a lot of servers or need to rebuild servers on a frequent basis. But the standard VMware ISO or vendor-customized ISO is probably sufficient for most companies.

ESXi Disk Layout

ESXi is loaded into memory when it boots up and runs from RAM. In memory, it uses tardisk archives that are static files and ramdisks that grow and shrink as they're used. In addition to the tardisks and ramdisks in memory, a number of disk partitions can be found. The way ESXi is deployed and the hardware that is discovered when it's deployed will determine what additional partitions are present and where they're stored. If these additional partitions exist, they're mounted in the filesystem. There is no way to manually define the partition layout during the install. Alternatively, if they aren't present, then the backing for the directories is more in-memory disk mounts. A detailed description of the deployment options for ESXi comes later in

the chapter; here, let's focus on probably the most popular type—how ESXi Installable is copied to a local hard disk—because it shows all the possible partitions. Figure 2.1 illustrates the typical layout.

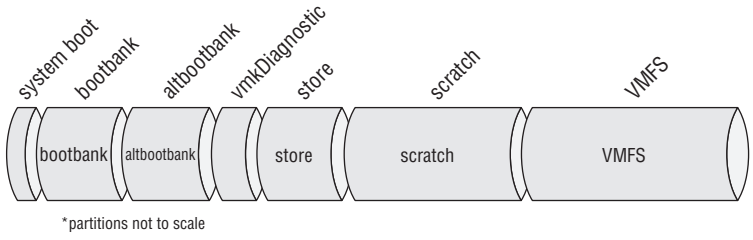


FIGURE 2.1 Typical partition layout of ESXi Installable on a local disk (minimum 5.2 GB)

A new ESXi 5 build now uses a GUID Partition Table (GPT) partition table as opposed to previous versions that used the older master boot record (MBR) partitioning. To view the list of partitions, instead of `fdisk`, use the `partedUtil` command.

System A small 4 MB bootloader partition resides at the start of the disk. This system disk is responsible for bootstrapping one of the two bootbanks.

Bootbank The bootbank partition is the primary system image. This partition is mounted in the filesystem in the `/bootbank` directory. The image in this partition, specifically the `s.v00` file, is uncompressed into RAM during boot time and becomes the root of the running filesystem.

Alt Bootbank The altbootbank partition is reserved for a backup image. This alternative bootbank can be used in case of problems during an upgrade. The altbootbank stores the "last known good configuration" as a fail-safe. During boot time, you can flip back to this predecessor using the Shift+R option. This partition is mounted as `/altbootbank` in the running filesystem.

vmkDiagnostic This partition stores the dump of the core memory if the host crashes with a Purple Screen of Death (PSOD) failure. This can then be retrieved and analyzed for troubleshooting. This partition is normally empty and not mounted to the filesystem.

Store The store partition keeps the VMware Tools ISO files and driver floppy images ready to mount to VMs. It's used to store any auxiliary files. It's mounted as `/store`. `/locker` is symbolically linked to this directory.

Scratch A *scratch partition* is a 4 GB virtual file-allocation table (VFAT) partition that's created by default during a regular install if a local disk of at least 5.2 GB is found on the first boot. The scratch partition persists the state archive and captures running state files such as the logs and diagnostic bundles (`vm-support`) that you can create if you're troubleshooting an issue. Userworld swap can also reside here. If the scratch partition doesn't exist and an alternate local location isn't found, the scratch directory is redirected to `/tmp/scratch` on a ramdisk. This means the contents of the scratch partition won't survive a reboot. It also means that up to 4 GB of RAM can be committed just with these files.

Scratch partitions are created if there is space on the local boot disk or another local disk. If not, then ESXi attempts to create a folder on a local Virtual Machine File System (VMFS) datastore if available; failing that, it uses a ramdisk. You can redirect the scratch directory to a persistent VMFS or NFS volume if the install doesn't create the partition (although VMware

doesn't recommend using an NFS volume). Scratch partitions are never created on what ESXi considers remote disks such as "boot from SAN" logical unit numbers (LUNs) and some SAS disks that are labeled as remote. Additionally, ESXi never creates scratch partitions on USB flash drives or SD cards, even if they have the capacity, because the potentially heavy disk I/O from the userworld swap could damage them.

VMFS Datastore If sufficient space exists during the first boot, a VMFS-5 datastore is created. This datastore is sized to fill the rest of the remaining boot disk.

Tardisks and Ramdisks

ESXi uses both tardisks and ramdisks to create its running filesystem. As the system boots up, tardisks are mounted as directories within the base VMkernel's tardisk, which is uncompressed into memory. Unlike regular filesystems, these tardisks aren't untarred and the resulting copy of the files used. Instead, ESXi keeps the tardisks intact and mounts them at points as if they were disks. The physical partitions described in the last section are also mounted in the running filesystem. Run `ls -lah /` to see the symbolic links for directories that are redirected to partitions (this is an easy way to check whether the running `/scratch` directory is on a local partition, VMFS, or ramdisk). The tardisks are laid out onto the filesystem in order as the system boots; once the process is finished, the filesystem you can see and interact with in the ESXi Shell is assembled. The tardisks are read-only and can't be changed. A tardisk in the image can overlay another one in the running filesystem, but it can't be modified.

Ramdisks provide storage for files that need to be created or modified while ESXi is running. For example, configuration settings and logs need to be maintained. If a modification to a file on a tardisk is required, ESXi uses a branching technique to create a working copy of it on a ramdisk. This is used extensively for the `/etc` folder. It uses a file's sticky bit to designate that it's *allowed* it to be branched off in this way. But these changes can't be written back to the tardisk files because they're read-only, so to persist this information it's written to the state file: `state.tgz` (most files are actually stored in `local.tgz`, which is inside `state.tgz`).

ESXi Deployment

ESXi can be deployed in a multitude of ways and will dynamically self-select its configuration options depending on the environment it finds. This gives you several deployment options, each subtlety suited to different scenarios. The consequences of each delivery mechanism in different hardware situations will drive an ESXi server's base design.

Hardware Requirements

To install ESXi, you need to consider several factors. First, the hardware should match the required specifications, and you should verify that all the components are supported on the HCL. Chapter 4, "Server Hardware," looks in detail at hardware requirements. However, as a basic starting point, the server should have 64-bit x86 CPUs with a minimum of 2 GB RAM. 1 GB of bootable disk space is required to complete a local installation.

ESXi Flavors: Installable, Embedded, and Stateless

ESXi comes in three flavors that dictate how it can be deployed. ESXi Installable is currently the most common and allows you to install the hypervisor on your own server hardware. ESXi

Embedded is an OEM option that you can purchase preinstalled on new servers. ESXi Stateless describes booting the hosts each time from a special PXE service called Auto Deploy.

The following section discusses ESXi Installable, ESXi Embedded, and ESXi Stateless and how these elements affect the design of a host deployment and your selection of hypervisor.

ESXi Installable

ESXi Installable is the version of ESXi that you install yourself to run on a server. It doesn't come pre-embedded on the hardware. Various options exist for the installation; and despite the name, you can boot and run it without installing it to a local drive.

One significant difference from most other OS deployments is that the system image is copied to the install location, and no installation per se is required. This makes the process significantly faster. The following explains the primary design factors in an ESXi Installable deployment:

Booting the Installer You can start the ESXi installer from several locations. The most common method is to download the installer from the VMware website, create a bootable CD, and start booting the server from the CD. But the server can also boot from USB (as long as the server's BIOS supports this) or via a PXE boot from one of the server's network cards. Any of these options can take advantage of a customized image created via the Image Builder tool, instead of VMware's vanilla download.

PXE booting is invoked when the server starts and is going through its Power On Self Test (POST). Usually a particular keystroke is required; or, often, if a server doesn't find a suitable bootable image on one of its storage devices, it will automatically try to find a PXE server. The network card broadcasts, looking for a response from a suitable PXE server. If discoverable, the PXE server replies to requests and can be configured to offer up the ESXi media as a bootable device. PXE servers offer a convenient solution to centrally storing installation media in a LAN environment. If you have several servers in one location, you can use PXE-based ESXi media depots to speed up installations, because each server doesn't need to be physically touched as it's built (or rebuilt). However, if you have several sites with WAN interconnects, you may need a PXE server at each location. Installing an OS of several hundred megabytes over a WAN link is probably impractical. Such an installation would be slow, could saturate the WAN link, and would probably fail before it finished.

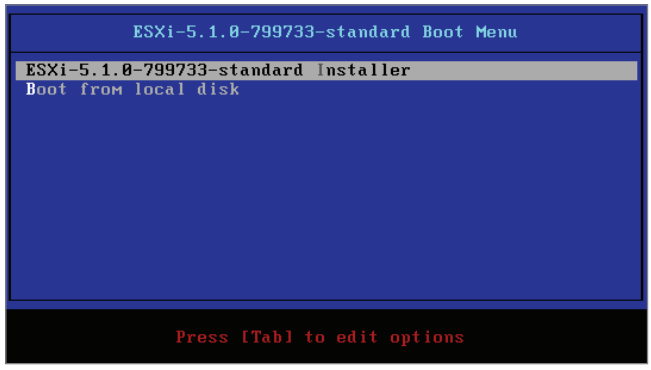
An installable deployment can also be started from an Auto Deploy server. As we'll look at more closely in the upcoming Auto Deploy section, this service is primarily used to deploy Stateless servers. But since vSphere 5.1, Auto Deploy can also be used for what is known as *stateful installs*. A stateful install uses Auto Deploy for PXE services and copies the images to the destination as installable images. It doesn't continue to boot from the Auto Deploy server like a regular Stateless install but merely uses the environment as a deployment tool. Once the image is copied to the destination, it behaves exactly as an installable ESXi image.

Installation Instructions You can run the installation in interactive mode or script it via an answer file:

Interactive Install Interactive mode, which is the default installation method, is a simple text-based routine. You select the answers to different options at the server's

console (or via a remote hardware console such as Integrated Lights Out [iLO], Remote Supervisor Adapter [RSA], or a Dell Remote Access Card [DRAC]) while the software is being set up. Figure 2.2 shows the format of the text-based installer.

FIGURE 2.2
ESXi text installer



Unlike the interactive install of ESX classic or most modern OS installs, which default to a GUI-based install, an ESXi install is much simpler. There are no complex partitioning questions to answer and surprisingly few options to work through.

Scripted Install Alternatively, you can specify a file that supplies the installation options automatically. You provide the answers for the installation options in a text file and feed that to the installer at the start. The syntax for ESXi answer files mimics Red Hat's kickstart scripts; even their default name is `ks.cfg`. A scripted install can produce the same result as an interactive install. The kickstart script file can be located on a CD, a USB flash drive, FTP, HTTP, HTTPS, or NFS export.

Kickstart file-scripted installations provide several benefits:

- Provides a perfectly repeatable and consistent process, should you need to rebuild the server
- Makes it faster to build and rebuild a server
- Creates a standardized build that can be applied to multiple servers
- Provides additional installation options

The last point is worth highlighting. Most vSphere architects and engineers will recognize the upsides of creating scripted OS installs; but with the ESXi scripted install, you can run post-install commands to automatically configure much of the server.

When you combine a kickstart script with a PXE boot-installer startup and a centralized media depot, then a practically hands-free install is possible. Obviously, some effort is required to set up, test, and suitably customize each part to make this work, so these types of build environments are best suited to larger enterprises that are provisioning tens or hundreds of servers. As usual, you must weigh the benefits of a scripted build against the development time required to create one.

A couple of community-driven projects aim to simplify the PXE server, media depot, and kickstart scripts. Both projects are similar but offer a slightly different experience. They're freely downloadable virtual appliances that provide web-based consoles to create customized scripted builds:

- ESX Deployment Appliance (EDA): `www.virtualappliances.eu/`
- Ultimate Deployment Appliance (UDA): www.ultimatedeployment.org/

Unfortunately, neither of these projects has maintained the enthusiasm they garnered before the arrival of the Auto Deploy feature. Auto Deploy, particularly with the stateful installs available in vSphere 5.1, can replace all these pieces. The PXE server, media depot, scripted install, and post-install configuration can all be achieved via Auto Deploy. But Auto Deploy is only available with the top-tier Enterprise Plus license, so many organizations are still using their own PXE servers and scripted installs. Arguably, most vSphere users that are large enough to consider setting up an alternative to Auto Deploy are the users with the Enterprise Plus licenses.

Destination The system image can be deployed to several locations. Ordinarily, you deploy it to a local hard drive. ESXi includes SSD drives in this category. Some drives, predominantly SAS drives, can be incorrectly recognized as remote disks. An install to what is considered a remote disk means no scratch partition is created, and the scratch directory is created purely in memory.

ESXi Installable can also be deployed to a USB flash drive or an SD card mounted in the server. These locations are both considered *removable* by ESXi and treated as if they were full-blown ESXi Embedded. If the USB drive or SD card is regarded by the server vendor as a supported device, then this is by VMware's definition an ESXi Embedded build, although the server vendor might not see it that way. Whether validated or not, the impact of installing ESXi to USB/SD is the same. See the upcoming *ESXi Embedded* section to understand the associated consequences.

You can also install the system image on a SAN LUN. The SAN LUN can be FC, FCoE, or iSCSI. ESXi can boot from an iSCSI LUN using an independent hardware iSCSI card (iSCSI HBAs) or a software initiator with a dependent hardware iSCSI adapter. To use a software initiator to boot from SAN, the network card must support an iSCSI Boot Firmware Table (iBFT). The iBFT format allows the iSCSI parameter to be saved in the network card's onboard memory.

VMware Go

An alternate to the regular install techniques is VMware's SaaS-based Go service. It's a cloud-based deployment and management solution for which a customer can sign up. Among other options, VMware Go can remotely install ESXi onto existing Windows servers that can hit the web service via the Internet. It checks the server hardware to ensure that it will be compatible and then streams the ESXi image onto it. It's primarily targeted at the SMB market, with a free version and an inexpensive subscription service with more features that is currently included in the Essentials licensing. If the deployment is for a small company that could benefit from the additional support and management options, then this may be something to recommend.

To reiterate the possible options, the ESXi Installable image can be provisioned in the following ways:

Booted from

- CD
- USB
- PXE server
- Auto Deploy server

Gets its instructions

- Interactively
- Scripted

Boots its image from

- Local disk (including SSD)
- USB or SD (acts like ESXi Embedded)
- FC, FCoE, or iSCSI SAN LUN

ESXi Embedded

The concept behind ESXi Embedded is that it's a version of ESXi that server vendors can sell onboard their equipment at the time of purchase. The ESXi image is preinstalled as firmware in the factory or burned to a USB flash drive or SD card and installed in an internal socket on the main system board. Usually, the ESXi image includes the hardware drivers and CIM plug-ins that vendors provide in their customized versions of ESXi Installable. The software is installed out of the box and is only awaiting setup details after it's first powered on. In reality, many server vendors will ask you to buy one of their certified USB drives, which they ship with a copy of the vendor-customized image that is normally available online as a customized ESXi Installable image.

The important point is that everything else about the image is the same. The hardware may vary, the manufacturer's warranty and support details may be different, but the software image of ESXi Embedded is the same image used in ESXi Installable. But in theory, ESXi Embedded is only an option on new hardware. Officially, you can't retroactively add it to an existing server. That said, because the image is the same, by installing ESXi Installable to a USB flash drive, you're effectively getting the same thing.

ESXi Embedded considers its media *removable* and treats it somewhat differently. It only uses the first 1 GB of the drive and doesn't have a scratch partition unless it can find a local disk partition or local VMFS datastore to store the scratch directory. Otherwise it will default to a memory-resident a memory-resident ramdisk. The impact is that its content will be lost during a reboot or server failure. The risk can be mitigated by creating a scratch folder on a remote datastore or by redirecting the logs to a centralized syslog server, as explained in the "Postinstallation Design Options" section of this chapter.

If you're procuring new server hardware and have decided on ESXi, then what are the key decision points for ESXi Embedded?

Advantages of ESXi Embedded

- No installation is required.
- Servers are potentially cheaper, because manufacturers don't need to configure them with RAID cards and local disks.
- Servers have an appliance-style nature, so you know all hardware will work out of the box and be fully supported.

Disadvantages of ESXi Embedded

- You can use Installable or Auto Deploy across all your existing servers, so you have a unified installation method and can continue to purchase the same hardware as before.
- ESXi's regular HCL is much more extensive, meaning you have more choices to customize the hardware to your exact requirements.
- Servers bought for Installable or Auto Deploy can be repurposed in other server roles, because you know they're off-the-shelf servers.
- Servers that were purchased for Installable are likely to have local disks, meaning they can offer local scratch and VMFS partitions.

ESXi Stateless

ESXi Stateless was introduced as a supported deployment in vSphere 5.0. As opposed to ESXi Installable and Embedded, Stateless hosts don't normally boot from a disk. Currently all Stateless hosts take advantage of their memory-resident nature and the small size of ESXi by streaming their image over the network every time they boot. This is dissimilar to PXE booting an install of Installable, because those hosts copy the ESXi image to a disk, and all subsequent boots are from the disk not the network.

ESXi Stateless Doesn't Equal Auto Deploy

ESXi Stateless leverages Auto Deploy as its deployment mechanism. It wouldn't be possible to boot a Stateless host without Auto Deploy. But it's incorrect to think that Auto Deploy equals ESXi Stateless and vice versa: Auto Deploy is the infrastructure mechanism, and Stateless is the way the host is configured. How to use Auto Deploy is explained later in this chapter.

The defining characteristic of Stateless hosts is that the physical servers themselves aren't *authoritative* for their state. This means despite where the servers boot from, they always get their state information from a third-party authority. So Stateless hosts do have a state, but it's reapplied from another source whenever they boot. As Stateless hosts boot, their state is streamed down to them from several sources. Table 2.1 shows where each part of a host's state is derived.

TABLE 2.1: Components that provide system state to stateless hosts

COMPONENT	STATE PROVIDED
Image profile	The ESXi image from which the host boots. Customized with Image Builder to include drivers.
Host profile	Configuration settings.
Host profile answer file	Configuration settings specific to that host, such as IP address.
vCenter	List of VMs stored on the host, HA state, licensing, VM autostart information, and distributed power management (DPM) details.

By default, the runtime state information such as log files is stored in the hosts' ramdisk and lost during each reboot.

Advantages of Stateless Hosts

Why would you want a server that forgets all its settings every time it reboots? Stateless servers receive their state from a central authority, which makes hardware much more interchangeable and allows for easy deployment and redeployment. Applying new ESXi images with host updates, improved drivers, or a tweaked configuration only requires a host reboot. In the same way you can consider hosts to be compute resources that you give to the cluster to be distributed to VMs, you have the ability to automatically *rebuild* hosts in minutes en masse to suit new cluster configurations with new network and storage settings.

Stateless servers also help to prevent configuration drift, because each server is refreshed back to the corporate standard after each reboot. No need to troubleshoot an individual host's configuration state anymore.

There is no need for Stateless hosts to have any local storage. The image is pulled from the network every time it boots up and is loaded straight into memory. If there is no local disk present, then you should redirect the logs and core dumps to remote collectors to protect them; but if local storage is available when the server boots, ESXi will take advantage of this and mount the scratch partition there. The concept of having no local disks and no local state information makes the ESXi servers more like hardware appliances than you might typically think of your servers. The nature of a Stateless deployment means the ESXi image and all the host's configurations can be managed centrally. In a large environment, this is more efficient than kickstart-style scripting installs.

Stateless Caching

vSphere 5.1 introduced a new option called *Stateless caching*. This follows the same ethos as a regular Stateless deployment, but during each boot as the image and state are being pulled down from the Auto Deploy server, they're also being copied to disk. The disk could be a local disk, a removable USB key, or even a boot-from-SAN LUN. If there is ever a problem with the Auto Deploy infrastructure, then the host can boot from its "locally" cached copy. In normal circumstances, a Stateless caching host boots from the network image. It only drops to the cached image as a fail-safe option. Having the local copy allows the server to power on even if

there is a problem with the Auto Deploy server, and get the VMs started. It can then help troubleshoot the Auto Deploy problem. When the Auto Deploy infrastructure is available and configured correctly again, the next server reboot will see the host take its image from the network again (and update its cached copy again). To enable this mode, a small change to the host profile is required, and the server's BIOS needs to attempt to boot from the disk if the PXE boot fails.

The regular Stateless mode without local caching enabled, which has been available since vSphere 5.0, still exists as the default mode in vSphere 5.1. Stateless without caching enabled retains the advantage that no disks need be involved.

Auto Deploy Infrastructure

Auto Deploy is a deployment tool to automate the provisioning and configuration of ESXi hosts. It can simplify large deployments by centralizing the images and configuration details. All hosts, at least initially, boot via PXE and connect to an Auto Deploy server. The Auto Deploy server selects an ESXi image via a set of rules and streams the images to the physical server. vCenter supplies the host with its configuration details via the Host Profiles feature.

Auto Deploy can dispense Stateless hosts as described in the previous section, "ESXi Stateless." In this case, the hosts come back to the Auto Deploy mechanism on every reboot. Auto Deploy can also be used as a pure installation mechanism for what are known as *stateful hosts*. Statefully deployed hosts are in effect regular ESXi Installable images. The only difference is that they were initially provisioned via the Auto Deploy tools, so they used an Auto Deploy image and host profile information to configure the hosts. This means the host can be rebuilt quickly if required as the information is retained in the Auto Deploy system. But once a stateful host is deployed, it's no longer dependent on the Auto Deploy server. It maintains its own state information, doesn't boot from the network, is patched in a normal fashion, and is to all intents and purposes an ESXi Installable host.

Auto Deploy Components and Process

Auto Deploy needs several components available and configured to provision ESXi hosts:

PXE Environment The host is set to PXE boot from a network adapter. A DHCP server responds to the request with a reserved IP address that you've set and provides DHCP option 66, which tells the server which TFTP server to go to, and option 67, which is the filename of the iPXE file (vSphere 5.0 used gPXE). The host contacts the named TFTP server and grabs the iPXE boot loader and iPXE configuration file. At this point, the host can make a request to the Auto Deploy server for an ESXi image.

Auto Deploy Server The Auto Deploy server is a service that comes preinstalled on the vCenter Server Virtual Appliance (VCSA); although it's disabled by default, and is also an installable option on the Windows vCenter ISO. The Auto Deploy server carries out two tasks. First, according to predefined rules that you configure, it matches the attributes that the hosts provide to an image file, a host profile, and the correct vCenter object location to join. Second, it provides the host a bootable image via its web service.

The Auto Deploy server can use the Image Builder tool mentioned earlier in the chapter to customize and update ESXi as required. The rules can match a number of machine attributes such as MAC address, IP address, and SMBIOS details. You use PowerCLI to create the working rules and then activate the rules.

vCenter The image that the Auto Deploy server provides to the host includes the host profile configuration that vCenter is responsible for maintaining. This host profile provides all the configuration details that the server needs beyond the base ESXi image. The host profile can be augmented by a host-specific answer file. The host then attaches to its vCenter as it boots and joins the appropriate cluster as a host resource ready to perform compute workloads.

The basic process to create the working rules and activate the rule set is as follows (the commands provided are just pointers; additional switches and parameters are required):

1. Connect to the software depot generated by Image Builder (`Add-EsxSoftwareDepot`).
2. Assign an image profile, assign a host profile, and assign a cluster rule (`New-DeployRule`).
3. Add the working rule to the active rule set (`Add-DeployRule`).

VMware has released a *fling*, a free unsupported tool, called Auto Deploy GUI that you can find at `http://labs.vmware.com/flings/autodeploygui`. The Auto Deploy GUI simplifies the process of configuring the Auto Deploy service. Although VMware flings aren't officially supported, as long as the hosts are tested once they're deployed, then this won't affect their operational running.

Deployment Modes

Currently, Auto Deploy supports three distinct deployment modes. Hosts will deploy via the first option unless specifically set to a different mode in their host profile (under System Image Cache Configuration):

Stateless Stateless Auto Deploy is the classic mode that has been available since vSphere 5.0. A host set in Stateless mode PXE boots from the network every time. Each boot follows the same process. If the server needs to be patched, update the common image and the rule set and reboot. The host is dependent on the Auto Deploy infrastructure every time it boots (although not while it's running).

Stateless Caching Stateless caching mode is similar to the Stateless mode except every time the image is streamed down to the server, it's cached to a disk. This disk can be a local disk, a remote boot-from-SAN disk, or a USB drive. If you've selected the USB option, it overwrites the first USB drive it finds attached to the server. This image is only cached; therefore as long as the Auto Deploy infrastructure remains available, the server continues to boot from the latest matching network image. The key difference is that during periods when the Auto Deploy infrastructure is down, the host server can still boot up. As soon as Auto Deploy is available again, the next reboot sees that the host receives a network image. vSphere 5.1 is required for this mode.

Stateful Install The stateful install mode only uses the Auto Deploy infrastructure the first time it boots to receive the image. That deployment is copied to the server's disk (local, boot-from-SAN, or USB) and boots from that image. Again, if you've selected the USB option, it overwrites the first USB drive it finds attached to the server. All subsequent reboots are from the disk image, not the Auto Deploy infrastructure. This is achieved by setting the boot order

of the server in the BIOS to boot from local disk first and a PXE boot second. Patching the host is done via conventional methods, and the configuration isn't updated unless a newer host profile is manually applied from vCenter. This can lead to configuration drift, and many of the advantages associated with Auto Deploy are lost. vSphere 5.1 is required for this mode.

Auto Deploy Recommendations

You should follow several general recommendations when designing your Auto Deploy infrastructure:

- Consider keeping your Auto Deploy servers in a separate dedicated management cluster, analogous to recommendations made for vCloud Director deployments. This prevents chicken-and-egg situations when a datacenter-wide outage shuts down all the physical servers and the VMs required to boot up the hosts can't be brought back online. Create a management cluster where the hosts aren't deployed in Stateless mode (although it's feasible to use Auto Deploy to build them in stateful mode).
- If you opt to use non-Stateless hosts to house the Auto Deploy infrastructure, ensure that all the pieces needed are running on this cluster. At least one DHCP server, the TFTP server, the Auto Deploy server, the vCenter server (along with database, SSO, and web client server), DNS, and almost certainly a domain controller should be there.
- Although you could install these components on physical hardware, it's recommended that they're installed on VMs. This provides hardware-failure protection through HA, hardware independence via vMotion and Storage vMotion, and resource protection and load-balancing with DRS.
- Build at least one host with a full ESXi image including the VM tools. These tools can be copied to a shared locker for all the hosts. Don't use this full image for all the hosts because it puts unnecessary load on the network and the Auto Deploy server.
- Remember to create a centralized persistent storage location for the Stateless hosts' logs and core dumps, or redirect these to remote collection servers: a syslog server and a dump collector. The dump collector will only work with a vSphere 5.1 vNetwork distributed switch (vDS).
- If the hosts are to be joined to Active Directory (AD), configure the AD Authentication Proxy service (available on the Windows vCenter install). This proxy service prevents domain administrator credentials from being stored in the host profile.

Comparing Deployments Options

Each of the described deployment options will provision serviceable ESXi hosts. There are advantages and disadvantages to each technique, and different methods suit some environments better than others. Figure 2.3 summarizes these options and how they affect where the servers boot their image from.

FIGURE 2.3 Deployment options and the resulting boot images

	Installed			Embedded	Stateless	
Deployment mode	Interactive	Scripted	Stateful	Purchased from Manufacturer	Stateless	Stateless Cached
Deployment image	CD, PXE boot, USB key		Auto Deploy	n/a	Auto Deploy server	
Boot image	Local disk, USB/SD, Boot from SAN			Embedded (firmware chip or internal USB/SD)	Auto Deploy server	
Failback boot image	altbookbank if media available		Auto Deploy server	altbookbank if available, some servers have dual-banks	none	Local disk, USB/SD, Boot from SAN

How Was This Server Deployed?

If you already manage some ESXi servers and are unsure how they were provisioned, a simple CLI tool can help you discover this. At the ESXi Shell, run the following command: `esxcli-info -e`. The resulting output provides the answer:

- `visor-thin`—An installable host. This also includes an Auto Deploy stateful install.
- `visor-usb`—An embedded host. This also includes self-installed ESXi on USB flash drives or SD cards.
- `visor-pxe`—A Stateless host, including those delivered in Stateless caching mode.

However, if a Stateless caching host can't find the Auto Deploy server when it boots, it will boot from its locally installed copy. During this time it will report as visor-thin.

The deployment strategy chosen is often decided by the size of deployment, the skill level within the organization, and how rapidly the servers need to be deployed (and possibly redeployed). The WAN topology and vCenter implementation also affect how servers are deployed: there can only be one Auto Deploy server per vCenter, which may limit where Stateless hosts are provisioned or encourage more vCenters to be introduced.

Scaling Deployments

A small company with only a handful of host servers is likely to opt to manually install each server using a bootable CD and the interactive install routine. As the company grows and the level of automation increases, the company may look at scripting the installs. In companies with most servers in one location, or locations with good WAN links, then PXE boot servers are a convenient way to centralize the image. This approach combined with kickstart scripts allows for a scalable option, providing quick and centrally managed install points. The largest companies, with vSphere Enterprise Plus licensing, can then consider automated post-install methods to make initial configurations and maintain these environments through host profile policies.

If the company already has a custom PXE boot environment and has amassed some knowledge around its scripted installs, then it may want to stick with this. But companies looking to set up automated installs from the ground up are best advised to implement this as an Auto Deploy infrastructure. In principle, the custom PXE booting with scripted installs is similar to

Auto Deploy stateful; but why build this yourself when VMware has done much of the hard work for you? The Auto Deploy solution is fast becoming a standard across companies and will be easier to maintain than a custom solution. When the organization needs the maximum flexibility and to reduce reprovisioning times, Auto Deploy Stateless builds show their worth. VMware is continuing to develop Auto Deploy, a Web Client GUI for configuration is almost certain to appear shortly, and the Host Profiles feature is maturing with each new release.

Companies may choose multiple techniques for different situations. For example, a company may use Auto Deploy Stateless in its primary datacenter, stateful installs for its management cluster, a handful of Embedded servers in a remote site managed by an external contractor, and Installable on local disks for servers to be built and sent to country-wide small offices.

For very large installs that use Auto Deploy, the infrastructure can be scaled out to an extent. The current limiting factor is that you can have only one Auto Deploy server per vCenter. If you need more, then you may need to split out to multiple vCenters. Then link them through the SSO service so they can be managed through the same Web Client server (or linked mode if you need shared licenses and roles, or across WAN links).

Each Auto Deploy server can boot around 40 servers concurrently with vSphere 5.1. This is based on using the smaller ESXi image without the VMware Tools. You should plan for the worst-case scenario of a boot storm (for example, after a power outage) and consider the minimum number of hosts you would want to power on initially. You can add more web proxy servers in front of a load balancer to distribute the web service load on the Auto Deploy server, which should allow for some level of linear scaling (another 40 hosts per additional web server), but you'll eventually hit a limit depending on the CPU power of the Auto Deploy server. As always, only testing in your own environment can prove these measures. For any Auto Deploy infrastructure of size, you shouldn't install the Auto Deploy service on the same server as vCenter, because both servers get hit fairly hard when multiple servers reboot.

Impact of Image Location

Even though all the deployment methods give you a running version of ESXi on your hosts, it's important to understand the impact of the different configurations you can end up with. Local disks are better for small deployments if you already have the servers and they come with disks. The disks are normally reliable, and the install has good defaults that require less post-install work to correct any deficiencies. Taking the most common image location for a deployment, the server's local disk, let's look how each alternative compares:

Removable Boot media considered removable, which are USB/SD installs and ESXi Embedded, don't have a scratch partition. If there is a VMFS volume on a local disk, then ESXi will use that to create a scratch directory. If not, ESXi will store the scratch directory in a ramdisk. The best solution is repointing the scratch in the hosts' advanced settings to a remote VMFS volume. Alternatively, you can set the syslog service and dump service to redirect them to remote syslog and dump collector servers. If you're using a USB/SD card for the installs, you're advised to buy good-quality media. Don't spend thousands of dollars on expensive server hardware to shortchange yourself with a $5 boot disk. Although the server won't crash if the boot media fails, you'll need to replace the media and reboot the server sooner rather than later. Cheap USB/SD media won't last very long.

Boot from SAN Booting from SAN provides some advantages such as making the servers moderately cheaper to purchase; using SAN disks, which have greater consolidation

and greater failure protection; and making the servers practically Stateless so a failed server can be replaced and repointed to the SAN LUN. But a number of issues remain, which you should consider before choosing this approach:

- Boot-from-SAN configurations are more complicated to configure, requiring individual fabric zoning for each server and potentially complex HBA/CNA configuration.
- SAN storage is usually more expensive than local storage, so any saving on server storage is lost on the extra SAN disks.
- A SAN LUN needs to be provisioned and managed for every server, which can create significant additional work for a storage team.
- Periods of heavy VMkernel disk swapping I/O can affect the VM's disk performance, because they share the same disk I/O channels.
- VMs configured with Microsoft Clustering (MSCS or Failover Clustering) aren't supported on boot-from-SAN configurations.

Booting from SAN is still popular in some datacenters, particularly blade shops, but Auto Deploy (particularly with its improvements in 5.1) is arguably a better option for large single-site datacenters. The relative complexity of boot-from-SAN in comparison to the ease of ESXi installs these days means it doesn't offer the same value as physical Windows or Linux boot-from-SAN configurations.

Stateless Classic Stateless servers have no underlying disks so everything runs from ramdisks. There are no mounted disks for the scratch partition or the store partition. There is no opportunity to boot back into an alternate bootbank (although you can offer a different image if you change the active rule set).

Upgrading ESXi

Upgrading ESXi hosts from ESXi 4.x can be tackled several ways. ESXi Installable 4.x hosts can be upgraded in one of three ways: an interactive upgrade via an ESXi Installable image, a scripted upgrade, or VUM. ESXi 5.0 hosts can be upgraded to 5.1 using these tools and can also use the `esxcli` command-line tool; or if you deployed with Auto Deploy, you can simply apply a new image and reboot.

When upgrading hosts from ESXi 4.x to 5.x using any of the three supported methods, 50 MB must be available on the local VMFS volume. This space is used to store the server's configuration data temporarily.

An alternative strategy to running in-place upgrades across the hosts is to perform a clean install on each one. The install will run through faster if you can apply the same post-install configuration steps as you did for the previous build, using the same script or host profile. This provides the opportunity to standardize your host fleet if there has been any configuration drift among them. This is also the chance to migrate to a new deployment strategy, such as using Auto Deploy.

However, in-place upgrades are faster if no existing configuration is set in host profiles or post-install scripts. VMs on local datastores don't need to be migrated or restored, and VUM can orchestrate the entire upgrade. There are at least two small resulting differences when

upgrading hosts instead of rebuilding them. First, the local VMFS volume remains as VMFS-3. This can later be upgraded nondisruptively to VMFS-5, but it remains an "upgraded VMFS-5" volume. The impacts of this aren't terribly significant; see Chapter 6, "Storage," for the gory details, but it may create a small mismatch between the subsequently provisioned new hosts and the upgraded ones. The second difference is that the boot-partitioning scheme remains MBR-style instead of the new standard, which is GPT. The GPT partitioning allows local partitions to be greater than 2 TB in size. This isn't that important on local ESXi disks but again will introduce variances in your host design going forward, which may confuse troubleshooting, for example.

Upgrading ESXi 5.0 Stateless hosts to 5.1 is much simpler because the single image profile just needs to be updated and the active rule set pointed to the new image. Upon the next reboot, the hosts will be upgraded. And in reality, the hosts won't even be upgraded but freshly rebuilt.

One point worth mentioning about updating and patching Stateless hosts is related to VIBs that don't need a reboot to apply. Any patch that isn't applied to the image profile gets lost after a reboot. So even if a patch doesn't require a reboot, it's important to update the image; otherwise, the patch will drop off after the next reboot. However, it can be handy to quickly apply host patches directly via VUM if doing so addresses a security vulnerability and you want to immediately protect your hosts before you have a chance to reboot them all. Just install the nondisruptive patch and then apply it to the image profile in preparation for the next planned (or even unplanned) reboot.

Migrating from ESX

A project still common among businesses with existing vSphere environments is migrating from ESX to ESXi. These conversions undoubtedly require testing and planning, but it's important to go back to the fundamental design of your vSphere hosts to ensure that the principal objectives are still met and to determine whether any additional improvements can and should be made. Although ESX and ESXi have more commonalities than differences, it's beneficial to understand how you can transition an existing ESX design to ESXi.

Testing

Prior to any redeployment of ESX hosts to ESXi, you should run a testing phase that looks at the existing disposition and examines whether each element is suitable for ESXi. Obviously, a pilot is a good way to test the sociability of ESXi in your existing configuration. However, it's difficult to test every possible scenario via a small pilot, because usually large enterprises have many different types of hardware, versions of hardware, connections to different network and storage devices, backup tools, monitoring tools, and so on.

It may be prudent to provide a contingency plan in case ESXi doesn't fit in a particular situation; the plan may provide financing to replace equipment, or it may specify that you keep ESX classic in that circumstance and migrate later when the equipment is due for replacement. It's feasible to mix ESX and ESXi hosts during a migration, even within the same cluster. If you're thinking of making some ESX hosts a more permanent fixture, you may want to consider some of the side effects of supporting mixed hosts, such as maintaining two patching cycles, troubleshooting two types of hosts, and collecting hardware-monitoring data in different ways.

You need to look at your server hardware carefully to be sure it's compatible with and fully supported for ESXi. The HCL listing for ESXi Installable is now greater than that of ESX classic,

but it still lacks some of the older servers. You should check that the servers and their add-on components are listed on the HCL.

If there are hosts that you can't migrate for whatever reason, consider treating them like ESXi hosts as much as possible with regard to how you manage them. Most of the VMware and third-party tools can now connect to either type of host. You can use tools that replace some Service Console functionality, such as the vMA and PowerShell commands, to manage ESX classic hosts. You can also use ESXi tools in a mixed environment and no longer have to rely on the Service Console.

Deployment

ESX servers can be upgraded or rebuilt. ESX 4.x server upgrades can be undertaken by the ESXi interactive or scripted install mechanism, or via VUM. If the ESX 4.x servers have previously been upgraded from ESX 3.x, then the partitioning configuration brought across might not allow a subsequent upgrade to ESXi. There needs to be at least 50 MB free in the host's local VMFS volume to store the previous configuration. If you're using VUM to upgrade, the `/boot` partition needs to have at least 350 MB free. If you don't have this much space, you can still use the interactive update because it doesn't need this space for staging the upgrade files. Any VMFS volumes on the local disk that need to be preserved must start after the first 1 GB; otherwise the upgrade can't create the ESXi partitions it requires.

If you have any ESX 3.x hosts, they must be upgraded to 4.x first and then upgraded to 5.x. You really have to question whether a dual upgrade is better than a single fresh install. Also, any host that is still ESX 3.x may struggle to match the HCL for ESXi 5, so migrating to new server hardware with clean builds is probably more appropriate.

There are several impacts from upgrading instead of completely rebuilding your ESX servers. Many of them mirror the considerations faced with ESXi 4.x upgrades that we discussed in the last section. In summary, the disks will keep their MBR scheme, limiting them to 2 TB; no scratch partition will be created, and the VMFS-3 volume will remain (although it can be upgraded afterward to VMFS-5, it remains an "upgraded" VMFS-5 volume). However, upgrading from ESX 4.x, as opposed to from ESXi 4.x, is a much bigger jump in platforms, and it's only reasonable to assume the results won't be as smooth.

Alternatively, a rebuild strategy might actually prove to be simpler and certainly generates a more consistent, dependable end result. During a rebuild, all settings are lost, and the newly rebuilt servers must be reconfigured. Larger companies with dedicated server staff are probably familiar with rebuilding OSes instead of upgrading them. But smaller companies that may have used external consultants to initially deploy ESX may be less prepared to run full installs across all their hosts.

If you completely rebuild the hosts, you should move all the data off any local VMFS volumes—VMs, templates, ISO files, and so on. It's possible to leave data on an existing local data store and do a fresh install of ESXi to the remaining space, but you risk losing it if something goes wrong, and you won't get the immaculate install for which you chose this option. Check the filesystem for files in the `/home` directories, `/tmp`, and anywhere else a local user may have saved files. Finally, you may wish to back up the files, particularly those in the `/etc` directory for configuration settings and the `/opt` directory for third-party installed agent software.

The VMware fling called *ESX System Analyzer* is a tool that can scan across ESX hosts attached to a vCenter and create pre-migration reports. It provides details such as HCL compatibility for the physical servers, VMs registered on local storage, Service Console modifications,

and version details for datastores and VMs. These reports can be invaluable when you're planning a migration strategy; this tool makes light work of an otherwise onerous task. You can download ESX System Analyzer at `http://labs.vmware.com/flings/esx-system-analyzer`.

If you wish to avoid VM downtime, you must have access to shared storage and at least the equivalent spare capacity of your most powerful server in each cluster. That way, you should be able to migrate the VMs around the hosts as you remove one at a time from the cluster to rebuild it.

Fortunately, many of the deployment methods you may have used for your ESX classic hosts are reusable. ESXi can use PXE servers you've already set up—you just need to copy the new images to the server. You need to modify ESX kickstart scripts for ESXi, but they can use largely the same syntax for the configuration; and usually you must remove unneeded lines rather than add new lines.

Management

One of the more obvious techniques to smooth a transition is to begin using the newer cross-host management tools as early as possible. Most of the utilities available for ESX classic hosts work with ESXi hosts. vSphere Client, vCenter Server, vCenter Web Client, vSphere command-line interface (vCLI), and PowerCLI are host-agnostic and will make any transition less disruptive. There is no reason not to start working with these tools from the outset; even if you decide not to migrate at that stage, you'll be better prepared when ESXi hosts begin to appear in your environment.

The primary management loss when you replace ESX is the Service Console. This is the one tool that, if you use it regularly, must be replaced with an equivalent. There are two main contenders: the vCLI and the ESXi Shell. The vCLI provides the same Linux-style commands as the Service Console. The easiest way to get it is to download the vSphere Management Assistant (vMA) virtual appliance. This includes a Linux shell and all the associated environmental tools you'd expect. Generally, anything you can do at a Service Console prompt, you can do in the vMA. Most scripts can be converted to vMA command syntax relatively easily.

The second option is the ESXi Shell. Although it's a stripped-down, bare-bones environment, it provides the majority of vSphere-specific commands that you'd otherwise find in the Service Console. Some of the more Linux-centric commands may not be available; but it provides a more familiar feel than the vMA, because it's host-specific and therefore the syntax is closer to that of the Service Console.

In addition to rewriting scripts, you'll need to replace other services that ESX includes in the Service Console. For ESXi hosts without persistent storage for a scratch partition, it's important to either redirect the logs to a remote datastore or configure the host to point to a remote syslog server. ESXi hosts don't have their own direct Web Access as ESX hosts do. It's unlikely that this is a regularly used feature, but if you've relied on it in certain circumstances, then you need to get accustomed to using the Windows client to connect to the host. Finally, if you've used the Service Console for host monitoring via SNMP or a server vendor's hardware agent, then ESXi has built-in CIM agents for hardware on the HCL, and many vendors can supply enhanced modules for their particular hardware. With these CIM modules, you can set up alerts in vCenter, and some third-party hardware-monitoring software can use the information. ESXi also provides some SNMP support, which you can use to replace any lost functionality of Service Console agents.

If your hosts are licensed for Enterprise Plus, host profiles can provide a convenient method of migrating host settings from ESX to ESXi. You can capture an existing ESX classic host in a

cluster as a reference host, rebuild one of the servers to ESXi, and apply that profile to receive a common set of configuration settings. The new ESXi host with the cluster's profiles applied can then be the basis of a new profile that you can apply to all the other hosts as they're rebuilt. If you're migrating existing servers whose design doesn't need to change, host profiles can be an excellent time saver.

You also need to replace any third-party applications that used the Service Console. Most add-ons are now available in ESXi-friendly versions, so you should be able to continue to use the applications you rely on, although you may need to upgrade them to the latest version. Most of these tools use a common set of APIs that works equally with either host. Check with your vendors to be sure they're ESXi host compatible; if they don't offer this, consider migrating to an equivalent tool from another vendor that does.

Postinstallation Design Options

From a design perspective, a number of configurations are important for your deployed hosts. You can include many of these in the post-install section of a scripted install, use a separate vMA or PowerShell script, push them out through host profiles, or configure them manually. But you should set these, because your host design isn't complete without them.

Here is a list of common post-install tasks that you should consider for an ESXi server deployment:

Hostname and IP Addressing During an interactive ESXi install, the hostname or IP address isn't specified. If you're using Auto Deploy, then these details are provided during the provisioning process. You can allow DHCP to configure the network settings for you, or you can specify a static IP address. It's always advisable to use static IP addresses for ESXi hosts with the corresponding hostname registered in DNS, because so many functions rely on name resolution and dependable IP connectivity. After a manual install, log into the DCUI and assign the static IP address, default gateway, subnet mask, and VLAN ID if this is a trunked connection. Set the server's DNS servers and fully qualified domain name (FQDN), and run the Test Management Network option. However, you may prefer to use DHCP reservations to assign the same DHCP leased address to the server via its MAC address. Using purely dynamically leased DHCP addresses, which have the potential to change regularly, isn't advisable for ESXi hosts.

Networking Configuration Post-install, you should configure the host's networking for vMotion, fault tolerance (FT) logging, NFS or software iSCSI connections, VM port groups, and so on. Chapter 5, "Designing Your Network," looks carefully at host networking design.

NTP You should configure the ESXi host to point to an authoritative time source or Network Time Protocol (NTP) server. Various aspects of the VMkernel rely on accurate time, such as logging, performance charting, and AD authentication, and VMs can use this time to synchronize their time via VMware Tools.

Host Certificates Each ESXi 5 host creates its own unique certificate based on its FQDN. It's considered good security practice to replace the default certificates. These certificates ensure that each host and vCenter trust each other and can encrypt the traffic between them. The easiest time to replace the default certificates is before you join each host to vCenter, because if you leave this until afterward all the hosts can become disconnected when updating the vCenter certificate. Each ESXi host certificate (and all vSphere 5 certificates) is an X.509 v3 base 64 encoded SSL certificate.

Connecting to vCenter If licensed for vCenter, you need to add the host to the datacenter, folder, or cluster of your choice. Connecting the ESXi host to vCenter automatically creates the local vpxuser account and activates the vmware-vpxa daemon. Depending on the location and settings in vCenter, additional software such as HA's fdm daemon may be configured at this stage as well.

Licensing Each ESXi host needs a valid license key. The host will run in evaluation mode for the first 60 days, after which a license is required. Ordinarily, vCenter is used to centrally manage the license keys; when the host is added to vCenter, you can apply the license. But if no vCenter is available, you can apply license keys directly through the vSphere Client. ESXi hosts are licensed by the number of physical sockets used in the server. If no license is applied, then after 60 days the host will drop to the limited vSphere hypervisor level, which was described earlier in the chapter.

Patching After you deploy an ESXi host is an opportune time to make sure all the latest patches are applied, before you launch the server into production. The host can be patched via the VUM or the ESXi Shell.

Storage Post-install, you also need to configure the host's storage and the connections. Chapter 6 examines host storage design.

Scratch Partition As you've seen in this chapter, there are several circumstances in which a scratch partition isn't created and the scratch directory is backed only by a volatile ramdisk. You should check your hosts' builds on the equipment you're using and take remedial action if required. The primary option is creating a shared scratch directory on a VMFS volume and pointing all the hosts to it. Use the advanced setting `ScratchConfig.ConfiguredScratchLocation` to set this option for each host.

Remote Logging If the server is considered Stateless or removable, and no local storage is available, the logs won't survive reboots. One solution is to provide a folder on a datastore to store each server's log files. You can do this using the advanced setting `Syslog.global.logDir`. An alternative is to configure the ESXi server's syslog daemon to ship the logs to a remote syslog server. It can be beneficial to send all your servers' log files to a centralized tool, even stateful ones with persistent local storage, to analyze them collectively and have one place to go. To do this, use the advanced setting `Syslog.global.logHost`. An improvement in vSphere 5 is that there is now an option to send syslog traffic via SSL. You can find all the syslog settings in `http://kb.vmware.com/kb/2003322`. The vCenter server installation wizard has a link to a Windows-based syslog server application that can be installed on the vCenter server (or any Windows server). The VCSA also has a syslog service that can be enabled and used to collect host syslog data.

Dump Collector On stateful installs, the vmkDiagnostic partition is created to capture a dump of the kernel's memory should the host unexpectedly fail (PSOD). If you prefer to centrally store these dumps for debugging later, and particularly if the host is Stateless and therefore wouldn't retain them, you can install the Dump Collector service on a Windows server from the vCenter Server installation media. Alternatively, the VCSA has a dump collector service that can be enabled.

vSphere 5.0 had a problem when exporting dumps remotely when hosts used a vDS. This created a particular issue because Auto Deployed hosts were often connected to a vDS type

switch and needed somewhere to remotely store dumps. This has been resolved in vSphere 5.1—another reason you should look at using 5.1 if you're considering Auto Deploy.

To enable a host for the Dump Collector service, in the host profile go to Network Configuration ➢ Network Coredump Settings ➢ Fixed Network Coredump Policy.

Shared VMware Tools Directory When deploying ESXi with Auto Deploy, it's common practice to use an image that doesn't have the VMware Tools components to reduce the image's size. If you do this, it's important to create a centralized VMware Tools directory that the hosts can reach, and reconfigure those hosts appropriately.

The following is a brief overview explaining how to create a shared directory and configure the hosts:

1. Copy the contents of the `/productLocker` directory from an ESXi server that has the VMware Tools folder to a suitably created folder on a shared datastore that all the hosts can access.
2. Set the `UserVars.ProductLockerLocation` attribute in the hosts' host profile to point to the shared datastore folder.
3. Reboot the servers so they take the new host profile setting.

ESXi 5.0 GA and Shared VMware Tools Directory

There is a known issue with the GA release of ESXi 5.0 using a shared VMware Tools directory. Make sure you're running at least 5.0 update 1 to avoid this defect.

SNMP Hardware Monitoring In addition to the CIM agentless monitoring available, ESXi can also use SNMP to send monitoring traps about the server's hardware and VMs. vSphere 5.0 embedded the SNMP agent in the hostd service, but in 5.1 this has been decoupled and runs as its own daemon. vSphere 5.1 has also been upgraded to run SNMP v3 (5.0 used SNMP v2), which improves the security options. To enable SNMP monitoring of your hosts, you need to use the `esxcli system snmp` command (`vicfg-snmp` on ESXi 5.0 hosts).

Local User Permissions There are several options here. You allocate permissions to most users via vCenter Server, but you can also create local user accounts. Local user accounts are obviously important if the design won't have access to a vCenter, but their management doesn't scale well past a handful of users and servers. Local users are configured on a per-host basis and provide user roles akin to vCenter roles, to graduate levels of access and control. Local host access is important if users will connect directly to the host via the vSphere Client, the DCUI, or the ESXi Shell. In vSphere 5.1, the requirement to use the root account when working on the ESXi Shell has been removed. Local users who are assigned the Administrative role on the host get full shell access. This means all sessions can be logged and audited against named local users. Host profiles can also be used in 5.1 to create named local user accounts and assign privileges on every host.

Active Directory Authentication Integrated AD authentication provides a secure, convenient way to manage local access to ESXi hosts. Using AD authentication requires that the

host be joined to AD and the required users or groups be given the desired permissions. A default AD group entitled ESX Admins simplifies the process, because it's automatically added to each host and given full administrator access to each host.

To restrict access, you may wish to create specialized AD groups and provide only a certain level of access and only to specific hosts. An advanced setting can also be set on hosts to prevent the ESX Admin group from getting full administrative access: `Config.HostAgent.plugins.hostsvc.esxAdminsGroupAutoAdd`.

If you're deploying hosts via Auto Deploy, a small tool called the AD Authentication Proxy eliminates the need to store domain admin credentials in the host profiles. This authentication proxy tool is a ready-to-use service on the VCSAs or a Windows-installable application available on the vCenter ISO image.

Lockdown Mode Lockdown mode is a vCenter feature that can increase the security of ESXi hosts by forcing all interactions through vCenter. It disables remote access, except for vCenter's vpxuser account. This means it can take advantage of the centralized nature of vCenter's roles and permissions and makes vCenter audit all remote access. The root user can still access the ESXi host locally on the DCUI interface.

Lockdown mode doesn't disable local ESXi Shell or remote ESXi Shell services, but it does prevent any user (including root) from accessing them because the authentication permissions are locked.

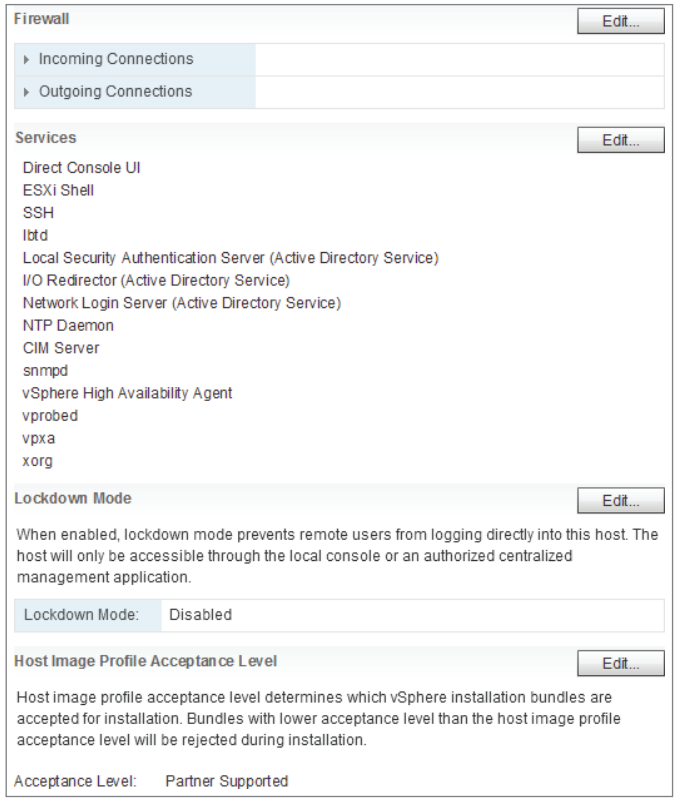
You can only enable lockdown mode on vCenter-connected hosts. It's disabled by default but can be enabled via the vSphere Client or the DCUI. Figure 2.4 shows the lockdown mode interface on the vSphere Client.

Table 2.2 shows how lockdown mode affects each remote-access method for hosts.

TABLE 2.2: Lockdown mode impact

ACCESS	DEFAULT MODE	LOCKDOWN MODE
DCUI	Root and users with admin privileges	Root only
Local ESXi Shell	Root and users with admin privileges	No access
Remote ESXi Shell (SSH)	Root and users with admin privileges	No access
vSphere Client direct to host	Root and users with admin privileges	No access
vSphere Client via vCenter	Users with admin privileges	Users with admin privileges
vCLI/vMA script to host	Root and users with admin privileges	No access
PowerCLI script to host	Root and users with admin privileges	No access

FIGURE 2.4
Lockdown mode



Third-party monitoring software that uses the CIM broker on each host is also affected by lockdown mode. A CIM customer must connect directly to hosts to get the hardware information, but in lockdown mode it must get an authentication ticket from vCenter. This allows it to use the vpxuser credentials to gather the details from the host's CIM interface.

Lockdown mode is reversible and can be disabled on a host-by-host basis. Despite lockdown mode being disabled by default, it's worth considering enabling it across an environment during your design. If you need remote access to the host, you can temporarily disable lockdown while you perform administrative local tasks, and then re-enable it afterward. However, enabling lockdown mode via the DCUI causes the local user and group permissions to be lost. To retain these local permissions, be sure you enable lockdown mode via vCenter.

For very security-conscious settings, you can take an additional step known as *total lockdown mode*. This is a combination of enabling lockdown mode and disabling the DCUI. Access can only be granted via vCenter, preventing anyone with the root password from disabling lockdown mode locally via the DCUI. But if the DCUI is disabled, and vCenter isn't available and can't be restored, then your only recourse to gain administrative access to the host is to reinstall ESXi.

Security Profile The security profile allows configuration of ESXi's internal firewall, which protects the management network port. By default, all ports are closed except those considered essential services.

On ESXi hosts, you can enable and disable access to certain remote access services, as you can see in Figure 2.4. But you're merely specifying that listed daemons can start or stop, and the setting of their runlevels. The most common services are listed explicitly in the web client, but additional ports can be manually added as regular firewall rules. Access can also be restricted to an IP address or a subnet. In addition to using the web client, you can adjust the security profile via PowerCLI, ESXCLI, or host profiles.

Firewall Ports Finally, you may need to open a number of ports if your hosts and other infrastructure pieces are divided with firewalls. Don't confuse this with ESXi's local security profile firewall mechanism just described. The primary ports required are listed in Table 2.3. Other ports that may be required are 88/389/445/464/1024 AD, 161/162 SNMP, 445 SMB, 514/8001 syslog, 111/2049 NFS, 3260 iSCSI, 6500/8000 Dump Collector, 6501/6502 Auto Deploy, 8100/8200 FT, and 8182 HA. You can find an excellent article that details firewall ports for vSphere and many other VMware products at `http://kb.vmware.com/kb/1012382`. Table 2.3 shows the ESXi host firewall requirements.

TABLE 2.3: ESXi host firewall requirements

PORT	SOURCE	DESTINATION	DIRECTION RELATIVE TO HOST	PROTOCOL	DESCRIPTION OF ESXI SERVICE
22	SSH client	ESXi server	Inbound	TCP	SSH access to server
53	ESXi server	DNS servers	Outbound	UDP	Name resolution
68	ESXi server	DHCP servers	Outbound	UDP	Dynamic IP address
80	vSphere Client	ESXi server	Inbound	TCP	Browser redirect to HTTPs
123	ESXi server	NTP	Outbound	UDP	NTP (time)
427	Other hosts and vSphere Clients	ESXi server	Inbound	UDP	Service location (SLPv2)
427	ESXi server	Other hosts and vSphere Clients	Outbound	UDP	Service location (SLPv2)
443	Other hosts, vSphere Clients, and vCenter	ESXi server	Inbound	TCP	Host management (HTTPS) and VM operations
902	Other hosts	ESXi server	Inbound	TCP/UDP	Provisioning and migration
902	ESXi server	vCenter	Outbound	TCP/UDP	Heartbeat to vCenter

TABLE 2.3: ESXi host firewall requirements *(continued)*

PORT	SOURCE	DESTINATION	DIRECTION RELATIVE TO HOST	PROTOCOL	DESCRIPTION OF ESXI SERVICE
902	vSphere Clients	ESXi server	Inbound	TCP	VM console (MKS)
5900–5964	Other hosts	ESXi server	Inbound	TCP	Management tools (VNC)
5900–5964	ESXi server	Other hosts	Outbound	TCP	Management tools (VNC)
5988, 5989	vCenter and CIM server	ESXi server	Inbound	TCP	CIM updates
5989	ESXi server	vCenter	Outbound	TCP	CIM updates
8000	Other hosts	ESXi server	Inbound	TCP	vMotion requests
8000	ESXi server	Other hosts	Outbound	TCP	vMotion requests
8301, 8302	Other hosts	ESXi server	Inbound	UDP	Distributed virtual switch (DVS) port information
8301, 8302	ESXi server	Other hosts	Outbound	UDP	DVS port information

Management Tools Overview

Once any ESXi host is deployed and configured, several management tools exist to monitor and maintain it. Many of these are discussed in depth in the next chapter, but let's review the full gamut of tools available and how they help specifically with the corralling of your ESXi hosts. The tools' functionality can fundamentally affect your design of a vSphere environment and how it will be maintained, and therefore your hypervisor architecture.

Host-Management Tools

Here are some tools you can use to manage hosts directly without any access to vCenter. These are often useful when first configuring hosts, when troubleshooting hosts, or while there are problems with vCenter.

vSphere Client

The Windows vSphere Client, often referred to as the C# or legacy client these days, can connect to an ESXi host directly in a way similar to how it can connect to vCenter. When directly connected, it uses the local ESXi accounts.

DCUI

The DCUI is the BIOS-styled yellow menu tool that appears on an ESXi server's console screen. Figure 2.5 shows a typical DCUI screen.

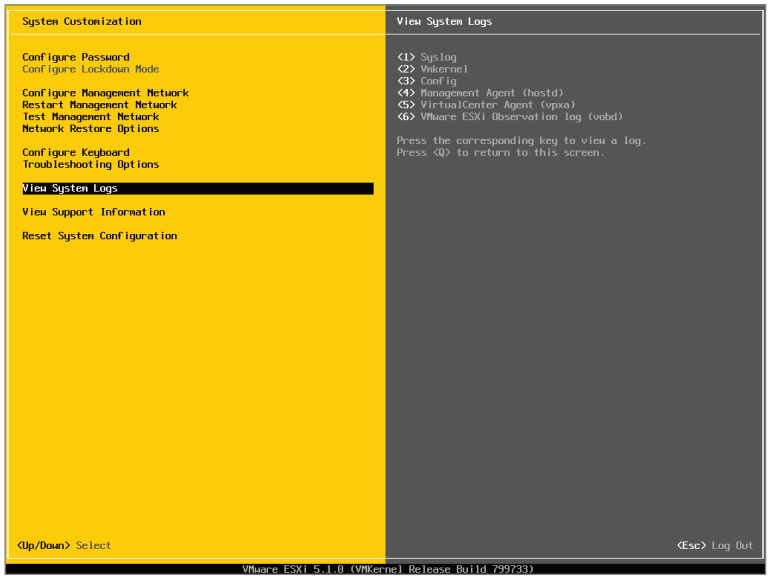


FIGURE 2.5
DCUI interface

You can use the DCUI to initially configure the management network's details, set the administrative password, monitor and restart management agents, view logs locally, enable access to the ESXi Shell, and turn on lockdown mode. It allows onsite first-line staff to set basic configuration options and perform rudimentary troubleshooting actions. This allows more complex configuration and management tasks to be performed remotely. The DCUI is focused on all the tasks that can prevent successful remote connections, such as network-configuration errors and failed management agents.

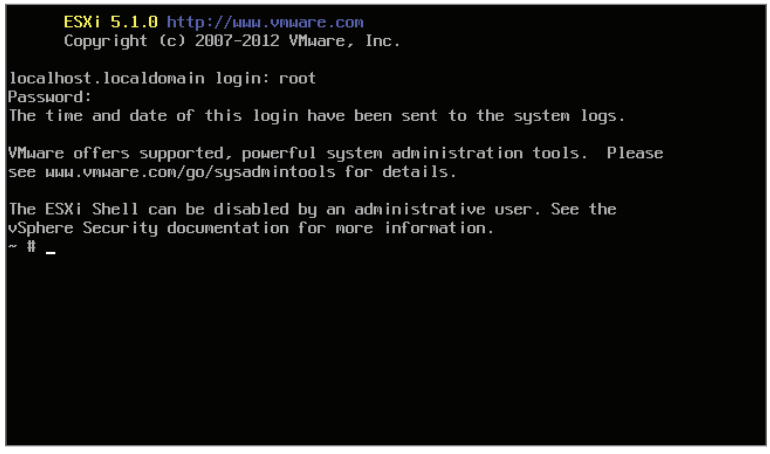
ESXi Shell

The ESXi Shell is a simple shell that provides a local console on which you can perform advanced troubleshooting and technical support. Regular maintenance tasks are still best executed with tools such as the vSphere Web Client and the vMA. In addition to local access, the tool offers remote ESXi Shell mode, which allows remote connections through SSH. Figure 2.6 shows an active logged-in local ESXi Shell session.

The ESXi Shell isn't a regular Linux shell and doesn't give you exactly the same environment that the Service Console used to. It uses the ash shell, instead of the more common bash shell, and it doesn't include all the normal Linux commands. For example, the current ESXi Shell mode doesn't include man (manual) pages for commands. Those commands that are present may not offer the same arguments and options, may be limited, and may not operate as you expect. The ESXi Shell is still a very useful tool for quick break-fix situations. Most important, the ESXi Shell gives you access to the complete `esxcli` command set and `esxtop`, the real-time

resource management tool for the console. The ESXi Shell doesn't allow any scheduled scripting jobs and is focused only on that single host—this is where the vMA shines.

FIGURE 2.6
ESXi Shell console



The ESXi Shell is disabled by default and must be turned on via the DCUI or vSphere Client before use. All commands are logged by the syslog service, so a full audit trail remains; and a timeout is available for both local ESXi Shell and remote ESXi Shell modes to ensure that no sessions stay logged in.

The ESXi Shell can be very useful if management agents are unresponsive, because at that point the vSphere Client, vCenter, vMA, or PowerCLI tools won't help. You can log in to the DCUI and try to restart the agents; if there are further problems, the ESXi Shell gives you console access to troubleshoot the issue. The ESXi Shell is usually quicker to use than some of the remote tools, so it's ideal when you need a fix as soon as possible. However, the filesystem presented in the ESXi Shell is formed from relatively small ramdisks; you must be careful not to copy large files to it as you may do on the vMA, because filling any of the partitions is likely to make the entire server unstable. Also remember that the system's files are laid out as it boots from the system image, so files that you change or add may not survive a reboot. The ESXi Shell is a user world on ESXi and so is limited by the resource constraints the VMkernel sets. This means the console can run out of memory if too many commands are run at once.

Local ESXi Shell and remote ESXi Shell access are both available to local users who are granted the Administrator role. The modes must be enabled for use, and access can be affected by the lockdown mode discussed earlier in the chapter.

vCLI and vMA

The vCLI is a Perl-based set of scripts that mimics the commands available at the ESXi Shell. The major difference between vCLI commands and ESXi Shell commands is that in the command syntax, you need to specify which host or vCenter you wish to direct the command to and with what credentials. The vCLI is packaged for both Linux and Windows.

The vMA is a small, Just enough OS (JeOS) prepackaged Linux virtual appliance that comes with the vCLI installed and ready to use. It's a convenient way to get the vCLI toolkit up and running quickly. You can also use the vMA as a syslog server for your ESXi hosts. And you can

use it to run scripts, create scheduled tasks with cron, and centrally run commands to several hosts at once. Both the vCLI and the vMA can be used against ESXi hosts and vCenter Server.

PowerCLI

The PowerCLI is similar to the vCLI, except that instead of Perl and Unix style syntax, it's Microsoft PowerShell based. It's a toolkit that you can install on Windows PCs with PowerShell installed. You can use it to run remote commands and scripts against ESXi hosts and vCenter. Like the vCLI, it has a vibrant community around it with many sample code snippets and scripts openly available.

Browser-Based Tools

A notable deficiency of ESXi at the time of writing (5.1 GA) is the lack of Web Client access. As long as the ESXi host is attached to vCenter, you can control VMs through vCenter's Web Client.

A nice web-based feature that is available on all vSphere hosts is a listing of the configuration files and log files. Although very simplistic, it gives a quick one-stop view of a host's settings. Figure 2.7 shows an example of the file listing, which is available through `https://<hostname>/host`.

There is also a similar web-based access to view a host's datastores on `https://<hostname>/folder`.

Centralized Management Tools

In addition to the stand-alone direct tools we just covered, there are several centralized management tools to nurture your ESXi hosts. vCenter is pivotal to these tools and is covered heavily in Chapter 3, "The Management Layer." Suffice it to say most of the following are dependent on vCenter for their operation.

vSphere Web Server

The vSphere Web Server client is the new interface of vCenter operations. First introduced in 5.0 as a basic VM management tool, the 5.1 release promoted it to full status as a vSphere client. It now supplants the Windows client as the principal tool, with all new features from 5.1 only available through it. It's undoubtedly the forward direction for vSphere, and the race is on for third-party plug-ins and remaining features still attached to the Windows client to get on board.

With the release of vSphere 5.1, the main sticking point is the Web Client's inability to connect directly to hosts. We'll have to wait for a future release until the ESXi hosts can be recognized by the SSO system to allow this functionality.

VUM

VUM is a vCenter plug-in that can scan hosts for missing updates, upgrades, and third-party driver and CIM modules, and centrally push out those patches and coordinate their installation. It's very useful for large environments where patching hosts regularly can otherwise prove to be an onerous task.

Host Profiles

Host profiles are a vCenter feature that can check for and apply consistency across ESXi hosts. This feature allows you to set a standard configuration to a single host or a cluster of hosts and

also automatically check for compliance to that standard. It reduces the management overhead associated with maintaining host configurations, helps to prevent misconfigurations, and alleviates some of the repetitive burden of setting up each host. Hosts must have Enterprise Plus licensing to use host profiles.

FIGURE 2.7
Web browser access to configuration and log files

Host profiles have a growing relevance to ESXi hosts moving forward. As organizations look to automatic host deployments and more Stateless hosts, host profiles provide an especially useful tool to automate the post-install configuration.

Host profiles are also valuable to companies with very large numbers of hosts, to keep their settings consistent. You can create scheduled tasks in vCenter to check on host compliance. By

default, a check for compliance is set to run once every 24 hours when a host profile is applied. Alarms triggers can also be applied, to alert you that profiles are being applied or non-compliance exists. You can only schedule tasks around compliance, not tasks to apply profiles.

Host profiles are particularly suited to new host installations, because this is when most of the work involving configuration-setting takes place, and host profiles work best on clusters of identical hardware. Slight differences between hardware can be tolerated, but it's advisable to start with the oldest server with the fewest hardware options (such as additional PCI cards). The profile won't work well if the reference host has more settings configured than a recipient host can apply.

Each profile is captured from a reference host when the settings of one host that has been configured properly are recorded. You can export each profile to a file with the extension `.vpf`; it's a simple XML-formatted file. To modify and update a profile, you can either change the settings on the reference host and recapture the profile or use the Profile Editor in vCenter, which lets you apply more advanced options. When you use the Profile Editor to change an existing profile, it's advisable to export the original first, rename it, and import it back in. Then you can work on a duplicate profile, test it before it's applied to an entire cluster, and still have the base profile to roll-back to.

Profiles are flexible enough to allow per-host settings, so even though most of the configuration options will be exactly same for every host, each host can vary. The options available for each setting are as follows:

Fixed Configuration Every host will be identical.

Allow the User to Specify Before applying a profile, the user is asked for the value. This is useful for per-server settings like hostname and IP address.

Let vCenter Pick The best value is chosen by vCenter. This option is often used for network settings and selecting which adapters to use.

Disregard Setting The setting will be ignored by the profile.

Although you can apply host profiles just to individual hosts, doing so limits the usefulness of host profiles to *compliancy*. To make the most of this feature, you should try to apply profiles at the cluster level. If you have multiple vCenters in linked mode, then the host profiles aren't available across them. You can export a profile from one vCenter to use in another, but it won't remain synchronized.

Try to minimize the number of different configurations where possible, and group like hosts together. You can apply a profile to a mixed cluster containing legacy vSphere 4.x hosts (ESX and ESXi). It's possible to apply ESX classic profiles to ESXi hosts but not vice versa. The Host Profiles feature can translate the Service Console and VMkernel traffic settings to ESXi management networks, but not the other way around. Therefore, if you have clusters with mixtures of ESX and ESXi hosts, you should use one of the ESX hosts as the reference host to avoid issues.

To apply a profile to a host, that host must first be in maintenance mode. For that reason, you must maintain sufficient redundant capacity with the servers to apply profiles without causing VM outages. Clusters have inherent compliance checks for special features such as DRS, DPM, HA, and FT. You don't need host profiles to keep these settings consistent.

Hardware Monitoring

ESXi's primary hardware monitoring is based on CIM. CIM is an open standard that allows information to be exchanged and can allow control of managed items. For example, Windows

Management Instrumentation (WMI) is a CIM implementation, and storage vendors have a version of CIM known as Storage Management Initiative-Specification (SMI-S).

ESXi's CIM usage provides an agentless service to monitor hardware resources. It consists of two pieces: the CIM providers that allow access to particular device drivers and hardware, and the CIM broker that collects the information from all the providers and makes it available via a common API. In addition to the default VMware CIM providers for common hardware on the HCL, you can install supplementary server-vendor, and OEM manufacturer specific CIM providers to extend the hardware-monitoring capabilities.

Because CIM is a common standard, most server-monitoring software can collect information from ESXi servers. vCenter can use the CIM information from the ESXi brokers. You can view this from connected vSphere Clients, and you can set vCenter alarms to warn of failures.

Logging

The collation and retention of logs is not only vital for troubleshooting host issues but also usually necessary for legal compliance reasons. It's important that you configure the hosts correctly for time synchronization with a suitable NTP time source, to ensure that the logs are accurate. Locally, all ESXi server logs are stored in `/var/log`. An improvement with vSphere 5 is that the logs that were previously combined into just a handful of files are now split into multiple single-purpose logs. This makes finding the pertinent information when troubleshooting that much more straightforward. The logs of primary interest are as follows:

- `auth.log` ESXi Shell authentication
- `esxupdate.log` ESXi patches/updates
- `fdm.log` HA logs
- `hostd.log` Host management
- `shell.log` ESXi Shell usage
- `sysboot.log` VMkernel and module startup
- `syslog.log` Management service initialization, watchdogs, scheduled tasks, and DCUI
- `vmkernel.log` Core VMkernel logs (devices, storage/network device/driver events, and VM startup)
- `vmkwarning.log` VMkernel warnings and alerts
- `vmksummary.log` ESXi startup/shutdown, uptime, VMs running, and service usage
- `vpxa.log` vCenter vpxa agent

A little known gem included in the new 5.1 Web Client is the integrated Log Browser. From it you can not only view all the pertinent logs centrally in a simple GUI interface, but it has advanced filtering and searching, can compare multiple logs to each other, and highlights key words and entries. You can review the ESXi logs in a number of other ways:

- vSphere Web Client's excellent log browser
- DCUI: Select View System Logs (can be seen in Figure 2.5)

- ESXi Shell console (local or remote): `tail` each log file
- vMA/vCLI/PowerCLI: Programmatically retrieve files
- vSphere Windows Client (connected to host or vCenter): Export diagnostic data
- Web browser: `http://<hostname>/host` (seen in Figure 2.7)
- Syslog server

Summary

Host design is an important foundation for any vSphere deployment. The host is the core component; the efficiency and effectiveness of any design depends on this key underpinning. A deeper understanding of its internal make-up and moving parts helps to ensure that you can respond to the associated design questions.

We've looked at the many decision points regarding ESXi deployment and upgrades. The selection depends on many diverse factors such as the size of the deployment, geographical distribution, available equipment, skills of the support staff, delivery expedience required, frequency of builds and rebuilds, and resilience of the infrastructure. It's feasible that your deployment solution will encompass more than one technique to suit your disparate environment.

The other important aspect of the design, once the deployment is complete, is a grasp of the outcome. Each host's layout and configuration is not a result of happenstance but of several competing factors weighing in, and the upshot can be complex to predict. The image used, hardware, delivery system, destination, and settings applied all determine how a host will react. How the servers cope with key infrastructure pieces like vCenter and the Auto Deploy hosts being offline, the ability of hosts to maintain log files after a crash, and how your operations team needs to subsequently patch the hosts need to be tested thoroughly to vindicate the design.

Chapter 3

The Management Layer

In this chapter, we'll discuss the points you should take into account when you're designing your management layer. We'll examine the components of this layer and how you incorporate them into your design.

The management layer comprises several components. In this chapter, we'll address what you should consider for your design regarding these items, among others:

- Operating system and resources to use for your vCenter Server
- Deciding whether your vCenter Server should be physical or virtual
- Providing redundancy for the vCenter Server
- Planning for the security of the management layer

Reviewing the Components of the Management Layer

What is the management layer? It's definitely not the executives or the board of directors of your company. We're talking about the components you use to manage your entire virtual infrastructure on a day-to-day basis. In this section, we'll provide a quick overview of vSphere's management components. We'll start with the main component, vCenter Server.

VMware vCenter Server

vCenter Server (which was once known as VirtualCenter Server) is one of the most centrally critical elements of your virtual infrastructure. It's the management application you'll likely use to manage your virtual datacenter. You'll create datacenters, clusters, resource pools, networks, and datastores; assign permissions; configure alerts; and monitor performance. All of this functionality is centrally configured in the vCenter Server. You should therefore dedicate part of your design to building a robust and scalable vCenter Server.

Compatibility Matrix

Always check the current VMware compatibility matrix before deciding which platform you'll install vCenter Server on. VMware will provide support only if your infrastructure is installed on supported software. This includes the hardware for your ESXi hosts, vCenter Server, underlying database, vSphere Client, vSphere Update Manager, web browser, and so on.

As we'll show you shortly, vCenter Server comes in a couple of different forms (either as a Linux-based virtual appliance or as a Windows application you install on Windows Server). In either case, you can download the most up-to-date version of vCenter from the VMware site, but be warned: depending on which version of vCenter Server you're downloading, it can be a pretty large download (almost 4 GB for the Linux-based virtual appliance, for example).

vCenter Server Components

Prior to vSphere 5.1, vCenter Server was largely a monolithic application. There were really only three major components involved with vCenter Server in vSphere 4.x and vSphere 5.0:

- An operating system instance. This could be either a Windows OS (a domain member server—it couldn't be a domain controller) or a preinstalled instance of Linux bundled with the vCenter Server Virtual Appliance (vCSA).
- Access to a back-end database. The local computer where vCenter Server is running could host this database, or a remote computer could host the remote database instance.
- vCenter Server itself, either installed onto an instance of Windows or preinstalled onto Linux as part of the vCSA.

With the introduction of vSphere 5.1, VMware has broken the monolithic vCenter Server into a number of different components. In addition to the three components listed previously, vCenter Server 5.1 also includes

- vCenter Single Sign On, a centralized authentication service that enables authorized vCenter Server users to access multiple vCenter Server instances with a single login.
- vCenter Inventory Service, a service that stores vCenter Server application and inventory data and that works across linked vCenter Server instances. Strictly speaking, this component isn't new to vSphere 5.1; what VMware has done with the 5.1 release is separate the Inventory Service so it can be installed on a separate computer for greater scalability.

The introduction of these additional components in vSphere 5.1 means you also have to decide if you'll run all these components on a single computer or break the roles apart on separate computers.

Let's take a closer look at each of the components in vCenter Server.

Operating System Instance and vCenter Server

We've grouped vCenter Server and the OS instance together because these two components are directly linked to each other. vCenter Server comes in two basic flavors: as an installable application running on an instance of Windows, or as a preinstalled application running on Linux as part of the vCSA. As you can see, choosing Windows as the OS on which to run vCenter Server means you're choosing the installable application version. Choosing Linux as the OS, on the other hand, means you're choosing the preinstalled version of vCenter Server in the virtual appliance. At the time of this writing, there was no option for a Windows-based virtual appliance or an installable version of vCenter Server that ran on Linux.

Later in this chapter, we'll discuss the considerations that go into selecting either the installable version of vCenter Server or the vCSA; we'll also discuss the considerations for choosing a version of Windows on which to run vCenter Server. (Note that you can't choose a Linux distribution or version; this is preinstalled as part of the virtual appliance.)

Back-End Database

Regardless of which form of vCenter Server you choose, you'll need a back-end database. One question that always comes up during the design phase is, "Do I use a central database server or install the database locally on the vCenter Server?" This question ranks right up there with similar questions like, "Should I use full-blown Microsoft SQL Server or Oracle as my database engine?"

We'll examine these questions and others in greater detail later in this chapter. In the section "Examining Key Management Layer Design Decisions," we'll discuss the various reasons you might choose one database over another (vCenter Server supports several different database servers) and whether you should host the database locally or remotely.

At this stage, though, you just need to know that vCenter Server requires a back-end database. As a result, you'll need to account for the additional resources that a back-end database requires, and you'll need to plan for the maintenance and operation of the back-end database server. In the section "Creating the Management Layer Design," we'll review how you can ensure the proper availability, manageability, performance, recoverability, and security for the back-end database. (Recall that these principles—introduced in Chapter 1, "An Introduction to Designing VMware Environments," and referred to as AMPRS—help guide your design decisions.)

vCenter Single Sign On

With the release of vSphere 5.1, VMware introduced a new component to vCenter Server known as vCenter Single Sign On. This component introduces a centralized authentication service that vCenter Server uses to allow for authentication against multiple back-end services, such as Active Directory and LDAP. For smaller environments, vCenter Single Sign On can be installed on the same system as the rest of the vCenter Server components; for larger environments, it can be installed on a separate system. vCenter Single Sign On also supports a variety of topologies, including a single server, a cluster of servers, and a multisite topology.

vCenter Inventory Service

To enable greater scalability for vCenter Server, in vSphere 5.1 VMware also split off the inventory portion of vCenter Server into a separate component. The vCenter Inventory Service now supports the discovery and management of inventory objects across multiple linked vCenter Server instances. As with vCenter Single Sign On, you can install vCenter Inventory Service on the same system as the other components (in what is called a *Simple Install*), or you can split the Inventory Service onto a separate system for greater scalability (and perhaps high availability/redundancy).

vSphere Web Client Server

To enable the next-generation vSphere Web Client—something we discuss in the next section—vCenter requires a server-side component referred to as vSphere Web Client. In reality, it should be called the vSphere Web Client server, because this is the server-side component that enables the use of a web browser to manage vSphere environments. This component was introduced in vSphere 5.0, but in vSphere 5.1 it takes on much greater importance because some features and tasks in vSphere 5.1 are accessible only from the next-generation web client.

Now, let's shift our focus to the client side.

vSphere Client and vSphere Web Client

Traditionally, VMware administrators have used a Windows-based application known as the vSphere Client to perform the vast majority of the day-to-day management tasks. This is true for VMware vSphere 4.*x* as well as VMware vSphere 5.0. In vSphere 5.0, VMware introduced a rudimentary Web Client; in vSphere 5.1, the vSphere Web Client (sometimes referred to as the Next-Generation Client [NGC]) takes its first steps toward becoming the primary way of managing VMware vSphere environments.

The Windows-based vSphere Client can be installed on almost any Windows OS:

- Windows XP Pro, SP3
- Windows XP Pro 64-bit, SP2
- Windows Server 2003, SP1
- Windows Server 2003, SP2
- Windows Server 2003 Standard, SP2
- Windows Server 2003 Enterprise, SP2
- Windows Server 2003 R2, SP2
- Windows Vista Business, SP2
- Windows Vista Enterprise, SP2
- Windows Vista Business 64-bit, SP2
- Windows Vista Enterprise 64-bit, SP2
- Windows 7 Client (32-bit and 64-bit)
- Windows Server 2008 Enterprise, SP2
- Windows Server 2008 Standard, SP2
- Windows Server 2008 Datacenter, SP2
- Windows Server 2008 Enterprise 64-bit, SP2
- Windows Server 2008 Standard 64-bit, SP2
- Windows 2008 R2 64-bit

The minimum resources required for the Windows-based vSphere Client are as follows:

- 266 MHz or faster Intel or AMD processor (500 MHz recommended)
- 1 GB RAM
- 1 GB free disk space for a complete installation, which includes the following components:
  - Microsoft .NET 3.5 SP1
  - Microsoft Visual J# 2.0 SE

You'll also need 400 MB free on the drive that has the `%temp%` directory during installation.

If all the prerequisites are already installed, 300 MB of free space is required on the drive that has the `%temp` directory, and 450 MB is required for vSphere.

- A gigabit network connection is recommended.

For the vSphere Web Client, the following browsers are supported:

- Microsoft Internet Explorer 7, 8, and 9
- Mozilla Firefox 3.6 and higher
- Google Chrome 14 and higher

TIP As of this writing, Apple's Safari web browser was not supported for use with the vSphere Web Client.

As we stated earlier, the vSphere Web Client takes on significant new importance in vSphere 5.1 environments. To take advantage of vCenter Single Sign On, for example, you must use the vSphere Web Client. If you use the traditional Windows-based vSphere Client, authentication won't use the new vCenter Single Sign On component. You'll note that throughout this book we make an effort to show all screenshots from the new vSphere Web Client as opposed to the older, Windows-based vSphere Client (which remains largely unchanged from previous versions).

These components make up the core of the management layer for a vSphere implementation. However, VMware also offers a number of optional components that you can also choose to include in your design. The first of these that we'll examine is vSphere Update Manager.

vSphere Update Manager

vSphere Update Manager (VUM) is an add-on that VMware provides in order to update your ESXi hosts and VMs. VUM (`www.vmware.com/products/update-manager`) is VMware's patch-management solution for your virtual environment. It's included with most tiers of vSphere.

In versions of vSphere prior to vSphere 5.0, VUM provided a patch-management solution not only for your ESXi hosts but also for the OSes and certain applications in supported VMs. With the release of vSphere 5.0, VMware discontinued the patch-management component for the OSes and applications in the VMs. This allows organizations to use existing patch-management solutions in place to patch their VMs the same way they patch their physical machines.

A VUM installation requires a few components to get started:

- A 64-bit instance of a Windows OS. This must be a domain member server (not a domain controller).
- Access to a database. This can be a remote database (Oracle or Microsoft SQL), or it can be installed locally on the VUM server.

You can install VUM on the same instance of Windows as vCenter Server, if you like; or, for greater scalability and security, you can install VUM on a separate instance of Windows. Note that although VUM requires a 64-bit instance of Windows, VUM itself is a 32-bit application.

With regard to the VUM database, VUM can reside side-by-side with the vCenter database on the same database server, but it can't be the same database as the vCenter Server. In addition to setting up a separate database server (or using an existing separate database server), you also have the option of using an embedded SQL Server 2008 R2 Express database on the VUM server. As soon as you exceed 5 hosts and 50 VMs, though, you'll need to step up to a separate SQL Server or Oracle server for the VUM database.

When you install vCenter Server, you create a system data source name (DSN) entry for the database. You need to create an additional DSN entry for the VUM database as well, as you can see in Figure 3.1. Note that because VUM is a 32-bit application, this needs to be a 32-bit DSN.

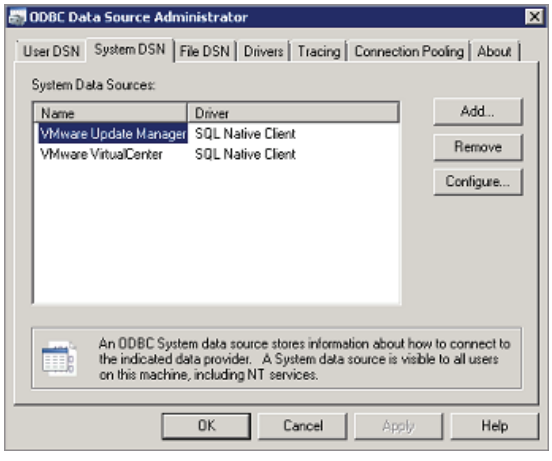


FIGURE 3.1
vSphere Update Manager requires an additional ODBC connection.

Once you've installed VUM, you configure it for the updates you want to download, schedule the automatic download of patches, set up notifications, scan your ESXi hosts, and update them. Optionally, if the environment necessitates, you can set up a dedicated download server to download patches and distribute them to your VUM server(s). This is particularly applicable in high-security environments, where only certain server are permitted to access external resources on the Internet.

NOTE The plug-in that VUM uses is only compatible with the vSphere Client. Thus, in vSphere 5.1 environments, you must use the vSphere Client—not the vSphere Web Client—to perform VUM configuration and VUM-related tasks.

Now let's look at some other applications that are also part of the management layer.

Management Applications

Logging in to each host to perform a management task—such as configuring a network vSwitch or setting the maximum number of NFS mounts you can connect on an ESXi host—can be a tiresome and repetitive task. It may become a nightmare if (and when) your infrastructure grows.

Let's start with the basics. ESXi does have a management console, although its use is generally discouraged in favor of other solutions (which we'll discuss later in this section). Prior to vSphere 5.0, vSphere included both ESX (which offered a Service Console based on a customized

Linux kernel) and ESXi (which offers a management console based on a BusyBox environment). With the release of vSphere 5.0, vSphere now only includes ESXi.

To connect these consoles remotely, you need an SSH client. Most administrators prefer using PuTTY (`www.chiark.greenend.org.uk/~sgtatham/putty`). Several other tools provide the same functionality, and every administrator has a preference. Administrators using Mac OS X or Linux can also use the preinstalled SSH client that is available via their native terminal application. In vSphere 5.0 and 5.1, you'll also need to enable SSH access to the ESXi management console (it's disabled by default). This is done from the Direct Console User Interface (DCUI).

Management consoles aren't the end of the story, though. ESXi and vCenter have an extensive API you can use to perform remote management. Building on this API, VMware provides several tools to manage vCenter and your ESXi hosts remotely:

- vSphere command-line interface (vCLI)
- PowerCLI
- vSphere Management Assistant (vMA)

Note that we haven't included orchestration tools such as vCenter Orchestrator in this list, nor is vCloud Director included. Although they could be considered management tools, our focus here is on day-to-day management tools at the vSphere level. Note that we do discuss some vCloud Director design considerations in Chapter 12, "vCloud Design."

The good thing about the different tools is that you don't have to choose between them: you can use them all. For example, both the Perl and PowerShell platforms have dedicated followers who constantly expand the ways you can use these tools in your environment.

Let's start with a look at the vCLI.

vSphere Command-Line Interface (vCLI)

The vCLI command set allows you to run common system-administration commands against ESXi systems from any machine with network access to those systems. You can run most vCLI commands against a vCenter Server instance and target any ESXi system that the vCenter Server instance manages. Because vSphere 5.0 doesn't include ESX, vCLI commands are especially useful for passing commands to ESXi hosts where direct use of the management console is generally discouraged.

vCLI commands run on top of the vSphere SDK for Perl. vCLI and the vSphere SDK for Perl are included in the same installation package.

The supported platforms for vCLI are as follows:

- Windows Vista Enterprise SP1 (32-bit and 64-bit)
- Windows 2008 (64-bit)
- Windows 7 (32-bit and 64-bit)
- Red Hat Enterprise Linux (RHEL) 5.5 (32-bit and 64-bit)
- SUSE Linux Enterprise Server (SLES) 10 SP1 (32-bit and 64-bit)
- SLES 11 and SLES 11 SP1 (32-bit and 64-bit)
- Ubuntu 10.04 (32-bit and 64-bit)

The commands you run on vCLI for management aren't exactly the same as the commands you can run on the ESXi console (or the commands you might have used in previous versions with the ESX Service Console), so you may have to adjust to the difference. Table 3.1 shows a couple of examples.

TABLE 3.1: vCLI syntax as compared to console syntax

vCLI	ESXi Shell
`vicfg-vmknic`	`esxcfg-vmknic`
`vicfg-nics`	`esxcfg-nics`

VMware KB Article 1008194 (`http://kb.vmware.com/kb/1008194`) gives a list of some of the differences/similarities between vCLI commands and ESX/ESXi console commands.

When you're connecting to a remote host, you must—at minimum—specify the following parameters: server, username, password, and command to perform. Authentication precedence is in the order described in Table 3.2.

TABLE 3.2: vCLI authentication precedence

Authentication Method	Description
Command line.	Password (`--password`), session file (`--sessionfile`), or configuration file (`--config`) specified on the command line.
Configuration file.	Password specified in a `.visdkrc` configuration file.
Environment variable.	Password specified in an environment variable.
Credential store.	Password retrieved from the credential store.
Current account (Active Directory).	Current account information used to establish an SSPI connection. Available only on Windows.
Prompt the user for a password.	Password isn't echoed to the screen.

Let's consider an example to see how these work. The environment is built as detailed in Table 3.3.

TABLE 3.3: Example for vSphere environment

vCenter Server	`vcenter.design.local`
Username	`viadmin`
Password	`a:123456`

You need to create a configuration file to define these settings. The file must be accessible while running your vCLI session. A session file for the configuration in Table 3.3 looks like Figure 3.2.

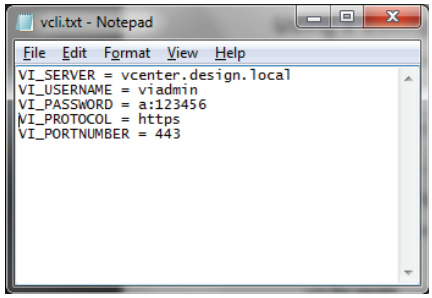


FIGURE 3.2: This configuration file supplies the necessary parameters for vCLI commands.

WARNING This file isn't encrypted in any way, so all passwords in the file are stored in plain text. Access to this file should be limited!

You can run your commands against the hosts defined in the configuration file without having to input the credentials each time you connect to a server. Here's an example of using the vCLI on a Windows-based system:

```
vicfg-nas.pl --config c:\users\administrator\vcli.txt --vihost esxi51-01.design.
local -a -o storage1.design.local -s /shared NFS_datastore1
```

`vicfg-nas`—The command to configure NFS storage (the equivalent in the ESXi shell is `esxcfg-nas`)

`--config`—The path to the configuration file containing your stored credentials

`--vihost`—The name of the specific ESXi host in vCenter to which this command should be directed

`-a`—Adds a new datastore

`-o storage1.design.local`—The fully qualified domain name (FQDN; you can also enter an IP) of the storage device to which you're connecting

`/shared`—The mount point to which you're connecting

`NFS_datastore1`—The name given to the datastore

PowerCLI

PowerShell is becoming the default scripting language for all Windows applications. Many articles explain why it's easier, better, and simpler to use PowerShell to manage your environment. The good thing is that VMware has made a strategic decision to follow suit and provide a PowerShell management environment for vCenter and ESX: PowerCLI.

PowerCLI has hundreds of cmdlets (pronounced "command-lets") that deal with practically all the elements of your infrastructure. You can configure and manage ESXi hosts, VMs, the OSes of the VMs—more or less anything. A vibrant community is constantly developing ways you can use PowerCLI to allow administrators to perform their jobs more easily.

For a list of all the cmdlets, see the online cmdlet reference:

```
http://pubs.vmware.com/vsphere-50/topic/com.vmware.powercli.cmdletref
.doc_50/Overview.html
```

NOTE The URL provided for a list of all PowerCLI cmdlets is valid for vSphere 5.0 Update 1, which was the latest version of the documentation as of the writing of this book.

What can you do with PowerCLI? In addition to making your toast in the morning, pretty much anything. Seriously, though, anything that is exposed in the vSphere SDK, you can access (and therefore manipulate).

Just as you create in vCLI a configuration file to store your credentials so you don't have to enter them each time you connect to your vCenter/ESXi host, you can do the same in PowerCLI. This isn't a unique feature of PowerCLI—it's more a PowerShell feature—but because you're using PowerShell, you can use it. Let's see how. First, let's look at the code for saving the credentials:

```
$vicredential = get-credential
$vicredential.Password | ConvertFrom-SecureString | Out-File c:\temp\viadmin.txt
$VIcred = New-Object System.Management.Automation.PsCredential "viadmin@design.
local", (Get-Content c:\temp\viadmin.txt | ConvertTo-SecureString)

Connect-VIServer -Server vcenter.design.local -Credential $VIcred
```

`Get-Credential` is a PowerShell cmdlet that lets you store credentials in an object for use without having to expose the contents of the password. Here, you store it in a variable named `$vicredential`. Next, you export the password from the variable into a text file. Don't worry, unlike in vCLI, the password isn't in plain text; rather, it looks something like this:

```
01000000d08c9ddf0115d1118c7a00c04fc297eb01000000c39f99b56e206a40a56c6e8e4ebc6e
c0000000000200000000000003660000c000000010000000395d0ba992b59f39e42e30a30e9
c972b0000000004800000a0000000100000008deee87fd1ebd9d74990d6ed44d984d018000000494b8
c989a3f55018cd9b7a743450f1d09a214843fda25b4140000005226217
e4587317d235557ad8e5177541859094b
```

That is pretty hard to decipher. You export the password because you'll use this credential more than once—instead of having to insert the password each time, you can create a variable to store it. This is done in line 3: a variable named `$VIcred` is created with an already known username, but the password is imported from the file you saved. This is what the variable holds:

```
UserName                                          Password
--------                                          --------
viadmin@design.local                    System.Security.SecureString
```

Last but not least, you connect to your vCenter Server with the credentials supplied in the variable.

After you've connected, let's use the same example as before with vCLI:

```
New-Datastore -Nfs -VMHost (get-vmhost) -Name NFS_datastore1 -Path "/shared"
-NfsHost $storage1.design.local
```

`New-Datastore`--The command to configure storage (the equivalent in the ESXi shell is `esxcfg-nas`; the equivalent in vMA or vCLI is `vicfg-nas`)

`-NFS`--Parameter for the kind of datastore (VMFS/NFS)

`-VMHost`--The host on which this datastore will be created (in this case, all hosts registered under vCenter)

`-Name NFS_datastore1`--The name you give to the datastore

`-Path "/shared"`--The mount point to which you're connecting

`-NfsHost storage1.design.local`--The FQDN (you can also enter an IP) of the storage device to which you're connecting

A Good PowerCLI Reference

For more detailed information on using PowerCLI, we recommend *VMware vSphere PowerCLI Reference: Automating vSphere Administration*, also published by Sybex.

vMA

vMA is a Linux-based virtual appliance provided by VMware that includes prepackaged software; specifically, it includes a preinstalled version of vCLI, which in turn includes the vSphere SDK for Perl; an authentication component (`vi-fastpass`) that allows a centralized direct connection to established target servers; and a logging component (`vi-logger`) that lets you collect logs from ESX/ESXi and vCenter Server systems and store the logs on vMA for analysis. Administrators and developers can use vMA to run scripts and agents to manage both ESX/ESXi and vCenter Server systems.

With the full transition to ESXi only in vSphere 5.0, vMA is much more relevant than it might have been in past versions of vSphere. Although ESXi includes a console shell, users are strongly encouraged to use vMA instead of the ESXi shell. In some cases, customers may prefer to use vCLI installed on their own Linux distribution; in other cases, deploying vMA might be easier. Both options allow administrators and other users access to run management and configuration commands without needing the ESXi shell.

Now, let's shift our focus to how you assemble these components to satisfy the requirements of the design. To help you better understand what's involved in creating the management layer design, we'll first look at key design decisions involved in creating a management layer design.

Examining Key Management Layer Design Decisions

In this section, we'll examine some—but not all—of the key decisions involved in crafting the management layer design for your VMware vSphere implementation. Specifically, we'll examine four key decisions:

- Virtual or physical vCenter Server?
- vCenter Server on Windows, or vCenter Server appliance?
- Local or remote database server?
- Which OS for vCenter Server?

Let's kick things off by tackling the decision whether to run vCenter Server on a physical system or in a VM.

Virtual or Physical vCenter Server?

There is a long-standing discussion in the VMware world about whether you should install vCenter Server in a VM or on a physical server. In the following section, we'll cover the support for both sides of the argument.

Physical Server

This book is about designing a virtual infrastructure environment, so why are we talking about physical hardware? Well, a certain amount of hardware has to be involved: your ESXi hosts, at a minimum. But why would you consider installing vCenter on a physical server? Here are some possible reasons.

The Chicken and the Egg We're sure you're familiar with the chicken-and-egg dilemma. Some people use this analogy for vCenter as a VM.

Let's say you have a large environment—100 ESXi hosts. For some reason, you have a serious outage. For example, you lose the LUN on which the vCenter VM is stored. For all sorts of reasons, you won't have the VM backup for another 4–6 hours. You may say, "OK, no problem"—until something goes seriously wrong with your environment. A VM starts to not behave due to high CPU RAM usage, so you try to find it in the mass of VMs among your 100 ESXi hosts and 2,000 VMs. Are you thinking "nightmare"? You locate the VM. Lucky you—it was on the third ESXi host you checked. And you want to vMotion the VM, but hey, you have no vCenter to perform the migration.

This is one example of what can happen if your vCenter is a VM. Another could be in a VMware View or vCloud Director environment—you can't deploy any new VMs because your vCenter is down. You can't perform certain actions to restore your environment because you have no vCenter, and you have no vCenter because your environment isn't functioning correctly. As you can see, serious issues may arise if your virtual vCenter Server isn't available.

Separation of Duties Certain organizations insist that because of issues like those just discussed, thou shall not have the management application of your environment running as part the environment itself. This doesn't mean you can't run vCenter as a VM—it can be run on a stand-alone host—but you'll lose a certain number of features, as you'll see shortly.

Amount of Resources vCenter Server can be relatively resource-intensive. If you run vCenter Server 5.0 and a back-end database server in the same VM, you could easily need 4 vCPUs and 8 GB of RAM. In vSphere 5.1, the same applies if you run vCenter Single Sign On, vCenter Inventory Service, vCenter Server, vCenter Web Client, and the back-end database server in the same VM. In fact, the requirements might be even higher. Do you want to use that amount of resources in your infrastructure on a single VM?

Virtual

We've discussed why it could be necessary, or better, for your environment to have vCenter on physical hardware. Now, let's look at the other side of the coin: why you could choose to go for vCenter as a VM. Note that VMware now lists the use of vCenter as a VM as a best

practice; however, as we stated in Chapter 1, it's imperative you understand *why* something is recommended as a best practice:

The Chicken and the Egg With proper planning and some preparation, you can mitigate all the problems that may arise from the earlier scenario. It's all about how to find the issues that may occur and provide a solution to address those issues if or when they happen. Let's take the issue of the LUN going down.

vCenter is an application. The guts of the environment is the database on which the vCenter relies. So in the earlier case, there is no way you should be down for 6 hours. If you separate the vCenter database from the vCenter Server, they're on separate servers. If they're VMs, they should be kept on separate ESXi hosts.

Table 3.4 shows how you can address some of these issues.

TABLE 3.4: Risk mitigation for a virtual vCenter Server

POTENTIAL RISK	MITIGATION ACTION
vCenter is lost when the database fails.	Separate the SQL instance on a different server to enable the use of high-availability features such as clustering.
Both vCenter and SQL are VMs on the virtual infrastructure.	Place them on separate hosts with anti-affinity rules.
Both vCenter and SQL are on the same storage.	Place the VMs on different LUNS or different storage devices.
SQL data corruption occurs on the LUN.	Plan for database snapshots at regular intervals.
vCenter/SQL suffers a performance hit due to other VMs.	Ensure that your vCenter/SQL has guaranteed resources with reservations.
vCenter completely crashes.	Install a new VM to replace the failed VM and attach to the database, and you're back in business.

Server Consolidation The whole idea of using a virtual infrastructure is to consolidate your physical servers into VMs running on ESXi hosts. And here you're doing exactly the opposite. With the proper planning, there is no reason not to virtualize any workload. It's also officially a VMware best practice.

Snapshots Suppose you're about to patch your vCenter Server, add a plug-in, or make a configuration change to a `.cfg` file. One of the greatest advantages you have with vCenter as a VM is the built-in ability to snapshot the VM before making any changes. If something goes drastically wrong, you can always revert to the snapshot you took before the change.

Portability A VM can be replicated to your disaster recovery (DR) site if need be. You can duplicate the vCenter Server easily if you want to create a test environment. You can even keep a cold backup of the vCenter Server in the event of a vCenter crash—you can bring the machine back up running on anything from VMware Player to Server to Workstation to a

stand-alone ESXi server. (Of course, this could result in some level of data loss, depending on how old the cold backup is.)

Redundancy As soon as you install vCenter on a vSphere HA-enabled cluster, you're automatically making the vCenter resilient to hardware failure. In order to get the same level of redundancy with physical hardware, you'd need a Microsoft cluster—and that would only be for the SQL database. vCenter isn't a cluster-aware application. Several third-party tools can provide this level of redundancy, but they aren't cheap.

And today, with vSphere Fault Tolerance (FT), you can provide a higher level of redundancy for your vCenter. At the present time, vSphere FT doesn't support more than one vCPU; therefore you can't protect the vCenter this way. Support for multiple vCPUs will be available in the future.

Support for Fault Tolerance with Multiple vCPUs

At both VMworld 2011 and VMworld 2012, VMware demonstrated and discussed the development of a version of vSphere FT that supports multiple vCPUs. However, VMware has not (as of the time of this writing) provided a date for when to expect this feature in a released version of vSphere.

Eating Your Own Dog Food How can you as the virtualization administrator say to all your clients and customers that you have faith in the platform, its capabilities, and its features, when you aren't willing to put one of your critical servers on the infrastructure? We all know that, with the correct planning, the product will perform just as well as a physical server, and you'll get the huge additional benefit of running the server as a VM. If you believe in the product, then you should use it.

vCenter Server on Windows or vCenter Server Appliance?

This question is a relatively new addition to the list of commonly asked management design questions. With vSphere 5.0, VMware introduced the vCenter Server virtual appliance (vCSA), a Linux-based virtual appliance that comes with vCenter Server preinstalled. Prior to the introduction of the vCSA, you had only a single choice: a Windows-based installation of vCenter Server. Now you have to decide: Windows-based vCenter Server or vCSA? Let's look at a few reasons in favor of each option.

Windows-Based vCenter Server

Here are some reasons you might deploy the Windows-based vCenter Server into your design:

The Devil You Know Many customers choose to deploy vCenter Server on Windows simply because it's "the devil you know." vCenter Server on Windows is the option that has been around for the longest, and it's therefore the option more people are familiar with and comfortable deploying. Although this reasoning might seem a bit arbitrary, we'll remind you that it's important to keep operational considerations such as this in mind when crafting your design. Deploying an option that your staff already knows and understands might be the best option for your environment.

Support for Linked Mode If you need support for Linked Mode instances of vCenter Server, then the Windows-based installation of vCenter Server is the only way to go. No, really—the virtual appliance version doesn't support Linked Mode.

Strong Operational Support for Windows Servers If your environment has strong operational support for Windows servers—meaning you have a solid patching solution in place, good monitoring and management options running, and a proactive team of Windows admins—then deploying the Windows Server-based version of vCenter Server makes more sense than introducing a Linux-based virtual appliance that may or may not integrate as well with your existing operational support systems.

vCenter Server Virtual Appliance

Now that we've discussed some reasons for deploying the Windows Server–based version of vCenter Server, what are some reasons for deploying the vCSA? Here are a few that you might consider:

It's Not Windows The very fact that the vCSA *isn't* Windows might be enough for some environments. Perhaps you work in a Linux/Unix-heavy environment and therefore don't have strong operational support systems in place for Windows. Perhaps your administrators are more comfortable managing and maintaining Linux-based systems than Windows-based systems. Perhaps your organization just has an ideological opposition to Windows. The vCSA also doesn't require monthly patches (although this doesn't imply that it will never need to be patched), and the vCSA doesn't require a Windows license. For any of these reasons, a Linux-based virtual appliance might make more sense.

Simplified Deployment In smaller environments and organizations, the ease with which an instance of the vCSA can be deployed is a very attractive benefit. Simply deploy the Open Virtualization Format (OVF) package, configure the vApp as it's being deployed, and then hit the web-based management tool afterward.

Local or Remote Database Server?

Here's a topic for endless discussion: "Where do I install the database: on the vCenter Server or on a remote server?" Let's go into the rationale behind both options.

Local

Having your database *local* means you've installed it on the same OS as your vCenter Server. Here are some of the benefits of having all the components on one server:

Bundled Database For the Windows-based version of vCenter Server, Microsoft SQL Express is bundled with the vCenter Server installation. The software doesn't cost anything, but it isn't suitable for large-scale enterprise environments. On the vCSA side, you can use a bundled version of DB2 (in vSphere 5.0) or PostgreSQL (in vSphere 5.1). This can be suitable for small environments or test environments.

Full Database Installation You can install a full database server on your vCenter Server—be it Oracle or Microsoft SQL—assuming that you're using the Windows-based vCenter Server installation. This provides enterprise-level functions that are suitable for a production

environment. Note that you can't install a full-blown instance of a database engine on the vCSA.

Faster Access to the Database Having the data local on the same box is in most cases quicker than accessing the data over the network. You don't have to go over the wire, and you're not dependent on factors such as network congestion.

An All-in-One Box You know where your weaknesses lie. You won't be affected by other applications abusing resources on the network on the shared database server (be it Oracle or SQL or DB2).

Backup and Restore A sound and solid backup infrastructure doesn't come cheap. Having all the components on one server saves you from having to back up multiple servers and track which part of the infrastructure is located in which part of your enterprise.

Remote

As opposed to local, here we're talking about installing vCenter software on one server and connecting to a database that doesn't reside on the same machine. The reasons to choose this option are as follows:

Central Database Server You already have a central location for the databases in your organization. If this is the case, you shouldn't start spreading around additional database servers for each and every application. Your DBA won't like you.

Deploying the vCSA You've chosen to deploy the vCSA, but you need a bit more horsepower than the bundled database can provide. In this case, you'll need to have a separate computer running the database server software.

Separation of Duties between Databases and Applications This is perceived as best practice. Servers have dedicated roles, and these roles should be on separate boxes. vCenter software and VUM software should be installed on one server, and the database on which these applications rely should be on a different machine. This ensures that loss of data on one of the servers doesn't cause lengthy downtime. This separation of duties can also, under the right circumstances, provide greater scalability for the management layer.

Corporate Policies (Separate DBA Team) Your organization has an established database administrator. In most cases, they will probably be more knowledgeable about performance, optimization, and troubleshooting of any issues that arise in the future. So, a database installed on the central server is usually maintained in a better fashion than you could administer it on your own. The Virtual(ization) Infrastructure Admin (VIAdmin) usually gets busy very quickly and doesn't have the time or the resources to acquire the knowledge needed to manage and maintain your database server in addition to all the new duties you inherited with the system.

Fewer Resources Needed for vCenter Server If you're going to join the roles of application and database on the same box, you need larger resources for the vCenter Server. (See the next section, "Local or Remote from a Resource Perspective.")

Providing Clustered Services (Redundancy) for the Database SQL and Oracle can be clustered. Doing so provides resilience in case of a database server failure. You can't do this if the database resides on the same server as vCenter.

Local or Remote from a Resource Perspective

The major database vendors—Microsoft, Oracle, and IBM—provide recommended resource configurations for optimal performance of their database servers. For example, Microsoft recommends a minimum of 1 GB of RAM and recommends 4 GB for optimal performance for SQL Server. In addition, take into account the resources needed for the vCenter service, web services, and plug-ins—you're looking at another 2–4 GB RAM for the vCenter host. It's pretty likely that in an enterprise environment, you have a properly sized database server that can accommodate the vCenter database without any major issues.

The following article provides more information about the recommended resources for SQL Server 2012:

`http://msdn.microsoft.com/en-us/library/ms143506.aspx`

This article explains that you need at least 1 GB RAM with at least one 1.4 GHz CPU. But more realistically, the recommended resources are as follows:

- 1 CPU, 2.0 GHz and up
- 4 GB RAM

Taking this into account, if you install both the vCenter and the database server on the same box, you need approximately 8 GB RAM and at least 4 CPUs (either a quad core physical machine or a VM with four vCPUs configured). Later in this chapter, we'll discuss some sizing recommendations, and you'll see more information about the resources that would be required for vCenter based on the anticipated size of the environment it will manage.

Local or Remote with Redundancy in Mind

We'll get into this topic a bit later in the chapter. But in short, there's no point in providing a clustered solution for only one database. If you're setting up a cluster to protect the database, you may as well protect more databases at the same time. Thus it's useless to install the database locally.

Which Operating System for vCenter Server?

You don't see this question as much, but it's a valid design consideration. Naturally, vCenter is a server and should be installed on a server OS. But which server OS?

As of vSphere 4.1, the requirement for the Windows-based version of vCenter Server is a 64-bit build of Windows Server. This aligns well with Microsoft's stated OS strategy, which declares that Windows 2008 R2 and all subsequent server OSes will be 64-bit only. Given that you have to go with a 64-bit version of Windows Server, then the consideration of which edition of Windows Server is essentially irrelevant—the 64-bit Standard Edition of Windows Server 2008 and Windows Server 2008 R2 are both capable of addressing up to 32 GB of RAM. (32-bit versions of Windows Server 2008 Standard were limited to 4 GB of RAM, which didn't make it an ideal platform for vCenter Server.)

More information on the memory limits for various versions of Windows can be found here:

`http://msdn.microsoft.com/en-us/library/windows/desktop/aa366778(v=vs.85).aspx`

With these considerations in mind, it seems reasonable that selecting Windows Server 2008 R2 Standard Edition should be fine for deploying the Windows-based version of vCenter Server.

If you've chosen to go down the vCSA route, then this question doesn't apply to you because the vCSA comes with Linux preinstalled. At the time of this writing, there was no supported way to roll your own Linux-based vCenter Server installation.

There are many, many more potential management layer design questions, but in the interest of time (and pages!), let's move on to a discussion of actually creating the management layer design.

Creating the Management Layer Design

In this section, we'll focus on creating the design for the management layer. To structure the discussion, we'll use the basic principles of design as described in Chapter 1—availability, manageability, performance, recoverability, and security—as a framework around which we'll organize the information. We'll start with a discussion of availability.

Availability

The first basic principle of design to consider when creating the management layer design is availability. Availability, as we discussed in Chapter 1, encompasses metrics like uptime and, in some cases, performance (if the application is so slow as to be unusable, is it still available?).

There are a couple of ways to provide a level of availability for your vCenter Server:

- vSphere HA
- vCenter Server Heartbeat

Before we talk about how to provide availability, you need to understand the implications of not having your vCenter available. Table 3.5 lists the functions that will or won't be affected.

TABLE 3.5: Functions not available when vCenter fails

ENTITY	FUNCTION		REMARK
HA	Restart VM	Yes	Full functionality.
	Admission control	No	vCenter is required as the source of the load information.
	Add a new host to the cluster	No	vCenter is required to resolve IP addresses of cluster members.
	Host rejoins the cluster	Yes	Resolved host information is stored in `/etc/FT_HOST`.
DRS	Manual	No	Impossible without the vCenter.
	Automatic	No	Impossible without the vCenter.
	Affinity rules	No	Impossible without the vCenter.
Resource pools	Create	No	Meaningless without the vCenter.

Entity	Function			Remark
	Add a VM	No		Meaningless without the vCenter.
	Remove a VM	No		Meaningless without the vCenter.
VMotion		No		No vMotion.
ESXi host	Shutdown	Degraded		Through the direct connection to the ESXi host server only.
	Startup	Yes	E	
	Maintenance	Degraded		Meaningless without the vCenter.
	Deregister	No		Meaningless without the vCenter.
	Register	No		Meaningless without the vCenter.
Virtual machine	Power on	Degraded	E	Through the direct connection to the ESXi host server only.
	Power off	Degraded		Through the direct connection to the ESXi host server only.
	Register	No		Meaningless without the vCenter.
	Deregister	No		Meaningless without the vCenter.
	Hot migration	No		No VMotion.
	Cold migration	Degraded		Within one ESXi host only.
Template	Convert from VM	Degraded		Direct connection to host only/ meaningless without vCenter.
	Convert to VM	Degraded		Direct connection to host only/ meaningless without vCenter.
	Deploy VM	No		No VM deployment.
Guest	All functions	Yes		No impact.
Alarms	All functions	No		Unless you have direct agents on the ESXi hosts.
Statistics	All functions	No		Not collected during the outage.
Yes	*Same functionality as without VCenter*			
Degraded	*Functionality degradation without vCenter*			
No	*Functionality lost without vCenter*			
E	*Functionality will expire after the 14-day grace period.*			

VMware

As you can see from the table, most of the management tasks related to vCenter won't work or will only partially function. Although a number of management functions are impacted, the good news is that not all VMs are affected.

Availability should be divided into two parts: the vCenter application and the database. Let's examine how you can provide availability for vCenter—first, the easiest way.

Providing Availability for vCenter

The first component you'll be protecting is vCenter. You should consider several options; it all depends on what level of redundancy is necessary for your environment and how critical it is to have your vCenter available the entire time.

vSphere HA

By running vCenter as a VM on a vSphere HA-enabled cluster, you automatically improve availability by protecting against hardware failure. This is true for both the Windows version of vCenter Server as well as the vCSA. If your ESXi host fails, then vCenter will automatically restart on another host within a short period of time and reconnect to the database. This is a very acceptable solution in the case of host hardware failure; you shouldn't have any major downtime.

As we mentioned earlier in this chapter, current limitations on vSphere FT (limited to a single vCPU only) prevent its use with vCenter Server, given that vCenter would typically require more than a single vCPU.

If you're running vCenter Server on a physical system, then vSphere HA isn't an option, and you'll have to look at vCenter Server Heartbeat instead.

vCenter Server Heartbeat

vCenter Server Heartbeat is a relatively new product from VMware that's specifically targeted at organizations that can't afford even the slightest downtime or failure of their vCenter Server. The product provides automatic failover and failback of VMware vCenter Server using failure detection and the failover process, making manual recovery and failover process definitions unnecessary. It enables administrators to schedule maintenance windows and maintain availability by initiating a manual switchover to the standby server. You can find vCenter Server Heartbeat at

`www.vmware.com/products/vcenter-server-heartbeat/`

The product also provides protection for the vCenter database, *even if the database is installed on a separate server*. The technology is based on a product from Neverfail (`www.neverfailgroup.com`). Figure 3.3 shows the product design.

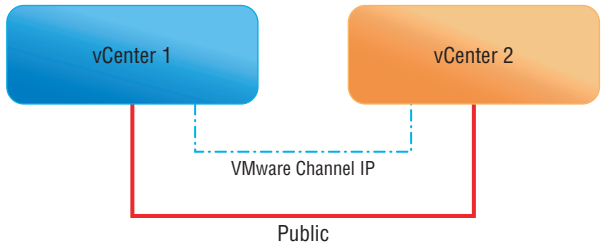


Figure 3.3
This diagram provides a conceptual overview of vCenter Server Heartbeat.

The great thing about vCenter Server Heartbeat is that it can be placed across the WAN. If you have two datacenters (US and Europe), you can place each node in a different location to provide potentially greater levels of availability.

You'll need to address certain prerequisites and limitations:

- vCenter Server Heartbeat can't be installed on a domain controller, global catalog, or DNS. Of course, vCenter Server can't be on a domain controller, either.
- It only protects Microsoft SQL Server applications.
- No other critical business applications should be installed on the SQL Server besides the vCenter database.
- Both primary and secondary servers must have identical system date, time, and *time zone* settings.
- Latency on the link between the two locations must not fall below the standard defined for a T1 connection.

You can install the configuration in three ways:

- Virtual to virtual (V2V)
- Physical to virtual (P2V). If you're setting up a P2V configuration, then
  - The systems should have similar CPUs.
  - The systems should have identical amounts of memory.
  - The secondary VM must have sufficient priority in resource-management settings to assure the performance of the VM.
  - Each virtual NIC must use a separate virtual switch.
- Physical to physical (P2P). In a P2P setting, then
  - The primary server must meet certain hardware and software requirements.
  - The secondary should be equivalent to the primary server to ensure adequate performance.
  - Advanced Configuration and Power Interface (ACPI) compliance must match the primary server.

We won't go into the details of how to install, because this isn't a step-by-step book (if you haven't noticed by now).

At this point we do need to point out a few things. First, the product is relatively new in comparison to the rest of the vSphere product suite. We haven't met anyone who has implemented it in their environment, because it's targeted at the highest percentile of customers who can't—at any cost—have their vCenter down. It also isn't a cheap solution: it starts at approximately $12,000 per license. Finally, vCenter Server Heartbeat is only supported for the Windows-based version of vCenter Server, not the vCSA.

Also, judging from the community activity (or lack thereof), vCenter Server Heartbeat hasn't been implemented widely. Or perhaps it works like a charm, so no one has any issues with it. See

```
http://communities.vmware.com/community/vmtn/mgmt/heartbeat
```

Now, on to the second component: the database.

Providing Availability for the SQL/Oracle Database

There are several ways to protect the database for your vCenter Server. Database protection is important because the database is the heart and soul of your virtual environment. All the settings (vSphere DRS, vSphere HA, permissions—basically, everything) are stored in the database.

vSphere HA

The vCenter Server can be protected with vSphere HA, and the same goes for your database server. But the chance of data corruption is much higher with a database server. If the server crashes during a write operation, the data can become corrupt, and your database may be rendered unusable. In this case, a single database on a vSphere HA-enabled cluster might cause you problems.

Microsoft Cluster/Oracle Cluster

Both Microsoft and Oracle provide their own solution for active-active and active-passive clusters for databases. The great thing is that you can combine these two options: vSphere HA and Microsoft/Oracle Cluster.

You can create a highly available database on vSphere. VMware has best practices for both platforms:

- Oracle Databases on VMware Best Practices Guide: `www.vmware.com/files/pdf/partners/oracle/Oracle_Databases_on_VMware_-_Best_Practices_Guide.pdf`
- Setup for Failover Clustering and Microsoft Cluster Service: `pubs.vmware.com/vsphere-50/topic/com.vmware.ICbase/PDF/vsphere-esxi-vcenter-server-50-mscs-guide.pdf`

Here are a few considerations and best practices you should take into account when virtualizing these applications, distilled from the documents just referenced:

Upgrade to the Latest Version of vSphere By upgrading to new versions of vSphere, you can—in some instances—gain a 10–20% performance boost on the same hardware.

Create a Computing Environment Optimized for vSphere You should set the BIOS settings for ESXi hosts accordingly. These recommendations include the following:

- Enable virtualization technology to support running 64-bit guest OSes.
- Enable Turbo Mode to balance the workload over unused cores.
- Enable node interleaving if your system supports non-uniform memory architecture (NUMA).
- Enable VT-x, AMD-V, EPT, and RVI to use hardware-based virtualization support.
- Disable C1E Halt State to prefer performance over power saving.
- Disable power-saving to prevent power-saving features from slowing down processor speeds when idle.

- Enable HyperThreading (HT) with most new processors that support this feature.
- Enable Wake On LAN to support the required features for the Distributed Power Management feature.
- Set Execute Disable to Yes for the vMotion and Distributed Resource Scheduler features.

Optimize Your OS Remove nonvital software, and activate only required services.

Use as Few vCPUs as Possible One of the biggest mistakes administrators make is over-allocating resources to a VM. When you allocate four vCPUs to a VM that barely uses one vCPU, not only will the performance of the VM be degraded, but you can also seriously impact the rest of the VMs running in the same host.

Enable HT for Intel Core i7 Processors This is a new recommendation for the Xeon 5500 family of processors. Until now, VMware had no recommendation about enabling HT.

Allow vSphere to Choose the Best Virtual Machine Monitor Based on the CPU and Guest OS Combination Workloads like those from a database that has a large amount of page-table activity are likely to benefit from hardware assistance. But it's still best to leave the BIOS settings as recommended above.

Use the VMXNET Family of Paravirtualized Network Adapters The paravirtualized network adapters in the VMXNET family implement an idealized network interface that passes network traffic between the VM and the physical network interface cards with minimal overhead.

Enable Jumbo Frames for IP-Based Storage iSCSI and NFS This will reduce load on the network switches and give a certain percentage of performance boost. Note that this feature must be enabled end-to-end (VMkernel, vSwitch, physical switch, and storage) for it to work properly.

Create Dedicated Datastores to Service Database Workloads It's a generally accepted best practice to create a dedicated datastore if the application has a demanding I/O profile, and databases fall into this category. The creation of dedicated datastores lets you define individual service-level guarantees for different applications and is analogous to provisioning dedicated LUNs in the physical world.

It's important to understand that a datastore is an abstraction of the storage tier and, therefore, is a logical representation of the storage tier, not a physical representation of the storage tier. So, if you create a dedicated datastore to isolate a particular I/O workload (whether log or database files) without isolating the physical storage layer as well, you won't get the desired effect on performance.

Make Sure VMFS Is Properly Aligned We'll get into disk alignment in more detail in Chapter 6, "Storage," and Chapter 7, "Virtual Machines." But note that the performance hit of not aligning your VMFS/NFS datastores can be significant.

vCenter Server Heartbeat

As noted earlier, this product provides redundancy for an SQL database only. That database must be the only one on the server in order for this to work.

Availability is obviously a key principle to keep in mind when creating the management layer design. The second principle we're going to discuss is manageability, and that's the topic of the next section.

Manageability

In Chapter 1, we stated that the principle of manageability includes such concepts as compatibility, usability, interoperability, and scalability. Some of these concepts are outside of your control. For example, you can't control the usability of most portions of the management layer, because VMware drives that. If your users find the vSphere Web Client to be unintuitive or the syntax of the vCLI to be unintelligible, there's very little you can do to alter or address those complaints. Other aspects of manageability, however, are under your control:

Compatibility and Interoperability If the environment requires that vCenter Server interoperates with other management components, then you might need to add third-party components to bridge any compatibility or interoperability concerns. Integration with a product like Microsoft Systems Center Operations Manager (SCOM) is one such example; third-party products like Veeam Management Pack for VMware provide the required interoperability and compatibility between vCenter Server and SCOM. Similar solutions exist to provide greater compatibility and interoperability between vCenter Server and storage, networking, and server hardware. Where dictated by the design factors, these additional components need to be factored into the management layer design.

Scalability Another aspect of manageability is scalability. If the management layer can't scale to handle the anticipated growth of the environment, the environment will become unmanageable. This aspect of manageability is closely related to performance, and so we'll address it later in this section under "Performance."

Scalability is an aspect that can be addressed in a couple of different ways. One of these ways, Linked Mode, is a solution that will have a number of different design impacts and therefore deserves a bit more consideration.

Linked Mode

In this section, we'll go into detail about vCenter Linked Mode as one potential method of providing scalability to the management layer design. VMware added this feature in the release of vSphere 4. It solves the issue of very large environments that need multiple vCenters to manage the infrastructure due to the sheer size and number of hosts and VMs.

The limit on the number of ESXi hosts that one vCenter can manage is 1,000; the limit for powered-on VMs is 10,000 per vCenter (15,000 total). For many organizations, this is more than sufficient—they can only dream of reaching that number of VMs. But for others, this limit is too small.

In comes Linked Mode. By joining your vCenter instances together, you can expand your infrastructure to 10 vCenter Servers, 3,000 hosts, and 30,000 powered-on VMs (50,000 total).

NOTE You need to be aware of a licensing caveat here. Linked Mode can only be used with the Standard Edition of vCenter Server. If you've purchased an Essentials or Foundation Bundle (which doesn't have a Standard vCenter license), you can't join the vCenter instances together in Linked Mode.

Prerequisites

To use Linked Mode for additional scalability to the management layer design, keep in mind the following requirements. These requirements apply to each vCenter Server system that will be a member of a Linked Mode group:

- DNS infrastructure must be operational for Linked Mode replication to work. Each member of the Linked Mode group must be able to resolve the names of the other members in the group.
- The vCenter Server instances in a Linked Mode group can be in different domains as long as the domains have a two-way trust relationship.
- When adding a vCenter Server instance to a Linked Mode group, the installer must be run by a domain user who has administrator credentials on both the machine where vCenter Server is installed and the target machine of the Linked Mode group.
- All vCenter Server instances must have network time synchronization. The vCenter Server installer validates that the machine clocks aren't more than 5 minutes apart. Within a single Active Directory domain, this is normally handled automatically. (Keep in mind that Linked Mode is only available for the Windows versions of vCenter Server, so naturally Active Directory is almost always involved.)

Design Considerations

You should take into account several considerations before you configure a Linked Mode group:

- Each vCenter Server user sees the vCenter Server instances on which they have valid permissions. If you wish to block certain parts of your environment from other users, you need to specifically deny permissions on that section.
- When you're first setting up a vCenter Server Linked Mode group, the first vCenter Server must be installed as a stand-alone instance because you don't yet have a remote vCenter Server machine to join. Any vCenter Server instances thereafter can join the first vCenter Server or other vCenter Server instances that have joined the Linked Mode group.
- If you're joining a vCenter Server to a stand-alone instance that isn't part of a domain, you must add the stand-alone instance to a domain and add a domain user as an administrator. Linked Mode isn't supported in workgroup environments—but then again, why would you use Linked Mode in a workgroup?
- The vCenter Server instances in a Linked Mode group don't need to have the same domain user login.
- The vCenter service can run under different accounts. By default, it runs as the machine's LocalSystem account.
- You can't join a Linked Mode group during the upgrade procedure when you're upgrading from VirtualCenter 2.5 to vCenter Server 4.1. Only after you've completed the upgrade can you join Linked Mode.
- Linked Mode groups can only contain vCenter instances running the same version of vCenter Server. If you have vCenter 5.0 and want to add vCenter 5.1, you'll need to upgrade to vCenter 5.1 on all systems before you can create a Linked Mode group.

With these considerations in mind, let's take a peek under the covers of vCenter Server's Linked Mode groups.

Under the Covers

How does Linked Mode work? VMware uses Active Directory Lightweight Directory Services (AD LDS, previously known as Active Directory Application Mode [ADAM]).

Instead of using your organization's Active Directory database to store the directory-enabled application data, you can use AD LDS to store the data. AD LDS can be used in conjunction with AD DS so you can have a central location for security accounts (AD DS) and another location to support the application configuration and directory data (AD LDS). By using AD LDS, you can reduce the overhead associated with Active Directory replication, you don't have to extend the Active Directory schema to support the application, and you can partition the directory structure so the AD LDS service is deployed only to the servers that need to support the directory-enabled application.

Linked Mode behaves like an Active Directory application. Let's connect to it and see what information is inside. Figure 3.4 shows an example of using `ldp.exe` (`http://support.microsoft.com/kb/224543`) to connect to a vCenter Server on port 389 (unless you've changed the default port).

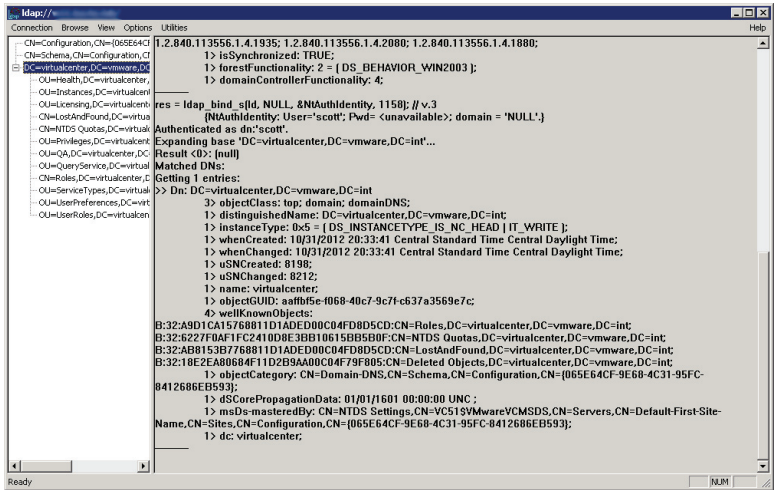


FIGURE 3.4
LDP shows the data found in a vCenter LDS instance.

It's easier to use Active Directory Services Interfaces (ADSI) Edit to see all the properties. Connect to your vCenter Server with the correct credentials, using the naming context `DC=virtualcenter,DC=vmware,DC=int` as shown in Figure 3.5 and Figure 3.6.

You can connect to three naming contexts:

- `DC=virtualcenter,DC=vmware,DC=int`
- `CN=Schema,CN=Configuration,CN={382444B2-7267-4593-9735-42AE0E2C4530}` (the GUID is unique to each vCenter installation)
- `CN=Configuration,CN={382444B2-7267-4593-9735-42AE0E2C4530}`

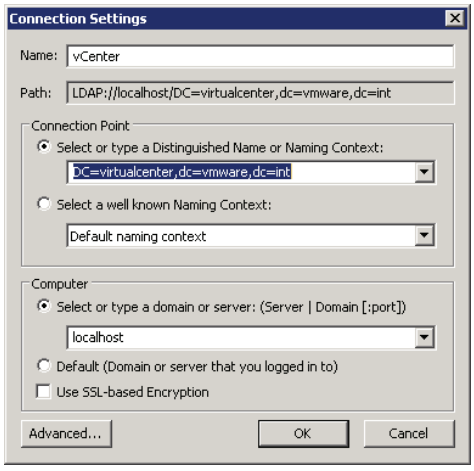


FIGURE 3.5
You can also use ADSI Edit to connect to a vCenter LDS instance.

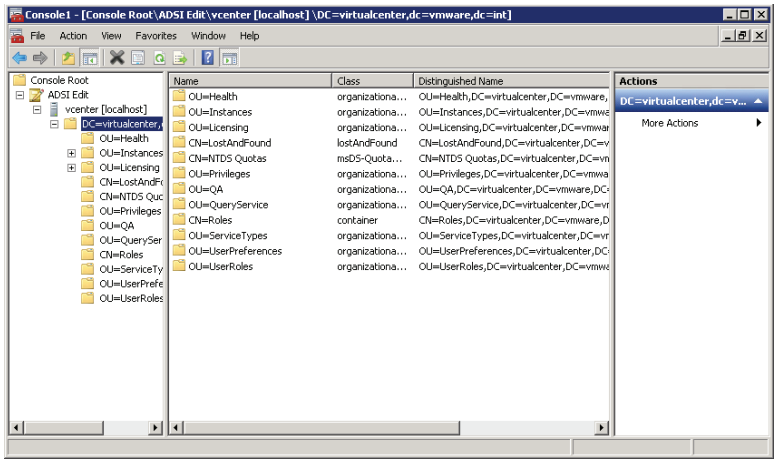


FIGURE 3.6
ADSI Edit shows the different vCenter LDS naming contexts.

It's a good idea to get acquainted with the structure of the schema and the configuration of the LDS instance. They aren't documented well, and the information may come in handy one day.

Considerations for vCenter Roles in Linked Mode

You need to know several things about roles when joining two or more servers in Linked Mode:

- The roles defined on each vCenter Server system are replicated to all the other vCenter Servers connected in the Linked Mode group.
- If the roles defined on each vCenter Server system are different, the role lists of the systems are combined into a single common list. For example, if vCenter Server 1 has a role named

ESX_Admins and vCenter Server 2 has a role named Admins_ESX, then both servers will have both ESX_Admins and Admins_ESX after they're joined in a Linked Mode group.

- If two vCenter Server systems have roles with the same name, the roles are combined into a single role if they contain the same privileges on each vCenter Server system. If the role exists on two vCenter Servers but they're assigned different privileges on the two servers, this conflict must be resolved by renaming at least one of the roles. You can choose to resolve the conflicting roles either automatically or manually.

  If you choose to reconcile the roles automatically, the role on the joining system is renamed to <*vcenter_name*> <*role_name*>, where <*vcenter_name*> is the name of the vCenter Server system that is joining the Linked Mode group and <*role_name*> is the name of the original role.

  If you choose to reconcile the roles manually, you have to connect to one of the vCenter Server systems with the vSphere Client and rename one instance of the role before proceeding to join the vCenter Server system to the Linked Mode group.

- If you remove a vCenter Server system from a Linked Mode group, the vCenter Server system retains all the roles it had as part of the group.

Linked Mode offers a great way to improve the manageability of a large vSphere environment by both enabling greater scalability as well as aggregating information from multiple vCenter Server instances. In fact, prior to vSphere 5.1, the only way to see information from multiple vCenter Server instances at the same time from the vSphere Client was using Linked Mode.

With the release of vSphere 5.1, though, architectural changes in vCenter Server—specifically, splitting the vCenter Inventory Service into a separate component that can service multiple vCenter Server instances—and the introduction of the new vSphere Web Client now allow environments running vSphere 5.1 to aggregate information from multiple vCenter Server instances in the next-generation Web Client without having to use Linked Mode groups.

Let's move on to the third principle of design, which is performance.

Performance

Invariably, most aspects of performance come down to sizing the components properly. So, as you consider the principle of performance when creating the management layer design, sizing the components properly will be a primary concern. With that in mind, let's discuss how you size the various management components.

Sizing vCenter Server

You need to consider the following elements while sizing your vCenter Server:

- OS
- Database placement
- Number of objects managed
- VUM

Let's look at each of these elements.

Operating System

As we described earlier in the section "Which Operating System for vCenter Server?" vCenter Server (if you're using the Windows version) must be installed on a 64-bit version of Windows. To avoid potential memory constraints, our recommendation is to use Windows Server 2008 R2.

To recap the reasons behind this recommendation, recall that one limitation of a Windows 32-bit OS is that it can only natively support 4 GB of RAM. Many vCenter Server instances were installed in the past with Enterprise Edition OSes to overcome that boundary. With a Standard license of Windows Server 2008 R2, you're limited to 32 GB of RAM, which should be more than sufficient for the vast majority of vCenter deployments. In addition, licensing for the higher versions of both the Windows OS and SQL Server is substantially more expensive than a standard license.

Database Placement

As we explained earlier in the section "Local or Remote Database Server?" the placement of the database for vCenter Server (and related components like vSphere Update Manager) has a significant impact on the resources that must be allocated to vCenter Server. In that section, we stated that the recommendations for Microsoft SQL Server, for example, are 4 GB of RAM and a 2.0 GHz or higher CPU. When you add these requirements to the requirements described next based on the number of managed objects, you can see that accounting for database placement is critical. If you're going to run the database local to vCenter Server, be sure to allocate appropriate resources. For maximum performance, we recommend splitting the database server onto a separate system.

Number of Objects Managed

VMware's documentation for installing and setting up vCenter provides the numbers in Table 3.6 as recommendations for optimal performance.

TABLE 3.6: Recommendations for optimal performance

	NUMBER OF CPUS	RAM (GB)	DISK (GB)
Up to 50 hosts and 250 powered-on VMs	2	4	3
Up to 200 hosts and 2,000 powered-on VMs	4	4	3
Up to 300 hosts and 3,000 powered-on VMs	4	8	3

```
http://pubs.vmware.com/vsp40/install/wwhelp/wwhimpl/common/html/wwhelp
.htm#href=c_vc_hw.html&single=true
```

You can clearly see that at most, you need four CPUs and 8 GB RAM. The sizing of your vCenter will depend on how big a deployment you're planning.

Update Manager

Will you be using your vCenter Server as a VUM server as well? Why does this make a difference?

In terms of CPU or RAM resources, you won't need additional resources if you don't have the SQL Server running on the vCenter Server. What you do need is disk space. VMware provides a tool that assists in sizing your installation: the VMware vCenter Update Manager Sizing Estimator, found at this URL:

```
www.vmware.com/support/vsphere4/doc/vsp_vum_41_sizing_estimator.xls
```

At the time this book was written, the VUM 4.1 document listed here was the latest version available, and it should be applicable to newer versions of VUM as well. Note that there are also VUM patch-management guidelines included in VMware's "Configuration Maximums" documentation for each vSphere release.

The vCenter Update Manager Sizing Estimator is an Excel spreadsheet that requests information about your environment:

- Version of hosts you'll be patching (3.0x/3.5x/4.x)
- Number of concurrent upgrades
- Number of hosts
- Number of VMs
- OS locale
- Service pack levels
- Frequency of scans for hosts, VMs, and VMware tools

The results you get from the spreadsheet will look like Table 3.7.

TABLE 3.7: Results from Update Manager Sizing Estimator

RESOURCE	INITIAL UTILIZATION MB	ESTIMATED MONTHLY UTILIZATION MB		
		Median	20%	–20%
Database space usage	150	133	160	107
Disk utilization—patch content	50	13,100	15,720	10,480

The biggest resource you need for VUM is disk space. Because of all the patches you'll download, the required disk space will increase depending on how many versions of the software you're downloading for (either version of ESX, and the different OS service packs).

It's recommended that you not install the patch repository in the default location provided during the installation (`C:\Documents and Settings\All Users\Application Data\VMware \VMware Update Manager\Data\`). Most administrators don't notice this question during the

installation; then, somewhere along the way, they begin to run out of space on the `C:` drive of their vCenter Server, and they wonder why. It's better to allocate a separate partition and folder for the downloaded updates, for these reasons:

Backup You don't always want to back up patches that are downloaded. The content hardly changes, and it can easily be downloaded again if needed.

Not Enough Space on the System Drive If you aren't careful, your system drive will fill up.

In addition to sizing vCenter Server itself, it's also critical to properly size the database, as we describe in the next section.

Database Sizing for vCenter and Update Manager

When you follow the previous recommendations and decide to use a database server that isn't on the same system as vCenter Server, you next have to plan the size of the database. Before we get into sizing, let's review the purpose of the database in a vCenter installation.

The database is the central repository of the logic and structure of your virtual infrastructure: resource pools, permissions on each item in vCenter, alarms, thresholds, the cluster structure, distributed resource scheduling (DRS), and, of course, the biggest consumer—statistics. All the statistics for every object and counter in your environment are stored in the database, including CPU, RAM, disk, network, and uptime. And each category has multiple counters associated with it.

VMware also provides the vCenter Server 4.x Database Sizing Calculator for Microsoft SQL Server (this was the latest version available at the time of writing):

`www.vmware.com/support/vsphere4/doc/vsp_4x_db_calculator.xls`

It's very similar to the calculator mentioned earlier for VUM. But in this case, a larger number of parameters are required in addition to those we've already covered:

- Number of NICs per host
- Number of NICs per VM
- Number of datastores per host
- Number of VMDKs per VM
- Number of physical CPUs
- Number of virtual CPUs

And most important, how long will you keep each level of statistics? You configure this setting in vCenter as shown in Figure 3.7.

The higher the level of statistics collected for each interval, the larger your database will become. VMware's report "VMware vCenter 4.0 Database Performance for Microsoft SQL Server 2008" includes the recommendations in Table 3.8.

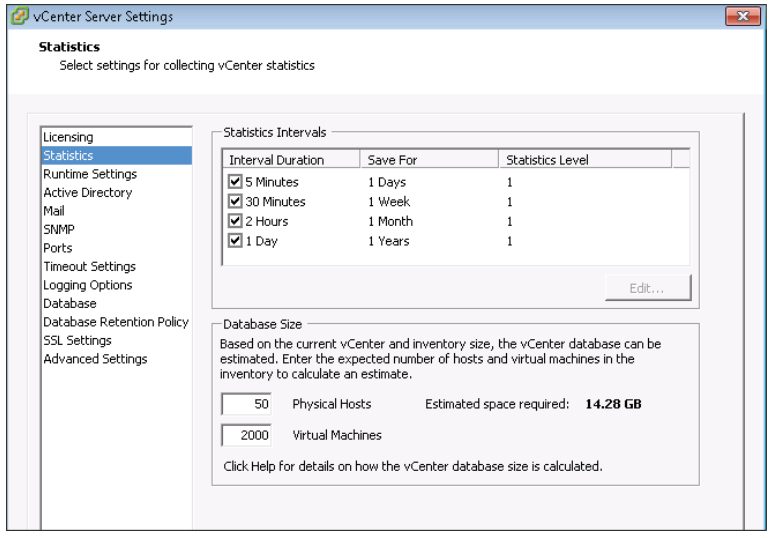


FIGURE 3.7
The vCenter statistics configuration has a profound impact on database size.

TABLE 3.8: VMware recommendations for database performance with Microsoft SQL Server 2008

STATISTICS INTERVAL	STATISTICS LEVEL
Past day	Level 2
Past week	Level 2
Past month	Level 1
Past year	Level 1

The document also provides a few additional recommendations, most of which we've already discussed or are general rules of thumb. You need to allocate enough RAM for your database server. One guideline for all relational databases is to allocate the server as much memory as necessary to allow for the caching of all the needed data in memory. You should refer to the documentation for the amount of memory to give to the OS and any other applications that run on the server you use for your database. You should go with a 64-bit server for the same reason we mentioned earlier regarding the 4 GB RAM memory limitation on Windows 32-bit OSes.

Another area to consider when performing sizing calculations for vCenter Server is the database recovery model. For example, with Microsoft SQL Server the default recovery model is Full. This means the database logs will grow endlessly if no backup is performed. If your organization doesn't have a DBA on staff to manage backups of the SQL Server database, then we recommend changing this to the Simple recovery model. If you still prefer the Full recovery model, be sure to account for the additional disk space that is required to store the logs between database backups.

Plug-ins

Plug-ins are great. They allow you to perform all the tasks you need in one management console without having to use a multitude of tools to manage your environment. As an example, most of the major storage vendors (NetApp, EMC, Dell, HP, and IBM) already have plug-ins that let you configure the virtualization portions of the storage array. Using these plug-ins, you can create new LUNs/datastores, view statistics of the VMs in correlation to the storage back-end, optimize the ESXi hosts according to vendor best practices, and more.

Unfortunately, with the advantages come some disadvantages that you didn't plan for—especially the additional resources required to run the plug-ins. You need to take into account the following resources:

- Disk space
- Memory

You must also consider whether these plug-ins are client-side plug-ins (designed to work with the Windows-based vSphere Client) or server-side plug-ins (designed to work with the next-generation vSphere Web Client). As you can probably surmise, client-side plug-ins have a resource impact on the clients where the plug-ins are installed, and server-side plug-ins have a resource impact on the vCenter Server or the vCenter Web Client server. Because the resource requirements vary greatly among plug-ins, we can't provide any specific recommendations here other than to consult with the plug-in provider to get complete details on the resource impact, configuration impact, and operational impact of the plug-ins.

Hardware Resources

All this talk of sizing and estimators is great, but ultimately it comes down to hardware resources (physical or virtual). According to the VMware documentation, the minimum resources required for vCenter are as follows:

- 2 CPUs (physical/virtual cores). A *processor* is defined as a 2.0 GHz or faster Intel or AMD processor. The requirements may be higher if the database runs on the same machine.
- 3 GB RAM. This requirement may be higher if the database runs on the same machine. VMware VirtualCenter Management Webservices requires from 128 MB to 1.5 GB of additional memory. The VirtualCenter Management Webservices process allocates the required memory at startup. The engine that drives these web services is Tomcat JVM.
- 2 GB hard disk space. The requirements may be higher if the database runs on the same machine and/or if you host the Update Manager database and store the updates on that machine. While installing vCenter, you'll need up to 2 GB in which to decompress the Microsoft SQL Server 2005 Express installation archive. Approximately 1.5 GB of these files are deleted after the installation is complete.
- A gigabit network connection is recommended.

These minimum requirements are fine for a test environment and for kicking the tires. But if you already know that you'll be deploying a large number of hosts and VMs, you'll need more than the minimums. In fact, we've already shared with you some recommendations based on a few key factors: OS, database placement, number of objects managed, and the presence

(or absence) of VUM. In selecting or allocating hardware resources for vCenter Server, you shouldn't plan for the immediate need, but rather for the expected or required expansion.

We've now considered three of the five principles of design: availability, manageability, and performance. Our next principle is recoverability.

Recoverability

When you design the management layer with recoverability in mind, you're designing with failure in mind. How quickly will you be able to recover a failed vCenter Server instance? How quickly will you be able to recover from a failed vCenter database? As we described in Chapter 1, recoverability describes several different ideas, including well-known metrics like Recovery Time Objective (RTO) and Recovery Point Objective (RPO) in addition to other measurements like Mean Time To Repair (MTTR). Concepts of disaster recovery and business continuity (DR/BC) are also included in this principle of design.

How does all this apply to the design of the management layer? As with all areas of vSphere design, the ideas of this principle are closely related to and heavily intertwined with the ideas in the other principles of design:

- Using vSphere HA or vCenter Server Heartbeat, as described when discussing availability, also has a direct impact on recoverability.
- Splitting the database onto a separate server, as described and recommended in the section on the principle of performance, can also help with MTTR by isolating different components and limiting the fault domain(s).

Backups are a key part of ensuring the recoverability of the management layer, and you should be sure that you're properly incorporating backup of the management layer into the design. You'll need to account for backups of all the various management layer components—vCenter Server, vSphere Update Manager (if present), other management applications, and the appropriate database servers—when crafting the backup strategy for the management layer.

Finally, complete and comprehensive documentation is key to recoverability. A measure of recoverability is not only the time required to recover from a failure or other event, but also the amount of effort required to recover. A well-crafted set of documentation that is both comprehensive and up to date can greatly reduce the amount of time required to recover the environment.

The fifth and final principle of design that you need to apply to the creation of the management layer design is security, and it's the focus of the next section.

Security

Perhaps this part of the chapter should have come first, but its current position has no reflection on its importance in the design process. Nine out of ten administrators would rank security as one of the most important considerations in virtual infrastructure design.

How can you design your vCenter Server for security? Here are three considerations you should keep in mind:

- Isolation
- Permissions
- SSL certificates

Let's look at each of these considerations in a bit more detail.

Isolation

Your vCenter Server, from a security perspective, is probably the most important component of your infrastructure. If someone compromises the vCenter Server, they can cause immense damage, from powering off VMs, to deleting data from a VM, to deleting a VM or—worse—a complete datastore.

Your vCenter Server shouldn't be accessible from all workstations in your corporate network. You can achieve this by using any of the following measures:

- Put the vCenter on a separate management VLAN.
- Put the vCenter behind a firewall.
- Define an access list on the network switches.
- Set firewall rules on the vCenter Server.

Permissions

By default, the Administrators Security group on a vCenter Server has full permissions to the entire environment. Do you always want the administrators on the vCenter to control every VM? This isn't always the best idea. Limiting the default Administrators group's access to the full vCenter can be achieved by doing the following:

1. Create a local security group on the vCenter Server (vi-admins).
2. Create a local user on the vCenter Server (viadmin-local). Then, if for some reason your domain account isn't available or the vCenter can't contact the domain, at least you'll have a way to control your environment.
3. Create a domain user (viadmin).
4. Add both the viadmin and viadmin-local accounts to the vi-admins local group on the vCenter Server.
5. Change the Administrator permissions from the default to the vi-admins group.

In addition, you should always follow the model of least-privilege permissions. Only give a user account the minimum required permissions needed to perform a task. There is no need to give a user the Administrator role if all they need is to access the VM console. It's a better idea to create a custom role that contains only the relevant permissions needed for the task and then assign that permission to that user.

One excellent example is VM access. Consider this: would you give any user who asked you the code for the server room so they could power on a server if they wanted to? That wouldn't be wise.

You should view your vCenter Server as a secure area of your datacenter. Not everyone should have access to the vSphere infrastructure. If you want to give users access to the server, they should access it through either Remote Desktop or an SSH session (depending on the OS).

It isn't recommended that you enable a VNC client on a VM; this requires additional configuration on each and every ESXi to allow for such remote management. In such a case, you're better off with a custom role.

SSL Certificates

Client sessions with vCenter Server may be initiated from any vSphere API client, such as vSphere Client and PowerCLI. By default, all traffic is protected by SSL encryption, but the default certificates aren't signed by a trusted certificate authority and therefore don't provide the authentication security you may need in a production environment. These self-signed certificates are vulnerable to man-in-the-middle attacks, and clients receive a warning when they connect to the vCenter Server.

If you intend to use encrypted remote connections externally, consider purchasing a certificate from a trusted certificate authority, or use your own PKI infrastructure in your domain to provide a valid certificate for your SSL connections.

You can locate the official "Replacing vCenter Server Certificates" guidelines here (this was the latest version available at the time of writing):

```
www.vmware.com/files/pdf/vsp_4_vcserver_certificates.pdf
```

Summary

The factors involved in the design of your vCenter Server shouldn't be taken lightly. You'll need to design the management layer and account for all five of the design principles: availability, manageability, performance, recoverability, and, last but not least, security.

Take into account how many hosts and VMs you have and what kind of statistics you'll need to maintain for your environment.

Size your server correctly from the ground up, so you won't need to redeploy when you outgrow your environment.

Remember that your vCenter Server should be separated from your database server, providing a separation of duties for the different components of the infrastructure. You should plan for redundancy for both components and take into account what kind of outage you can afford to sustain. When you have that information, you can plan the level of redundancy you need.

Don't be afraid to run your vCenter as a VM. In certain cases, you can provide a greater level of resilience, with more ease, than you can with a physical server.

Your vCenter is the key to your kingdom. Leaving it exposed places your kingdom out in the open. A great number of attacks today are carried out from within the network, for a number of reasons: disgruntled employees, malicious software, and so on. Protect that key with the methods we've discussed.

Chapter 4

Server Hardware

All vSphere hosts rely on the underlying hardware to provide a suitable environment to run the ESXi hypervisor and the guest virtual machines (VMs). The server hardware is the most significant factor that affects the host's capabilities, performance, and cost. A vSphere design will never be successful if the server hardware isn't fit for the task.

Often, during the initial stages of a design, hardware procurement is on the critical path, because the process of selecting, approving, and ordering hardware and having it delivered can take several weeks. However, as this chapter investigates, you must review many considerations before embarking on the right server choice.

The chapter is split into the following sections:

- The importance of hardware, and the factors that influence and constrain your choices
- How vendors differ and the options to consider when choosing among them
- Which server components are most important to a vSphere installation and why
- Scale up a design with large powerful servers, or scale out with more agile servers
- Choosing between rack servers and blade servers
- Understanding new consolidated datacenter approaches
- Alternatives to buying servers

Hardware Considerations

A host's hardware components are critical to the capabilities of your vSphere environment. Selecting the correct mix and ensuring sufficient capacity will determine how much guest consolidation is possible. Unlike most of the other chapters in this book, which discuss software options, this chapter examines hardware, which is much more difficult to change after the initial implementation. Most software choices can be reconfigured after the fact, albeit with perhaps short outages, if something doesn't work as expected or you find an improvement to the design. With hardware selection, however, it's crucial that the architecture can survive not only the proposed requirements but any likely changes.

A design that tries to plan for any eventuality will end in overkill. You can't expect to cover every possible variant. Your goal should be a design that isn't overly specific but that covers a few of the most likely contingencies. This section will help you think about what is required before you start to select individual components.

Frequently, in a vSphere project, the first item addressed is the purchase of hardware. When any project begins and new hardware is needed, a whole procurement cycle must begin. A basic design is provided, along with the justification; project managers push it forward, and managers who control budgets are involved. Often, due to the large expense associated with server hardware, several nontrivial levels of approval are required, requests for tender follow, and vendor negotiations ensue. The cycle repeats until everyone is satisfied. But that's usually only the start.

Once the servers are ordered, it's often several weeks before they're delivered. Then, the server engineers need to test and configure them. Prior to that, a pilot may be required. The hardware must be moved to the appropriate datacenter, racked, and cabled. Power, network, and storage need to be connected and configured appropriately. All this installation work is likely to involve several different teams, and possibly coordination among several companies.

This potentially drawn-out hardware procurement cycle can take months from start to finish. This is one of the reasons virtualization has become so popular. Many managers and solutions architects are beginning to forget how long and painful this process can be, thanks to the advent of virtual hardware and the subsequent almost-immediate provisioning that is now possible.

For this reason, server hardware is nearly always on the critical path of a vSphere deployment. It's important to start this process as quickly as possible to avoid delays. But so much relies on the hardware that until you make many of the other decisions covered in this book, it's impossible to correctly design the server configurations. Buying incorrectly specified server hardware is likely to cause a vSphere design to fail to meet expectations, and this issue can be very difficult to recover from. Probably more than any other factor, the server hardware must be designed properly before you rush forward.

It's possible to identify certain do's and don'ts for server hardware that reduce the likelihood of ordering the wrong equipment. The next section looks at what is likely to determine these choices in your individual circumstances.

Factors in Selecting Hardware

The physical hardware of vSphere servers plays an important role in several areas. These roles and their importance in a specific environment will shape the hardware's design. Some are hard requirements, such as particular hypervisor features that need an explicit piece of equipment or functionality in order for the feature to be available. Others are quantitative options that you can select and weigh against each other, such as the amount of RAM versus the size or speed of the hard drives.

Features

Several vSphere features have specific hardware requirements. These features may not be available or may not run as efficiently if the correct underlying hardware isn't present. It's therefore critical to review these features prior to purchasing equipment and decide whether you need any of them now or are likely to need them. Purchasing servers without the capability of running a required feature could be an expensive mistake.

These features are discussed in more depth in the subsequent chapters, but at this stage it's important for you to understand their hardware requirements:

vMotion vMotion relies on hosts having a level of similarity. The CPUs must be from the same vendor (either Intel or AMD) and of the same family providing the same hardware flags. This compatibility is particularly important within vCenter clusters, because this is the

level at which many of the additional features that use vMotion (such as distributed resource scheduling [DRS]) operate. vMotion is available between hosts in the same datacenter, even if they're in different clusters, so there is merit to host hardware consistency beyond the cluster if at all possible.

The following VMware article lists Intel CPU compatibility:

`http://kb.vmware.com/kb/1991`

And this VMware article lists AMD CPU compatibility:

`http://kb.vmware.com/kb/1992`

Chapter 8, "Datacenter Design," examines a technique known as Enhanced vMotion Compatibility (EVC), which can make servers with somewhat dissimilar CPUs be vMotion compatible.

Fault Tolerance VMware's fault tolerance (FT) has a number of hardware limitations. These are listed in detail in Chapter 8, but it's important to know that there are some strict CPU requirements.

The following VMware article lists CPUs compatible with FT:

`http://kb.vmware.com/kb/1008027`

Distributed Power Management DPM must have access to Intelligent Platform Management Interface (IPMI), Hewlett Packard's Integrated Lights Out (iLO), or a network adapter with Wake-On LAN (WOL) capabilities, to power on the server when the cluster requires it.

DirectPath I/O DirectPath I/O allows a special I/O passthrough to a VM from the NIC or possibly a storage host bus adapter (HBA). This feature relies on specific Peripheral Component Interconnect (PCI) cards being used; but more important, the CPU must support either Intel's VT-d or AMD-Vi (input/output memory management unit [IOMMU]).

SR-IOV Single Root I/O Virtualization (SR-IOV) is similar to DirectPath I/O but provides multiple virtual instances of the devices to be presented to VMs. To enable this, the server and its BIOS must support SR-IOV and IOMMU, and the PCIe card's driver and firmware must support SR-IOV.

Performance

The hypervisor's performance relates directly to the hardware. Other elements can have an effect, but the most significant performance enabler is derived from the hardware itself. The general premise of *more and faster* is better; but with other limiting constraints, you must usually choose what hardware gives the most performance bang for your buck.

In a vSphere design, the main performance bottlenecks revolve around CPU, memory, and I/O (particularly storage I/O). Therefore, in a server's hardware, the CPU and memory are critical in terms of its scalability. Most other server components are required in order to provide functionality, but they don't tend to limit the performance the same way. The CPU and memory rely on each other, so a good balance is required. The question of smaller but more numerous servers, as opposed to fewer servers that are more powerful, is examined in a later section in

this chapter; but either way, the server's CPU and memory ratio should correlate unless you have a particular need for more of one.

Other elements can limit performance, but most newly purchased up-to-date servers from mainstream vendors avoid the obvious bottlenecks unless you have unusual cases that demand special attention. This chapter looks closely at both CPUs and memory.

Reliability

The vSphere server hardware is likely to be a critical infrastructure piece in the datacenter and has the potential to make up a large part of a company's compute resources. It's obvious that the server's stability is paramount. Therefore it's important when you're selecting hardware that each component be thoroughly reliable. Although it's possible to find whitebox equipment that works with ESXi, and which may even be listed on the HCL, it's important to consider the server's reliability.

Servers for a production environment should be from a reputable vendor with a proven track record. Many companies avoid the first-generation series of a new server line, even from a recognized top-tier server vendor, because this is where any stability quirks in the BIOS code or hardware agents are most likely to be found.

Additionally, with each new server, a period of testing and *bedding-in* is normal and is part of checking for stability issues. Common approaches are discussed later in the chapter.

Redundancy

Along with general reliability, a server's components should provide sufficient redundancy to avoid outages during hardware failures. All components, even the most reliable, will fail periodically. However, a well-designed server can mitigate many of these failures with redundant parts. You should choose servers that are designed to take redundant parts and ensure that you order them with the extra parts to make them redundant.

These are the most common server parts that should be offered with redundancy:

- Hard drives with both RAID protection and hot spares
- Power supply units (PSUs) that not only protect from a failure of the PSU but also let you split the power supply across two separate circuits
- Multiple network and storage interfaces/cards, allowing connections to separate switches
- Several fans that prevent overheating, should one fail

Upgradability and Expandability

An important element, particularly in vSphere hosts, is the ability to expand the server's hardware options at a later stage. Server hardware is often purchased with an expected life cycle of three to five years, but rapid advances in hardware and software, and continuously falling prices, often make upgrading existing servers an attractive option.

It's somewhat unrealistic to expect to upgrade a server's CPU at a later stage, because the increase in performance is likely to be minimal in comparison to the additional cost. And you're unlikely to buy a server with excess sockets that aren't filled when the server is first purchased (not to mention the difficulty of finding the exact same CPU to add to the server). However,

RAM tends to drop significantly in price over time, so it's feasible that you could consider a replacement memory upgrade. Larger servers with extra drive bays offer the option for more local storage, although this is rarely used in vSphere deployments other than locations without access to any shared storage facilities.

The most likely upgrade possibilities that you may wish to consider when purchasing servers is the ability to fit extra PCI-based cards. These cards can add network or storage ports, or provide the potential to later upgrade to a faster interface such as 10GbE or converged network adapters (CNAs). This is one of the reasons some companies choose 2U-based server hardware over 1U-based rack servers. If space isn't an issue in the datacenter, these larger servers are usually priced very similarly but give you considerably more expandability than their smaller 1U counterparts.

Computing Needs

It's important to look carefully at the computing needs of the vSphere environment before you create a detailed shopping list of server parts. Although generalizations can be made about every vSphere deployment, each one will differ, and the hardware can be customized accordingly.

Hardware Compatibility List

VMware has a strict Compatibility Guide, which for hypervisor servers is colloquially known as the hardware compatibility list (HCL). It's now a web-based tool, which you can find at `www.vmware.com/go/hcl`. This is a list of certified hardware that VMware guarantees will work properly. Drivers are included or available, the hardware will be supported if there's an issue, and VMware has tested it as a proven platform.

Choosing hardware that isn't on the HCL doesn't necessarily mean it won't work with vSphere; but if you have issues along the way, VMware may not provide support. If a component that isn't on the HCL does work, you may find that after a patch or upgrade it stops working. Although the HCL is version-specific, if hardware has been certified as valid, then it's likely to be HCL compatible for at least all the subsequent minor releases.

For any production environment, you should use only HCL-listed hardware. Even test and development servers should be on the HCL if you expect any support and the business needs any level of reliability. If these nonproduction servers mimic their production counterparts, this has the advantage that you can test the hardware with any changes or upgrades to the vSphere environment that you plan to introduce. A disciplined strategy like this also provides warm spares in an emergency as onsite hardware replacements for your production servers.

Which Hypervisor?

Chapter 2, "The ESXi Hypervisor," discussed the newer ESXi hypervisor and the differences from the older ESX. Despite their similarity, the hypervisor does have some impact on the hardware. ESXi is less reliant on local storage but can still use it if required. ESX and ESXi have different HCLs, so if an upgrade project is considering reusing hardware originally designed for use with ESX classic, you should check to ensure that the proposed solution is still compliant.

ESXi combines the Service Console and VMkernel network interfaces into one management network, so you may need one less NIC if you use 1GbE. ESXi also uses particular Common

Information Model (CIM) providers, which allows for hardware monitoring. If you're using ESXi, you should confirm the availability of CIM providers for the hardware being used.

If you want to use ESXi Embedded, it will probably have a significant effect on your hardware selection, because vendors sell specific servers that include this. In addition, the HCL for ESXi Embedded is much smaller than the HCL for Installable/Stateless, so it may limit your choices for adding hardware.

Minimum Hardware

The minimum hardware requirements for each version of vSphere can differ, so be sure to consult the appropriate checklist. Most designed solutions are unlikely to come close to the required minimums, but occasional specific use cases may have minimal custom needs. You still need to hit VMware's minimums in order for the hypervisor to be VMware supported.

Purpose

It's worth considering the type of VMs that will run on the hypervisor. vSphere servers can be used not only to virtualize general-purpose servers but also for a variety of other roles. A server may be destined to host virtual desktops, in which case servers should be designed for very high consolidation ratios. Alternatively, the hypervisor may host only one or two very large VMs, or VMs with very specific memory or CPU requirements. Some VMs need high levels of storage or network I/O; you can fit more capable controller cards to provide for the VM's needs, with high I/O ratings or the ability to do hardware passthrough. The servers may need to host old P2Ved servers that have specific hardware requirements such as serial or parallel ports for software dongles, or to access old equipment like facsimile modems or tape backup units.

Scaling

Buying the right hardware means not only getting the right parts but also scaling properly for your capacity and performance needs. If you buy too much, then resources will lie idle and money will have been wasted. If you buy too little, then resources will be constrained and the servers won't deliver the expected levels of performance and may not provide the required level of redundancy. No one likes to waste money, but an under-resourced environment means trouble. First impressions last, and if virtualized servers are a new concept to a business, then it's important that it perform as expected, if not better.

Not every design needs its hardware requirements planned from the outset. If your company's procurement process is sufficiently flexible and expeditious, then you can add server nodes as they're demanded. This way, the quantity should always be suitable for the job. Despite the planning and testing you conduct beforehand, you're always making a best estimate with a bit added for good measure.

Hardware Consistency

If you're purchasing new servers to supplement existing equipment, it's important to ensure that certain components are sufficiently similar. This is particularly significant if the new hardware will coexist in the same cluster, because this is where VMs frequently migrate.

Consistency within the same hardware generation is also important, so wherever possible it's advisable to set a standard level of hardware across the servers. If some situations require more or less compute resources, then you may want to implement two or three tiers of hardware

standards. This consistency simplifies installation, configuration, and troubleshooting, and it also means that advanced cluster functions such as DRS, high availability (HA), and DPM can work more efficiently.

Consistency within the same type of servers is beneficial, such as populating the same memory slots and the same PCI slots. You should try to use the same NICs for the same purpose and ensure that the same interface connects to the same switch. This makes managing the devices much easier and a more scalable task.

Server Constraints

In any server design, you must consider a number of constraints that limit the possible deployment. Any datacenter will be restricted primarily by three physical factors: power, cooling, and space. vSphere servers have traditionally put a strain on I/O cabling, and host licenses can restrict what server hardware is utilized.

Rack Space

The most apparent physical constraint faced in any server room is that of rack space. Even though virtualization is known to condense server numbers and may alleviate the problem, you still can't fit servers where there is no available space. Co-locating datacenters is common these days, and customers are often billed by the rack or down to the single U; even if it isn't your datacenter to manage, it still makes sense to pack in as much equipment as possible.

Aside from virtualizing, there are two common approaches to maximizing space: minimize the height of the rack servers or switch to blade servers. Rack servers are traditionally multi-U affairs, with sufficient height to stack ancillary cards vertically. But all mainstream vendors also sell 1U servers to reduce space. Many opt for blade servers as a way to fit more servers into a limited amount of rack space. A regular rack can take up to 42 1U servers; but most vendors sell 10U chassis that can fit 16 half-height blades, meaning at least 64 servers with 2U to spare. Both thin rack servers and half-height blades are normally sold only as dual-socket servers, so these methods align themselves more closely with a scale-out model. Both rack versus blade and scale-up versus scale-out are debated later in this chapter.

Power

With denser hardware configurations and virtualization increasing consolidation levels, power and cooling become even more important. Smaller, more heavily utilized servers need more power and generate more heat. Cooling is discussed separately later in this section; but be aware that increased heat from the extra power used must be dissipated with even more cooling, which in turn increases the power required. Older datacenters that weren't designed for these use cases will likely run out of power well before running out of space.

As energy prices go up and servers use more and more power, the result can be a significant operating expense (OPEX). Power supply can be limited, and server expansion programs must consider the availability of local power.

Most of the world uses high-line voltage (200–240V AC) for its regular power supply, whereas North America's and Japan's standard for AC supply is low-line voltage (100–120V AC). Most datacenter customers in North America have the option of being supplied with either low-line or high-line for server racks. Vendors normally supply servers with dual-voltage PSUs that are capable of automatically switching. High-line is considered more stable and efficient, can reduce

thermal output, and allows for more capacity. However, whatever the available power supply is, you should check all server PSUs, uninterruptible power supplies (UPSs), and power distribution units (PDUs) to be sure they're compatible. Some very high-performance servers may require three-phase high-line power to operate.

The power input for each server is often referred to as its *volt amperes* (VA), and this is cumulatively used to calculate the power required in a rack for PDU and UPS capacity. PDUs shouldn't provide more than half of its capacity in normal operations to ensure that it can handle the excess required if one circuit fails. Also consider the number and type of sockets required on each PDU. Vertical PDUs help save rack space.

It isn't just the make and model of servers that affect the power estimate, but also how fully fitted the server is with CPUs, RAM, disks, I/O cards, and so on. The hypervisor's load affects power usage, so if you expect to run the servers at 40% or 80% utilization, that should be factored in. Most hardware manufacturers have downloadable spreadsheets or online calculators you can use to make more accurate estimates of server power and cooling requirements. Figure 4.1 shows an example of one offering, but all vendors have their own versions.

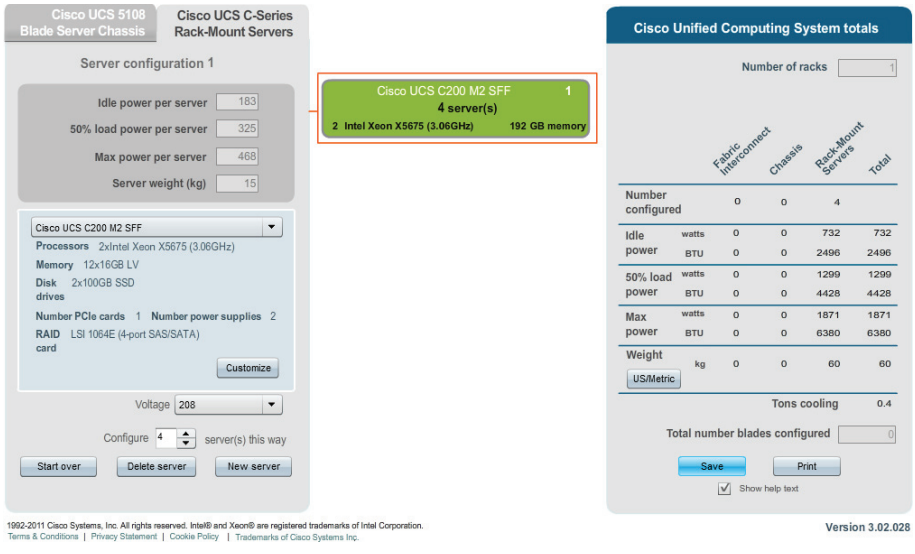


FIGURE 4.1
Example of a power and cooling estimation tool
Courtesy of Cisco

Server PSUs should come with inrush surge protection, because when power is initially applied to a server, it draws power momentarily on full load. This normally lasts only a few seconds but can use several times more current than normal. It's important to think about this with multiple servers in a cluster. Following a power outage, if all the servers in a cluster try to power back on at the same time, the result may be an inrush that can affect the whole room. When you're powering servers back on, consider waiting at least 10 seconds between servers. Most servers have power settings to automatically start up after a power failure, but look for those that can use a random time offset to help prevent inrush issues.

Power-design calculations are often made at the start of an initial deployment—for example, when a blade chassis is fitted and only semipopulated with blades. As time goes on, more blades are added. But because no extra power cables need to be fitted, the additional power

requirements are forgotten. If you're fitting a blade enclosure, then for design purposes, imagine it's fully populated.

UPS

UPSs are necessary to provide a clean, continuous supply of power to vSphere host servers. Any type of power failure or minor fluctuation can cause a shutdown. UPSs are designed to bridge the gap, automatically switching over to a battery bank until power is restored. UPSs can also filter out power spikes or sags, which can not only power off servers but also damage the PSUs and internal components. Many UPS systems provide automatic monitoring and alarming and can help with power capacity planning.

UPSs should be sufficiently rated to keep the servers powered on long enough to at least allow a clean shutdown of all VMs and hosts. Unlike regular servers, which only need to shut down one OS, hypervisors can be running tens of guests, which when all instructed to shut down at the same time can take several minutes to do so. Therefore, it's important to think about how long it may take banks of vSphere servers and all their VMs to go down cleanly.

For environments where uptime is absolutely crucial, UPS systems may only need to be enough to tide things over until a backup generator starts up. You should test any UPSs and standby power supply to ensure that all the equipment is suitably balanced and will be ready when required.

Cooling

All server hardware produces a lot of heat, which must be constantly dissipated to prevent the equipment from overheating. Cooling makes up a substantial amount of the power used in a datacenter, often over half the total power bill. Making that cooling more efficient means less cooling is required. Making the server's power usage more efficient also reduces cooling needs.

Cooling Measurements

Server heat is usually thought of in either watts (W), which is the amount of input power, or British Thermal Units (BTUs), which is the amount of cooling required for the power being consumed (BTU/hr = 3.4 × watts). In North America, cooling systems are often rated in tons of refrigeration (RT), where 1 ton is equal to the heat absorption of 3.5 kWh or 12,000 BTU/hr. This measure originally came from the amount of cooling energy found in one ton of ice.

When you're trying to minimize the amount of cooling that each server needs, think about the airflow through your servers from front to back. How much airflow do the rack doors allow? Is the cabling at the back impeding the flow? Are the side doors attached, servers stacked together, and blanking covers fitted to prevent hot and cold air mixing? With server cooling systems, it's important to think of the entire room, because the air isn't contained to one server or one rack. Use hot and cold aisles, and think about the placement of AC units, perforated floor tiles, and the use of overhead conduits even if you have raised floors, to split power from other cables and leave more room for cooling.

I/O Ports

In addition to the power cabling provided by PDUs, servers have a collection of cables that need to be connected. These can include Ethernet cables for both network and storage I/O, fiber optic cables, out-of-band management card connectors, and KVM cabling. Prior to fitting new servers,

it's critical that you consider the amount of cabling and the number of ports required to connect each one. This means you need to know how many switch ports, patch panel ports, fibre switch ports, and KVM ports are free and usable. Remember, some of these types of equipment also need licensing on a per-port basis.

Proper capacity management also means thinking about the I/O loads on these connectors, to ensure that additional workloads won't prevent linear scaling.

vSphere Licensing

Although it isn't a physical constraint, vSphere licensing can be an important consideration. Many vSphere features are only available with higher-tier licensing, but licensing may also restrict the hardware you can use. This is in addition to the fact that larger four- or eight-way servers need more licenses. In a new vSphere deployment, this becomes a project cost; but if you're supplementing an existing environment that already has a license agreement drawn up, your existing license may reduce your options.

vRAM Licensing

When vSphere 5.0 was released, VMware also introduced a new licensing model that included the concept of vRAM. vRAM was a measure of all the configured memory of powered-on VMs across the ESXi hosts. Depending on your level of vSphere license, you were suddenly restricted to a certain amount of memory. If you used any more, then you were expected to buy additional licenses. These changes had the potential to artificially affect host design. With the increase in core-count per socket and ever-dropping RAM prices, instead of balancing CPU and memory for the best performance and thinking about scale-up versus scale-out, the financial impacts of this new scheme changed the equation.

This move from VMware was extremely unpopular with vSphere customers. It was perceived (somewhat unfairly) as a money-grab from VMware. Shortly after the announcement, VMware doubled the allowed vRAM amounts within each licensing level to try to appease the dissenting voices. In reality, most vSphere licensees were never affected by the changes, but the damage had been done. VMware's competitors jumped on the opportunity.

One year later, alongside the release of vSphere 5.1, VMware announced the expunction of vRAM. VMware rescinded vRAM in 5.1 and retroactively annulled it from 5.0. vRAM is now relegated to the news archives and will no longer be bothering any vSphere designs. However, it's worth noting that if you created designs during those dark vRAM days, you might need to revisit them and reconsider the influence of this constraint. Perhaps there is scope to improve your host architecture on your next hardware refresh.

Differentiating among Vendors

Several vendors produce servers suitable for vSphere hypervisors. The Tier-1 companies commonly associated with ESXi servers are HP, IBM, and Dell, although Fujitsu-Siemens has a limited following in Europe. Cisco, well known for its networking equipment, burst onto the scene in 2009 with its new line of servers; it can also be considered a mainstream vendor, despite its infancy in the server market. Many other companies have products listed on VMware's HCL, but they're much less well-known and arguably less trusted.

An option for the most budget-conscious business is what is known as a *whitebox server*. These are computers that aren't sold by a recognized vendor and may even be configured from parts. Whitebox servers tend to lack the high-end features available from the main vendors, such as redundant parts and on-hand identical stocked replacements, and whitebox servers rarely scale beyond one or two CPUs. Such servers may appear on the HCL or meet the minimum requirements, but checking each part is left up to you.

It's difficult to recommend whitebox servers, although this approach has a popular community following and is frequently used for home test-lab type situations. A couple of excellent sites list tested whitebox equipment, although obviously VMware will only support those on its own HCL:

```
http://vm-help.com/esx40i/esx40_whitebox_HCL.php

http://ultimatewhitebox.com/systems
```

Both sites largely revolve around ESXi 4 compatibility, but the advice is still largely valid. The forum connected to the `vm-help.com` site has good 5.x device information.

The vast majority of vSphere installations happen on Tier-1 supplied hardware. The relative importance of hypervisor hardware in comparison to regular servers, largely in part due to its high consolidation workload, means most companies spend the extra dollars to buy trusted equipment. Another reason these vendors are so popular with vSphere is that it's still an enterprise-dominated product. Small companies that are more likely to use whitebox equipment haven't embraced hypervisors so readily. They often don't benefit as much from consolidation and lack dedicated IT staff with the skills to implement it.

In certain circumstances, you may be unable to choose a different vendor, because your organization has an approved supplier. This may be due to prenegotiated pricing, tender agreements, or a historical preference for one brand that makes continued support easier if everything remains the same. But given the opportunity to choose between vendors, beyond the raw computing power of their servers you may wish to consider the following points for hypervisor equipment. Many of them use the same underlying generic hardware, but these value-adds make them different and are particularly important for hypervisor servers, which usually have a very high criticality in a datacenter:

Warranty and Support Server warranties are commonly for three years, although they often can be extended on a year-by-year basis. Warranties are obviously important should a component fail, but it's also important for Tier-1 vendors to stock exact replacement parts. For example, in a multi-CPU server, if one CPU fails, only an identical CPU can be fitted to match the existing ones. If you have a cluster full of servers, a different replacement server won't suffice.

Support agreements vary between vendors, and often each vendor has different options available. Compare how they can offer support—telephone support, instant messaging, email, and so on—and what hours they're willing to provide support (such as business hours or 24/7). If you have multinational offices, be sure the vendor provides international support. Previous experience with a vendor will often give you a feel for the level of service you can expect. Agreements should also specify onsite support, detailing how quickly the vendor will fit replacement parts or be onsite to troubleshoot issues.

HCL Investment Top-tier vendors should be investing in ongoing certification work with VMware. Doing so ensures that their products continue to be supported under the HCL and

helps the vendors optimize their equipment for the hypervisor. This means drivers can be automatically included in vSphere build media, and the vendors have suitable hardware agents or CIM providers to enable hardware monitoring.

Technologies A lot of the hardware included in servers is fairly generic and not usually manufactured by the vendor. However, vendors try to distinguish themselves with newer technologies, such as the ability to pack in more memory, optimize internal buses, or be the first to market with a particular CPU.

Later in the chapter, we'll consider consolidated approaches that match networking and storage options to servers to provide all-in-one packages.

Hardware Management Most server vendors provide a centralized hardware-management tool, such as HP's System Insight Manager, IBM's Director, or Dell's OpenManage. It manages your hardware and provides reporting notification tools to trigger alerts when problems occur (such as failed disks). These tools often provide the capability to push out BIOS and agent updates from a central location. These products often come with additional licensing fees for extra functionality, although the base tool may come with the server.

Remote Management Another important server option that can differ between vendors is the availability and functionality of out-of-band management cards. HP uses iLO, IBM uses RSA (Remote Supervisor Adapter) cards, and Dell has Dell Remote Access Cards (DRACs). These can offer numerous remote-access tools, but the more important ones for vSphere servers are as follows:

- Remote console access
- Power-button access
- Virtual optical drives
- Hardware status and logging

Some vendors include base functionality while licensing the more advanced features; others sell add-on hardware cards. These management cards with remote console access are particularly useful for offsite datacenters or remote offices where onsite support is less likely to be able to deal with an ESXi console screen.

Server Components

Servers have a multitude of options available, and almost every component can be customized for your needs. vSphere host servers have particular needs; with careful consideration of each part, you can design a server to best fit its role as hypervisor. This section looks at each component important to virtualization, the impact it has, and where your budget should concentrate.

Before we explain the function of each component, remember the basic premise of type 1 hypervisors. vSphere ESXi virtualizes the CPU, memory, disk, and network I/O to maximize throughput, making as few changes as possible so as to not impede performance. Most other hardware functions are emulated in software, because they don't play a critical role in performance and are referenced relatively little. How these four elements are shared among the hypervisor and guests is critical in overall performance, but any improvement in hardware that can improve the efficiency and speed of the CPU, memory, and I/O is crucial.

CPU

VMware vSphere 5 hosts only run on top of 64-bit CPUs. The server's CPUs are critical in the performance of the VMs. Most servers come equipped with at least two CPU sockets, although four- and eight-way models are common as companies scale up. The most recent major advance is the use of multicore CPUs and the significant performance increases they can provide. CPUs used to be measured purely in MHz, but now vendors are packing in more punch by delivering CPUs with multiple cores. 4-, 6-, 8-, and 10-core CPUs are available now, and more are delivered in each generational refresh.

Multicore CPUs and Scheduling

A multicore CPU consists of a single socket processor with multiple core units. These cores can share some of the cache levels and can also share the memory bus. Each core has near-native performance to that of a single-core CPU, so a dual core is close to two single CPUs, and a quad core is close to four single CPUs or two dual-core CPUs. Sharing the same caches and buses can reduce performance when the VMs are particularly memory intensive, but otherwise multicore CPUs offer compelling performance for their modest increase in price.

Some Intel CPUs have a feature known as HyperThreading (HT) that allows each physical core to behave as two logical cores. HT allows two different threads to run on the same core at the same time. This may speed up some operations, depending on the software running at the time. The gains are likely to be marginal and certainly not as substantial as having additional physical cores. vSphere uses HT by default, as long as it's enabled in the server's BIOS. Since Intel's Nehalem chip, HT has been referred to as simultaneous multithreading (SMT).

The VMkernel employs a complex but extremely efficient CPU scheduler. Its purpose is to equitably share CPU resources between its own needs and those of all the running VMs. With default resources allocated, a vSphere host time-slices processing power equally among all the VMs as soon as the CPU resources are overcommitted. Ordinarily, the host needs to take into account VM shares, reservations, and limits; the number of allocated vCPUs (VM CPUs); and the varying demands made by each VM. A VM should be oblivious to the fact that it's running on virtualized hardware, so the scheduler needs to give the impression to the VM that it completely owns the CPU. This becomes increasingly complicated when VMs have multiple vCPUs that expect all their processors to compute at the same time and not to have to wait on each other. This synchronous use of CPUs is maintained in the VMkernel with a technique known as *co-scheduling*. The co-scheduling algorithms have steadily evolved with each ESX (and ESXi) release, with continuous improvements being made to how the CPU scheduler deals with symmetric multiprocessor (SMP) VMs.

The CPU scheduler must take into account the number of physical CPUs and cores, whether HT is available, the placement of logical and physical cores in relation to the CPU caches and their cache hierarchy, and memory buses. It can make informed choices about which core each VM should run on, to ensure that the most efficient decisions are made. It dynamically moves vCPUs around cores to yield the most efficient configuration with regard to cache and bus speeds. It's possible to override the CPU scheduler on a VM basis by setting the CPU affinity in a VM's settings. This process is explained in Chapter 7, "Virtual Machines." By pinning vCPUs to specific cores, you can optimize a VM's usage. However, the built-in CPU scheduler is incredibly efficient, and pinning vCPUs to cores can prevent simultaneous workloads from being spread among available cores. This may lead to the VM performing worse and will interfere with the host's ability to schedule CPU resources for the other VMs.

CPU Virtualization

CPUs from both Intel and AMD have continued to evolve alongside each other, mostly offering comparable features (albeit with each one pushing ahead of the other, followed by a quick period of catch-up). As each vendor's products are released, new features are added that can help to improve performance and capabilities while also potentially breaking compatibility with previous versions.

vSphere uses virtualization, rather than CPU emulation where everything runs in software and the underlying hardware is never touched. Virtualization is different in that it tries to pass as much as possible to the physical hardware underneath. This can result in significantly better performance and means VMs can take advantage of all the features the CPUs can offer. With regard to server hardware choices, the impact comes from compatibility between hosts. A VM runs on only one host at a time. However, when the host is a member of a cluster, the hosts must present similar CPUs to the guest VMs to allow vMotion. If a host exposes more (or fewer) features than another host in the cluster, you can't vMotion the VMs between them. This in turn affects other features that rely on vMotion, such as DRS.

You can configure clusters with Enhanced vMotion Compatibility (EVC) settings, which effectively dumbs down all the hosts to the lowest common denominator. This technically solves the incompatibility problems but can mask instruction sets from your new CPUs that the VMs might be able to take advantage of. If there are mixed hosts, then this is a useful technique to allow them to cohabit a cluster and prevent segmentation of compute resources. Enabling EVC is a balancing act between the flexibility and elasticity of your cluster resources, against the perhaps obscure potential to remove a performance enhancing feature that an application benefits from.

Also be aware that there is currently no compatibility between Intel hosts and AMD hosts. You should split these servers into separate clusters whenever possible. Incompatible hosts can still power on VMs moved from other hosts, so you can take advantage of HA if you have no choice but to run a mixed cluster.

FT also has specific CPU requirements, which you should account for if FT is part of your design. Chapter 8 provides more details about FT requirements and how they may affect CPU decisions.

VMware uses two types of virtualization in its vSphere 5 products:

Binary Translation VMware's original method of virtualizing guest OSes is binary translation (BT) or, as VMware recently began calling it, *software-based virtualization*. BT attempts to pass as much as possible directly to the host's CPU; but it knows which calls shouldn't be allowed through, intercepts them, and translates them in software. Inevitably, this method uses slightly more CPU cycles than native OS calls, but very few calls need to be translated. It's surprisingly efficient and is the basic technique that VMware also uses on its hosted Type 2 products.

Hardware-Assisted Virtualization With the advent of certain new processors, most of the system calls that can't be passed on directly can be intercepted in special hardware instead of software. This newer method of virtualization uses hardware-assisted CPU virtualization (HV). This reduces the associated CPU overhead and should improve overall processor efficiency. The introduction of HV-enabled servers has diminished the need for paravirtualization and is the main reason for it being retired.

In previous versions of vSphere, a third type of virtualization was supported, known as paravirtualization. *Paravirtualization* is a technique that is possible when a guest VM is aware that it's

virtualized and can modify its system calls appropriately. Because paravirtualization depends on guest OS cooperation, it could only be used with certain OSes. It was enabled on a per-VM basis with a feature known as Virtual Machine Interface (VMI). Support for paravirtualization was deprecated in vSphere 5.0 due to the advent of hardware-assisted CPUs and lack of OS support.

Virtualization Enhancements

Subsequent generations of CPUs from both Intel and AMD offer virtualization-specific enhancements. The first generation supported CPU improvements, the second generation of hardware advancements adds optimizations to the overhead associated with the memory management unit (MMU), and the third generation allows VMs direct access to PCI devices:

Hardware-Assisted CPU Enhancements The hardware-assisted CPU enhancements are available in all CPUs that have the Intel VT-x or AMD AMD-V flags. These CPUs allow the use of a HV Virtual Machine Monitor (VMM), which is more efficient than BT.

Hardware-Assisted MMU Enhancements Hardware-assisted MMU enhancements rely on a newer generation of CPUs. Intel packages this as Extended Page Tables (EPT) and AMD as Rapid Virtualization Indexing (RVI) or Nested Page Tables (NPT). These MMU improvements allow virtual-to-physical page mappings to occur in hardware, as opposed to being the responsibility of the hypervisor's MMU. CPUs with this feature can hold an additional level of page tables and avoid the need for the shadow page tables that the hypervisor normally maintains.

Hardware-Assisted I/O MMU Enhancements The latest hardware enhancement that can benefit a virtualized workload is I/O MMU, which is available on Intel VT-d or AMD-Vi systems. This chipset improvement means a VM can access the memory and interrupts of peripheral devices such as network adapters, HBAs, or graphics cards. It is this PCI passthrough technology that allows direct access, avoiding the hypervisor and enabling features such as DirectPath I/O and SR-IOV.

CPU Capacity

When you're selecting CPUs for your server hardware, there are several things to consider. The overall strategy of scaling up or scaling out may dictate the spread of CPUs to memory, which will be discussed in significantly more depth in the aptly named "Scale Up vs. Scale Out" section. Because CPUs are such a significant part of a server's ability to consolidate VMs, it's important to get the most powerful processors possible.

The general premise of faster, newer, and more is reasonable and won't see you wrong; but for virtualization-specific scaling, you should look a little further. The high core count on some CPUs yields massive improvements. Get the most cores possible, because other than scaling up to more CPUs, you'll achieve the greatest improvements. Any recently purchased CPUs should have the hardware-assisted CPU and MMU additions, but this is worth checking. Paying more for incrementally faster CPUs usually won't give you the same return as additional cores.

Scaling the server to the VMs depends largely on the workload of the VMs and the number of vCPUs per VM. As an approximate guide, you should expect to get at least four vCPUs per physical core. As the number of cores per CPU increases, your vCPU consolidation may drop slightly because the cores are getting smaller proportions of the CPU bus and shared memory cache. Some workloads can comfortably fit far more vCPUs per core, so if possible test the configuration with your own environment.

RAM

In additional to CPUs, host memory is critical to the performance and scalability of the server. With the core count on most servers rising rapidly, it's increasingly important to have enough memory to balance the equation. There is little point in cramming a server full of the latest CPUs if you have so little RAM that you can only power on a few VMs.

vSphere hypervisors are incredibly efficient with memory usage and have several methods to consolidate as many VMs onto the same host as possible. In order to make the most of the limited supply, you should understand the basic ways in which guest VMs are allocated RAM.

Memory Usage

vSphere hosts need memory for both the host and the VMs:

Host The host itself needs memory to run the VMkernel processes. This is used for the system, device drivers, and management agents. Since vSphere 5.1, you can manually create a system swap file, up to 1 GB in size. This allows the host to swap this allocated memory if it is under pressure. To create a system swap file, use the following command with its required parameters: `esxcli sched swap system.`

VMs VMs have memory allocated to them that is mapped through to guests' physical memory pages for use by the OS. Each VM also carries a certain amount of overhead that depends on the RAM allotted, the number of vCPUs, the video memory (by default only 4 MB, but can be more if you need higher resolutions and multiple screens—for example, VDI workstations), and the base VM hardware. Table 4.1 shows the memory overhead incurred, over and above the memory you allocate, for the most common VM configurations. The memory overhead is required for the VM to be powered on. vSphere 5 has made improvements that substantially reduce this overhead for VMs, which allows more memory to be available for use by VMs. The drop is likely to recoup several GBs of memory on most servers. For example, a 1 vCPU VM with 1 GB of memory consumed about 124 MB of memory as overhead when it ran on an ESXi 4.1 host, but on ESXi 5.1 this has dropped to 26 MB. An 8 vCPU VM with 16 GB of RAM would be allocated around 1 GB of memory overhead previously, and now with 5.1 uses only 169 MB.

Table 4.1: Memory overheads for common VM configurations

Allocated Memory	1 vCPU	2 vCPUs	4 vCPUs	8 vCPUs
1 GB	26 MB	30 MB	38 MB	54 MB
4 GB	49 MB	53 MB	61 MB	77 MB
16 GB	140 MB	144 MB	152 MB	169 MB

Memory Mapping

The hypervisor maps the host's physical memory through to each powered-on VM. Ordinarily, the memory is divided into 4 KB pages and shared out to the VMs, and its mapping data is

recorded using a page table. The guest OS is unaware that the memory is being translated via the hypervisor; the OS just sees one long, contiguous memory space.

vSphere can also use *large pages*. Large pages are 2 MB in size, and if a guest OS is able to and enabled to use them, the hypervisor uses them by default. Large pages reduce the overhead of mapping the pages from the guest physical memory down to the host physical memory.

Hardware-Assisted Mapping

The memory mapping is stored in a shadow page table that is then available to the host's MMU, unless the host has CPUs that are capable of hardware-assisted memory mapping. Hardware-assisted MMU virtualization is possible if the host is fitted with either Intel's EPT support or AMD's RVI support (AMD's RVI is also occasionally referred to as Nested Page Tables [NPT]). Using this additional hardware feature can improve performance compared to the shadow page table technique because it reduces the associated overhead of running it in software. Only some guest OSes can use hardware-assisted MMU; vSphere uses the shadow page table for those that aren't supported.

Memory Overcommitment

vSphere has a unique set of features to overcommit its memory. This means it can potentially provide more memory to its guests than it physically has on board. It can transfer memory to guests that need more, improving the server's overall memory utilization and increasing the level of consolidation possible.

Memory overcommitment is successful largely because at any one time, not all guests are using their full entitlement of allocated RAM. If memory is sitting idle, the hypervisor may try to reclaim some of it to distribute to guests that need more. This memory overcommitment is one of the reasons virtualization can use hardware more efficiently than regular physical servers.

Techniques to Reclaim Memory

Several methods exist in vSphere to reclaim memory from VMs, enabling more efficient guest memory overcommitment. These are the five primary methods used, in order of preference by the VMkernel:

Transparent Page Sharing *Transparent page sharing* (TPS) is the process of removing identical memory blocks and replacing them with logical pointers to a single copy. The process is similar to how storage products use deduplication to reduce storage costs. When VMs use memory blocks that have the same content between them, or the same in a single VM, then only one copy needs to be stored.

Using TPS results in less host memory being used and therefore more opportunities to consolidate more running VMs on the one host. TPS doesn't compare every last byte but instead uses a hash of each 4 KB page to identify pages that need closer inspection. Those are then compared to confirm whether they're identical. If they're found to be the same, then only one copy needs to be kept in memory. The VM is unaware that it's sharing the page.

Ballooning When the VMware tools are installed into a guest OS, it includes a spurious device driver that is used for memory *ballooning* (more correctly known as *vmmemctl*). Ordinarily, the hypervisor is unaware of what memory is most important to the guest, and

the guest doesn't know if the host is under memory pressure. The balloon driver is a mechanism the hypervisor can use to ask the guest to choose what memory should be released. The guest understands which pages are being used for what purpose and can make much better decisions about freeing up memory and swapping, so it has far less impact on performance.

When the VMkernel needs a guest to release memory, it *inflates* the balloon by telling the driver to try to consume more memory as a process in the guest. The guest OS then decides what is the least valuable to keep in memory. If the VM has plenty of free memory, it's passed to the balloon driver, and the driver can tell the hypervisor what memory pages to reclaim. If the VM doesn't have any free memory, the guest OS gets to choose which pages to swap out and begins to use its own pagefile (in the case of Windows) or swap partition/file (in the case of Linux). This means the balloon driver, and hence the hypervisor, can make full use of the guest's own memory-management techniques. It passes the host memory pressure on to the guests, which can make enlightened decisions about what pages should be kept in RAM and which should be swapped out to disk.

By default, the balloon driver only ever tries to reclaim a maximum of 65% of its configured memory. The guest must have a sufficiently large internal pagefile/swap to cover this; otherwise the guest OS can become unstable. As a minimum, you must ensure that your guests have the following available:

$$\text{Pagefile/swap} \geq (\text{configured memory} - \text{memory reservation}) \times 65\%$$

However, because there is the potential to change both the reservation and the pagefile after the VM is created, it's always advisable to make the guest's pagefile at least as large as the RAM allocated. Remember that if you bump up the VM's RAM, you need to also increase the internal guest pagefile for this reason.

We strongly recommended that ballooning not be disabled within VMs. If users are concerned about the potentially negative effect on their VMs, consider strategies other than disabling ballooning to prevent memory overcommitment. For example, applying a memory reservation will reduce the chance that any ballooning will occur (in fact, a full memory reservation means all the VM's memory is mapped to physical memory and no reclamation will ever happen).

Compression vSphere 4.1 introduced a memory-compression algorithm that analyzes each 4 KB page and determines whether it can compress the page down to at least 2 KB in size. If it can't compress the page that small, the page is allowed to be swapped to disk.

When the VM needs the page again, it decompresses the file back into guest memory. Despite the small latency and CPU overhead incurred by compressing and decompressing, it's still a considerably more efficient technique than host swapping. By default, the memory-compression cache is limited to 10% of guest's memory, and the host doesn't allocate extra storage over and above what is given to the VMs. This prevents the compression process from consuming even more memory while under pressure. When the cache is full, it replaces compressed pages in order of their age; older pages are decompressed and subsequently swapped out, making room for newer, more frequently accessed pages to be held in the cache.

Swap to Host Cache A technique introduced in vSphere 5.0 utilizes faster solid-state drive (SSD) disks for hypervisor swapping, reducing the swapping to VMs' dedicated `.vswp` files. The cache is created on a per-host basis and, although remote SSD disks can be used, locally

attached SSD disks are preferable to reduce the additional disk access latency incurred over the fabric/network. Dedicated host SSD disks are many times faster than the disk normally allocated to VMs and their `.vswp` files, which reduces the impact of host swapping.

A pool of cache for all the VMs on a host is created. This isn't the same as moving all the VMs' `.vswp` files to a SSD datastore, because the cache doesn't need to be large enough to accommodate all the `.vswp` files. If host swapping is necessary, then this host cache is used first; but once it's all allocated, the host is forced to use the VM allocated `.vswp` files. Regardless of this host cache being available, each VM still requires its own `.vswp` file. However, the larger the host cache available, the less extensively the VMs' `.vswp` files are used.

The swap-to-host-cache feature reduces the impact of host swapping but doesn't eliminate it unless each host has more SSD space than configured vRAM (fewer memory reservations). Remember to include HA failovers and host maintenance downtime if you're sizing SSD for this purpose. This is ideal but isn't always possible for servers with large amounts of memory and a small number of drive bays—for example, high-density scale-up blades.

Swapping When a VM is powered on, the hypervisor creates a separate swap file in the VM's working directory called *vmname*`.vswp`. It's used as a last resort to recover memory and page memory out to disk when the host is under heavy memory contention. The VMkernel forcibly moves memory pages from RAM onto disk; but unlike the action of the balloon driver, it can't use the intelligence of the guest's memory management and grabs chunks of memory randomly.

Host swapping leads to significant performance degradation, because the process isn't selective and will undoubtedly swap out active pages. This swapping is also detrimental to the host's performance, because it has to expend CPU cycles to process the memory swapping.

If the guest is under memory pressure at the same time as the host, there is the potential for the host to swap the page to the `.vswp` file, and then for the guest to swap the same page to the guest's pagefile.

Host swapping is fundamentally different than the swapping that occurs under the control of the guest OS. Ballooning can take time to achieve results and may not free up enough memory, whereas despite its impact, host swapping is an immediate and guaranteed way for the host to reclaim memory.

Preferred Swap File Location

By default, a VM's `.vswp` file is located in its working directory. But you can change this using a setting available in the cluster or host settings. Moving these swap files to an alternate location is often done to use less of the expensive shared storage by dumping the files onto the hosts' local disks. It can also be used to remove these volatile files from logical unit numbers (LUNs) that are snapshotted or replicated on the array.

Using local host disks has an impact on vMotion performance because these potential large files need to be moved with the VM for every transfer. This can obviously affect any services that rely on vMotions, such as the automatic host evacuation of Maintenance Mode, Update Manager, DRS, DPM, and so on. You should avoid using any thinly provisioned storage for alternate swap space, because an out-of-space condition will disrupt the VMs.

When Memory Is Reclaimed

Memory is only reclaimed from nonreserved memory. Each VM is configured with an amount of memory. If no memory reservation is set, then when the VM is powered on, the `.vswp` file is created as large as the allocated memory. Any memory reservation made reduces the size of the swap file, because reserved memory is always backed by physical host memory and is never reclaimed for overcommitment. It's guaranteed memory and so is never swapped out by the host:

$$\text{Swap file (vswp)} = \text{configured memory} - \text{memory reservation}$$

Because the host never attempts to reclaim reserved memory, that proportion of memory is never under contention. The VM's shares only apply to allocated memory over and above any reservation. How much physical memory the VM receives above its reservation depends on how much the host has free and the allocation of shares among all the VMs.

The use of the memory-reclamation techniques depends on the amount of free host memory. Some processes run all the time, and others are activated as less free memory is available. There are four defined levels of memory usage: High; Soft, which is two-thirds of High; Hard, which is one-third of High; and Low, which is one-sixth of High (the exact values are 64%, 32%, and 16% of High, respectively). Prior to vSphere 5, High was set by default at 6%, which meant Soft approximately equaled 4%, Hard 2%, and Low 1%. These levels are primarily in place to protect the VMkernel from running out of memory, but as host memory capacity has grown, the need to protect so much isn't as relatively great. For example, an ESXi 5.0 host can potentially run on hosts with 2 TB of RAM. Using these static values, the host would start reclaiming memory even when it still had over 100 GB of free memory. To set the levels more effectively, vSphere 5 adjusts the High level according to the amount of memory in the host. Assuming your host has more than 28 GB of memory installed, High is set at 900 MB plus 1% of all memory above 28 GB. Table 4.2 extrapolates the memory levels for some common server configurations.

TABLE 4.2: Memory reclamation levels

HOST MEMORY	HIGH	SOFT	HARD	LOW
48 GB	1,105 MB	707 MB	354 MB	177 MB
64 GB	1,269 MB	812 MB	406 MB	203 MB
96 GB	1,596 MB	1,022 MB	511 MB	255 MB
128 GB	1,924 MB	1,231 MB	616 MB	308 MB
256 GB	3,235 MB	2,070 MB	1,035 MB	518 MB
512 GB	5,856 MB	3,748 MB	1,874 MB	937 MB
768 GB	8,478 MB	5,426 MB	2,713 MB	1,356 MB

If the VMkernel is deemed to be in the High memory state—that is, it has more free memory than the High column in Table 4.2—it's considered not to be in contention. But when

the memory state is pushed above this level, the host begins to compare VM shares to determine which VMs have priority over the remaining memory.

If the amount of free memory drops lower as VMs consume more resources, then as it reaches each predetermined threshold, more aggressive memory reclamation takes place. Each level is designed to get the host back to a free memory state:

Regular Cycle (Regardless of Memory State) TPS runs regularly, even when there is no memory pressure on the host. By default, the host scans each VM every 60 minutes to find redundant pages. The one notable exception is after a Windows VM powers on, because the guest OS touches all of its memory as it boots up. vSphere runs TPS over those VMs immediately and doesn't wait until the next cycle.

However, TPS only scans 4 KB memory pages and not 2 MB large pages. This is because the large pages are far less likely to have identical contents, and scanning 2 MB is more expensive. The one thing it does continue to do, regardless of the host's memory state, is create hashes for the 4 KB pages in the large pages.

The other process that runs regularly is the calculation of *idle memory tax* (IMT). VMs' memory shares are used when the host hits levels of memory contention, to figure out which ones should have priority. However, a VM with higher levels of shares may be allocated memory that it isn't actively using. To rebalance the shares so those VMs that really need memory are more likely to get some, IMT adjusts the shares to account for the amount of unused memory. It "taxes" those VMs that have lots of idle memory. IMT is regularly calculated (every 60 seconds by default) despite the level of free memory on the host. Remember that the shares that are being adjusted are taken into account only when the host is in memory contention. IMT runs all the time but is used only when memory usage is above the High memory level.

Memory State Reaches High When the memory state hits High, the hypervisor calls TPS immediately even if it isn't due for another run. Ideally, this brings the host back under the High level. As it rises toward the Soft limit, it preemptively starts to use the balloon driver, knowing that it can take time to reclaim memory, using the shares (adjusted by IMT) to make sure those deemed more worthy are under less pressure.

Memory State Reaches Soft At the Soft memory state, ballooning is in full swing trying to reclaim memory from guests to recover the host back below High.

Memory State Reaches Hard If the Hard memory state is reached, the host starts to forcibly reclaim memory by swapping the VMs' memory to their `.vswp` files. At this point, compression kicks in to try to reduce the amount of data being swapped out. In addition, large pages begin to be broken down into regular 4 KB pages so they can be shared via TPS to avoid them being swapped to disk if possible. Ideally, all these measures recover the host's memory back to a free state.

Memory State Reaches Low If the host's memory usage rises above the Low memory state, the host stops creating new pages for VMs and continues compressing and swapping until more memory is freed up and the host recovers.

Figure 4.2 shows the levels at which the different memory-reclamation techniques kick in. As less memory is available for the VMs, the VMkernel becomes more aggressive.

The one exception to these memory state levels is if a VM has a memory limit set. If an artificial limit is set, then when the VM reaches it, the host begins to balloon and, if necessary,

swap the VM. The host does this even if it has plenty of free memory. Chapter 7 discusses the dangers of setting VM memory limits.

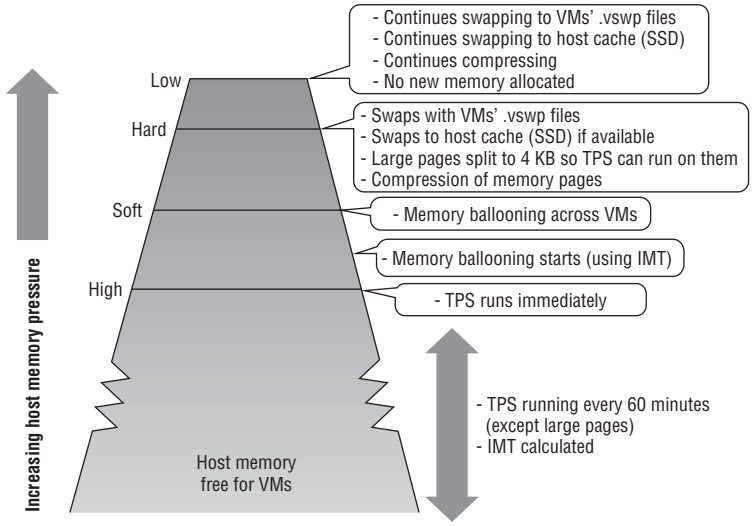


FIGURE 4.2
How memory is reclaimed

MEMORY CAPACITY

vSphere hypervisor servers demand more from their memory than the average general-purpose OS. Not only do they require swaths of capacity, but they often test the hardware more rigorously than other OSes.

Achieving the right balance of CPU cores to memory is important. A general rule of thumb is to make sure you have at least 4 GB RAM per core. Many of today's memory-hungry applications tilt this ratio, so you may need more memory than 4 GB per core, although this very much depends on workload. Remember, this ratio is related to the proportion of shared and reclaimed memory to the time-sliced co-scheduling of the CPU cores. It's used to understand what a balanced amount of RAM may be for an average server with a certain CPU configuration.

As the core density in modern servers grows, it can become increasingly difficult to fit in enough memory. At the time of writing, 16 GB modules are the sweet spot for memory, being only an additional 20% premium relative to 8 GB sticks. Jumping up to 32 GB modules can quadruple the price from 16 GB. However, if you have a 4-way, 12 -core server, you need at least 192 GB to make sure you have enough memory to back all the cores. If you want to set up a greater consolation ratio of 8 GB to each core, then unless you have a server with at least 24 DIMM sockets, you'll have to pay the extra for those costly 32 GB modules.

As 32 GB modules become the norm and CPUs gain more cores, you should reevaluate this guideline for your environment. But be aware that more dense form-factor motherboards can constrain overall system performance unless more expensive memory is fitted.

Fortunately, with the abolition of vRAM, vSphere licensing is based on CPU socket count, not RAM; if you have the slots available, buying the maximum amount of memory you can afford is probably worthwhile. Arguably, memory is the number-one scalability factor in any server.

Aside from capacity, the front-side bus speed on the memory is an important element of overall server performance. The speed of the bus and the attached memory often have as much impact as the CPUs driving the instructions.

NUMA

Most modern servers come with nonuniform memory access (NUMA), occasionally referred to as nonuniform memory architecture, enabled CPUs and motherboards. NUMA is available on all recent AMD and Intel CPUs, with support on current motherboard chipsets (although some vendors extol the virtues of their own specialized chipset support).

Multi-CPU servers, particularly those with multiple cores, face a bottleneck when so many processors simultaneously try to access the same memory space through a single memory bus. Although localized CPU memory caches can help, they're quickly used up. To alleviate this issue, NUMA-enabled servers' CPUs are split into nodes that have access to localized RAM modules that have much lower latency. NUMA combines CPU and memory allocation scheduling. But if VMs need access to nonlocal memory, this can actually increase latency beyond normal SMP-style architecture and degrade performance.

vSphere can use NUMA-enabled systems and has a specially tuned NUMA CPU scheduler to manage the placement of VMs. Each VM is allocated a home node and is then given memory from the same home node. The NUMA scheduler dynamically balances the home-node allocations every 2 seconds, rebalancing them as each VM's CPU and memory requirements change.

The NUMA scheduler uses TPS memory sharing on a per-node basis to prevent shared pages being matched from nonlocal memory. You can disable this feature if memory is particularly tight on the server or if many VMs are very similar and will benefit more from TPS.

One of the problems faced by the NUMA scheduler is VMs with more vCPUs than each node has cores. Also, if a VM is allocated more memory than a single node's local memory can provide, it must get some of its memory across an intersocket connection. vSphere 4.1 introduced *wide NUMA* scheduling to improve the placement of VMs with more vCPUs than a single node can hold, which prevents them becoming more scattered than they need to be and allocates memory as locally as possible. Of course, if you know how many vCPUs and how much RAM will be allocated to your larger VMs, you can scale your server design sufficiently to make sure that, where possible, the VMs will fit on a single NUMA node.

Often, NUMA is disabled in the BIOS. The setting Node Interleaving means the server ignores NUMA optimizations and doesn't attempt to localize the memory. To enable the use of NUMA, make sure Node Interleaving is set to Disabled. Normally, NUMA is only enabled on hosts with at least four cores across at least two NUMA nodes.

NUMA allocation is yet another reason it's advisable to have similarly specified servers in the same DRS cluster. Otherwise, VMs can vMotion between hosts where the source has one NUMA node size but there's a different node allocation on the next. The DRS mechanism is currently unaware of NUMA calculations and sizes on each host.

ESXi 5.0 introduced support for virtual NUMA (vNUMA). vNUMA presents the host's physical NUMA typology through to the guest OS and its applications, allowing them to participate in placement optimization. If a guest OS can understand the underlying NUMA structure, it can schedule its guest threads to better align with the NUMA nodes. vNUMA is enabled by default on VMs greater than eight vCPUs because those are the most likely VMs to naturally span more than one NUMA node. For a VM to be exposed to vNUMA information, the hardware compatibility must be set to ESXi 5.0 or greater (hardware version 8).

Arguably, NUMA as a feature is useful but probably not enough to make you buy servers because they include it. However, if you're deploying servers that have NUMA, various design options can take advantage of the NUMA scheduling and maximize local low-latency memory access for your VMs. If possible, it's preferable to size VMs' vCPUs as multiples of the host's NUMA node. The physical placement of RAM modules in the motherboard slots affects which CPUs use that RAM as local memory, so normally you should follow your vendor's advice and ensure that the RAM modules are spread evenly so each CPU receives an equal share.

Motherboard

The server's motherboard, sometimes known as the mainboard or system board, dictates what components can be fitted to the server and how many of each. The motherboard is designed to cope with the hardware, so it shouldn't be a bottleneck; but different vendors try to provide competing efficiencies because the motherboard is one of the few pieces particular to them.

One of the more crucial elements on the motherboard is the chipset that provides the interface between the CPU and its front-side bus (FSB) to the memory and peripheral buses. The motherboard and its chipset mandate the number and type of CPUs, RAM slots, and PCI slots. Given sufficient space in the case, how expandable the server is depends on its motherboard.

The motherboard can also be responsible for an onboard RAID solution, although it's more common to use a separate controller card in large servers. It monitors the temperature of components and can adjust the internal fans appropriately. Motherboards also provide integrated peripherals such as serial, parallel, and USB ports and usually have onboard diagnostics.

Normally, motherboards aren't marketed as options, but they're the main differentiators between a vendor's models. Choosing a particular model of server isn't just about the form factor of the case; primarily it's about the motherboard inside. From a vSphere design perspective, it dictates the expandability of the server and configuration maximums available. Generationally, newer boards allow you to connect newer hardware components. In addition to choosing the CPU family and number of sockets you need, along with the capacity to fit all the memory and cards required, you should look for designs with better bus speeds, the latest PCIe standards, and the largest system caches.

Storage

Storage in vSphere is a fundamental topic that is the basis for Chapter 6, "Storage." From a server hardware perspective, it revolves around two different areas: the local storage that the server commonly boots from and the way in which the server connects to any shared external storage.

Local storage is most often used for the ESXi boot image. Other options exist, such as Boot from SAN or Auto Deploy images, which can negate the need for any local storage. The local storage can also be physically external to the server itself, in an expansion shelf connected via a SCSI cable.

If the local storage will be used to run VMs on a Virtual Machine File System (VMFS) partition, the performance of the disks is important. In this case, the speed of the disks, interface connector (SAS, SATA, and so on), RAID type, number of spindles in the RAID set, and RAID controller are all factors in the VM's performance, because disk I/O is important. Local VMFS storage is often used in small offices and remote office locations where shared storage may not be available. It tends to be significantly less expensive and so can be useful to store less important data, or as an emergency drop location if there are issues with shared storage.

If local storage is only used to boot the vSphere OS, performance is arguably less important. The ESXi hypervisor is loaded from disk entirely into memory, so apart from minor differences in boot speed, faster disks won't improve performance. The expense of buying faster disks is probably better spent on more CPU cores or more RAM modules. Any production server should employ some sort of RAID redundancy protection, but an extra hot spare provides an inexpensive additional level of protection.

The more common storage for VMs is shared storage where the servers connect to centralized storage. The method a server uses to connect to that shared storage is dictated by the protocol and transport used by the storage array. The common connections are Fibre Channel (FC) host bus adapters (HBAs), Fibre Channel over Ethernet (FCoE) converged network adapters (CNAs), iSCSI hardware HBAs, and Ethernet network cards for both software iSCSI and Network File System (NFS). Because the speed and resilience of these connections can be paramount to VMs, the selection of the cards, their speed rating, the redundancy of ports, and the PCI connector type are all important. Select the best PCI card connection possible on the motherboard, because the storage cards should ordinarily take priority over any other I/O cards. It's advisable to buy two single-connector cards instead of one dual-port card if the budget allows, because this will help to spread the I/O across two PCI connectors and provide redundancy if a card fails or disconnects.

Network

Network I/O is also a cardinal vSphere design topic and is explained in depth in Chapter 5, "Designing Your Network," but several design choices with respect to server hardware are worth discussing at this juncture. First, although most servers have two or four onboard 1GbE network adapters, it isn't uncommon to see an extra two or even three four-port 1GbE PCI cards to cover all networking needs. If you're using any FC HBAs or CNAs, you should reserve the fastest PCI connections for them and then use the next available fastest PCI slots for your additional network connections.

But if there is no need for other storage bandwidth-intensive cards, or you're going to use 10GbE cards to aggregate storage and network traffic, these should be in the fastest slots possible. Although using 10GbE ports is likely to reduce the number of cables used, at the time of writing few servers come with onboard 10GbE; and like storage cards, you may choose to use two one-port cards instead of a single two-port card, so you still need at least two high-speed PCI slots. Try to get cards that support NetQueue, because it can improve 10GbE performance.

If a company has specific rules about DMZ cabling or doesn't use trunked network ports, you may need even more network cards.

PCI

PCI is a standard bus used by expansion cards to connect to a motherboard. The original PCI standard has evolved through numerous versions, including the PCI-X and PCI Express (PCIe) revisions. Table 4.3 shows the increased theoretical maximum bandwidth between the standards.

The PCI-X interface became increasingly popular with 1GbE cards because the cards couldn't saturate the bus link. Now the PCI Express standard brings bus speeds closer to the FSB speeds used by CPUs today. Most servers come with PCI Express slots, but you should check how many and of what type, because some have only one or two or may not be PCI Express version 2.0 or 3.0. Consider the number of high-speed PCI slots and the version number against your card

requirements. At the time of writing, only the latest-generation servers were shipping with PCIe 3.0 sockets, and network/storage cards were not yet available.

TABLE 4.3: PCI bus speeds

BUS	MAX BANDWIDTH
PCI	133 MB/s (although extended up to 533 MB/s for 64-bit at 66 MHz)
PCI-X	1,064 MB/s
PCI Express	250 MB/s per lane (8x is 2 GB/s, 16x is 4 GB/s, 32x is 8 GB/s)
PCI Express 2.0	500 MB/s per lane (8x is 4 GB/s, 16x is 8 GB/s, 32x is 16 GB/s)
PCI Express 3.0	1 GB/s per lane (8x is 8 GB/s, 16x is 16 GB/s, 32x is 32 GB/s)

DIRECTPATH I/O

DirectPath I/O is a technique employed by vSphere to directly connect up to two PCI devices to a VM, allowing the I/O to bypass the virtualization layer. This can potentially reduce latency and improve the performance of high-speed cards such as 10GbE and FC HBAs, but few cards are currently supported. To use DirectPath I/O, the host CPU must support Intel's VT-d or AMD-Vi (IOMMU).

This method places several restrictions on the VM, such as no vMotion, Storage vMotion, FT, or snapshots. And the performance gains are so slight that unless device performance is paramount, DirectPath I/O may be something you wish to avoid. It's unlikely to be useful unless there is such an extreme level of network I/O that it significantly affects the host's CPU usage. Cisco's Unified Computing Systems (UCS) platform has a special certification that removes the vMotion and snapshot restrictions.

You should make sure your highest-speed slots are used for your highest-bandwidth I/O cards, to avoid buses being a bottleneck. Usually, storage cards—whether FC HBAs, FCoE CNAs, or 10GbE interfaces—take precedence. If you're limited on PCI Express slots, ensure that these cards are fitted first. Less bandwidth-intensive workloads such as 1GbE network adapter cards can use less-well-specified slots. For a single-port 10GbE card, aim to use at least a PCI Express 2.0 x4 slot; and for a dual-port card, use a x8 as a minimum.

One last important design consideration for host PCI slots is consistency. Ensuring that servers have the same cards in the same slots can ease installation and configuration, particularly cabling, and make troubleshooting hardware issues considerably more straightforward. This becomes increasingly important as your deployment techniques mature with greater levels of automation. If all the servers in a cluster are the same make and model, then having the same I/O cards in the same slots means that each port gets the same vmnic or vmhba number. If you have a multisite rollout, and you're able to purchase the same server hardware for more than one location, think about all the sites' storage and networking requirements before choosing

which slot to use for which. For example, although you may have only a few sites with a FC SAN, with the rest using 1GbE-connected iSCSI or NFS devices, you may wish to always put the 1GbE cards into a slower slot. Even though at most sites you have one or two slots free that are very high performance, the 1GbE cards won't use the extra bandwidth, and you can keep all the servers consistent across the fleet.

SR-IOV

A new feature added in vSphere 5.1 is the support of Single Root I/O Virtualization (SR-IOV). It's easy to think of SR-IOV as the evolution of DirectPath I/O. Unlike DirectPath's one-to-one mapping through to a VM, SR-IOV allows a PCIe device to present multiple virtual devices, known as virtual functions (VFs), through the hypervisor. This allows multiple VMs direct access, and multiple instances within each VM. Each card's firmware and driver controls the physical functions (PFs) and maps them to VFs that are exposed to the VMs. Configuration settings can be made on the PF, which the VFs inherit. The guest OSes in the VMs must understand that the presented VFs aren't full PCIe cards and don't have the same level of configurability.

Much like DirectPath I/O, SR-IOV has some fairly limiting restrictions associated with its use:

- Very small number of PCIe cards supported
- Server hardware must specifically include support in the BIOS and have IOMMU enabled
- Only Red Hat Enterprise Linux (RHEL) 6.1 or Windows 2008 R2 SP1 guest OSes
- PCIe NICs can't be used as host uplinks (vmnics) when they're enabled for SR-IOV

When a VM is configured for one or more VFs, the following functions aren't available:

- vMotion, Storage vMotion, DRS, or DPM
- HA protection or FT support
- Standby/hibernate or suspend and resume
- Hot adding/removing of VM hardware devices
- NetFlow or vShield support

Basically, the VM loses its ability to participate in the cluster. It's locked to the host. vSphere 5.1 supports a limited number of VFs depending on which card is used. Again, like DirectPath I/O, these limitations on VMs mean SR-IOV is usefully only in corner cases. If you have VMs that are very latency sensitive (for example, a VOIP appliance) or that are driving an acute level of I/O (enough to affect the hypervisor's resources), then you may want to test SR-IOV.

Preparing the Server

After server hardware selection is complete, you should test the server's non-OS settings and hardware configuration prior to using them in production. You need to make several choices regarding their setup, and a good design should include a set of preproduction checks to make sure the hardware is deployed as planned. Hardware configuration and testing will affect the rollout.

Configuring the BIOS

Every server's hardware settings are primarily configured through its BIOS. Some of the default settings set by vendors are configured for general-purpose OSes and may not give you optimal performance with vSphere. To get the most from your servers, several settings should always be set:

Sockets and Cores Ensure that all occupied CPU sockets are enabled and all cores are enabled.

Hardware Assist Enable hardware-assisted virtualization features. For the CPU, this is Intel VT-x or AMD-V; for memory, it's Intel EPT or AMD RVI.

Execute Protection Enable the Execute Protection feature, because it's required for EVC. This is eXecute Disable (XD) on Intel-based servers or No eXecute (NX) on AMD.

Node Interleaving Disable node interleaving, or NUMA management won't be used. (IBM refers to this setting as Memory Interleaving.)

HyperThreading HT should be enabled.

The following are settings you may wish to consider changing:

Power Settings Servers often have power-management technologies that attempt to save power while they aren't being fully utilized. But some users report that these settings reduce vSphere's performance, stepping down the CPUs unnecessarily. Some of the newer servers now include OS Control as an option, which allows the hypervisor to control CPU throttling. This tends to provide better results than letting the system's firmware moderate it. Consider disabling this setting, because performance should almost always be more important than saving power.

Turbo Mode Enable any Turbo Mode settings, which can temporarily increase the CPU clock speed when more performance is required and thermal limits allow. Intel uses the moniker Turbo Boost for this feature.

Extraneous Hardware Consider disabling any hardware that is unused, such as legacy serial and parallel ports. These continue to create unnecessary hardware interrupts while they're enabled and cost CPU cycles.

In general, you should aim to keep as many settings as possible at the manufacturer's default. Use the server installation as a good opportunity to update the BIOS to the latest version available. Do this first, before taking the time to change any options, because re-flashing the BIOS may reset everything. Strive to have all BIOS firmware levels and settings identical across every host in a cluster.

Other Hardware Settings

In addition to the BIOS settings, remember that you should set several other settings according to the designed environment:

RAID Controller and Disk Configuration Before installing the hypervisor, you need to configure the local disks into a usable RAID set. It's advisable to reserve one local disk as a hot spare.

I/O Cards Each I/O card, such as the network and storage PCI cards, has its own firmware that should be updated and configured. Normally, these are set up with a keystroke during

the server's Power On Self Test (POST). You should consult not only the I/O card's manufacturer but also the destination device's recommended practices.

Remote Access Cards Most servers at least have the option of an out-of-band management card. Prior to being ready for production, you should set this device's IP address, hostname, password, and so on.

Burn-in

Before each server is unleashed under a production workload, you should test it extensively to ensure that any initial teething problems are identified and rectified. Most servers have a hardware diagnostic program in the BIOS, during the POST, or on a bootup CD. This utility runs a series of stress tests on each component to make sure it can cope with a full load.

For vSphere, the server's memory is the most critical thing to test thoroughly, because it's used so much more intensively than normal. A useful free tool to test memory is Memtest86+, which you can download from `www.memtest.org` and burn to a CD. We recommend that you boot new servers off the CD and let the utility run its memory testing for at least 72 hours before using the server in production.

Preproduction Checks

Finally, after the server's hypervisor is installed and before the server is brought into a working cluster and allowed to host VMs, you should perform several checks:

- Memory and I/O cards have been fitted to the correct slots.
- The server is racked properly, and the cabling is correctly fitted. Using cable-management arms not only helps improve airflow but also allows access to the servers without shutting them off to change hot-swap items and check diagnostic lights.
- The storage cards can see the correct datastores. If you're using NFS, the server has write access to each datastore.
- The network cards can see the correct subnets. Move a test VM onto each port group, and test connectivity.
- This is a good opportunity to make sure the hypervisor is patched.
- NTP is working properly.
- The server has the appropriate vSphere licensing.

Scale-Up vs. Scale-Out

vSphere allows administrators to spread the x86 infrastructure across multiple physical hosts, with the ability to consolidate several workloads onto each server. Each VM's hardware layer is virtualized, abstracting the underlying physical compute resources such as CPU and memory from each VM's allocation. This abstraction allows you to separate the decisions around VM scaling from those of the host servers. The process of virtualizing the guest systems gives rise to an important design decision: how much consolidation is desirable. The ever-expanding capabilities of today's hardware allows an unprecedented level of VMs to hypervisors; but as an

architect of vSphere solutions it's important to understand that just because they can be larger doesn't necessarily make them the most desirable configuration for your business.

The *scale-up versus scale-out* argument has existed as long as computers have. Virtualized infrastructure has its own implications on the debate; and as hardware evolves, so do the goal posts of what scale-up and scale-out really mean. Essentially, a *scale-up* design uses a small number of large powerful servers, as opposed to a *scale-out* design that revolves around many smaller servers. Both aim to achieve the computing power required (and both can, if designed properly), but the way in which they scale is different.

The classic scale-up scenario was based around server CPU sockets; in general computing circles during the ESX virtualization era, this usually meant one or two sockets for scale-out and four or eight sockets for scale-up. But in the last few years, this definition has been significantly blurred, primarily due to a couple of hardware advances. First, the size of RAM modules in terms of gigabytes and the number of DIMM sockets per motherboard has increased massively. Even in relatively small servers with one or two sockets, the amount of memory that can be fitted is staggering. Smallish blade servers can take large amounts of RAM. For example, some of Cisco's UCS blade servers can handle up to 1.5 TB of RAM! Second, the number of physical sockets on a server no longer necessarily dictates the CPU processing power, because the advent of multicore CPUs means a 2-way Intel server can have 20 CPU cores, and an 8-way server can have a colossal 80 cores. At the time of writing, AMD had 16-core CPUs available, and undoubtedly these two vendors will only continue apace.

These monstrous memory and core levels rewrite the rules on scale-up and scale-out and reiterate the message that the design is no longer based only on socket numbers. But the underlying premise still holds true. Scale-up is a smaller number of more powerful servers; scale-out is about lots of smaller servers. It's just that the definitions of large and small change and are based on differing quantifiable means.

With regard to vSphere servers, the scale-up or scale-out debate normally revolves around CPU and memory. I/O is less of a performance bottleneck, and storage and networking requirements are more often an issue of function rather than scale. These things work or they don't; it isn't so much a matter of how well they work. We're talking about the server hardware, not the switches or the storage arrays themselves. Obviously, storage can be a performance bottleneck, as can the backplanes on the switches; but with regard to the server hardware, we mean the I/O cards, adapter ports, and transport links. These adapters rarely dictate a server's level of scalability. There can be clear exceptions to this, such as security-focused installations that require unusually large numbers of network ports to provide redundant links to numerous air-gapped DMZ switches, or hosts that need to connect to several older, smaller SANs for their collective storage. Create a rule, and there will always be an exception. But generally speaking, scale-up versus scale-out server design is concerned with CPU and memory loading.

As this chapter has identified, CPU and memory are both important performance characteristics of a host server. It's important to note that for the *average* VM workload, you need to maintain a good balance of each. Even though RAM modules are inexpensive and a two-socket server can fit a very large amount of memory, they may not help much if the VMs are CPU-constrained. Similarly, even if it's comparatively cheap to buy eight-core CPUs instead of four-core CPUs, if you can't afford the extra memory to fit alongside them, the extra cores may be wasted. The CPU and memory requirements must normally be in balance with each other to be truly effective.

A common misconception is that scaling up means rack servers and scaling out means blades. The densities that can be achieved in both form factors mean that scaling up and out

decisions aren't necessarily the same discussion. To understand a business's requirements, you should examine each independently. Although there is potential for crossover in arguments, one certainly doesn't mean the other. Blades and rack servers have their own interesting architectural considerations, and we'll look at them in the next section.

Now that you understand the basic definitions of scaling up and out, we can compare each approach.

Advantages of Scaling Up

The advantages of scaling up are as follows:

Better Resource Management Larger servers can take advantage of the hypervisor's inherent resource optimizations, such as TPS or CPU co-scheduling (but remember that by default, TPS on NUMA servers only shares pages on the same NUMA node, not across the entire server). Although scaling out can use DRS to load balance, it doesn't make such efficient use of resources.

Larger servers can cope with spikes in compute requirements much more effectively, whereas smaller servers have to react by using load-balancing techniques that incur significant delay.

Cost This is an interesting advantage, because the classic scaling based on CPU sockets meant that scaling up used to be more expensive. Generally, a four-way SMP server was much more expensive than four one-socket servers. However, with the changes in server components, scaling up often means adding more and more cores and RAM modules; and by scaling up instead of buying more and more smaller servers, you're likely to achieve some savings. By scaling up, the RAM or processors need not be more expensive, so scaling can be linear, and you save on the number of PSUs, I/O cards, case components, and so on.

Fewer Hypervisors With fewer servers loaded with ESXi, you have fewer hypervisors to manage. Despite the number of VMs running, each physical server needs to have its hypervisor OS installed, maintained, patched, upgraded, monitored, and so on. Scaling up means fewer instances to manage.

Lower Software Licensing Costs VMware licenses its software on a socket basis, so having servers with more cores and more memory means fewer socket licenses to buy. This makes scaling up an attractive option when your business is paying for every server it adds. When the vRAM licensing appeared at the time of vSphere 5.0's release, it had a heavy impact on the design options for scaling up. Fortunately, now that the vRAM licensing model has been withdrawn, this no longer artificially skews the hardware choices.

Additionally, many businesses license their guest OS software on physical servers. You can buy Microsoft server licensing to cover unlimited guest copies per host hypervisor. The fewer hosts, the fewer licenses needed.

Larger VMs Possible Large servers give you more flexibility with VM scaling. A two-way quad-core server can only accommodate a VM with eight vCPUs, and even that isn't particularly desirable. If you stick to the rule that you should always have more cores than the largest VM has vCPUs, then such a server should only host VMs with six vCPUs. If you can feasibly think that some elements of your business will need VMs with 16 vCPUs, then you may want to consider hosts with at least 20 cores (2-way with 10 cores, or 4-way with 6 cores) At the time of writing, Intel does produce 10-core CPUs, but only on its most premium

models. The practical option for most are the 8-core models, which means that for 16 vCPUs cases, you may need to consider a 4-way server instead of a 2-way.

Less I/O Cabling Each vSphere host is likely to need a similar number of network and storage cables attached. By using fewer but more powerful servers, you cut down the switch ports, fabric ports, and cabling needed. This in itself may reduce the number of fabric switches or switch blades, further reducing the amount of infrastructure provisioning required. The network team has fewer server ports to manage, and the storage team doesn't need to maintain so many zones, masks, redundant links, and so on.

Less Power and Cooling Generally speaking, scaling up uses less power per VM, and needs less cooling, than a scale-out approach. Although smaller servers use fewer watts, a server with half the number of cores or RAM won't reduce the power consumption by 50%.

Advantages of Scaling Out

In comparison to larger servers, more servers that are less powerful have the following advantages:

Less Impact during a Host Failure Having fewer VMs per server reduces the risk if a physical host failure should occur. The old adage of not putting all your eggs in one basket is the point here, and this is one of the predominant advantages to scaling out.

Arguably, having twice as many servers, each with half the parts, should mean you get twice the number of failed hosts on average. But the reality is that you'll have fewer outages per VM. Hardware failures are likely to account for relatively few host failures. Server components are so reliable and are backed by so many redundant subsystems that they don't often collapse in a heap. User error is a far more likely cause of host failure these days.

Although scaling out may reduce overall VM outages per year, that's not the main point. The real importance of scaling out is the impact of a single host failure. When a host fails (and they will fail occasionally), fewer VMs will fail at once. One of the greatest worries that companies have about virtualization is that host failures can have a significant effect on services. By scaling out to many more servers, fewer VMs are affected at once.

Although HA is a great recovery mechanism, reducing the time VMs are offline, it doesn't prevent VM outages when a host fails. With fewer VMs per host, HA should recover those VMs far more quickly. In a scale-out situation, if a host fails, HA has fewer VMs to recover and also has more hosts on which to recover them. Generally, the VMs are up and running again much more quickly. After the VMs are brought back up, DRS can load-balance those VMs more effectively than if there is a much smaller pool of hosts.

Less Expensive Host Redundancy Most companies have a policy of host redundancy of at least *n+1*. With a scale-out approach, this is significantly cheaper to accomplish.

Easier Continuation of Scaling over Time When the servers are small, it's more straightforward to add nodes as demanded. The significant cost of scale-up hosts can make it difficult to add another host; but a scale-out host gives you more granularity, meaning you can slowly add hosts as you go. Scale-up clusters are more likely to be replaced wholesale after they reach their useful limit.

More Efficient Bus I/O With an increase in the number of cores and the amount of RAM in each server, the various internal buses come under increasing pressure. Larger servers have

solutions such as NUMA to try to deal with this issue, but that can create significant performance compromises for the hypervisor. Smaller servers have more bandwidth available on these buses for throughput, which can lead to reduced latency for the CPU-to-memory bus. Despite a reduction in possible TPS and CPU coscheduling opportunities, more scaled-out servers may provide better performance in your environment.

Scaling Is a Matter of Perspective

The classic picture of scale-up and scale-out being one to two sockets versus four or eight isn't appropriate in many situations. It's all about the interpretation of what is *large* and what is *small*. For example, compared to six one-socket servers, three two-socket servers is scaling up. And instead of four eight-way servers, using eight four-way servers is scaling out. The same can be said of cores versus sockets or the size of the RAM modules. It's all a matter of perspective.

Whatever a business considers scaling up or out is always tempered somewhat by its VM requirements. It's often not out or up, but what is right for the business. It's unlikely that any company will only opt for two very large servers, because if it's hoping to have *n*+1 redundancy, it effectively needs two hosts that are so large the company could run everything from one. That means buying double the capacity the company needs. Neither is a company likely to decide that 32 hosts per cluster is a good idea, because the company will lose any efficiencies of scale, and such a solution doesn't leave any room for host growth in the cluster.

Sizing hosts appropriately depends on the work you expect. Hosts will vary if you're going to run hundreds of workstation VDI desktops or a handful of very large Tier-1 multi-vCPU workhorses. Generally, large VMs need large hosts, and lots of small VMs work best with scaled-out hosts.

Risk Assessment

The biggest fear with a scaled-up architecture is the large impact created by a single host failure. This risk is largely dependent on the business and the applications running on the hosts. Is such aggressive consolidation worth the associated risk? If the server hosts business-critical VMs, then perhaps reducing the consolidation ratio to limit the risk of an outage is justified. Some companies split their resources so that most VMs run in large, consolidated scaled-up hosts; other, more important VMs run in a cluster that is designed with smaller hosts that have consolidation ratios normally found in a scale-out approach.

You should consider the risk of a failure of a particularly heavily loaded host in the context of its likelihood. Yes, there is always the chance that a host will fail. However, think of some of your organization's other critical infrastructure pieces. For example, many businesses run all of their main datacenter's VMs from a single storage array. That one array is stacked full of redundant parts, but it's still a single infrastructure piece. Unfortunately, servers don't have the same level of redundancy. But if you manage your hosts properly, with good maintenance and change-control procedures, host failures should be very rare.

Fear still drives a lot of companies away from scale-up designs. If an application is that critical to the business, much more than server hardware redundancy and vSphere HA should be in place to protect it. With the correct level of software insurance, only the most extremely risk-averse situations should shy away from scale-up servers for fear of host outages. Many options exist, such as VMware's FT, guest OS-based clustering, and failover written into applications. These extra levels of protection should supplement the most important VMs.

It's important to think about how applications interoperate, as well. There is little point in having clustered VMs run across servers that are scaled up so much that a single server failure will apply enough pressure on the remaining nodes that the application becomes unusable. On the other hand, scaling out servers to minimize the risk to an application won't help if all the VMs need to be online for the application to work. A single host failure, however scaled out, will still bring down the entire system. This is where you can design VM affinity and anti-affinity rules, along with host sizing, to protect your VMs.

If you're less concerned with the potential risk of large hosts, it's worth considering that these days, application owners think less and less about redundancy and failover. This is largely due to the success of virtualization and improvements to hardware and guest OS stability. In the days before mainstream x86 virtualization, application owners thought carefully about hardware redundancy and what would happen if a server were to fail. But with the ease of provisioning new virtual servers, and the belief that vSphere hosts with DRS and HA features are infallible, many application designers assume that their failover needs are taken care of. They don't realize that hosts can still fail—and that when they do, their VMs will go down. This means that more and more, it's up to those designing the vSphere layer to understand the applications that run on it and the level of risk associated with an outage.

Choosing the Right Size

Getting the right scaled hosts is usually a good balance of risk versus cost efficiencies. Scaling up saves money on OPEX and licensing. Larger servers used to be prohibitively expensive, but this is no longer the case. Most costs are fairly linear. Adding more cores is now often cheaper than scaling out; and because servers have increasingly large DIMM banks, there is less need to buy very expensive RAM modules. CAPEX-wise, price your scale-up and scale-out options, because they will depend on your definition of *up* and *out*. You may be surprised to find that scaling up is no longer the more expensive option.

Another issue that used to plague scale-up solutions was rack space, because most four-socket servers were at least 3Us, and often 4Us or 5Us. Today, with such dense core packages on the processors, your scale-up option may be on a 1U server or even a blade.

Look at the VM workload and the number of VMs, and consider what you think the pain points will be. Sometimes different situations require solutions that most would consider lopsided. You may have very high CPU requirements or unusually high memory requirements. The scale-up and scale-out approaches may also differ within an organization. The company's main datacenter probably has very different compute requirements than one of its branch offices. For example, a design that chooses a scale-out approach in the datacenter may want a scale-up for its smaller sites. The fact that the scaled-out servers are larger than the remote office's scaled-up servers is a product of the situation.

When you're considering the desirable consolidation ratio for VMs on each host, it's important to remember that after a design is implemented, the ratio will very likely change naturally. Undoubtedly, more VMs will appear, and it may be some time before the hosts are scaled up or out further to accommodate the extra VMs. Consolidation ratios should be designed with the expectation that they will be stretched.

It's easy for this to become a "religious" debate, and all too often architects have strong opinions one way or the other. These preferences can often cloud the best decision for a business, because every situation is different. Remember the conceptual design behind the functional and nonfunctional requirements, and how that affected the logical design. Chapter 1, "An

Introduction to Designing VMware Environments," delved into the process of deriving the physical design from a logical design first. Each company has its own unique requirements, and only by revisiting the base differentiators can you make an objective, agnostic decision every time. It's important to remember that as hardware capabilities constantly change and evolve, this decision should be continually reviewed. Although you had a preference for scale-out last year, this year you may think differently for the same business and opt for scale-up (or vice versa).

CPU to Memory Design Ratio

One frequently debated area of host design is the ratio of CPU to memory levels. Of course, the correct answer lies in the oft-used reply *it depends*, which is true: it does rely on the VMs that will run on the hosts. As you'll see in the next section, sizing the hosts doesn't need to be a black art. It's a fairly straightforward process to right-size a cluster of hosts for a particular workload. However, it can be instructive to talk in generalities and understand common server workload ratios.

When looking at the ratio of CPUs to memory in a host, the CPUs are best described in terms of cores. Although HT increases the number of logical processors, they don't provide the full power of a separate core. Factors such as HT, speed in GHz, bus speed, and NUMA design affect performance, but the number of cores is the best unit for this primitive comparative analysis.

An understanding of this type of analysis is useful is because it can be applied to existing configurations. Once you compare a few clusters, particularly clusters that are starting to create performance concerns, you begin to get a feel for vCPU consolidation ratios and CPU to memory levels in your environment.

Memory levels in a host are fairly easy to understand. Memory maps through to VMs on a one-to-one basis initially. Memory-reclamation techniques such as page sharing allow for some level of overcommitment, but it's unusual to see anything more than 30% in general server VM clusters. Many enterprises strive to avoid overcommitment, using it instead as a soft limit for their capacity (not that we advocate this as a good method of capacity management). When considering failover capacity in a cluster, this further reduces the chances that the hosts are overcommitted for memory. For planning purposes, a one-to-one mapping of VM memory to host memory isn't uncommon.

CPU sizing is far less tangible. There is an expectation that you can over-allocate CPUs many times over. As opposed to memory, which a guest *uses* all the time, a CPU is sent instructions to process on a more intermittent basis. The level of CPU cores to vCPUs will affect the VMs' performance once the CPUs become overloaded. The old rule of thumb was that around four vCPUs to each core was reasonably safe. As CPU technology has progressed, many users see 6:1 or even 8:1 as acceptable in their environment. Obviously each circumstance is different, and just as CPU power has increased, often so has the applications' demands as SMP tasking becomes more efficient in guests. For mission-critical operations where even small amounts of CPU latency are unacceptable, 2:1 or even 1:1 may be appropriate.

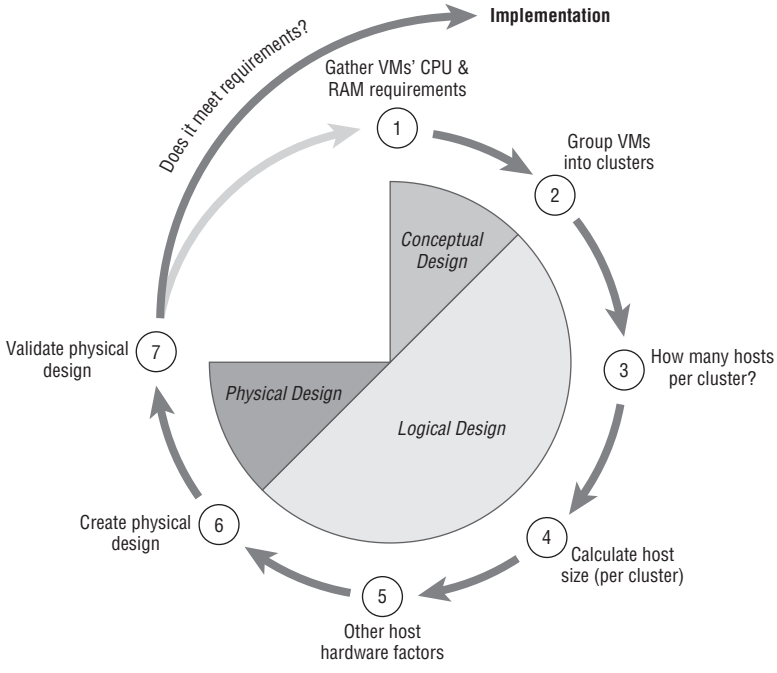
In new configurations, experience can help to make design assumptions about the potentially nebulous vCPUs-to-core ratio and the CPU-to-memory balance. In an existing environment, it's possible to see whether the existing levels of consolidation are within norms. Unfortunately, the vSphere Client's CPU usage graphs for the hosts are somewhat misleading. Look at any heavily used host, and it will almost always appear that the memory usage is high and the CPU is hardly being taxed. But the CPUs could be the performance bottleneck if the vCPUs-to-core

ratio is too high and there are insufficient scheduling opportunities for the VMs. A host's CPU allocation to each VM is affected by such things as the hypervisor's SMP coscheduling, time slicing, and NUMA locality. A heavily consolidated host can equate to VMs that suffer from CPU queuing. Check the host's CPU Load Average and VMs' CPU Ready figures to see if it's being affected by processor latency.

Sizing the Hosts

A possible approach to right-sizing your hosts could be as follows. Figure 4.3 shows an assessment cycle to determine the CPU and memory sizing for hosts and the number of hosts. It follows the process from VM requirements through the logical design phase, finishing in a physical design. If followed properly, the logical design is where the majority of the work is done, meaning the physical design is only a final step.

FIGURE 4.3
Host sizing cycle



1. Gather the VMs' CPU and memory requirements.
2. Group the VMs into clusters.

   As you'll see in Chapter 8, cluster design can revolve around many factors. In an ideal world, clusters would be dictated by aligning similar VMs. Collocating VMs of approximately equal size means cluster functions such as DRS, HA, and Storage DRS work most efficiently. However, being pragmatic, clusters are often split out for other important

technical and business reasons, such as connection to storage arrays, security zones, business units, projects, tenancy, application ownership, business criticality, and DR rules.

3. Decide how many hosts are desirable per cluster.

   The previous sections covered the scale-up versus scale-out arguments. Clusters can only have up to 32 hosts. Consider the level of redundancy you need: the failure domain size the business is comfortable with. Most consider creating multiple general-purpose clusters smaller than five hosts a waste of redundancy resources. It's common to see clusters up to 8 or 12 hosts. Additional redundancy should be factored in. The standard *n*+1 is a common approach but may not be sufficient. On the other hand, some businesses are willing to accept a certain level of performance degradation or allow some test VMs to be offline during short outages.

4. Calculate the host sizes for each cluster.

   Once you know how many hosts you'd like to have in each cluster and the VMs you hope to have in each cluster, then it's fairly simple math to add up the VMs' vCPUs and RAM. You need to make some assumptions about the vCPUs to cores and memory commitment levels (see the previous section). You can then calculate the number of cores and GB of RAM required in each host. Your calculations should take into account the expected growth of the environment during the life-cycle of the hardware.

   You may find that the levels are too high or too low and that hosts don't come in those types of specifications. If that is the case, then you'll need to revise the number of hosts in each cluster or be prepared to split or consolidate the clusters.

5. Consider other server hardware factors.

   Once you have the appropriate CPU and memory requirements for your servers, you can turn your attention to all the other hardware choices available. These include things like PCI slots, form factor (blades half-height, full-height, 1U, 2U, and so on), vendors, and any ancillaries and accoutrements.

6. Create the physical design.

   At this point you can look at the specifics of the physical servers. These include model numbers, parts, memory modules, and PCI cards.

7. Validate the physical design against the logical design points, and accept it or reanalyze.

   Once the hardware has been chosen, you should re-review all the conceptual design requirements and each of the logical design steps to validate that the final decision is appropriate. If need be, another cycle back through the circle can help to resolve outstanding issues and gives you an opportunity to finesse the physical design.

Blade Servers vs. Rack Servers

In addition to the continual debate over scaling up or out, the intertwined argument over blade or rack servers continues unabated. This is also a contentious issue, but it's slightly more quantitative. There are different opinions about what a scale-up or scale-out solution is, but a server

is definitively either a blade or a rack. Each manufacturer has its own twist, but generally a rack server is a stand-alone unit that is horizontally mounted. A blade server uses a chassis or enclosure with certain shared components such as power, backplane, and I/O units. It's common to see blade servers mounted vertically with half- or full-height options.

The blade versus rack discussion used to be far more closely aligned with scale-up and scale-out solutions, because blades were always considered to be a very dense, lower-performing option. But these days, that categorization isn't as applicable. You can find blade servers in four-way configurations with the potential for several hundred gigabytes of memory. Also, some manufacturers are beginning to produce rack servers that can compete with blades in the areas of power efficiency and reduced cabling.

With that caveat understood, you can generalize about blades and take them as a scale-out option, the advantages and limitations of which we covered in the last section. Our discussion here will focus on the differences inherent in the two form factors.

Servers have also long come in tower models. Towers serve a purpose and shouldn't automatically be discounted even by large enterprises. But their large size usually prevents them from being space-efficient enough in all but the smallest deployment. Also, remember that tower servers should be specified just like any other production server with adequate hardware redundancy and quality components. Towers aren't often used in virtualization environments because they tend to be very underpowered; but if a small branch office needs a stand-alone server, and you want to consolidate its small workload on a hypervisor, a tower can prevent the cost and space requirements that come with a half-height rack installation.

Blade Servers

Both blades and racks are viable options with vSphere. Despite a blade's limitations, you can make it work well in a virtualized environment. Although rack servers are very much the predominant force in the server market, blades have always attracted those interested in hypervisors. Often, the same people who can see the distinct advantages of consolidation, energy efficiency, and modular hardware options in vSphere can see obvious synergies with blade servers as their form factor of choice.

However, those who dabbled with blades when they first arrived were often hit by first-generational teething problems: poor redundancy, lack of management tools, BIOS issues, no training, excessive heat, and so on. This has tarnished the view of many server engineers, and some are vehemently opposed to using blades. But in many situations, using blade servers offers real gains.

The Case for Blade Servers

After a blade chassis has been fitted to a rack and cabled in, adding/removing/replacing blades is trivial. It takes only minutes to fit or unrack a server and move it to new chassis. The modular design is one of its key strengths. This can be one of most obvious OPEX cost-reduction effects of using blade servers.

Their combined midplane reduces the amount of cabling, which not only reduces the OPEX but can also reduce the CAPEX resulting from cabling costs in a datacenter. Fewer power cables are required in racks and therefore fewer PDU connectors. The reduction in the number of Ethernet and FC cables going to the next hop switch cuts cabling costs and can also substantially reduce the number of ports on network and storage switching equipment.

Blade chassis can potentially allow for a reduction in rack-space usage. You can typically fit at least 50% more servers into the same area than even 1U rack servers. If you lease your rack space, this can be quite a cost saving. Most of the current generation chassis offer very advanced management tools, which can surpass those found on rack servers; and unlike the rack equivalents, which can be licensed extras, the blade's tools are usually included in the price of the chassis. These advanced management tools can provide remote management (power options and remote consoles), virtual hardware device emulation (mounting remote media), midplane configuration, BIOS management, automated provisioning, and so on.

Traditionally, blade servers have offered power-efficiency savings. By combining several servers' PSUs together, a chassis can reduce the number of PSUs well below the normal minimum of two per rack server and still provide hardware redundancy. Modern blade chassis use incredibly efficient PSUs, to try to reduce heat generation as much as possible. They can automatically step down their power consumption depending on how full the enclosure is.

With the increasing use of 10GbE in servers, storage, and networking equipment, blades become even more viable. The additional bandwidth they can provide means that much higher I/O is possible down to each server. The interconnects on many of the chassis backplanes are 10GbE, with extremely fast interserver traffic possible. This removes one of the biggest drawbacks that blades posed, particularly to virtualized workloads that needed excessive Ethernet cabling.

With the increase in CPU core density, the availability of four-way blades, and large DIMM socket density, it's possible to build very powerful blade servers. Considering that vSphere servers aren't usually built to depend heavily on local storage, blades can make excellent hypervisors. Additionally, often companies don't virtualize everything they have, and it's possible to mix non-ESX servers into chassis to spread the HA and I/O load.

The Case against Blade Servers

One of the biggest constraints with blade servers that deters many companies is the much higher initial entry cost. Before deploying your first blade server, you must buy and fit a full chassis with power supplies, network, and possibly FC switches. This can effectively make the first server very expensive. If you buy several blades at once, this cost can be absorbed. Either you need a flexible budget in which you can offset CAPEX investments for subsequent blades, or you must plan to purchase quite a few blades in the first order to make the cost more palatable. The tipping point for blade servers is usually somewhere around seven or eight per chassis, at which point you can begin to see reasonable unit costs per server. Anything less means each server will seem very expensive.

Server technology churns frequently, with new models and features arriving all the time, so you shouldn't expect to wait several years to fill each chassis—those empty slots may become useless because the new blades may need newer chassis. Make sure the chassis, all its integrated equipment, and the change in infrastructure provide a suitable ROI. The chassis are proprietary to the vendor, so you're locked into those blades and their add-on options after you buy the chassis. You're entirely reliant on the vendor's hardware plans, so in the future you may be limited to the technology roadmap the vendor chooses. If it delays moving to the next CPU architecture, doesn't offer the latest chipset options, or doesn't provide a new I/O protocol or transport, there is little you can do.

Another frequently quoted concern about blades is the chance of an entire chassis failure. Such failures are regarded as very rare, but the thought is enough to dissuade many businesses.

If you've ever experienced a complete chassis outage that takes all the blades down at once, it's likely to make you think twice about purchasing blades again. People dismiss scaling up their vSphere servers to two or three times their usual size, for fear of losing all their VMs in one go. Imaging losing 16 scaled-out servers for a period of time. Although vendors describe chassis failures as extremely unlikely, there are always single points of failure involved; and if that risk is too much, this is a very real barrier to adopting blades.

This possibility of a single point of failure also influences certain aspects of your vSphere design. For example, vSphere 5 clusters still have no understanding of the physical server placement within blade chassis (or racks for that matter). Ensuring that a redundant pair of VMs do not run on the same host can be achieved with a single VM-to-Host anti-affinity rule. However keeping those two VMs in separate chassis isn't as easy to accomplish.

If you've ever stood behind several fully loaded blade chassis, then you know the tremendous amount of heat they can generate. Although high server density is an obvious advantage of using blade servers, you need to be prepared for the very high power and subsequent cooling requirements. Today's datacenters are struggling to keep up with these physical demands, and the power or cooling available to you may be limited. Although blades shouldn't produce any more heat than their rack equivalents, they allow increased concentration. If you expect to take the same amount of space and fill it with blade servers, then you'll need more power and more cooling. Think about the hot spots blades can create and how they affect your hot and cold aisle distribution. You may need additional cooling units and extractors over particular areas to account for uneven heat distribution.

When introducing blades, especially in larger environments, you may find that internal teams need to work more closely than they're accustomed to. If there are separate server, storage, and networking teams, you'll need buy-in from all parties. They may be accustomed to physical separation of equipment, but with blades chassis, the management tools and rack space are shared. You may have to change internal processes for teams that are used to being independent. Often, these teams have disparate purchasing cycles and separate budgeting models, so the financial logistics may need to change. To make this happen, a cultural shift is often required, which can be more difficult to achieve than you expect. Or, as sometimes happens, the server guys may need to come up to speed with network switches and storage equipment very quickly.

One often-overlooked aspect of using blades is training. Rack servers are a known entity, and it's fairly straightforward for an engineer to maintain a new model or even move to a new vendor's equipment. But blade servers are different beasts and need a lot of internal management that may require additional training. Even moving from one blade vendor to another can involve an uphill learning curve, because blades are much more proprietary than rack servers. A deeper understanding of network and storage switches is often needed, and non-server engineers may have to familiarize themselves with new management tools.

Blade chassis used to suffer terribly from a lack of I/O in comparison to their corresponding rack server models. 10GbE has resolved many of these issues; but unless you already have a 10GbE uplink switch to connect to, this can require expensive upgrades to your switching infrastructure. Most vendors should have 10GbE and FC mezzanine cards available, but you may not always find the 16 Gbps FC option, PCIe flash accelerator, PCoIP cards, or even a compatible connector for your requirements. Even when it comes to changing from one standard to another, you normally have to do it en masse and exchange the entire chassis's I/O options at once.

Even with 10GbE, you'll still find I/O limitations with vSphere. You'll almost certainly need to rely on VLANing to separate the traffic to reduce the broadcast domains and provide basic

security segregation. vSphere's network I/O control (NIOC) feature can help provide simple quality of service (QoS) while aggregating the traffic onto a smaller number of cables and maintaining redundant links. But blade servers can never match rack servers with their full PCI slots for I/O expansion options.

Despite the fact that the blades have access to very fast interconnects, chances are that most of the traffic will still exit each blade, go out through the network cards, and hit a physical network switch where the default gateway sits. Localizing traffic in a chassis usually depends on the network cards' functionality and how capable and interested the network team is.

Blades are considerably more powerful than they used to be, with four-socket CPU configurations possible and many more DIMM slots than previously. The increase in core density has also improved the viability of blades, increasing the compute density. However, even though some 2-way blades offer up to 32 DIMM sockets, to really scale out in terms of some of the large 4-way rack servers is very expensive, because you have to use large, costly memory modules. It's difficult to fit enough memory to keep up with the number of cores now available. Even the most powerful blades can't compete with the levels of scalability possible with rack servers. For your most ambitious scale-up architectures and your most demanding Tier-1 applications, you may find that blades don't measure up.

Rack Servers

An often lauded downside of rack servers is that they consume more physical rack space. You literally can't squeeze as many of them into as small a space as you can with dense blade servers. However, if you're not space constrained, this may not be such a disadvantage. Blades tend to increase heat and power usage in certain areas in your datacenter, creating hotspots. Often, virtualization projects free up space as you remove less powerful older servers and consolidate on fewer, smaller, and often more energy-efficient servers that take up less space than their predecessors. If you have plenty of space, why not spread out a little?

One advantage of rack servers is that they can be rolled out in small numbers. Unlike blades, which need several servers to be deployed at a minimum to make them economical, rack servers can be bought individually. In small offices that need one or two vSphere servers, you wouldn't even consider a blade chassis purchase. By choosing rack servers to also use in your larger datacenters, you can standardize across your entire fleet. If you don't need to manage two different form factors, then why would you?

You can also redeploy rack servers to different locations without having to provide an accompanied chassis. They're stand-alone units, so you can more easily redistribute the equipment. Rack servers also provide opportunities for upgrading, because they're much more standardized and usually take standard PCI cards. If you need to add the latest I/O card to support new network or storage technologies, then doing so will always be easier, and probably cheaper, on a set of rack servers.

If you need to be able to scale up, then rack servers will always be the obvious choice. Although blades can scale up to respectable levels these days, they can never match the options with rack servers. Many datacenter backbones are still 1GbE, and it isn't uncommon to see vSphere host servers with the need for two quad NIC cards and a pair of FC HBA ports. In addition to the number of ports, if you have very heavy I/O loads and bandwidth is particularly important, you're likely to opt for rack servers, because blades can't offer the same level of bandwidth across the entire chassis. Most blades still come as two sockets, so even four-way servers are usually rack servers, not blades. If you need to scale up beyond four sockets, rack servers are really the only option.

Blade servers have long been admired for their cable consolidation, but with 10GbE PCI cards, you can consolidate all networking and storage demands down to similarly small numbers on rack servers as well. Even some of the advanced management features that blade chassis enjoy are starting to be pushed back up, allowing the management of rack servers in common profile configurations.

Form-Factor Conclusions

Both blades and rack-mounted servers are practical solutions in a vSphere design. Blades excel in their ability to pack compute power into datacenter space, their cable minimization, and their great management tools. But rack servers are infinitely more expandable and scalable; and they don't rely on the chassis, which makes them ultimately more flexible.

Rack servers still dominate, holding more than 85% of the worldwide server market. Blades can be useful in certain situations, particularly in large datacenters or if you're rolling out a brand-new deployment. Blades are inherently well suited to scale-out architectures, whereas rack servers can happily fit either model. Blades compromise some elements. But if you can accept their limitations and still find them a valuable proposition, they can provide an efficient and effective solution.

A reasonable approach may be to use a mixture of small rack-mounted servers in your smaller offices, have some blade chassis making up the majority of your datacenter needs, and use a few monster rack servers to virtualize your largest Tier-1 VMs. After the blade chassis are filled, the cost per VM works out to be very similar. Neither blades nor racks are usually very different in cost per vCPU and gigabyte of memory. As long as each solution can provide sufficient processing power, memory, I/O, and hardware redundancy, either form factor is acceptable. Much of the decision comes down to personal preference drawn from previous experiences and seeing what works and doesn't work in your environment.

Alternative Hardware Approaches

Before trotting down to your local vendor's corner store with a raft of well-designed servers on your shopping list, you may wish to consider a couple of alternatives: cloud computing and converged hardware. Both approaches are in their relative infancy, but momentum is starting to grow behind each one. Although neither is likely to displace the role of traditional server purchases any time soon, it's worth considering where you might benefit and look at how they could replace elements of your existing design.

Cloud Computing

Much is being made of the term *cloud computing* these days, but confusion often remains about its place in traditional infrastructure models. Cloud computing is really a generic term used to describe computing services provided via the Internet. Rather than buying servers, storage, and networking in-house, you work with an external company that provides a service offering. Such offerings can usually be classified into three common models:

Infrastructure as a Service Infrastructure as a Service (IaaS) provides the basic hardware infrastructure required, so you can install your own OS and applications. This typically includes servers, storage, datacenter space, networking equipment, and bandwidth.

You may be offered two different versions of IaaS. With the first, the hosting company provides you with a dedicated physical server, and you're responsible for the hypervisor, the VMs, the guest OSes, and the applications (similar to the traditional hosted model). With the second type of IaaS, you get only the virtual infrastructure. The hosting company manages the virtualization layer, and you control your VMs and templates on the provided hypervisor or transfer them from your existing infrastructure. This is what most people consider IaaS in the newer cloud-centric viewpoint, and it's the basis that third-party vCloud providers supply.

Platform as a Service Platform as a Service (PaaS) gives you everything up to a working OS and may offer preinstalled frameworks. Most companies offer either basic Linux or Windows platforms. You can log in to your server and install and configure your software. Examples of current PaaS offerings are Microsoft Azure and Google App Engine. VMware has its own Cloud Foundry toolkit that can offer PaaS; but it's flexible in that it can be hosted privately in-house or run on an IaaS offering. The choice of PaaS offering is usually dependent on the choice of developer framework, such as .NET, Java, Node.js (JavaScript), Ruby, or Python.

Software as a Service Software as a Service (SaaS) is the most hands-off approach. The external company runs everything behind the scenes, and your users merely have the ability to log in to the remote software and use it via their web browser. SaaS is particularly popular for CRM, accounting, enterprise resource planning (ERP), HR, and financial software. Examples of SaaS products are Google Docs, Salesforce, and Microsoft Exchange Hosted Services and Office 365. VMware's SlideRocket is its first foray into this bigger market with an online presentation tool.

Clearly IaaS has an obvious correlation to an existing or a newly designed vSphere environment. When you're considering new hardware, it's becoming more feasible to consider external providers to provide that hardware and manage it for you. VMware's own vCloud Director is a product that gives external service providers the tools to more easily provide these IaaS services around the vSphere hypervisor. Chapter 12, "vCloud Design," discusses vCloud Director and its design impacts. The vCloud Connector is a plug-in for vCenter that allows VMs to be moved into and out of publicly hosted cloud offerings. Amazon is the clear market leader with its Xen-based Elastic Compute Cloud (EC2) model. Some of the other prominent players in this IaaS space currently are the large web-hosting companies such as Rackspace and AT&T.

IaaS can be a boon for small business and startups, because minimal technical knowledge is needed to get going. It can be instantly deployed and accessed worldwide. It's completely self service, so users can create instances themselves. IaaS is usually pay-as-you-go, so you only have to pay for the time it's been used—you don't need to justify it through an entire hardware lifecycle. And IaaS is instantly scalable; if you need more compute power, you just add more.

The biggest concern for any company regarding these cloud computing offerings is security. Most businesses are rightfully wary of giving up all their precious data to sit in someone else's datacenter, relying on their backups and entrusting an external company with the keys.

Despite the considerable hype surrounding cloud computing and the ongoing growth in this sector, there will always be a substantial need for in-house vSphere deployments on privately owned servers. Don't worry: despite the pundits' warnings, our jobs are safe for now. However, this is an increasingly interesting market, which gives you more options when designing how to deploy VMs. You may wish to investigate external hardware IaaS solutions prior to making any large investment, to compare costs and see if IaaS is a viable option.

Converged Hardware

Most server vendors are starting to come to market with their versions of a converged solution that combines servers, virtualization, networking, and storage in various ways. This may be a combination of the vendor's own home-grown equipment, because some vendors have expanded or bought their way into diversified markets; or it may be the result of a coalition between hardware partners. The coalitions are often the outcome of *coopetition*, where companies in related fields such as servers, storage, and networking work together in some parts of the market where they don't directly compete but continue to vie in other areas.

For example, here are some of the more popular current examples of these converged market products:

VCE (EMC/Cisco) Vblocks Vblocks are a product of the Virtual Computing Environment (VCE) coalition. VCE involves Cisco and EMC selling their combined products, with Cisco networking gear and servers, and EMC storage, for VMware-specific markets.

Vblocks are predesigned systems that have a fixed hardware makeup. They're sold in tiered units, scaled for particular purposes, and come prebuilt as new equipment, racked and ready to install. They provide an easy, quick way to deploy new environments. You're literally purchasing capacity en masse.

NetApp/Cisco FlexPod FlexPod is NetApp's response to EMC's VCE coalition. Rather than a prebuilt system, FlexPod is a reference architecture using NetApp storage and Cisco network and server equipment. There are FlexPod architectures for vSphere, but also for Citrix, Microsoft, and RHEL among other nonvirtualized application specific roles.

Unlike Vblocks and other more traditional rigid converged infrastructure models, FlexPod is something you need to design and build yourself. Doing so in accordance with the FlexPod reference architecture produces a combined, supportable solution. It's more customizable; and because it isn't sold as a unit, it gives you the opportunity to reuse existing equipment. This makes the FlexPod solution more flexible but also more complex to build.

Cisco UCS Servers Cisco's Unified Computing Systems (UCS), not to be confused with its Unified Communications System, is a play from the world's largest networking hardware provider to expand into the server market. The company's blade enclosures move a lot of the management from the blade chassis up to a top-of-the-rack management switch. Cisco has also expanded into rack servers.

HP HP as a server vendor has long been expanding into several lateral markets. It has its own storage lines, such as Modular Smart Arrays (MSAs) and Enterprise Virtual Array (EVAs), and it recently bought out LeftHand's virtualized storage and 3PAR. HP also has ProCurve network switches and has bought several high-profile networking manufactures such as 3Com. HP has a very complete offering in the converged market space, including several products; its Converged Systems line is its prepacked converged infrastructure offering.

Oracle When Oracle bought out Sun Microsystems, it became another big player among the converged equipment providers. It owns a large stack from servers to storage, a Unix OS (Solaris), and virtualization products based on Xen and VirtualBox. In addition to its sizable existing middleware and database software products, Oracle added Java and MySQL. Oracle's recent acquisition of Xsigo Systems introduces more technologies to the company's Exalogic toolkit.

Dell Dell obviously sells servers, but it also sells its own LSI-based storage along with the popular Storage Center (from the Compellent acquisition) and EqualLogic SANs. Dell has added vStart as its own converged infrastructure stack options in addition to adding file-based solutions to its storage portfolio.

IBM IBM has long sold servers and its own LSI-based storage devices. It also rebrands storage from NetApp and resells networking equipment. The PureFlex System is IBM's converged solution.

HDS At the time of writing, HDS, known for its SAN equipment, has announced its new converged system called the Unified Compute Platform (UCP). UCP Pro is an integrated and prepackaged offering, whereas UCP Select is HDS's reference architecture to provide more flexibility.

These are just some of the more popular and well-known examples of continuing converged solution sets from vendors. Buying equipment this way has the advantage of being generally a more simplified total solution that needs less design and that provides end-to-end accountability with components that are certified to work together.

A converged solution reduces the amount of homework you need to do to get something on the ground if you need to produce a more immediate solution with less risk. However, by purchasing your equipment this way, you risk the dreaded vendor lock-in, making it harder to switch out to a competitor when you want to. You have to accept the vendor's opinion regarding the correct balance of equipment. If your needs are less typical, then a preconfigured arrangement may not fit your workload. When buying converged packages, you'll always have an element of compromise: you reduce the opportunity to get the best of breed for your environment in each area. Such solutions are useful for new greenfield deployments or companies that need to scale up with a new solution very quickly.

At the end of the day, converged equipment is just another way for vendors to sell you more *stuff*. It can simplify the procurement process; and if you're already tied to a vendor, or you have a particularly good relationship with one (that is, they offer you a substantial discount over the others), it can make sense to explore these options as alternatives to server hardware in isolation.

Summary

Server hardware defines the capabilities, the performance, and a large proportion of the CAPEX of a vSphere implementation. It's crucial that you don't rush into the purchase, but first consider all the elements of your design. CPU, memory, and I/O requirements are fundamental to hypervisor servers, but more options exist and need to be thought out. Don't try to make the design fit the servers, but create the logical design around your needs and then figure out exactly what physical hardware is required. Some architects have very fixed opinions about scaling up or out, or blade versus rack servers, but each situation is different and demands its own solution.

If possible, standardize the hardware across your fleet. You may need to provide a tiering of two or three levels to accommodate more or less demanding roles. You may even be able to stick to one model while scaling up or down with more or fewer cores or memory modules. Try to select common I/O cards across the board and think of other ways you can simplify deployment, configuration, or management to reduce OPEX.

When comparing vendors or models, get some sample equipment and test it during any pilot phases. Most vendors will be only too happy to lend such equipment, and often you can learn a lot by spending hands-on time with it. Make sure you double-check the HCL for all server equipment you order.

Despite any preconceptions you may have with regard to hardware, try to think about your overall strategy and whether you want to generally scale up or out. Choose the approach you want for the size of servers before you even think about whether you want rack or blade servers. Your hardware needs can then influence your form-factor decision. Trying to fit your design to a prechosen vendor, form factor, or server capacity will only result in a compromise. Design first, and then choose your servers.

Chapter 5

Designing Your Network

Have you ever tried to make a cell-phone call in an elevator? Were you shocked when the call dropped in the middle? Some people get annoyed when that happens. But others plan ahead. They know the call will drop due to the lack of reception, so they end the call before they step into the elevator and continue it when they get out. That is called *proper planning*.

We won't go into the reasons why cell-phone coverage is bad in general or why you have to hold your phone a certain way (sorry, iPhone) when you talk. What we'll examine in this chapter are the factors you need to take into account when designing the infrastructure for virtualization. We guarantee that if your servers go down due to a network outage resulting from bad planning, more than one person will be annoyed!

The topics we'll discuss in this chapter include

- Redundancy (at all levels)
- Security (this time, it's moved higher in the list)
- Networking considerations for the different vSphere components (HA, IP storage, FT, and so on)
- Sizing: which NICs to use for which purpose
- Virtual switches
- Naming conventions
- Design scenarios

Examining Key Network Components

As we've done in other chapters, we'll begin by first reviewing the major components in any network design. Although you're probably familiar with many, if not all, of these components, reviewing them establishes a baseline knowledge level on which we'll build throughout the rest of the chapter.

Ready? Let's start with the most visible (both literally and figuratively) component, the physical connectivity.

Physical Connectivity

Naturally, a network isn't a network without all the Layer 1 stuff (the first layer in the Open Systems Interconnect [OSI] model, the physical layer). A number of items are important in this layer:

Cabling Will you use Gigabit Ethernet or 10 Gigabit Ethernet? Depending on the speed, you'll need different kinds of cabling, and the cabling you use will introduce different limitations or constraints on the design. For example, if you use Twinax cabling for your 10Gb Ethernet design, then you'll need to work around some cable length limitations. If you use 10GBase-T for your 10Gb Ethernet design, then you must ensure that all your network components—network interface cards (NICs) as well as switches—support 10GBase-T, and that your unshielded twisted pair (UTP) cabling is Category 6A (Cat-6A) cabling. In addition, Cat-6A cables have very stringent requirements on the number of twists per inch, which might make it practically impossible for datacenter operators to make their own cables (a common practice in some datacenters). Cat-6A cables also have limitations on the bend radius, meaning the cables can't be bent too sharply. We mention this because sometimes it's easy to overlook such details in the grander scheme of network design—but these important points must be considered.

Network Switches Consider the number, type, configuration, and features of the network switches that are currently in place or will be added as part of your vSphere design. As we'll discuss in greater detail later, the network switches will have a significant impact on your overall network design. We highly recommend that you get your networking team involved—if there is a networking team present—and keep them involved throughout the design process.

Network Topology The topology of the network—how the hosts and switches are connected via the cabling—is also very important and must be considered in the vSphere network design. Certain network topologies lend themselves very well to some traffic patterns/types but not other traffic patterns/types, so you'll want to understand the topology and the impact that topology has on the traffic moving among the various components in your vSphere design.

Traffic patterns and traffic types are also components in the network design. Let's take a closer look at some of the network traffic types you can expect to see in a typical vSphere deployment.

Network Traffic Types

Unlike a nonvirtualized datacenter, virtualization introduces some new and additional types of traffic to consider when creating a network design. In this section, we'll examine the three major traffic types:

- Management traffic
- VMkernel traffic
- Virtual machine (VM) traffic

Let's start with management traffic.

Management Traffic

The management network is the lifeline into your ESXi host. If that lifeline goes down, you can't remotely manage that host. The host will think it has lost connectivity and—depending on the configuration of vSphere HA—may power down all the VMs it has running. The other hosts in the cluster may think the host is down and try to restart the VMs that were running. In a nutshell, bad things happen. Clearly, management traffic is pretty important, and later in this chapter (in the "Availability" section) we'll discuss ways to ensure that you don't lose management connectivity to your ESXi hosts.

WARNING If your management network becomes isolated, you lose management of the ESXi host, and rightly so. The other hosts in the cluster think so as well, which causes a failover or isolation response event.

VMkernel Traffic

Both management traffic and VMkernel traffic are new to a virtualized environment; that is, there is no nonvirtualized equivalent to these types of traffic. We use the term *VMkernel traffic* to refer to traffic generated by the ESXi hypervisor and not by VMs hosted by the hypervisor. Technically, this includes management traffic as well; however, given the importance of management traffic we felt it deserved its own section. In this section, we're referring to the other kinds of traffic generated by the hypervisor: vMotion, fault tolerance (FT), and IP-based storage.

Although all these traffic types are generated by the hypervisor, each has its own characteristics and considerations. For example, providing redundancy for vMotion traffic is handled differently than for IP-based storage. In fact, providing redundancy for iSCSI is different than providing redundancy for NFS, so there are key differences even within these different traffic types. As you integrate or incorporate these traffic types into your vSphere design, each will have a different impact on the design for which you'll need to account.

Now let's turn our attention to the third and final traffic type: traffic generated by VMs.

Virtual Machine Traffic

In addition to traffic generated by the hypervisor, you need to account for traffic generated by the VMs hosted by the hypervisor. Depending on the workloads that are deployed in the vSphere environment, this could be almost any type of traffic—from remote desktop protocols (like RDP, HDX, PCoIP) to file transfer (FTP, SFTP, SMB, NFS) to application-specific protocols (MS SQL Server, Oracle, SAP, HTTP, email, calendaring).

One thing that can sometimes trip up vSphere architects is differentiating between hypervisor-generated NFS/iSCSI traffic and VM-generated NFS/iSCSI traffic. It's important to note that steps taken to provide security and redundancy for hypervisor-generated IP-based storage traffic won't necessarily apply to VM-generated IP-based storage traffic. These traffic types originate from different sources and therefore typically need to be handled separately.

The last network component we'll discuss before exploring the factors influencing the network design is the software component.

Software Components

In addition to physical components discussed earlier—like cabling and physical switches—some significant software components make up a vSphere network design. The software components that probably jump to mind immediately are VMware's software-based switches (*softswitches*) in the hypervisor, but you'll also want to ensure that you consider some other important software components:

- Potential third-party softswitches, like the Cisco Nexus 1000v or the IBM DVS 5000v
- Virtual NICs (vNICs) in the guest OS instances
- Virtual firewalls, like vShield App, vShield Edge, or Cisco ASA 1000v
- Virtual load balancers or network address translators, like vShield Edge

Not all of these components will be present in every design, but where they're present or are needed—as dictated by the functional requirements—you'll need to account for the impact of their presence in your design. Later in this chapter, in the section "vSwitches and Distributed vSwitches," we'll discuss some design considerations around the use of the VMware-supplied softswitches. In Chapter 12, "vCloud Design," we'll provide some design considerations particular to vCloud Director, which typically will include vShield Edge as a key network design component.

In the next section, we transition from examining the network components to exploring some of the various factors that will influence your network design.

Exploring Factors Influencing the Network Design

In the previous section, we reviewed some of the key components of a vSphere network design. In this section, we'll explore some of the major factors influencing how you, the vSphere architect, should assemble these components to create the network design. In the next section ("Crafting the Network Design"), we'll discuss specific ways in which you can build a vSphere network design according to the five principles of design: availability, manageability, performance, recoverability, and security (AMPRS).

The factors influencing the network design that we'll discuss in this section include

- Physical switch support
- vSwitches and distributed vSwitches
- 10 Gigabit (Gb) Ethernet
- Single Root I/O Virtualization (SR-IOV) and DirectPath I/O
- Server architecture

This isn't a comprehensive list of all the potential factors that will influence your network design, but it should be enough to get you thinking about how various items can affect and/or influence the network design. Let's start with physical switch support.

Physical Switch Support

The features, capabilities, and protocols supported by the physical switches you incorporate into your design will, quite expectedly, play a significant role in shaping your vSphere network design. Some of the more significant areas of support about which you should be aware include

- Link aggregation
- Private VLANs (PVLANs)
- Jumbo frames

In addition, we'll discuss the impact of 10Gb Ethernet in the section "10Gb Ethernet").

A Look Ahead

In addition to evaluating physical switch support for link aggregation, private VLANs, jumbo frames, 10Gb Ethernet, and Fibre Channel over Ethernet (FCoE), you'll also want to keep an eye on physical switch support for additional technologies that are likely to impact your datacenter in the near future. For example, at the time of this writing no physical switches supported VMware's network encapsulation protocol Virtual Extensible LAN (VXLAN). However, as VXLAN adoption grows, incorporating switches that have hardware VXLAN Tunnel End Point (VTEP) support might be beneficial. The same goes for other datacenter-related protocols like TRansparent Interconnection of Lots of Links (TRILL) and Edge Virtual Bridging/Virtual Ethernet Port Aggregator (EVB/VEPA). Where possible, try to keep a forward-looking approach when selecting or recommending switches to help preserve the investment and position your datacenter for future developments.

Let's start with a more in-depth look at link aggregation.

Link Aggregation

The vast majority of managed datacenter switches support link aggregation, so you might be wondering why link aggregation has its own section. Well, there are some key considerations around link aggregation that you'll want to address in your design. For example, you might want to ask yourself these sorts of questions about the link-aggregation support in your physical switches:

- Do the switches support any form of multiswitch link aggregation?
- What types of link aggregation does the switch support?
- What link-aggregation protocols are supported, if any?
- What load-balancing mechanisms are supported for placing traffic on the links in the aggregate?

The answers to these questions can have a dramatic impact on your vSphere network design. To understand why, however, we first need to ensure that everyone understands the basics of link aggregation.

Link aggregation is a mechanism whereby two devices on a network—these could be two switches, or an end-host and a switch—agree to treat multiple physical links as if they were a single logical link. For example, you could configure two switches in the network to communicate with each other over four physical links, and use link aggregation to have the switches treat those four physical links as a single logical link. Similarly, you could configure an end-host (like an ESXi host) to communicate with a switch over four physical links, and the two systems could agree to treat those four physical links as a single logical link.

What does this buy you? Well, for starters, it allows the two devices to potentially use multiple physical links without worrying about Spanning Tree Protocol (STP) blocking some of the links to prevent bridging loops. (Note that link aggregation isn't the only way to work around this potential STP issue.) Using link aggregation also helps provide redundancy; the logical link remains up as long as at least one physical link in the link aggregate remains up.

Link aggregation can be a pretty handy technology, but it does have its limitations and considerations. Let's take a look at some of these:

Multiswitch Link Aggregation Multiswitch link aggregation (MLAG) support is a big factor in network design. Normally, link aggregation is used between exactly two devices (two switches, or a switch and an end-host). This doesn't allow for redundancy of the physical devices, so in a scenario with an ESXi host using link aggregation the topology would look something like Figure 5.1.

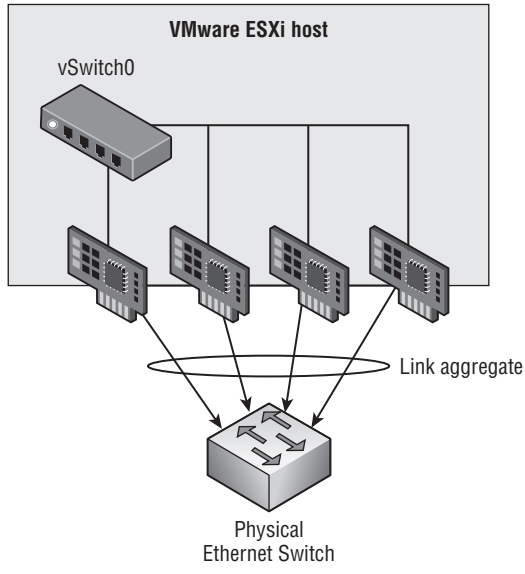


FIGURE 5.1 Link aggregation without MLAG can't offer full redundancy.

Even though you have redundancy at the link level, what you don't have in Figure 5.1—when the physical switches don't support MLAG—is switch redundancy. That upstream switch is now a single point of failure (SPoF), and that's not good. However, with MLAG support in the upstream physical switches, you could instead build a topology like that shown in Figure 5.2.

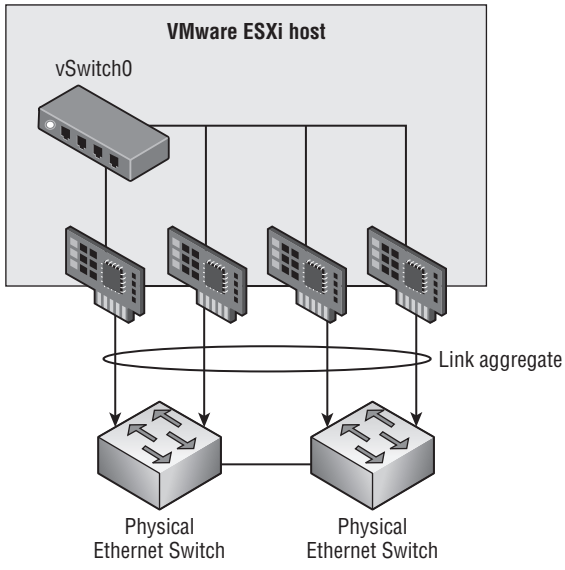


FIGURE 5.2
MLAG support offers both link redundancy and switch redundancy.

Much better! Now you have redundancy both at the link level and at the switch level. For this reason, if you're planning to use link aggregation in your network design, we highly recommend ensuring that your physical switches support MLAG.

MLAG Support in vSphere

As of the writing of this book, VMware's softswitches don't have any form of MLAG support. Fortunately, most upstream switches provide MLAG support in a way that is transparent to the hypervisor softswitches. For example, some stackable switches support MLAG, and technologies like Cisco's Virtual Port Channel (vPC) don't rely on any support in the hypervisor (or other end-host) in order to work.

Link-Aggregation Protocol Support Early forms of link aggregation utilized nonstandard and proprietary ways of managing the physical links as a single logical link. Later, as link aggregation became more prevalent, a standard protocol, known as Link Aggregation Control Protocol (LACP), was defined and adopted by the vast majority of network vendors. Sometimes referred to as 802.3ad, LACP is now defined by the IEEE 802.1ax standard.

However, prior to vSphere 5.1, vSphere didn't support any link-aggregation protocol, and so link aggregation had to be defined manually. This created some challenges at times and made troubleshooting more difficult. With the introduction of vSphere 5.1, VMware has added LACP support. (At this time, LACP is only supported with the vSphere Distributed Switch.)

If you're planning to use link aggregation in your design, we recommend selecting physical switches that provide full support for LACP. Even if you aren't going to use LACP right

away—perhaps because you aren't deploying vSphere 5.1 just yet—the physical infrastructure will be ready when you're ready.

Link Aggregation Load-Balancing Mechanisms Earlier we stated that one of the benefits of using link aggregation is that it allows devices to *potentially* use more than one link in the aggregate. Why only potentially? Devices that support link aggregation use one or more algorithms to determine how to place traffic on the physical links in the aggregate. VMware only supports a single algorithm—it hashes the source and destination IP address to determine which physical link should be used. Physical switches might support a number of different algorithms, but when used with vSphere they generally should be configured to use the same algorithm (a hash of source and destination IP address). This means you'll want to ensure that your physical switches support IP hashing as their link aggregation load-balancing mechanism (most do) and that the physical switches are configured to use IP hashing as their load-balancing mechanism (not the default setting on some switches).

One other side effect of these load-balancing mechanisms is important to note: because the algorithm performs some sort of mathematical calculation to determine which link to use (like hashing the source and destination IP address), two results occur.

First, traffic between two endpoints will *never use more than a single link*. If VM A is communicating with physical Server B, then there are only two endpoints—VM A and Server B—and the mathematical hashing will always produce the same result, causing the switches to select the same link for every single packet. The end result is that only a single link will ever be used for the communication between these two endpoints.

Second (and this is an offshoot of the first point), traffic between two endpoints will *always use the same link*. If the mathematical calculations for the traffic between VM A and Server B result in the selection of link #1 in the aggregate, then every packet between those two endpoints will always use link #1.

The results of the load-balancing mechanisms have some significant impacts on certain traffic types, especially IP-based storage. We'll discuss that in more detail later in this chapter in the "Availability" section of "Crafting the Network Design."

Whereas link aggregation serves to combine physical links together as logical links, sometimes vSphere architects need to provide a greater degree of separation between certain types of traffic. This is where VLANs and PVLANs can be helpful.

VLANs and Private VLANs

You're probably already familiar with the concept of VLANs, which allows users to logically subdivide physical network segments into separate broadcast domains. Although it isn't required, VLANs are typically associated with an IP subnet so that each VLAN = an IP subnet. For systems in different VLANs to communicate, they must pass their traffic through a Layer 3 device (a router).

By and large, VLANs are pretty well understood, and therefore the potential impacts of using VLANs in a vSphere environment are also pretty well understood. However, we want to point out a few things that are important to note:

- Many physical switches must be specifically configured to treat a link as a VLAN trunk (a link that will pass VLAN tags to connected systems). If you don't configure the links going to your vSphere hosts as VLAN trunks, then VLANs just won't work.

- Most VLAN implementations offer a single VLAN that doesn't carry any VLAN tags (even across a VLAN trunk). Depending on the vendor, this VLAN might be referred to as the *native* VLAN or the *untagged* VLAN. Regardless of what it's called, any vSphere port groups that should receive traffic for this particular VLAN shouldn't have a VLAN ID specified. It makes sense, if you stop to think about it (it *is* the untagged VLAN, after all).
- Building on the idea of the native VLAN (also called the *untagged* VLAN), it's also important to note that many switches can have different native VLANs on each port. This could create unpredictable results, so be sure to consistently assign the native/untagged VLAN across all switch ports in order to get consistent connectivity results.
- Different switch implementations support different ranges for VLAN IDs. Although the VLAN specification calls for VLAN IDs all the way up to 4094, some switches might not support that broad a range. Be sure your physical switches will provide the necessary VLAN ID support end-to-end, or your design's connectivity could be compromised.
- Because VLANs require a Layer 3 router to pass traffic from one VLAN to another, be sure to keep this in mind when evaluating availability and performance. (For example—have you provided any form of redundancy for the Layer 3 router that connects your VLANs? What if it goes down? What will happen to your vSphere environment?)

Although VLANs are fairly widely deployed, a related technology isn't quite so pervasive. Private VLANs (PVLANs) are related to VLANs but share some unique advantages over "regular" VLANs. For example, VLANs are typically associated with an IP subnet, and there is no easy way to restrict communications in a given VLAN/subnet.

PVLANs, on the other hand, enable users to restrict communication between VMs on the same VLAN or network segment, significantly reducing the number of subnets needed for certain network configurations. PVLANs add a further segmentation of a logical broadcast domain to create private groups. *Private* in this case means that hosts in the same PVLAN can't be seen by the others, except those selected in a promiscuous PVLAN. (We haven't introduced the term *promiscuous* yet; hang tight for a second and we'll explain what that means in the context of a private VLAN.)

A PVLAN is divided into these two groups:

Primary PVLAN The original VLAN that is being divided into smaller groups is called the primary. All the secondary PVLANs exist only in that primary.

Secondary PVLANs The secondary PVLANs exist only in the primary. Each secondary PVLAN has a specific VLAN ID associated with it. Each packet traveling through it is tagged with a VLAN ID as if it were a normal VLAN. The physical switch associates the behavior (isolated, community, or promiscuous) depending on the VLAN ID found in each packet.

Secondary PVLANs are further divided into these three groups:

Promiscuous We mentioned this term earlier; now it's time to explain what it means. A node attached to a port in a promiscuous secondary PVLAN may send and receive packets to any node in any other secondary PVLAN associated with the same primary. Routers are typically attached to promiscuous ports so all hosts in any of the secondary PVLANs can communicate through the router to other devices in other VLANs.

Isolated A node attached to a port in an isolated secondary PVLAN may only send packets to and receive packets from the promiscuous PVLAN. It may not communicate with other ports in the same isolated secondary PVLAN or with other ports in a

community PVLAN, thus providing the additional segmentation functionality we mentioned earlier. (If you're trying to think of a good use case here, think of a demilitarized zone [DMZ] where hosts in the DMZ should be able to communicate with the firewall but not with any other hosts in the DMZ.)

Community A node attached to a port in a community secondary PVLAN may send packets to and receive packets from other ports in the same secondary PVLAN, as well as send to and receive from the promiscuous PVLAN. It may not communicate with nodes in other community secondary PVLANs, nor may it communicate with nodes in an isolated secondary PVLAN.

Figure 5.3 shows the connectivity among these three groups.

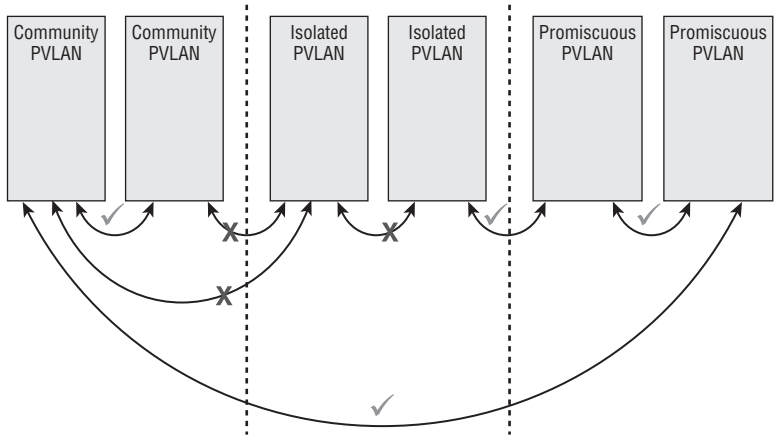


FIGURE 5.3 Systems in a PVLAN can only communicate with certain other nodes.

Although we haven't explicitly stated it yet, if you want to use PVLANs in your environment, you'll need physical switch support. Not all physical switches support PVLANs, so be sure to verify PVLAN support when selecting the switches for your vSphere design. Without physical switches that support PVLANs, traffic between PVLAN-configured ESXi hosts simply won't work. Another feature that is dependent on physical switch support, like PVLANs, is jumbo frames.

JUMBO FRAMES

The term *jumbo frames* means that the size of the largest Ethernet frame passed between one device and another on the Ethernet network is larger than the default of 1500 bytes. Jumbo frames are often set to 9000 bytes, the maximum available for a variety of Ethernet equipment, although this is by no means a ratified standard across vendors and network hosts. ESXi supports jumbo frames out of the box, but jumbo frames aren't enabled by default. The same is also true for many physical network switches: many of them support jumbo frames, but jumbo frames aren't configured by default.

The potential benefit of jumbo frames is that larger frames represent less overhead on the wire and less processing required at each end to segment and reconstruct Ethernet frames into the TCP/IP packets used by network protocols. Even though enhancements in the ESXi network

stack keep reducing the CPU cycles needed to deal with jumbo frames, they're still often configured to extract even the slightest improvement in performance. You should always test your specific workloads and environment to see if the use of jumbo frames will give you added value and performance.

You must do two important things when working with jumbo frames:

Configure End-to-End Configuration has to be done on the ESXi host *and* the network switch *and* any other network devices (like a storage controller) in order for jumbo frames to work. If even one component isn't configured, the jumbo-frames configuration will cause more harm than good. Note that you need more than just support for jumbo frames—the devices need to be specifically configured to use jumbo frames.

The process for configuring a device for jumbo frames will vary from device to device and manufacturer to manufacturer, so we won't try to provide any instructions here. Refer to your vendor's documentation for complete details on how to configure a device to use jumbo frames.

Define ESXi Settings on All Components As we mentioned, it's critical that you configure jumbo frames support end-to-end. With regard to ESXi specifically, this means you'll need to configure the vSwitch (or distributed vSwitch), the VMkernel port (if applicable), and any VM NICs (if applicable). If you miss part of the sequence, it won't work.

The process for configuring the virtual NIC in a VM will vary from guest OS to guest OS, so we won't include that information here. Refer to the documentation for the guest OS installed in the VM for complete details on how to configure that OS for jumbo frame support.

We will, however, discuss how to configure jumbo frames on the hypervisor side. Prior to vSphere 5.0, setting a maximum transmission unit (MTU) of 9000 on a vSwitch or VMkernel interface must be done at the command line (either via the physical console or through the vSphere Management Assistant [vMA]). vSphere 4.1 offered a GUI for setting the MTU on a DVS, but not a vSwitch.

You must do the configuration both at the vSwitch level and at the port group/VMkernel. This example—which uses the vSphere command-line interface (vCLI) installed on a Windows-based system—creates a vSwitch named vSwitch1 and sets the MTU to 9000:

```
vicfg-vswitch.pl --config c:\users\administrator\vcli.txt \
--vihost esxi51-01.design.local -a vSwitch1 -m 9000
```

Prior to vSphere 5.0, when you create a VMkernel interface in the ESXi GUI, it's automatically created with an MTU of 1500. This can't be changed (you can see this by running `vicfg-vmknic` using the vCLI or vMA). So, even though the vSwitch (or DVS) supports jumbo frames, that particular VMkernel interface—which you might be using for IP-based storage, vMotion, FT, or even management (though there is really no benefit for management traffic)—won't support jumbo frames.

To fix this, you have to remove the VMkernel interface and re-create it at the command line with the correct MTU. This example shows how it would be done using the vMA, assuming a port group named VMkernel:

```
vicfg-vmknic --config /home/admin/vcli.txt -a -i 192.168.1.9 -n 255.255.255.0 \
VMKernel -m 9000
```

NOTE For more information on the vCLI and vMA, refer to sections with the same names in Chapter 3, "The Management Layer."

With the release of vSphere 5.0, VMware simply added a GUI option for specifying the MTU. This GUI option is present for vSwitches, dvSwitches, and VMkernel ports. You can't specify the MTU when creating these objects (it's not included in the wizard), but you can easily edit the MTU after the fact to enable jumbo frames.

We've discussed the physical switches and their support in the role of the network design, but it's time to shift gears and focus on the virtual side of the network with a discussion of vSwitches and distributed vSwitches. This is the topic of the next section.

vSwitches and Distributed vSwitches

Before virtualization, the access-layer switches were the "last mile" of the network. Now that virtualization has (rightfully) entrenched itself in the datacenter, the last mile has moved into our servers. VMware's softswitches are now the last hop before hitting servers, applications, and workloads. VMware has two types of softswitches:

- The vSphere standard switch, also known as the vSwitch
- The vSphere distributed switch, more commonly known as the distributed vSwitch or the dvSwitch

A Note on Terminology

VMware has gone through a couple different naming schemes for the virtual switches. In the vSphere 4.*x* releases, they were called the vNetwork Standard Switch and the vNetwork Distributed Switch. In the vSphere 5.0 release, VMware renamed them the vSphere Standard Switch and the vSphere Distributed Switch. In this book, we'll use vSwitch to refer to a vSphere Standard Switch and dvSwitch, distributed vSwitch, or VDS to refer to a vSphere Distributed Switch. Our use of *dvSwitch*, in this chapter in particular, is consistent with VMware's use of the term *dvPort group* to refer to a port group on a distributed vSwitch.

Each type of virtual switch has its own strengths and weaknesses. One key factor in favor of the vSwitch is that vSwitches are available in every ESXi version from the vSphere hypervisor (free ESXi) all the way up to vSphere Enterprise Plus. dvSwitches, on the other hand, are only available on vSphere Enterprise Plus.

That distinction aside, let's start by comparing vSwitches to dvSwitches. The following features are available on both types of virtual switches:

- Can forward L2 frames
- Can segment traffic into VLANs
- Can use and understand 802.1q VLAN encapsulation
- Can have more than one uplink (NIC teaming)
- Can have traffic shaping for the outbound (TX egress) traffic

The following limitations or restrictions are also applicable to both types of virtual switches:

- Won't forward packets received on one uplink out another uplink (prevents bridging loops)
- Only supports IP hashing algorithms for link aggregation

The following features are available only on a dvSwitch:

- Can shape inbound (RX ingress) traffic
- Has a central unified management interface through vCenter
- Supports PVLANs
- Supports LACP for dynamic link aggregation configuration
- Supports load-based NIC teaming
- Uses persistent network statistics (sometimes referred to as network vMotion)
- As of vSphere 5.1, the ability to import/export VDS configuration

In this section, we're not going to dive deep into all these features; instead, we'll focus on how the differences between vSwitches and dvSwitches will impact your vSphere design. Specifically, we'll focus our discussion around two key topics:

- Centralized management
- vCenter Server as the control plane

Centralized Management

The larger your environment grows, the more dynamic it becomes, and the harder it gets to manage the network configuration and keep it consistent across all the hosts in your cluster. Consistency across the network configuration is important because a lack of consistency makes troubleshooting difficult and can introduce unexpected results. For example, a mismatched VLAN configuration between two hosts can result in VMs "dropping off the network" after a migration. As vSphere environments continue to scale, this challenge won't go away.

The dvSwitch helps address this challenge by treating the network as an aggregated resource. Individual, host-level vSwitches are abstracted into a single large dvSwitch that spans multiple hosts at the datacenter level. Port groups become distributed virtual port groups (dvPort groups) that span multiple hosts and ensure configuration consistency for VMs and virtual ports necessary for such functions as vMotion and network storage. The end result is reduced management overhead and improved consistency for network configuration.

Whereas the control plane for a vSwitch was found in an ESXi host, the control plane for a dvSwitch resides in vCenter Server. Moving the control plane into this one centralized place is what enables dvSwitches to provide the centralized management we've been discussing so far. (Although the control plane is centralized in vCenter Server, the switching plane/data plane is still locally situated in each ESXi host.) However, centralizing the control plane in vCenter Server is not without some concerns—and we'll discuss these concerns in the next section.

vCenter Server as the Control Plane

Putting the control plane for the dvSwitch into vCenter Server impacts your design in a number of ways. Let's talk about some of the potential ramifications you'll want to be sure to consider:

vCenter Server Availability What happens if you need to modify or update your dvSwitch, but vCenter Server isn't available? Now that vCenter Server is the sole point of management for the dvSwitch, making sure that vCenter Server is highly available becomes even more important.

Virtual vCenter Server "Chicken-and-Egg" Scenario What if you need to modify the dvSwitch while vCenter Server is down, but in order to get vCenter Server back up you need to modify the dvSwitch? This "chicken-and-egg" scenario is one common reason that vSphere architects choose to use a hybrid vSwitch-dvSwitch approach, with management traffic residing on a vSwitch.

VMware significantly wounded this potential issue in vSphere 5.1 with the introduction of the automated rollback and recovery feature. With this new feature, if management traffic is interrupted, then vSphere will roll back the last change in order to try to restore management traffic. Although this doesn't necessarily address all scenarios, it does address many scenarios in which architects might have been worried about running a virtual vCenter Server on top of a dvSwitch.

NOTE We also discussed some of these ramifications in Chapter 3. Refer to the section "Examining Key Management Layer Design Decisions" for more information.

Aside from these considerations, the primary determinant of whether you should use a vSwitch or a dvSwitch really comes back to—you guessed it—the functional requirements. Does your design need a feature or function that is only supported by a dvSwitch? Then incorporate the dvSwitch into your design, accounting for the design impacts along the way. The same goes for using only vSwitches, using third-party virtual switches, or any combination of these.

In addition to virtual switches, another factor that will affect your network design is the use of IP-based storage, as well as the type of IP-based storage. The next section explores this factor in greater detail.

IP-Based Storage

For the most part, the type of network traffic isn't nearly as important as the volume (in megabits or gigabits per second), performance (in latency), or security requirements. As the saying goes, "There's an exception for every rule," and it's true here as well. IP-based storage—specifically, the use of NFS and iSCSI by ESXi—has some specific considerations that could impact your network. Recall from our earlier discussion that NFS and iSCSI from the hypervisor are different than NFS and iSCSI from a guest VM, and that statement is applicable here as well. In this section, we'll focus specifically on IP-based storage being used by the hypervisor.

Although both NFS and iSCSI fall into the category of IP-based storage, the way in which these protocols operate is very different, so we need to discuss them differently. Let's start with iSCSI.

iSCSI

One of the key differences between iSCSI and NFS lies in how iSCSI and NFS handle *multipathing*. Multipathing—which we'll discuss in greater detail in Chapter 6, "Storage"—is the feature

whereby the hypervisor can understand and potentially use multiple paths to a single datastore. iSCSI, as a block protocol, handles multipathing above the network layer, meaning that vSphere's implementation of iSCSI can recognize and understand that multiple paths across the network exist, and utilize those paths accordingly. Specifically, vSphere uses MPIO (Multipath I/O), part of the block storage stack in ESXi, to recognize and utilize use various paths between the hypervisor and the storage array.

NOTE It's important to understand that MPIO is different than Multiple Connections per Session (MCS or MC/S). MCS is another way of providing multiple paths that are built into the iSCSI protocol itself. vSphere uses MPIO, but vSphere doesn't use MCS.

You might be wondering why this is important. We're glad you asked! Because iSCSI is handling the multipathing "above" the network layer, this allows iSCSI to use multiple VMkernel ports and multiple physical NICs without relying on network-level redundancy/availability features. As a result, the way you design your network to handle iSCSI traffic is affected, and is quite different from how you would design your network to handle NFS traffic (as we'll describe in just a moment).

This architecture is also why multipath iSCSI configuration is handled in a very specific way in vSphere. To recap the process for those who might be unfamiliar, here's a high-level look at how it's done:

1. Create multiple VMkernel ports. (Depending on your storage vendor, these VMkernel NICs might be on the same subnet, or they might be on different subnets. Refer to your storage vendor's recommendations.)
2. Configure each VMkernel port so that only a single physical NIC acts as an uplink for that VMkernel port. So, if you had two VMkernel ports for iSCSI, you would need two physical uplinks. Each physical uplink (physical NIC) is marked Active for only one of the two VMkernel ports, and the other is marked Unused. As a result of this configuration, you create a scenario in which each VMkernel port represents a physically separate path out to the network.
3. Bind iSCSI to each of the "paths" (the one-to-one VMkernel port to physical NIC mappings). From there, continue to configure iSCSI with the appropriate target addresses, and so on.

Take note that, in this process, you aren't using any network-level redundancy features like link aggregation. All the redundancy is being handled above the network layer. As we turn our attention to NFS, you'll see this is a key distinction between the two protocols (aside from the fact that iSCSI is a block protocol and NFS is a file-level protocol).

NFS

Unlike iSCSI, NFS doesn't use MPIO. In fact, NFS—in its current incarnation in vSphere, at least—doesn't recognize any form of multiple paths between the hypervisor and the storage. Instead, NFS relies on network-level redundancy features in order to provide multiple paths from the hypervisor to the NFS export. Although there are versions of NFS (think NFS v4.1, also known as pNFS) that do support multiple paths across the network, vSphere uses NFS v3—and NFS v3 doesn't support multiple paths across the network.

Instead, if you want to provide redundancy for NFS traffic, you have to use features like link aggregation (be sure to think about MLAG to avoid an SPoF!). As we discussed earlier in the

section "Link Aggregation," this has certain side effects as well—most notably in this case the fact that NFS won't be able to take advantage of more than a single physical link's worth of bandwidth for any given NFS datastore. In fact, if you need more bandwidth to an NFS datastore than a single 1Gb Ethernet link can provide, you have only one option: migrate to 10Gb Ethernet.

Later in this chapter, in the section "Crafting the Network Design," we'll discuss some ways you can ensure the appropriate levels of availability and performance for both NFS and iSCSI. For now, though, let's turn our attention to another factor that influences the network design: the use of 10 Gigabit (Gb) Ethernet.

10Gb Ethernet

Prices for 10Gb Ethernet ports have dropped significantly in the last few years, and it's now becoming much more common to see organizations deploying 10Gb Ethernet in their datacenters, especially in conjunction with vSphere. 10Gb Ethernet can offer a number of benefits over standard 1Gb Ethernet:

- Using 10Gb Ethernet reduces the total number of ports required when compared to 1Gb Ethernet. It's not uncommon to see 6, 8, or even 10 1Gb Ethernet ports in the back of a vSphere host. With 10Gb Ethernet, you can cut that down to only 2 ports (a reduction of 3x, 4x, or even 5x).
- Because fewer ports are needed, cabling is significant reduced, and that can have benefits for airflow and datacenter cooling as well.
- In some cases, using 2 10Gb Ethernet ports per ESXi host is cheaper than using 6, 8, or 10 1Gb Ethernet ports per server.

Looking even deeper and integrating some of the other things we've discussed in this chapter, we can find even more potential benefits of 10Gb Ethernet in VMware vSphere environments:

- Recall from the "Link Aggregation" section earlier that traffic between two endpoints would never be able to use more than a single link out of the aggregate. Even with four 1Gb Ethernet links bonded together, a single traffic flow between two endpoints will be constrained to a theoretical maximum of 1 Gbps. Moving to 10Gb Ethernet removes bandwidth constraints for traffic that needs more than 1 Gbps but is primarily point-to-point traffic. (IP-based storage, anyone?)
- Rather than having to use physical links as highly inflexible ways to partition and control traffic, vSphere designs can now use hypervisor-based tools like Network I/O Control (NIOC) to more efficiently partition and shape traffic. NIOC was enhanced in vSphere 5.0 to include user-created network resource pools, a feature that wasn't available in earlier releases. (Note that some vendors offer products that perform similar functionality—the ability to partition a single 10Gb Ethernet link—in hardware. HP's Flex-10 and IBM's Virtual Fabric are good examples, as is Cisco's Unified Computing System [UCS]).

Although 10Gb Ethernet can offer benefits to a vSphere environment, several considerations come to mind when you're planning the infrastructure for 10Gb Ethernet:

- Physical network cable
- Physical switches

- Server architecture
- Network partitioning in hardware

Some of these factors we've already discussed in great detail, so we won't repeat all that information here:

Physical Network Cable We talked about cabling earlier in this chapter, and we mentioned that 10Gb Ethernet has some very specific cabling requirements. You'll need to be sure to take cabling limitations into consideration in planning your network design, and you'll want to ensure you're using Cat-6A cabling. Yes, you could get by with Cat-6 for very limited distances, but do you really want to cut corners in your datacenter? We certainly wouldn't want to be the person behind the "we need to recable the datacenter" email thread.

Physical Switches Several vendors provide 10Gb Ethernet-capable switches today. When you're designing your virtual infrastructure, you should take into account your current environment and how these network components will fit into it. Usually, your network team will manage these components, so they should be part of the network design process.

Also, refer back to the "Physical Switch Support" section earlier in this chapter for other considerations you'll need to keep in mind when selecting the physical switches for use with your 10Gb Ethernet environment.

Server Architecture We'll talk in more detail about this topic later in the chapter (in the section titled, appropriately, "Server Architecture"), but the architecture of the server can have an impact on 10Gb Ethernet designs. How many PCI Express (PCIe) slots does the server have? Are the PCIe slots true x8 slots, which are required to allow a 10Gb Ethernet NIC to run at full wire speed? You'll want to have the answers to these sorts of questions.

Network Partitioning in Hardware Suppose you've opted to go for 10Gb Ethernet, but you don't necessarily want to allocate 10Gb Ethernet for vMotion or IP storage. As we mentioned earlier, some vendors offer the ability to partition a single 10Gb Ethernet link in hardware, so you could create something like this:

- 1 Gb—Management port
- 1 Gb—vMotion
- 2 Gb—VM traffic
- 2 Gb—FT
- 4 Gb—IP storage

Although this sort of functionality is pretty handy, there is one interesting side effect of which you should be aware. vSphere uses the link speed to determine the maximum number of concurrent vMotion operations it will support. For a 1 Gb NIC, that maximum is 4; for a 10 Gb NIC, the maximum is 8. If you partition a 10Gb Ethernet NIC to *anything* less than 10 Gb, then your maximum concurrent vMotion operations will drop from 8 to 4. Will this matter? Perhaps not, but you need to know about this impact and account for it in the design.

Other vendors offer the ability to create multiple instances of a NIC, but not necessarily to partition the traffic in some way. We'll discuss this functionality later in the section "SR-IOV and DirectPath I/O." Products like Cisco's Virtual Interface Controller (VIC) are examples of

this sort of functionality (although you should know that Cisco's VIC doesn't use SR-IOV as its partitioning mechanism).

Lots of vSphere architects are including 10Gb Ethernet in their designs to work around some of the inflexibilities introduced by using multiple 1Gb Ethernet NICs in their servers. There is, though, potentially another way to address those inflexibilities: a new technology called I/O virtualization.

I/O Virtualization

An emerging technology called *I/O virtualization* is essentially a virtual LAN in a box. You connect a hardware network component at extreme speeds to the backbone (for example, 780 Gbps); the network component serves as a robust I/O gateway for dozens of servers. Servers connect to the hardware network component through PCI Express bus extenders or InfiniBand.

Using this technology, you can create virtual host bus adapters (HBAs) and virtual NICs and present them to the host with a considerable amount of bandwidth. You can create profiles and allocate QoS to each vNIC and vHBA presented to the host. An example of such a vendor is Xsigo (`www.xsigo.com`), which offers this technology with its products (Xsigo was recently acquired by Oracle).

What are some of the considerations of using this sort of technology in your design? Here are a few that spring to mind (conveniently organized with our familiar AMPRS approach):

Availability/Redundancy Does the I/O virtualization product offer a solution to ensure that your network design remains highly available and redundant? If not, then this is probably not the sort of enterprise-class solution you want running your business-critical vSphere environment.

Manageability How easy is the I/O virtualization product to manage? Does it integrate with vSphere? If it introduces yet another management point, the potential benefits it offers might not be worth the additional operational overhead.

Performance Although the interconnect between your ESXi hosts and the hardware network component is typically PCIe or InfiniBand (and therefore offers sufficient bandwidth and low latency), what is the oversubscription ratio out of the I/O virtualization solution onto the rest of the network? How does traffic flow through this solution?

Recoverability Where is the configuration for the I/O virtualization solution stored, and can it be easily backed up and restored? If not, then this solution isn't very recoverable—and that's something you'll want to take into consideration.

Security Does this solution satisfy the security requirements for your design? Does it integrate with existing security frameworks, such as existing authentication mechanisms or account directories? If not, what is the operational impact of having yet another set of accounts and logins that must be managed?

I/O virtualization is a relatively new technology, so a lot of growth and development will still occur in this space. It's also not the only network-related virtualization solution you might want to consider in your design. SR-IOV is another I/O virtualization technology that might—based on the functional requirements—have a place in your design.

SR-IOV and DirectPath I/O

DirectPath I/O (more generically known as hypervisor bypass) is a technology that has existed in vSphere since the 4.0 release. Also referred to as VMDirectPath, DirectPath I/O is the idea of attaching a supported PCIe device directly to a VM, bypassing the hypervisor (hence the name *hypervisor bypass*). Naturally, DirectPath I/O has a number of drawbacks that limit its usefulness. What sort of limitations? VMs that are using DirectPath I/O can't

- Use vMotion (except in very specific circumstances involving Cisco UCS)
- Be protected using vSphere HA or vSphere FT
- Take advantage of Network I/O Control
- Use memory overcommitment

Those are some pretty significant limitations. Further, because DirectPath I/O involved directly assigning a PCI device to a VM, it wasn't very scalable—servers simply didn't (and still don't) have enough PCI slots to support high consolidation ratios when using DirectPath I/O. As a result of both of these factors, DirectPath I/O generally sees limited use.

With the vSphere 5.1 release, VMware adds support for SR-IOV, which helps address at least one of those limitations. SR-IOV is a PCI SIG standard that allows a single PCIe device to subdivide itself into multiple virtual instances. These virtual instances, more properly called virtual functions (VFs), can each be assigned to separate VMs and appear to the VM as its own individual NIC. It's pretty straightforward to see how the ability to create 16, 24, or 32 VFs on a single SR-IOV-enabled NIC (more properly called a physical function, or PF) addresses the scalability concerns of DirectPath I/O. What SR-IOV doesn't address, however, are the other concerns of DirectPath I/O, so you'll still have to sacrifice VM mobility if this is a feature that you want to use (as one example).

SR-IOV also introduces some considerations of its own:

- Many SR-IOV NICs include basic Layer 2 switching in hardware on the NIC. This means VM-to-VM traffic between two VFs on the same SR-IOV card is extremely fast (up to 40 Gbps).
- When traffic patterns change (changing from VM-to-VM traffic with both VMs on the same SR-IOV card to VM-to-VM traffic with both VMs not on the same SR-IOV card), performance will change dramatically. Depending on the path, VM-to-VM performance might drop all the way back to whatever the physical layer is (perhaps as low as 1 Gbps).
- If some VFs are used for DirectPath I/O but some VFs are used as uplinks for a dvSwitch, then NIOC can't/won't see the traffic on the VFs used for DirectPath I/O—even though the traffic on those VFs will affect the hypervisor-managed VFs (all the traffic flows through the same PF, after all). The same goes for the use of load-based teaming; you could see unexpected results when mixing bypassed and non-bypassed VFs with load-based teaming.

As you can see, although SR-IOV and DirectPath I/O offer some interesting possibilities, they also create situations that you'll need to carefully consider. SR-IOV (and DirectPath I/O) probably aren't technologies you'll see in every vSphere network design, but it's important to

understand what they are and how they can potentially be used to help your design fulfill the functional requirements.

Before we move on to a discussion of building (or crafting) the vSphere network design, let's wrap up this section with a review of the impact of server architecture on your network design.

Server Architecture

The architecture of your servers can impact your vSphere network design in a number of ways. Some of these impacts are obvious; some aren't quite so apparent.

Some of the obvious impacts to the network design include the following:

- The architecture of the server determines how many different network interfaces, and what types of network interfaces (1Gb or 10Gb Ethernet), are available.
- The architecture of the server determines how much redundancy, if any, you can provide for network connectivity.
- The architecture of the server (blade server versus rack-mount server) affects the overall network topology.

We won't go into much detail on the impacts listed; they've been discussed extensively in various forums and are reasonably well known.

Some of the not-so-obvious impacts include these:

- Chipset architecture affects PCIe slot performance.
- Mismatched PCIe connectors on some slots can affect expansion-slot performance.

These not-so-obvious impacts deserve a bit more attention:

Chipset Architecture and PCIe Slot Performance It's a little-known fact that different generations of processors and their chipsets have very different I/O structures. For example, the Intel Xeon 5500/5600 series of CPUs used an I/O hub (sometimes referred to as the *southbridge*) through which all PCIe I/O traveled. This meant that, regardless of the PCIe slot, you could expect reasonably consistent performance. The only performance limitations were the slot (x4 or x8) and the overall throughput to/from the southbridge.

However, the Intel Xeon E5 family assigns some PCIe lanes directly to CPU sockets, whereas other PCIe lanes travel through the C600 chipset. What does this mean? You could see dramatically different performance from different PCIe slots. If you put a 10Gb Ethernet NIC in a PCIe slot that runs through the C600 chipset, you'll be limited to a x4 connection—and that's not enough to sustain full wire rate on a 10Gb Ethernet card.

For these reasons, it's important to understand how the PCIe slots in your servers are assigned to CPU sockets. On the majority of Intel Xeon E5-based systems, slots 2, 3, and 5 are x8 slots that are connected directly to a CPU root complex (and are therefore capable of sustaining full line rate for a 10Gb Ethernet card).

Mismatched PCIe Connectors Some servers have PCIe connectors that don't match the capabilities of the underlying bus. For example, in some Intel Xeon E5-based servers on the market, slot 1 has a PCIe x8 connector, but the underlying PCIe bus supports only four lanes (a x4 connection). Users think they can install a x8 device (like a 10Gb Ethernet NIC) in the slot, and physically the slot will accept it; however, it won't perform to its fullest potential. Take the time to fully understand how the I/O moves through your server so you can architect your design to perform its best.

As you can see, quite a few factors will influence your network design. In some cases, these influential factors may actually be pushing the design in different directions! It's up to you, the vSphere architect, to reconcile the influence of these factors with the functional requirements, the constraints, and the risks as you build the network design. Speaking of building the network design, that's the focus of our next section.

Crafting the Network Design

We've finally arrived at the section you've been anxiously waiting to read—how do you take all the various components involved, consider the factors influencing the design, and then craft the network design for your vSphere environment? That's what we'll discuss in this section. Building on the information from the previous two sections, and using our familiar AMPRS organization, we'll discuss how you assemble a vSphere network design.

Availability

For the most part, ensuring the availability of the network is really about ensuring the proper redundancy for the various components that form the network. The dictionary definition of *redundant* is as follows: "Serving as a duplicate for preventing failure of an entire system (as a spacecraft) upon failure of a single component."

When you design your environment, you don't want it to include an SPoF. That is why you have two hard disks for mirroring, two power supplies, and two NICs—two and two and two. Of course, in reality the environment will be much more complicated and expensive, because you should have redundant storage arrays (or storage processors) and a redundant location (a disaster recovery/business continuity planning [DR/BCP] site) to bring everything up if one site fails.

Fortunately, many hardware vendors understand the need for redundancy as a way of providing availability. Show me a server (a brand name) that you can buy today that has one NIC—we'll bet you can't do it. The same goes for a server with Ethernet ports that are less than 1 Gb. Soon it will be the standard for all servers to have dual 10Gb Ethernet ports.

In this section, we'll discuss the various ways you can ensure that your design provides the appropriate level of network availability. We'll focus our discussion around these key areas:

- Management traffic
- VM traffic
- IP-based storage traffic
- vMotion traffic
- FT traffic

Before we examine the details of providing availability for the various types of traffic in your vSphere design, let's first get some obvious recommendations out of the way. These are things we've already mentioned, but we want to include them for greater clarity:

- Always use multiple physical NIC ports in your servers. Don't rely on a single physical NIC port, or you're relying on an SPoF.
- Ideally, use multiple, separate physical NICs in your servers (separate mezzanine cards, or built-in NICs in conjunction with a PCIe expansion card).
- Always use multiple physical switches on your network.

- Ideally, use physical switches that are themselves as redundant as possible (redundant power supplies, redundant fans, redundant supervisor modules, and so on).
- Always make sure your ESXi hosts are connected to more than one physical switch, preferably using ports from separate physical NICs in your hosts.
- Keeping operational aspects in mind, a solid network team is a must for a virtualized datacenter, especially one that is dependent on IP-based storage.

Figure 5.4 graphically summarizes these recommendations.

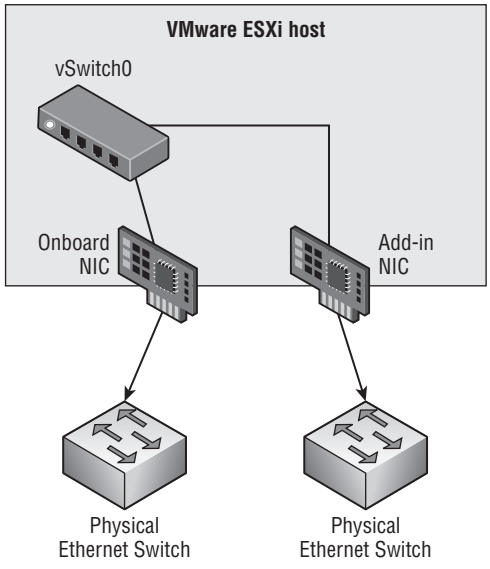


FIGURE 5.4
Use multiple connections from separate NICs in your ESXi hosts to separate switches for maximum redundancy.

Now, let's dive a bit deeper into protecting the specific traffic types in a vSphere environment. We'll start with management traffic.

MANAGEMENT TRAFFIC

You have two options for planning for the redundancy of your management traffic. You can have either two management network ports on two separate vSwitches or one management network port with redundant NICs on the same vSwitch. Let's start with an example of the first option, as shown in Figure 5.5.

The first management network port is named *Management Network*. It has an IP address of `192.168.165.1` configured on vSwitch0, which uses vmnic0. The second management network port is named *Management Network 2* and has an IP of `192.168.166.1` on vSwitch1 using vmnic1. By providing two management points into your ESXi host, you mitigate the risk of an SPoF for the management network. (Did you notice that the VM Network in Figure 5.5 has an SPoF? You should have!)

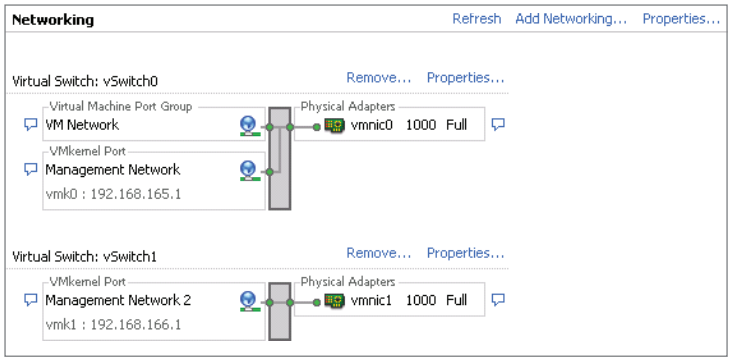


FIGURE 5.5
This ESXi host uses two management ports for redundancy.

A certain amount of additional overhead is required in this sort of configuration. You need to apply some configuration changes to your cluster to accommodate such changes. These changes are as follows:

`das.isolationaddress[x]` This option specifies that you now have more than one possible isolation address that should be available in the event of loss of network on one of the management ports. By default, with one management port, `das.isolationaddress` is set to the default gateway of the management port (in the previous case, `192.168.165.254`). In this case, the correct settings are

```
das.isolationaddress[0] 192.168.165.254
das.isolationaddress[1] 192.168.166.254
```

`das.failuredetectiontime` This option defines the interval that triggers an HA failure if there is no network connection during that period. The default setting is 15,000 milliseconds. In order to prevent the case of a false positive and having HA invoked because of an error, you should increase this to 30,000 ms:

```
das.failuredetectiontime 30000
```

Note that this setting is valid for vSphere 4.*x* environments, but not for vSphere 5.*x* environments. In vSphere 5.0, this setting has been removed and can't be set. In vSphere 5.1, VMware added back a related setting called `das.config.fdm.isolationPolicyDelaySec`. This setting allows you to change the number of seconds before the isolation policy is executed; the minimum value is 30 seconds. In most cases, you won't need to change this setting.

Now, let's look at the second option to provide redundancy for your management network, illustrated in Figure 5.6.

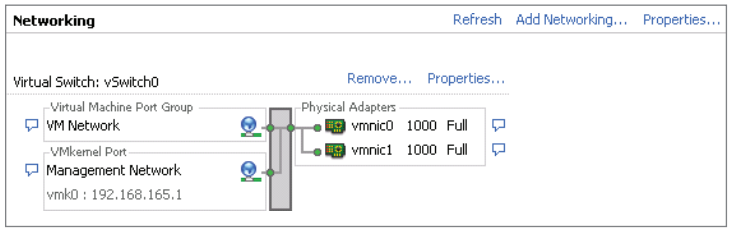


FIGURE 5.6
This ESXi host has one management port but uses two NICs for redundancy.

Here you use only one vSwitch and only one management port. This makes the configuration slightly easier and less complex. You don't need an additional IP address for the management ports, and you don't have to configure the additional settings required in the first option. You use two NICs that are set in an active-passive configuration. Each physical uplink (vmnic0 and vmnic1 in Figure 5.6) should be connected to a separate physical switch so that if one goes down, the other will continue to provide management connectivity for the ESXi host.

In vSphere 4.*x* environments, you should also set `das.failuredetectiontime` to 30000. Doing so prevents the occurrence of a false positive, which can cause you to end up with the VM being powered off (if you've set the configuration this way on your cluster settings) and not powered back up because the host no longer detects that it's isolated. For vSphere 5.*x* environments, see our earlier explanation of how this setting changed in vSphere 5.0 and vSphere 5.1.

Of these two options, what is the preferred configuration: two management ports or one (redundant management ports or redundant NICs)? That depends on a number of factors:

- How many NICs per host? If you're limited in the number of NICs available to you, then you probably can't dedicate two NICs to a management port.
- Can you make better use of the configuration by adding functions to the vSwitch (vMotion, for example)?
- Does your security policy allow for the mixture of management VLANs and other purposes (vMotion, IP storage, and VM traffic)? If not, then you'll need to use dedicated NICs.

In the end, it's a question of choices that depend on your environment.

Virtual Machine Traffic

Each ESXi host can run a multitude of VMs—the exact number always depends on the consolidation ratio you want to achieve and the capabilities of your infrastructure. A good portion of these VMs will need network connectivity outside the host onto your corporate network. So, when you're designing this portion of your infrastructure, you shouldn't be dependent on a single NIC or the connection to a single physical switch (advice we shared with you earlier, but it bears repeating). You'll need to plan for the redundancy of the VM traffic.

In the physical world, systems administrators would generally team the physical NICs either for load-balancing or for redundancy: basically, two network cables run into the server. You can do it that way in the virtual world, as shown in Figure 5.7.

Although certain applications or guest OS configurations might require multiple NICs and teaming, it's generally not the best way to provide redundancy. Fortunately, providing redundancy for the VM traffic is remarkably easy—just provide redundant uplinks out of the vSwitch or dvSwitch hosting the VMs, and you're done. Yes, it really is that easy—most of the time.

Because a vSwitch/dvSwitch doesn't forward packets it has received back out again, it doesn't create bridging loops and therefore doesn't need to participate in STP. This also means you can have multiple active uplinks out of a vSwitch/dvSwitch that—using the default setting "Route based on the originating virtual port ID"—don't require any additional configuration on the upstream physical switches.

For the vast majority of workloads, this configuration is sufficient. Each time a VM is powered on, it's assigned to an uplink and continues to use that uplink until it's power-cycled or until the uplink fails and network traffic is passed to one of the other active uplinks in the virtual switch (or the port group).

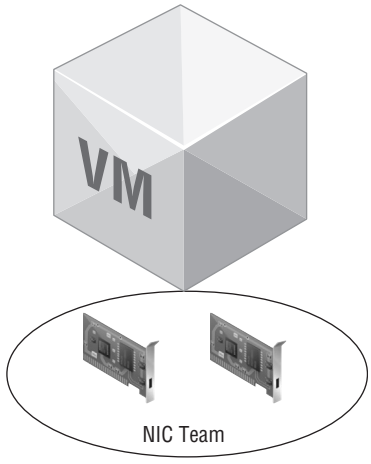


Image: ©VMware, Inc.

FIGURE 5.7
Configuring NIC teaming at the guest OS level often isn't the best way to provide redundancy.

Although this configuration is simple and provides adequate redundancy, it's not without its limitations. This setup doesn't provide efficient load balancing over all the uplinks in the virtual switch, and the traffic across the uplinks can become unbalanced in certain cases. In cases like that, or in cases where a VM's traffic patterns are predominantly many-to-one/one-to-many, then it might be beneficial to use link aggregation ("Route based on IP hash" is how link aggregation is denoted in the vSphere UI).

We discussed link aggregation extensively earlier (see the section "Link Aggregation"), so you know already that link aggregation isn't without its limitations. Notably, the traffic patterns need to be one-to-many/many-to-one; one-to-one traffic patterns won't benefit from link aggregation. Additionally, link aggregation requires support from the upstream physical switches, introduces additional complexity, and requires support for MLAG to avoid an SPoF.

If you're using a dvSwitch, you also have one other option: load-based teaming. With load-based teaming, no upstream switch configuration is required, and the dvSwitch will periodically evaluate physical NIC load to see if VM traffic needs to be rebalanced across the uplinks. It's a great feature and blends the best of both worlds. Unfortunately, it's only available with a dvSwitch, which requires Enterprise Plus licensing.

So which approach is best? Generally, we recommend keeping it as simple as possible. Unless the functional requirements drive the use of link aggregation, the default vSwitch/dvSwitch settings are generally acceptable for most installations.

NOTE The considerations for providing redundant uplinks out of a vSwitch or dvSwitch apply not just to VM traffic, but also to other types of traffic. The difference is that other types of traffic—like the management traffic or the IP-based storage traffic—generally introduce additional considerations that must also be taken into account.

IP Storage (NFS/iSCSI)

Earlier in this chapter, we discussed the differences between NFS and iSCSI as it pertains to the impact on your network designs, and we explained how iSCSI's use of MPIO allows iSCSI to use network architectures that wouldn't benefit NFS at all. In this section, we want to focus

specifically on the mechanics of how you go about providing availability (in the form of redundant connections) for both iSCSI and NFS.

For iSCSI, we outlined the process earlier. Let's repeat it here for completeness:

1. Create multiple VMkernel ports. (Depending on the storage vendor, these VMkernel ports might be on the same subnet, or they might be on different subnets. Refer to the storage vendor's recommendations.)
2. Configure each VMkernel port so that only a single physical NIC acts as an uplink for that VMkernel port. So, if you had two VMkernel ports for iSCSI, you would need two physical uplinks. Each physical uplink (physical NIC) is marked Active for only one of the two VMkernel ports, and the other is marked Unused. As a result of this configuration, you create a scenario in which each VMkernel port represents a physically separate path out to the network.
3. Bind iSCSI to each of the paths (the one-to-one VMkernel port to physical NIC mappings). From there, continue to configure iSCSI with the appropriate target addresses, and so on.

Note that there are no underlying dependencies on physical switch support for link aggregation or anything like that. All you need is IP connectivity across multiple physical connections; the block storage stack in vSphere handles the rest.

For NFS, though, the situation is a bit more complex. With NFS, the architecture for building redundant connections depends wholly on the network—and thus depends wholly on mechanisms like link aggregation. Let's look at two configurations for NFS: one with link aggregation, and one without.

NFS with Physical Switches That Support Link Aggregation

Suppose your goal is to have your ESXi host access more than one storage controller from different physical uplinks. In this case, you have to set up multiple IP addresses in the storage controller and configure link aggregation on your ESXi host (and upstream switch). Configuring link aggregation on the storage array is optional but most likely beneficial. Figure 5.8 illustrates an example of such a setup.

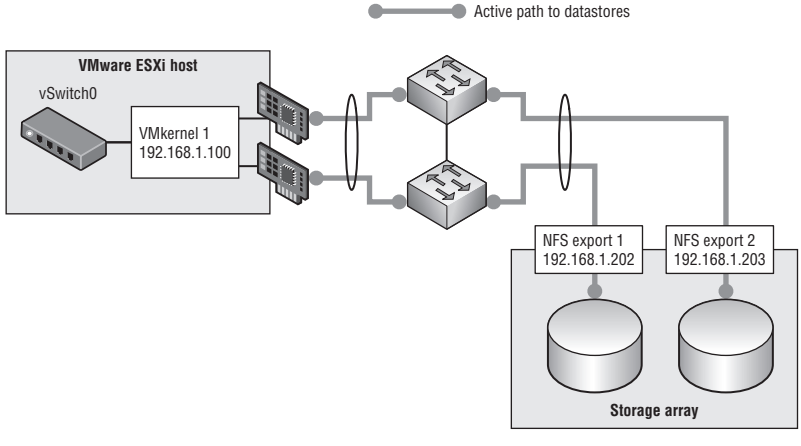


FIGURE 5.8 This configuration uses link aggregation for both the ESXi host and the storage array.

In this example, VMkernel 1 resides on a vSwitch that has been configured to use link aggregation. As a result, depending on the IP hash (recall that vSphere only uses a hash of source and destination IP to determine which link to use), traffic from VMkernel 1 could travel out either of the two uplinks.

If you used only a single target IP address hosting a single NFS export, you'd never use more than one of the two links. (If you don't understand why this is, go back and read the "Link Aggregation" section earlier in this chapter.) So, you need multiple target IP addresses. And because vSphere only associates an NFS datastore with a single IP address, you'll need multiple datastores. In Figure 5.8, you can see that the storage controller has two different IP addresses assigned to its interfaces—and has two different datastores that the ESXi host is accessing. Using link aggregation, each datastore is accessed on its own link.

There are a couple of key takeaways from this example:

- You'll need multiple target IP addresses on the storage system.
- Each NFS datastore will still only be associated with a single IP address (or DNS name); thus, each NFS datastore will be limited to a single link's worth of bandwidth.
- Although the aggregate bandwidth for all datastores increases (assuming the IP hashes spread the traffic evenly across the links), the individual bandwidth for a given NFS datastore won't increase (other than an increase due to potentially less contention on the network).
- If your switches don't support MLAG, you've introduced some SPoFs into your design—and that's not good. MLAG support is a must.

The nice thing about this approach is that it requires only a single VMkernel interface, and there aren't any strict requirements about the IP addresses that must be used. When the physical switches don't support link aggregation (or don't support MLAG), then the configuration looks quite different.

NFS with Physical Switches That Don't Support Link Aggregation

Now suppose the goal is the same, but the situation is more complicated than in the previous example. The storage controller still requires multiple IP addresses (and each NFS datastore on its own address), but they have to be on separate subnets. In addition, multiple VMkernel ports—also on separate subnets—are required. Figure 5.9 gives an example of such a configuration.

Why the need for multiple VMkernel interfaces, and why do they have to be on separate subnets? It's actually a simple IP routing issue—the VMkernel IP routing table can only select a single interface to use for any given IP subnet, so having multiple VMkernel interfaces on the same subnet means only one of the interfaces will be used. Using separate subnets introduces separate routes, one for each interface, and the VMkernel can then use all the interfaces. You only need to make sure you have an NFS datastore on the same subnet for each VMkernel interface.

Like link aggregation, though, this doesn't increase the per-datastore bandwidth, only the aggregate NFS bandwidth to all datastores. As we described in the "Link Aggregation" section, if you need more bandwidth to a single NFS datastore than can be provided by a 1Gb Ethernet link, your only option is 10Gb Ethernet.

Keep in mind that iSCSI and NFS implementations (on the storage target side) vary from vendor to vendor. Your vendor can provide the best practices for your specific environment.

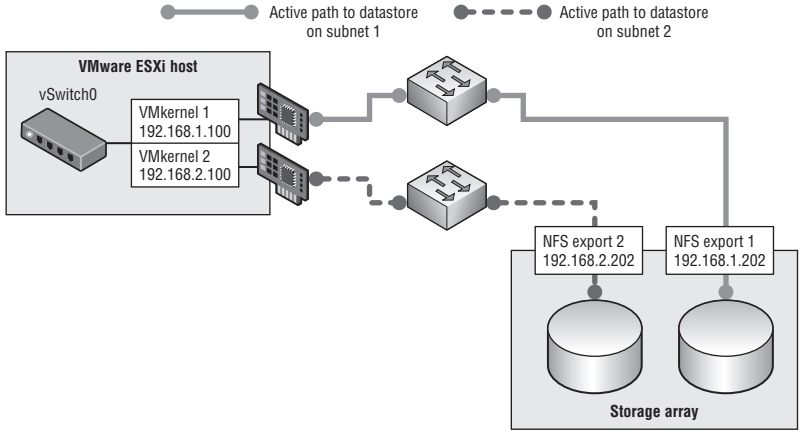


FIGURE 5.9 Without link aggregation, NFS redundancy requires multiple datastores and multiple VMkernel interfaces.

vMotion

Why should you have redundancy for your vMotion interface? You may think this isn't an essential part of your enterprise. Well, you're wrong. Without a working vMotion interface, you can't balance your cluster properly, you can't evacuate your host if you need to perform maintenance, and more. You definitely need redundancy for your vMotion interface.

Does this mean you must have two dedicated NICs for this purpose? Probably not. You can use one of the NICs from your management network or VM network to provide redundancy in the case of failover. Usually the degradation in performance is acceptable for the short time until you restore the primary uplink used for vMotion. If you do feel you need multiple NICs for vMotion, we'll discuss options for that in the "Performance" section later in this chapter.

vSphere FT

vSphere FT is a relatively new feature that was made available in vSphere 4.0. In order for FT to work, you need a dedicated uplink that replicates the state of the VM from one host to another. You can't afford to have that uplink fail—if that happens, your VM will no longer be protected; and in the case of host failure, the secondary VM won't be available. Therefore, in the case of FT, you should have two dedicated uplinks going to separate physical switches to ensure that FT traffic doesn't get interrupted.

Although the current requirements for FT indicate that a 1Gb Ethernet link is sufficient, future enhancements to FT might require a 10Gb Ethernet link, so be sure you plan accordingly.

That's it for the first design principle of availability. The next design principle we'll discuss in the context of network design is manageability.

Manageability

From a network design perspective, we can think of two major areas for ensuring that the network is as manageable as possible. Both of these areas fall into the operational facet of vSphere design:

- Interoperability with existing network equipment and staff
- Naming and IP conventions

Let's look at interoperability first.

Interoperability with Existing Equipment and Staff

It's highly likely that the vSphere environment you're designing will be added to an existing network—one that has established standards for equipment, processes, procedures, and configurations. How interoperable is the design you're proposing with that existing network? Here are some things to think about:

- Using VMware's vSwitch or dvSwitch means the existing network group loses control over the last mile of the network (the access layer). Is this an acceptable shift of responsibility? What additional impacts will this change in ownership have on operational processes such as troubleshooting and provisioning?
- If the network group wants to retain control over the access layer, then perhaps something like Cisco's Nexus 1000V might be the right solution. The network team continues to manage it in much the same way they manage the rest of the network, and the 1000V creates a nice provider/consumer relationship between the network group and the VMware group.
- Will the network design integrate with existing network management solutions?
- Certain configurations—like using IP hashing on a VMware vSwitch or dvSwitch—require matching upstream configurations. The same goes for VLAN trunking and private VLANs. How will this impact the networking team's existing processes and procedures? Are the necessary standards and protocols supported for this design to function as expected?

These are just a few of the questions you'll want to be sure you have answers for when you examine your network design for manageability.

The second area of manageability centers on naming conventions and IP address assignments.

Naming and IP Conventions

It would be great if we could give any names we wanted to network components. After all, we name our children the way we want, give our pets names, and perhaps name a boat. But a network isn't the same as a family. Naming your VLANs Tom, Dick, and Harry may be amusing, but it isn't the way to do things in the enterprise.

For maximum manageability and operational efficiency, you should label your network components properly. They should be clearly identifiable even for those who don't manage the environment on a day-to-day basis. Here are a few examples:

- iSCSI_VLAN_765
- VM_VLAN_55
- 755_NFS

- 123_vMOTION
- Mgmt_1

Choose names that that can be recognized easily and associated with the appropriate VLAN.

In addition, be sure to create IP addresses in a consistent manner across all of your hosts. Table 5.1 shows an example of how *not* to do it.

TABLE 5.1: How *not* to assign IP addresses

	IP ADDRESS
Management	192.168.1.4
NFS	192.168.6.54
vMotion	192.168.4.20
iSCSI	192.168.20.222

This list has no standardization. When the environment grows, you won't be able to manage anything.

How about the example in Table 5.2?

TABLE 5.2: Standardized IP addresses

	HOST 1	HOST 2
Management	192.168.1.1	192.168.1.2
NFS	192.168.6.1	192.168.6.2
vMotion	192.168.4.1	192.168.4.2
iSCSI	192.168.20.1	192.168.20.2

We hope you can see the difference and how much easier it is when you keep things nice and tidy. Although it's not always possible to maintain a perfect numbering strategy, it's worth the effort to keep things as organized and consistent as possible.

NETWORK DISCOVERY PROTOCOLS

Another area that can prove beneficial with regard to manageability is the use of network-discovery protocols. vSphere supports two network discovery protocols: Cisco Discovery Protocol (CDP) and Link-Layer Discovery Protocol (LLDP). LLDP support was added vSphere 5.0; CDP support has been around for a while. As the name implies, CDP is specific to Cisco environments, although a number of other vendors (besides VMware) also support it. LLDP is a standards-based protocol supported by a number of different vendors.

Both of these discovery protocols allow devices that support them to exchange information across the network, making it easier to determine which ports are connected where and what devices are neighbors to other devices. Unless functional requirements dictate otherwise, we recommend enabling CDP (for predominantly Cisco-based environments) or LLDP (for non-Cisco or heterogeneous environments) on all your ESXi hosts. The minimal added work to do so (a simple GUI change or CLI command) is far outweighed by the additional information that is made available, and you'll be thankful for it when you're trying to troubleshoot a difficult network problem.

Moving on from manageability, it's time to look at the third design principle: performance. The performance of the network design is critical—without a well-performing network design, the entire vSphere environment will suffer, and the whole project might fail. The next section addresses how to design your network for performance.

Performance

What speed should your NICs be? The question should more accurately be, to what ports should your NICs be connected? Ideally, your NICs should be as fast as possible for everything, but that isn't practical. You could have a 10Gb Ethernet NIC for your management network, another for redundancy, two (or four) for IP storage, more for your VMs, and others for vMotion, but that would probably be extreme overkill.

Why should you care? Because each port has a cost. Here's an example. Suppose your server racks are equipped with a patch panel that goes to your corporate Tier-1 switches, and in addition an older Tier-2 10/100 Mb managed switch is used for all the remote control cards and backup ports for each server. The NICs connected to Tier-1 ports are more expensive than Tier-2. In most cases, a 10/100 Mb port is more than sufficient for remote-control cards, management ports, and backup NICs in a team that is mostly dormant. Your regular production traffic will go over the Tier-1 ports, and in the case of an outage on the primary NIC/port it will fail over to the Tier-2 port.

But what do you actually need? Let's look at the following components:

- Management network
- vMotion
- IP storage
- VM networking

Management Network

What level of traffic goes through your management port? If it's a dedicated management port used only for ESXi management, then not that much traffic goes through. In theory, a 10 Mb port would be more than enough, but finding such a port today is relatively impossible. You'll probably go for a 100 Mbps port, although even those can be scarce in some datacenters. A 1Gb Ethernet port is more than sufficient.

If you do decide to use a 100 Mb NIC, note that it's possible to saturate the throughput of that NIC. There are several ways to do that; the most common is to import a VM into your environment. The process when you're importing or converting a VM uses the bandwidth on the management network.

Figure 5.10 shows an example of an ESXi host running without any unusual load on vmnic0 (the management port).

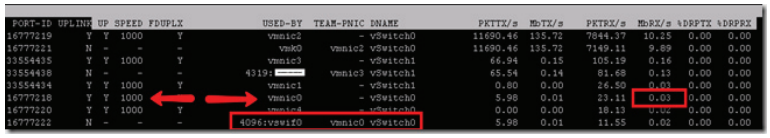


FIGURE 5.10
esxtop is showing a normal network load on an ESXi host's management port.

When you begin an import process on a VM through the Enterprise Converter or the VMware stand-alone converter, all import traffic goes through the management port. As you can see in Figure 5.11, the traffic on that port can easily hit 200 Mbps, in which case a 100 Mbps port won't be sufficient.

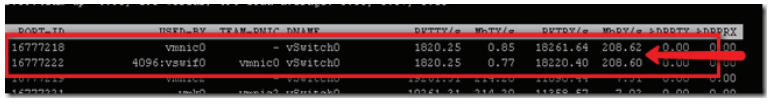


FIGURE 5.11
When converting a VM, esxtop shows heavy usage of the management port.

If you're planning to import a number of VMs into your environment or you plan to perform a large number of physical to virtual (P2V) conversions, you should definitely allocate a 1Gb Ethernet port for the management network.

vMotion

How fast do you want your host evacuated? How fast do you want your migrations to work? These are the questions that will drive how you provision your vMotion network.

Let's consider the following scenario. One of the hosts in your cluster just reported that a power supply failed. If you planned correctly, you have two power supplies in your host, so the server can run with only one power supply for a while; but server redundancy is now degraded, and you don't want to leave your server vulnerable. You should evacuate the host as soon as possible. So, you start to vMotion off your VMs (you have 60 on the host). Running on a 1 Gb NIC, you can perform four simultaneous migrations. Each migration takes 1 minute; that means you can vacate the host in 15 minutes, which is more than acceptable in our book. But using a 100 Mb NIC, that number will be more like 150 minutes, which isn't so acceptable. (As we noted earlier, it's becoming increasingly difficult to find ports that only run at 100 Mb in enterprise datacenters, so this isn't likely to be an issue. Further, vMotion requires 1 Gb NICs.)

You want the migration to go as quickly as possible. So, you could go for 10Gb Ethernet, but you should take into account the following. Starting with 4.1, the number of concurrent vMotions was increased to eight from four, and the speed cap on a 10Gb Ethernet link for vMotion was raised to 8 Gbps. If you aren't careful, you may saturate the 10 Gb NIC with only vMotion. This isn't a good thing. You'll generally be using the NIC for other purposes as well (VM traffic or IP storage). If this is the case, you'll have to incorporate some kind of network QoS on the NIC to ensure that the vMotion interfaces don't saturate the interface.

Also recall from our earlier discussion of 10Gb Ethernet that a 10 Gb NIC that has been subdivided into separate logical NICs will be treated as a 1 Gb NIC for the purposes of calculating simultaneous vMotion operations (so the limit will drop back to four).

In addition, newer versions of vSphere introduced support for multi-NIC vMotion, which allows vSphere to use multiple NICs simultaneously to further speed up VM migrations. If you

want faster migrations but can't afford (or don't want) to migrate to 10Gb Ethernet, this might be an option to incorporate into your design.

IP Storage

IP storage has grown from being a second-level storage platform to being a more mainstream enterprise-grade storage platform. The I/O performance you can achieve from a 10 Gb NIC is no worse than the speeds you can achieve with Fibre Channel (FC) storage. But the number of 1 Gb NICs needed to achieve performance equal to FC is considerably higher, and the configuration is much more complex to achieve the same results. Further, as we described earlier (in the "Link Aggregation" and "Availability" sections), no matter how many NICs you throw at NFS, it will only use one link per datastore.

Therefore, the default choice for IP storage should be 10 Gb NICs. This is especially true for compute nodes with limited slots, like blades. Add in NIOC to maximize performance and options for fan-in growth in your datacenter. This, of course, assumes you have the infrastructure in place—if not, then you should plan to make this your standard for the future.

One final consideration regarding IP-based storage is in regard to the use of DNS for NFS mount points. If your design uses a scale-out NFS storage platform (there are a number of examples on the market, such as EMC Isilon and NetApp in Cluster mode), then using DNS round robin—a single DNS name backed by multiple IP addresses—for your NFS mount might improve performance by spreading the workload across multiple connections. Each ESXi host will still use only a single connection, but aggregate traffic from multiple hosts will potentially benefit.

VM Networking

Each VM needs a certain amount of bandwidth, and there is no one-size-fits-all solution. You'll have do your homework and measure (or estimate) the amount of network traffic each VM will use. If you're converting a physical server, then collecting the data beforehand should be part of your policy before you migrate the server.

We won't go into the details of how this can/should be performed on Windows/Linux servers; we'll leave that to you and your corporate policies and procedures. But when you have this data, you can estimate how many VMs can reside on each NIC. You then size your host accordingly, taking into account the network information. It may be that one 1 Gb NIC will suffice for your environment, or it may be that two 10 Gb NICs won't suffice.

The bottom line, for all the different vSphere traffic types, is to plan according to your sizing needs. ESXi can accommodate your needs regardless of the speed of your NIC.

With availability, manageability, and performance out of the way, let's now discuss recoverability.

Recoverability

To ensure that your network design is recoverable, you'll want to do the following things:

1. If the organization doesn't already have established procedures for making and storing backups of the network device configurations, create those procedures as part of the vSphere design. You don't want to suffer an outage simply because you didn't account for making a backup of the configuration of a key network switch. You can easily incorporate services like TFTP into the vSphere design to help with this process.

2. If you're using vSwitches in your design, there's no way to back up that configuration. Instead, focus on creating scripts that can easily re-create the vSwitch configuration in exactly the same manner. This helps in two ways: first, it helps with recoverability; second, it helps with adding new ESXi hosts to the environment.
3. If you're using dvSwitches prior to vSphere 5.1, you don't have a way to back up the dvSwitch configuration. Again, your next-best alternative might be a series of scripts that configure the dvSwitch exactly the way you want—thus making it easy to re-create in the event of an emergency. vSphere 5.1 adds the ability to back up and restore the dvSwitch, so incorporate this functionality into the operational facet of your design. The Cisco Nexus 1000V can also make backups of its configuration.
4. Document, document, document! Be extremely detailed in your documentation—make notes of exactly which ports are plugged in where and why, and keep hard copies of the configurations just in case. You never know when this information might be handy.

The last design principle we'll discuss is security. Although it's listed last, it's certainly not least, as you'll see in the next section.

Security

Security shouldn't be an afterthought—you should take the time to look at the components of your design from the point of view of security. Always consult the VMware site for updated recommendations on securing your vSphere infrastructure. VMware currently offers a vSphere 4.0 Security Hardening Guide:

```
www.vmware.com/files/pdf/techpaper/VMware_vSphere_HardeningGuide_
May10_EN.pdf
```

There is a Security Hardening Guide for vSphere 4.1 as well:

```
www.vmware.com/resources/techresources/10198
```

VMware also has a Hardening Guide for vSphere 5.0:

```
http://communities.vmware.com/docs/DOC-19605
```

At the time of this writing, a Security Hardening Guide for vSphere 5.1 had not been released.

With regard to a vSphere network design, the security focus is primarily on the various network traffic types. We'll be discussing the following kinds of traffic in this section:

- Management network traffic
- VM traffic
- IP storage traffic
- vMotion and FT traffic

Management Network

Do your domain controllers and mail servers sit on the same subnet as your desktop computers? We hope not! They shouldn't, because your corporate servers should be separate from your end users. This arrangement provides the option to protect your server farm from outside attacks. We don't mean attacks from outside your network but rather from inside your network due to a computer being compromised and acting as an attack point into the server farm.

Some enterprise organizations have their server farms behind a firewall with IPS and IDS systems protecting their critical servers.

Your vSphere environment should definitely be treated as a critical server. The risk of the environment being exploited if it's compromised is potentially disastrous. If someone takes control of your vCenter Server, they gain control of every VM in your environment. If they take control of one host, they have control of every VM running on that host.

Therefore, your management network should be separate from the rest of your virtual environment. And it isn't the only element that should be separated. vMotion, IP storage, and FT should be separate as well; we'll get to those later in this section.

How do you separate your management network? Dedicate a network segment specifically for this purpose. Some enterprises have dedicated subnets for out-of-band (OOB) management devices such as Integrated Lights Out (iLO) ports. Some define the management network on an ESXi host as an OOB port, because the management network is there only to provide management to either the vSphere Client or the vCenter Server on that host. You can also provide a new dedicated subnet, depending on your corporate policy.

With this segregation, you can provide the correct network-access lists on this segment to secure your environment even further.

VM Traffic

When you start, you'll be hosting a few VMs. Then the number will grow. In the not-too-distant future, you'll be providing virtualization services for a great number of VMs. You don't want to have all those VMs (production servers, desktops, lab machines, test and development machines, and so on) on the same subnet. You segregate traffic on the physical network exactly the same way you should separate it on a virtual network: production servers on this subnet, desktops here, and so on. VMware makes this extremely easy with VLAN tagging on VM port groups. All the VMs may be running on the same two physical uplinks, but they're on different VLANs with different IP addresses.

You must make sure that all the VLANs are trunked correctly to the appropriate ports on the physical switch and that the port groups are defined on all the hosts that are in the same cluster. Otherwise, the VMs will disconnect from the network if an uplink fails or they're migrated to another host.

You should work with your network team to define a solid policy that will work well in your environment. The option of assigning all VLANs up front to every port is very appealing because it would require a one-time configuration for each ESXi host. But in some environments, having all the VLANs open on the uplinks will cause problems. Multicast network traffic is a good example.

vMotion and FT Traffic

vMotion of a VM between hosts is a necessity for most enterprise environments, whether for planned maintenance or to balance machines with distributed resource scheduling (DRS). When you migrate the machine from one host to another, the VM disk isn't migrated over the network—only the VM's live running state (CPU execution, memory contents, and network). None of this traffic is encrypted, so someone could eavesdrop on the traffic and acquire the information flowing at that time, which could be a security risk.

If you're performing a Storage vMotion, which takes much longer, the powered-off VM is transferred over the network (unless your array supports vSphere APIs for Array Integration [VAAI]). This information can also be compromised.

In this case, the earlier solution of network segregation and a nonroutable VLAN will minimize the attack surface and provide a level of protection. This segregation should also be done with FT traffic.

IP Storage Network Traffic

When we say *IP storage*, we're talking about NFS or iSCSI. There are good use cases for both network protocols (as we've already discussed), but this traffic isn't encrypted, and it travels over the wire. Anything that travels over the wire can potentially be tapped with a packet sniffer—and may compromise your security.

How can you protect this traffic? We'll discuss three ways. They aren't either/or solutions but can be used together in the appropriate use cases:

- VLAN isolation and a nonroutable VLAN
- NFS export (`/etc/export`)
- iSCSI CHAP authentication

VLAN Separation This approach is valid for both iSCSI and NFS (as well as all the other types of vSphere traffic). Your IP storage should be on a separate network segment. This helps by segregating IP storage from the rest of your network and thus limiting the attack surface to only that segment. An attacker must have an interface on that segment to eavesdrop on the traffic.

To extend this concept, the VLAN should be nonroutable. This ensures that only interfaces on the same network can reach the VMkernel and IP storage interfaces.

NFS Exports This approach applies only to NFS. When you create the NFS export on your storage, be it an enterprise-grade storage array or a Linux server providing NFS, you should always limit the hosts that are allowed access to NFS shares. Doing so limits who can access the filesystems where your VMs are stored.

You do this using the exports file. The `/etc/exports` file specifies remote mount points for the NFS mount protocol per the NFS server specification. On some arrays, you use the GUI rather than configuring the file itself.

Let's look at an example. If you want to define the export to folder `/vol/nfs1` to all the hosts on the `192.168.0.0/24` segment (254 hosts), you configure the export as shown here:

```
192.168.0.0/255.255.255.0 (rw,no_root_squash)
```

To export `/vol/nfs2` only to a host that has IP address `192.168.0.45`, configure the export as follows:

```
192.168.0.45 (rw,no_root_squash)
```

In this case, any other IP trying to mount the NFS share will be denied access.

CHAP Authentication This approach is relevant only to iSCSI. For the most part, iSCSI operates as a clear-text protocol that provides no cryptographic protection for data in motion during SCSI transactions. Because of this, an attacker can listen in on iSCSI Ethernet traffic and potentially do the following:

- Rebuild and copy the files and filesystems being transferred on the network.
- Alter the contents of files by injecting fake iSCSI frames.
- Corrupt filesystems being accessed by initiators and exploiting software flaws.

ESXi supports bidirectional CHAP authentication. This means the target and the host initiators authenticate with each other, providing a higher level of security to the IP storage protocol.

Before we wrap up this chapter, we'd like to take all the information we've presented and pull it together with a few network-design scenarios.

Design Scenarios

Last but not least in this chapter, we'll provide some scenarios of ESXi host configuration with two, four, six, or eight NICs. In all the design scenarios, each host has the following:

- Management port
- VMkernel for IP storage
- VMkernel for vMotion
- VM port group 1
- FT port

We assume the use of multiple physical switches upstream to ensure that there is redundancy at the physical network layer as well.

Two NICs

This isn't a good idea. You can't provide proper performance, security, isolation, or redundancy with only two NICs. Can such a design be done? Yes. Do we recommend it? No. With that off our chest, let's start.

VM traffic goes through vmnic1; management, vMotion, IP storage, and FT are on vmnic0, as you can see in Figure 5.12.

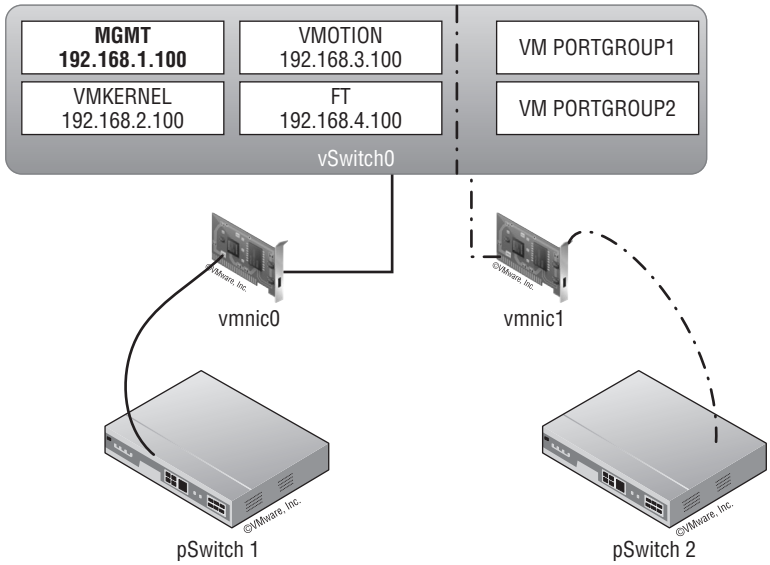


FIGURE 5.12
This figure shows a sample configuration with two NICs.

The only time two NICs would be acceptable is if you were using two 10Gb Ethernet NICs. In that case, the configuration would be as described previously: a single vSwitch with two uplinks, and traffic split across the uplinks by setting the NIC failover order on each port group.

Four NICs

In this case, traffic is split between two virtual switches, each with two uplinks. vSwitch0 handles management, IP storage, vMotion, and FT; vSwitch1 handles VM traffic. The port groups on vSwitch0 are configured to use a custom NIC failover order so that management and IP storage run on one of the uplinks and vMotion and FT run on the other uplink (with the ability for either group to fail over to the other uplink if necessary). Figure 5.13 illustrates this configuration.

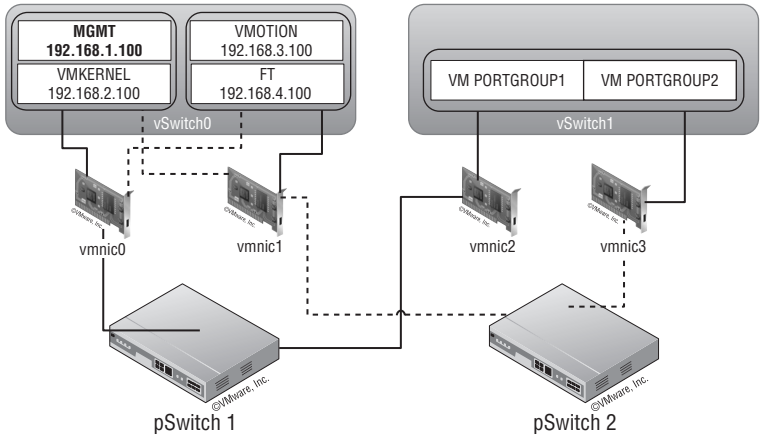


FIGURE 5.13 This ESXi host has two vSwitches, each with two uplinks.

The solid lines represent the primary links for each component, and the dotted lines represent the backup links. Here you provide redundancy for the VM traffic and some sort of redundancy for console and other traffic. This isn't ideal because too many components are competing for the same resources on vSwitch0.

NOTE Another way of handling a configuration with four NICs would be to use a single vSwitch with four uplinks, and use a different NIC failover configuration for each of the different types of traffic. This approach might provide a bit more flexibility than the two vSwitch/two uplink configuration.

Six NICs

Here, VM traffic goes through vSwitch1 (vmnic2 and vmnic3, both active). vSwitch0 handles management traffic on vmnic0 (active; vmnic1 on standby) and vMotion on vmnic1 (active; vmnic0 on standby). On vSwitch2, IP storage goes through vmnic4 (active; vmnic5 on standby) and FT through vmnic5 (active; vmnic4 on standby). This is all illustrated in Figure 5.14.

You have redundancy for all components, but the FT and IP storage traffic have only one NIC each. Depending on the size of the environment, this might not provide enough throughput for either of the two functions.

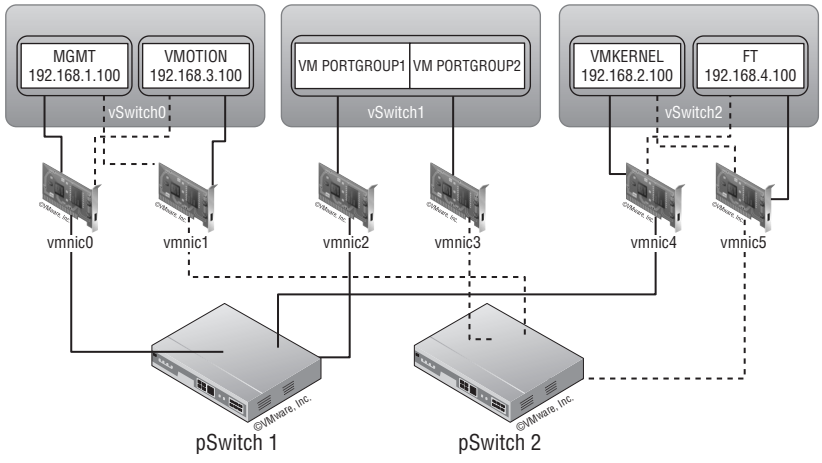


FIGURE 5.14
A six-NIC configuration provides redundancy for all components.

Eight NICs

In this scenario, VM traffic goes through vSwitch1 (vmnic2 and vmnic3, both active). On vSwitch0, management traffic goes through vmnic0 (vmnic1 on standby) with vMotion on vmnic1 (vmnic0 on standby). vSwitch2 handles IP storage (vmnic4, vmnic5, and vmnic6 active with vmnic7 on standby) and FT (on vmnic7, with vmnic6 and vmnic5 on standby). Figure 5.15 shows this configuration.

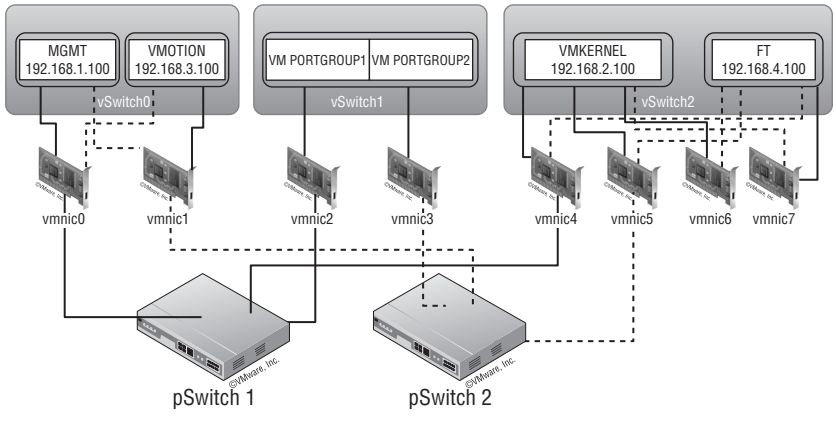


FIGURE 5.15
A configuration with eight NICs provides both redundancy and scalability.

Here you have redundancy for all components, and IP storage traffic has reasonable throughput using three NICs (keep in mind that, depending on the type of IP storage, it might be difficult to get it to fully utilize all three NICs). FT is limited to one NIC, which will limit the number of FT VMs you can host.

We hope these design scenarios give you some ideas of the flexibility you have in creating your vSphere network design. Although many different factors and considerations will shape your network design, vSphere offers a range of solutions and features to help ensure that you're able to satisfy the functional requirements.

Looking to the Future

Naturally, we must focus the majority of our discussion on what is available today for you to use in your vSphere design. However, it's also important to take a quick look at the near future and examine some standards and protocols that might affect how you do vSphere network designs:

Network Virtualization VMware's announcement of VXLAN at VMworld 2011, along with its 2012 purchase of Nicira, indicates that network virtualization in vSphere environments is something that is likely to become a reality in the near future. Network virtualization, in its simplest form, provides abstraction of the physical network topology to allow organizations to create multiple logical network topologies. Protocols such as VXLAN, Network Virtualization using Generic Routing Encapsulation (NVGRE), and a forthcoming IETF standard help enable this functionality. Through the creation of multiple logical network topologies, vSphere architects can connect geographically dispersed resources as if they were on the same Layer 2 segment and provide multi-tenancy support. We'll discuss a bit about VXLAN in Chapter 12, where we discuss design considerations for vCloud Director.

TRILL and Datacenter Fabrics TRILL is an emerging standard for Layer 2 multipathing, enabling an any-to-any datacenter fabric with multiple paths between nodes. No longer will vSphere architects and network engineers need to worry so much about STP; there is no need for STP on network segments where TRILL is used. Many of the techniques that we discussed in this chapter for optimizing certain types of traffic would no longer be necessary in TRILL-enabled environments. Although TRILL is a standard, adoption of the standard is still fairly low in most network environments.

LISP and Workload Mobility Locator/ID Separation Protocol (LISP) is another emerging IETF standard that addresses IP routing concerns arising from workload mobility (think vMotion). As many organizations look to enhance their ability to migrate workloads among multiple datacenters—sometimes geographically dispersed datacenters—the impact of that workload mobility on IP routing mechanisms has not yet been addressed. (There are a variety of other challenges today as well, but we're talking about the future, remember?) LISP attempts to provide a mechanism whereby these IP routing impacts are mitigated. It's unclear whether LISP (and related protocols) will see broad adoption; the rise of network virtualization and its ability to create logical network topologies somewhat duplicates LISP's functionality.

These are just three potential areas that might affect your vSphere network designs in the near future. We encourage you to stay closely connected to developments in the networking industry so you're prepared for the impacts to your designs as these developments unfold.

Summary

Designing any part of your virtual infrastructure isn't easy. It's a lengthy and complicated process, with many parameters that have to be taken into account.

Plan for that rainy day when things go bad. You may receive a small token of appreciation if you save a few bucks, but we assure you that you won't receive flowers if your environment crashes because you don't have the proper redundancy measures in place. You must consider your network standard—1Gb or 10Gb Ethernet—and learn the best way to set up your network for your specific environment.

Now, we'll move on to another critical part of the infrastructure: storage.

Chapter 6

Storage

The storage component of any design of a vSphere environment is commonly regarded as one of the most crucial to overall success. It's fundamental to the capabilities of virtualization. Many of the benefits provided by VMware's vSphere wouldn't be possible without the technologies and features offered by today's storage equipment.

Storage technologies are advancing at a tremendous pace, particularly notable for such a traditionally staunch market. This innovation is being fueled by new hardware capabilities, particularly with widespread adoption of flash-based devices. Pioneering software advancements are being made possible via the commoditization of many storage array technologies that were previously only available to top-tier enterprise storage companies. When VMware launched vSphere 5, it was colloquially referred to as *the storage release*, due to the sheer number of new storage features and the significant storage improvements it brought.

The storage design topics discussed in this chapter are as follows:

- Primary storage design factors to consider
- What makes for an efficient storage solution
- How to design your storage with sufficient capacity
- How to design your storage to perform appropriately
- Whether local storage is still a viable option
- Which storage protocol you should use
- Using multipathing with your storage choice
- How to implement vSphere 5 storage features in your design

Dimensions of Storage Design

In the past, those who designed, built, configured, and most importantly, paid for server storage were predominantly interested in how much space they could get for their dollar. Servers used local direct attached storage (DAS), with the occasional foray into two node clusters, where performance was limited to the local bus, the speed of the disks, and the RAID configuration chosen. These configurations could be tweaked to suit all but the most demanding of server-room requirements. If greater performance was required, companies scaled out with multiple servers; or if they had the need (and the cash!), they invested in expensive dedicated Fibre

Channel storage area network devices (SANs) with powerful array technologies. Times were relatively simple. CIOs cared about getting the most bang for their buck, and $/GB (cost per gigabyte) was what was on the storage planning table.

With the advent of virtualization, storage is now much more than just capacity. Arguably, the number of terabytes (TBs) that your new whizzy storage array can provide is one of the lesser interests when you're investigating requirements. Most shared storage units, even the more basic ones, can scale to hundreds of TBs.

Some of the intrinsic vSphere capabilities mean that storage is significantly more mobile than it was previously. Features such as Storage vMotion help abstract not just the server hardware but also the storage. Upgrading or replacing storage arrays isn't the thing of nightmares anymore; and the flexibility to switch in newer arrays makes the situation far more dynamic. Recent vSphere additions, such as Storage Distributed Resource Scheduler (Storage DRS) and Profile-Driven Storage, allow you to eke out even more value from your capital expenditure. Some of the innovative solutions around flash storage that are now available provide many options to quench virtualization's thirst for more input/output operations per second (IOPS).

Rather than intimidating or constraining the vSphere architect in you, this should open your mind to a world of new possibilities. Yes, there are more things to understand and digest, but they're all definable. Like any good design, storage requirements can still be planned and decisions made using justifiable, measurable analysis. Just be aware that counting the estimated number and size of all your projected VMs won't cut the mustard for a virtualized storage design anymore.

Storage Design Factors

Storage design comes down to three principle factors:

- Availability
- Performance
- Capacity

These must all be finely balanced with an ever-present fourth factor:

- Cost

Availability

Availability of your vSphere storage is crucial. Performance and capacity issues aren't usually disruptive and can be dealt with without downtime if properly planned and monitored. However, nothing is more noticeable than a complete outage. You can (and absolutely should) build redundancy in to every aspect of a vSphere design, and storage is cardinal in this equation. In a highly available environment, you wouldn't have servers with one power supply unit (PSU), standalone switches, or single Ethernet connections. Shared storage in its very nature is centralized and often solitary in the datacenter. Your entire cluster of servers will connect to this one piece of hardware. Wherever possible, this means every component and connection must have sufficient levels of redundancy to ensure that there are no single points of failure.

Different types of storage are discussed in this chapter, and as greater levels of availability are factored in, the cost obviously rises. However, the importance of availability should be overriding in almost any storage design.

Performance

Performance is generally less well understood than capacity or availability, but in a virtualized environment where there is significant scope for consolidation, it has a much greater impact. You can use several metrics, such as IOPS, throughput (measured in MBps), and latency, to accurately measure performance. These will be explained in greater depth later in the chapter.

This doesn't have to be the black art that many think it is—when you understand how to measure performance, you can use it effectively to underpin a successful storage design.

Capacity

Traditionally, capacity is what everyone thinks of as the focus for a storage array's principal specification. It's a tangible (as much as ones and zeros on a rusty-colored spinning disk can be), easily describable, quantitative figure that salesmen and management teams love. Don't misunderstand: it's a relevant design factor. You need space to stick stuff. No space, no more VMs. Capacity needs to be managed on an ongoing basis, and predicted and provisioned as required. However, unlike availability and performance, it can normally be augmented as requirements grow.

It's a relatively straightforward procedure to add disks and enclosures to most storage arrays without incurring downtime. As long as you initially scoped the fundamental parts of the storage design properly, you can normally solve capacity issues relatively easily.

Cost

Costs can be easy or difficult to factor in, depending on the situation. You may be faced with a set amount of money that you can spend. This is a hard number, and you can think of it as one of your constraints in the design.

Alternatively, the design may need such careful attention to availability, performance, and/or capacity that money isn't an issue to the business. You must design the best solution you can, regardless of the expense.

Although you may feel that you're in one camp or the other, cost is normally somewhat flexible. Businesses don't have a bottomless pit of cash to indulge infrastructure architects (unfortunately); nor are there many managers who won't listen to reasoned, articulate explanations as to why they need to adjust either their budget or their expectations of what can be delivered.

Generally, the task of a good design is to take in the requirements and provide the best solution for the lowest possible cost. Even if you aren't responsible for the financial aspects of the design, it's important to have an idea of how much money is available.

Storage Efficiency

Storage *efficiency* is a term used to compare cost against each of the primary design factors. Because everything relates to how much it costs and what a business can afford, you should juxtapose solutions on that basis.

Availability Efficiency

You can analyze availability in a number of ways. Most common service-level agreements (SLAs) use the term *9s*. The 9s refers to the amount of availability as a percentage of uptime in a year, as shown in Table 6.1.

TABLE 6.1: The 9s

AVAILABILITY %	DOWNTIME PER YEAR
90%	36.5 days
99%	3.65 days
99.5%	1.83 days
99.9%	8.76 hours
99.99%	52.6 minutes
99.999% ("5 nines")	5.26 minutes

Using a measurement such as the 9s can give you a quantitative level of desired availability; however, the 9s can be open to interpretation. Often used as marketing terminology, you can use the 9s to understand what makes a highly available system. The concept is fairly simple.

If you have a single item for which you can estimate how frequently it will fail (mean time between failures [MTBF]) and how quickly it can be brought back online after a failure (mean time to recover [MTTR]), then you can calculate the applicable 9s value:

$$\text{Availability} = ((\text{minutes in a year} - \text{average annual downtime in minutes}) / \text{minutes in a year}) \times 100$$

For example, a router that on average fails once every 3 years (MTBF) and that takes 4 hours to replace (MTTR) can be said to have on average an annual downtime of 75 minutes. This equates to

$$\text{Availability} = ((525600 - 75) / 525600) \times 100 = 99.986\%$$

As soon as you introduce a second item into the mix, the risk of failure is multiplied by the two percentages. Unless you're adding a 100 percent rock-solid, non-fallible piece of equipment (very unlikely, especially because faults are frequently caused by the operator), the percentage drops, and your solution can be considered less available.

As an example, if you have a firewall in front of the router, with the same chance of failure, then a failure in either will create an outage. The availability of that solution is halved: it's 99.972 percent, which means an average downtime of 150 minutes every year.

However, if you can add additional failover items in the design, then you can reverse the trend and increase the percentage for that piece. If you have two, then the risk may be halved. Add three, and the risk drops to one-third. In the example, adding a second failover router (ignoring the firewall) reduces the annual downtime to 37.5 minutes; a third reduces it to 25 minutes.

As you add more levels of redundancy to each area, the law of diminishing returns sets in, and it becomes less economical to add more. The greatest benefit is adding a second item, which is why most designs require at least one failover item at each level. If each router costs $5,000, the second one reduces downtime from a 1-router solution by 37.5 minutes (75 – 37.5). The third will only reduce it by a further 12.5 minutes (37.5 – 25), even though it costs as much as the

second. As you can see, highly available solutions can be very expensive. Less reliable parts tend to need even more redundancy.

During the design, you should be aware of any items that increase the possibility of failure. If you need multiple items to handle load, but any one of them failing creates an outage, then you increase the potential for failure as you add more nodes. Inversely, if the load is spread across multiple items, then this spreads the risk; therefore, any failures have a direct impact on performance.

Paradoxically, massively increasing the redundancy to increase availability to the magic "five 9s" often introduces so much complexity that things take a turn south. No one said design was easy!

You can also use other techniques to calculate high availability, such as MTBF by itself.

NOTE Remember that to a business, *uptime* may not mean the same thing as *availability*. For example, if performance is so bad as to make a solution unusable, then no one will be impressed by your zero-downtime figures for the month.

Also worthy of note is the ability to take scheduled outages for maintenance. Does this solution really need a 24/7 answer? Although a scheduled outage is likely to affect the SLAs, are there provisions for accepted regular maintenance, or are all outages unacceptable? Ordinarily, scheduled maintenance isn't considered against availability figures; but when absolute availability is needed, things tend to get very costly.

This is where availability efficiency is crucial to a good design. Often, there is a requirement to propose different solutions based on prices. Availability efficiency usually revolves around showing how much the required solution will cost at different levels. The 9s can easily demonstrate how much availability costs, when a customer needs defined levels of performance and capacity.

Performance Efficiency

You can measure performance in several ways. These will be explained further in this chapter; but the most common are IOPS, MBps, and latency in milliseconds (ms).

Performance efficiency is the cost per IOPS, per MBps, or per ms latency. IOPS is generally the most useful of the three; most architects and storage companies refer to it as $/IOPS. The problem is, despite IOPS being a measureable test of a disk, many factors in today's advanced storage solutions—such as RAID type, read and write cache, and tiering—skew the figures so much that it can be difficult to predict and attribute a value to a whole storage device.

This is where lab testing is essential for a good design. To understand how suitable a design is, you can use appropriate testing to determine the performance efficiency of different storage solutions. Measuring the performance of each option with I/O loads comparable to the business's requirements, and comparing that to cost, gives the performance efficiency.

Capacity Efficiency

Capacity efficiency is the easiest to design for. $/GB is a relatively simple calculation, given the sales listings for different vendors and units. Or so you may think.

The "Designing for Capacity" section of this chapter will discuss some of the many factors that affect the actual usable space available. Just because you have fifteen 1 TB disks doesn't mean you can store exactly 15 TB of data. As you'll see, several factors eat into that total significantly; but perhaps more surprising is that several technologies now allow you to get more for less.

Despite the somewhat nebulous answer, you can still design capacity efficiency. Although it may not necessarily be a linear calculation, if you can estimate your storage usage, you can predict your capacity efficiency. Based on the cost of purchasing disks, if you know how much usable space you have per disk, then it's relatively straightforward to determine $/GB.

Other Efficiencies

Before moving on, it's worth quickly explaining that other factors are involved in storage efficiencies. Availability, performance, and capacity features can all be regarded as capital expenditure (CAPEX) costs; but as the price of storage continues to drop, it's increasingly important to understand the operational expenditure (OPEX) costs. For example, in your design, you may wish to consider these points:

Watts/IOPS How much electricity does each disk use? Flash drives, although more expensive per GB, not only are cheaper per IOPS but also use significantly less electricity per IOPS.

Rack Usage Now that it's more common to use co-lo facilities, businesses are aware how much each U in a rack costs. Solutions that can combine very dense capacity (SATA) and very dense performance (flash) in the required mix can save leased space.

Management Overhead The cost of managing the storage may be difficult to quantify, but it can have a significant effect on the OPEX associated with a storage design. Later, this chapter discusses different designs and protocol choices. As you'll see, the protocol you opt for often comes down to its integration in your existing environment.

Flexibility One thing is certain: your design will never be perfect and must remain flexible enough to adapt to an ever-changing environment. Can the design accommodate more disks, different disks, multiple protocols and transport, upgrades, more hosts, and so on? If not, future OPEX costs may be more expensive than previously planned.

vSphere Storage Features

As vSphere has evolved, VMware has continued to introduce features in the platform to make the most of its available storage. Advancements in redundancy, management, performance control, and capacity usage all make vSphere an altogether more powerful virtualization stage. These technologies, explained later in the chapter, allow you to take your design beyond the capabilities of your storage array. They can complement and work with some available array features, making it easier to manage, augment some array features to increase that performance or capacity further, or simply replace the need to pay the storage vendor's premium for an array-based feature.

Despite the stunning new possibilities introduced with flash storage hardware, the most dramatic changes in storage are software based. Not only are storage vendors introducing great new options with every new piece of equipment, but VMware reinvents the storage landscape for your VMs with every release.

Designing for Capacity

An important aspect of any storage design involves ensuring that it has sufficient capacity not just for the initial deployment but also to scale up for future requirements. Before discussing what you should consider in capacity planning, let's review the basics behind the current storage options. What decisions are made when combining raw storage into usable space?

RAID Options

Modern servers and storage arrays use Redundant Array of Independent/Inexpensive Disks (RAID) technologies to combine disks into logical unit numbers (LUNs) on which you can store data. Regardless of whether we're discussing local storage; a cheap, dumb network-attached storage (NAS) or SAN device; or a high-end enterprise array, the principles of RAID and their usage still apply. Even arrays that abstract their storage presentation in pools, groups, and volumes use some type of hidden RAID technique.

The choice of which RAID type to use, like most storage decisions, comes down to availability, performance, capacity, and cost. In this section, the primary concerns are both availability and capacity. Later in the chapter, in the "Designing for Performance" section, we discuss RAID to evaluate its impact on storage performance.

Many different types of RAID (and non-RAID) solutions are available, but these examples cover the majority of cases that are used in VMware solutions. Figure 6.1 compares how different RAID types mix the data-to-redundancy ratio.

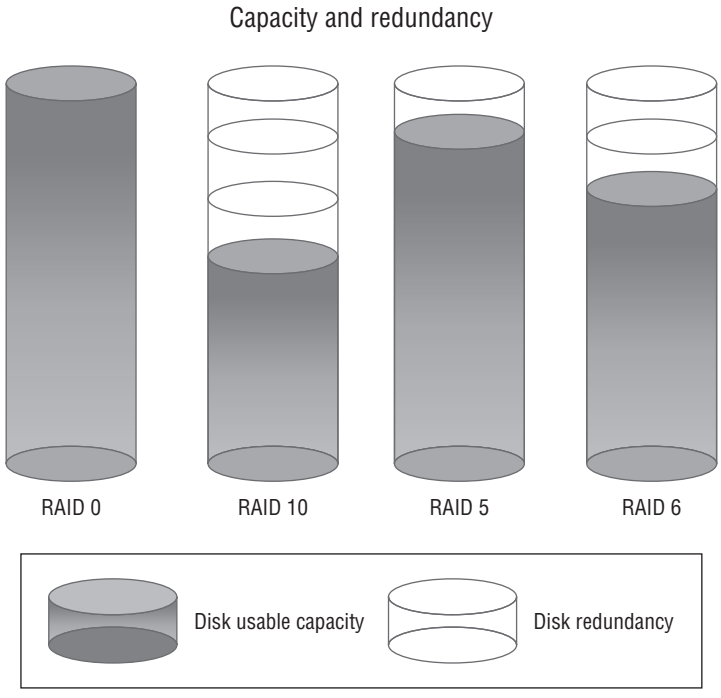


FIGURE 6.1 Capacity versus redundancy

RAID 0

RAID 0 stripes all the disks together without any parity or mirroring. Because no disks are lost to redundancy, this approach maximizes the capacity and performance of the RAID set. However, with no redundancy, just one failed disk will destroy all of your data. For this reason, RAID 0 isn't suitable for a VMware (or almost any other) production setting.

RAID 10

RAID 10 describes a pair of disks, or multiples thereof, that mirror each other. From an availability perspective, this approach gives an excellent level of redundancy, because every block of data is written to a second disk. Multiple disks can fail as long as one copy of each pair remains available. Rebuild times are also short in comparison to other RAID types. However, capacity is effectively halved; in every pair of disks, exactly one is a *parity* disk. So, RAID 10 is the most expensive solution.

Without considering performance, RAID 10 is useful in a couple of vSphere circumstances. It's often used in situations where high availability is crucial. If the physical environment is particularly volatile—for example, remote sites with extremes of temperature or humidity, ground tremors, or poor electrical supply—or if more redundancy is a requirement due to the importance of the data, then RAID 10 always provides a more robust solution. RAID 1 (two mirrored disks) is often used on local disks for ESXi's OS, because local disks are relatively cheap and capacity isn't normally a requirement when shared storage is available.

RAID 5

RAID 5 is a set of disks that stripes parity across the entire group using the equivalent of one disk (as opposed to RAID 4, which assigns a single specific disk for parity). Aside from performance differences, RAID 5 is a very good option to maximize capacity. Only one disk is lost for parity, so you can use $n - 1$ for data.

However, this has an impact on availability, because the loss of more than one disk at a time will cause a complete loss of data. It's important to consider the importance of your data and the reliability of the disks before selecting RAID 5. The MTBFs, rebuild times, and availability of spares/replacements are significant factors.

RAID 5 is a very popular choice for SCSI/SAS disks that are viewed as fairly reliable options. After a disk failure, RAID 5 must be rebuilt onto a replacement before a second failure. SCSI/SAS disks tend to be smaller in capacity and faster, so they rebuild much more quickly. Because SCSI/SAS disks also tend to be more expensive than their SATA counterparts, it's important to get a good level of capacity return from them.

With SAN arrays, it's common practice to allocate one or more spare disks. These spare disks are used in the event of a failure and are immediately moved in as replacements when needed. An advantage from a capacity perspective is that one spare can provide additional redundancy to multiple RAID sets.

If you consider your disks reliable, and you feel that two simultaneous failures are unlikely, then RAID 5 is often the best choice. After all, RAID redundancy should never be your last line of defense against data loss. RAID 5 provides the best capacity, with acceptable availability given the right disks and hot spares.

RAID 6

An increasingly popular choice among modern storage designs is RAID 6. It's similar in nature to RAID 5, in that the parity data is distributed across all member disks, but it uses the equivalent of two disks. This means it loses some capacity compared to RAID 5 but can withstand two disks failing in quick succession. This is particularly useful when you're creating larger RAID groups.

RAID 6 is becoming more popular as drive sizes increase (therefore increasing rebuild times), because MTBF drops as physical tolerances on disks become tighter, and as SATA drives become more pervasive in enterprise storage.

Other Vendor-Specific RAID Options

The basic RAID types mentioned cover most scenarios for vSphere deployments, but you'll encounter many other options when talking to different storage vendors. Many of these are technically very similar to the basic types, such as RAID-DP from NetApp. RAID-DP is similar to a RAID 6 group, but rather than the parity being distributed across all disks, RAID-DP uses two specific disks for parity (like RAID 4). The ZFS file system designed by Sun Microsystems (now Oracle), which includes many innovative storage technologies on top, uses a self-allocating disk mechanism not dissimilar to RAID 5, called RAID-Z. Although it differs in the way it writes data to disks, it uses the premise of a disk's worth of parity across the group like RAID 5. ZFS is used in many Solaris and BSD-based storage systems. Linux has a great deal of RAID, logical volume manager (LVM), and file system options, but due to licensing incompatibilities it has never adopted ZFS. Linux has a new file system called BTRFS that is set to compete directly with ZFS and is likely to be the basis of many new storage solutions in the near future as it stabilizes and features are quickly added.

Some storage arrays effectively make the RAID choices for you, by hiding the details and describing disks in terms of pools, volumes, aggregates, and so on. They abstract the physical layer and present the storage in a different way. This allows you to select disks on more user-friendly terms, while hiding the technical details. This approach may reduce the level of granularity that storage administrators are used to, but it also reduces complexity and arguably makes default decisions that are optimized for their purpose.

Basic RAID Storage Rules

The following are some additional basic rules you should follow in vSphere environments:

- Ensure that spares are configured to automatically replace failed disks.
- Consider the physical location of the hardware and the warranty agreement on replacements, because they will affect your RAID choices and spares policy.
- Follow the vendor's advice regarding RAID group sizes and spares.
- Consider the importance of the data, because the RAID type is the first defense against disk failures (but definitely shouldn't be the only defense). Remember that availability is always of paramount importance when you're designing any aspect of storage solutions.
- Replace failed disks immediately, and configure any phone-home options to alert you (or the vendor directly) as soon as possible.
- Use predictive failure detection if available, to proactively replace disks before they fail.

Estimating Capacity Requirements

Making initial estimates for your storage requirements can be one of the easier design decisions. Calculating how much space you really need depends on the tasks ahead of you. If you're

looking at a full virtualization implementation, converting physical servers to VMs, there are various capacity planners available to analyze the existing environment. If the storage design is to replace an existing solution that you've outgrown, then the capacity needs will be even more apparent. If you're starting anew, then you need to estimate the average VM, with the flexibility to account for unusually large servers (file servers, mailbox servers, databases, and so on).

In addition to the VMDK disk files, several additional pieces need space on the datastores:

VM Swap Space By default, a VM uses some disk space to create a swap area that is equal to the size of the VM's allocated RAM. Technically, if you a set a memory reservation on the VM, this swap allocation is reduced. This topic will be discussed more in Chapter 7, "Virtual Machines," but for the purposes of capacity planning, you can ignore the reservation.

Snapshots Many advanced features in vSphere use snapshotting functionality in addition to manually created snapshots. Backup tools, Storage vMotion, and others also create snapshots, which use extra space in the datastores. New to vSphere 5, each VM's disk snapshots are held in the same Virtual Machine File System (VMFS) volume as the disks themselves.

Templates and ISOs For even the smallest deployments, the use of templates provides a convenient method of creating new VMs and consistent builds. Storing the templates and ISO files on shared storage allows for all hosts in a cluster to use a single set, minimizing the need for individual copies on every host (which would increase the maintenance overhead).

TIP A good rule of thumb for estimating overhead is to add 25 percent to the size of the datastore to account for this overhead.

VMFS Capacity Limits

VMFS isn't the only storage option available for VMs, but it's by far the most popular. You can make different assumptions when using Network File System (NFS) datastores, and they will be discussed later in the chapter in "Choosing a Protocol." Using raw device mapping disks (RDMs) to store VM data is another option, but this is out of the scope of this chapter. Chapter 7 will look at RDMs in more detail; for the purposes of capacity planning for a storage array, consider their use frugally and note the requirement for separate LUNs for each RDM disk where needed.

VMFS itself is described in a little more detail in the "vSphere Storage Features" section later in this chapter, but it's worth detailing the impact that it can have on the LUN sizing at this point. VMFS-5 volumes can be up to 64 TB in size (as opposed to their predecessor, which was ≈ 2 TB), which allows for very high consolidation ratios. Whereas previously, anyone who wanted very large datastores looked to NFS options (although concatenating VMFS extents was technically possible), now block storage can have massive datastores, removing another potential constraint from your design. In reality, other performance factors will likely mean that most datastores should be created much smaller than this.

With VMFS-3, *extents* could be used to effectively grow the smaller datastores up to 64 TB. Extents are a concatenation of additional partitions to the first VMFS partition. This is no longer required for this purpose, but extents still exist as a useful tool. With VMFS-5, 32 extents are possible. The primary use case for extents today is to nondisruptively grow a volume. This can be a lifesaver if your storage array doesn't support growing LUNs online, so you can expand the VMFS volume. Instead, you can create additional LUNs, present them to the same hosts that can see the first VMFS partition, and add them as extents.

There are technical arguments why extents can be part of a design. The stigma regarding extents arose partially because they were used in cases where planning didn't happen properly, and somewhat from the belief that they would cause performance issues. In reality, extents can improve performance when each extent is created on a new physical LUN, thereby reducing LUN queues, aiding multipathing, and increasing throughput. Any additional LUNs should have the same RAIDing and use similar disk types (same speed and IOPS capability).

However, despite any potential performance benefits, there are still pitfalls involving extents that make them difficult to recommend. You must take care when managing the LUNs on the array, because taking just one of the extent LUNs offline is likely to affect many (if not all) of the VMs on the datastore. When you add LUNs to the VMFS volume, data from VMs can be written across all the extents. Taking one LUN offline can crash all the VMs stored on the volume—and pray that you don't delete the LUN as well. Most midrange SANs can group LUNs into logical sets to prevent this situation, but it still remains a risk that a single corrupt LUN can affect more VMs than normal. The *head* LUN (the first LUN) contains the metadata for all the extents. This one is particularly important, because a loss of the head LUN corrupts the entire datastore. This LUN attracts all the SCSI reservation locks on a non-VAAI-backed LUN.

Datastore clusters are almost the evolution of the extent, without the associated risks. If you have the licensing for datastore clusters and Storage DRS, you shouldn't even consider using extents. You still get the large single storage management point (with individual datastores up to 64 TB) and multiple LUN queues, paths, and throughput.

The most tangible limit on datastores currently is the size of the VMDK disks, which can only be up to 2 TB (2 TB minus 512 KB, to be exact). VMDKs on NFS datastore are limited in the same way. If you absolutely must have individual disks larger than 2 TB, some workarounds are as follows:

- Use the VM's guest OS volume management, such as Linux LVM or Windows dynamic disks, to combine multiple VMDK disks to make larger guest volumes.
- Use physical RDMs, which can be up to 64 TB (virtual RDMs are still limited to 2 TB).
- Use in-guest mounting of remote IP storage, such as an iSCSI LUN via a guest initiator or a mounted NFS export. This technique isn't recommended because the storage I/O is considered regular network traffic by the hypervisor and so isn't protected and managed in the same way.

Large or Small Datastores?

Just how big should you make your datastores? There are no hard-and-fast rules, but your decision relies on several key points. Let's see why you would choose one or the other:

A Few Large Datastores At first glance, having a few very large datastores would seem an obvious choice:

- You have fewer datastores and array LUNs to manage.
- You can create more VMs without having to make frequent visits to the array to provision new LUNs.
- Large VMDK disk files can be created.
- It allows more space for snapshots and future expansion.

A Lot of Small Datastores There are also some very good reasons not to max out every datastore:

- Better control of the space:
  - Having fewer VMs in each datastore means you have more granularity when it comes to RAID type.
  - Disk shares can be apportioned more appropriately.
  - ESXi hosts can use the additional LUNs to make better use of multipathing.
  - Storage DRS can more efficiently balance the disk usage and performance across more datastores.
- There is less contention on individual LUNs and storage processors (SPs), making for more balanced use of array performance.
- It lowers the likelihood that an out-of-control snapshot will take down large numbers of VMs.
- Arguably, you waste less space if the datastores are created more specifically for VMs. But depending on how you reserve space for snapshots, this can be negated by the requirement to keep a certain percentage or amount free on each datastore.

In reality, like most design decisions, the final solution is likely to be a sensible compromise of both extremes. Having one massive datastore would likely cause performance issues, whereas having a single datastore per VM would be too large an administrative overhead for most, and you'd soon reach the upper limit of 256 LUNs on a host.

The introduction of datastore clusters and Storage DRS helps to solve some of the conundrum regarding big or small datastores. These features can give many of the performance benefits of the smaller datastores while still having the reduced management overheads associated with larger datastores. We delve into datastore clusters and Storage DRS later in the chapter.

The size of your datastores will ultimately be impacted primarily by two elements:

Size of the VM's Disk Files If your VMs are very large, you'll need larger datastores. Smaller VMs need smaller datastores; otherwise, you might see overcommitment-based performance issues.

I/O Levels of Your VMs If you have VMs that use elevated amounts of I/O—for example, databases or Exchange or SharePoint servers—then you should reduce the number of VMs on each datastore (and in turn reduce their size) to avoid I/O contention and protect the VMs.

vSphere 5 *limits* your datastores to a voluminous 2,048 VMs, but consider that more a theoretical upper limit and not the number of VMs around which to create an initial design. Look at your VMs, taking into account the previous two factors, and estimate a number of VMs per datastore that you're comfortable with. Then, multiply that number by your average estimated VM size. Finally, add a fudge factor of 25 percent to account for short-term growth, snapshots, and VM swap files, and you should have an average datastore size that will be appropriate for the majority of your VMs. Remember, you may need to create additional datastores that are specially provisioned for VMs that are larger, are more I/O intensive, need different RAID requirements, or need increased levels of protection.

Fortunately, with the advent of Storage vMotion, moving your VMs to different-sized datastores no longer requires an outage.

VMFS Block Sizes

vSphere hosts prior to version 5 used VMFS-3, which could have one of several partition block sizes: 1, 2, 4, and 8 MB blocks. With most file systems, if you pick a large block size, then every file that's created, no matter how small, uses an entire single block. Ordinarily, a file system has to hold potentially millions of files, which can lead to excessive waste if you choose a block size that's too large for your needs. Because VMFS was designed specifically for storing VMs, the number of files usually numbers in the hundreds at most, so this wasn't too much of a concern.

Choosing a small block size with VMFS-3 limited your options when attempting to grow the VMs beyond the limit imposed by the smaller block size (for example, datastores with a 1 MB block size could only hold files up to 256 GB in size). This could prevent future growth and create problems committing snapshots that caused the disk to grow over the allowable size. Also, after you created the partition, you couldn't change it. To make a datastore bigger, you had to evacuate everything first, and then delete and re-create the entire datastore.

Another side effect of the differing block sizes is that Storage vMotion performance can be severely hampered if you're moving VMs between datastores that have different block sizes.

VMFS-5, the native datastore file system in vSphere 5, has a unified block size of 1 MB. There are no options—no decisions to make. This simplifies the datastore creation process and ensures that all datastores are created equally. Previous limits around the maximum file sizes are gone, and Storage vMotions can run unimpeded. It's one less design impact to ponder.

Thin Provisioning

The ability to thin-provision new VM disks from the vSphere client GUI was introduced in vSphere 4. You can convert existing VMs to thinly provisioned ones during Storage vMotions. Chapter 7 explains in more depth the practicalities of thin-provisioning VMs, but you need to make a couple of important design decisions when considering your storage as a whole.

Thin provisioning has been available on some storage arrays for years. It's one of the ways to do more for less, and it increases the amount of usable space from disks. Since the support of NFS volumes, thin provisioning has been available to ESXi servers. Basically, despite the guest operating system (OS) seeing its full allocation of space, the space is actually doled out only as required. This allows all the spare (wasted) space within VMs to be grouped together and used for other things (such as more VMs).

The biggest problem with any form of storage thin-provisioning is the potential for over-commitment. It's possible—and desirable, as long as it's controlled properly—to allocate more storage than is physically available (otherwise, what's the point?). Banks have operated on this premise for years. They loan out far more money than they have in their vaults. As long as everyone doesn't turn up at the same time wanting their savings back, everything is okay. If all the VMs in a datastore want the space owed to them at once, then you run into overcommitment problems. You've effectively promised more than is available.

To help mitigate the risk of overcommiting the datastores, you can use both the Datastore Disk Usage % and Datastore Disk Overallocation % alarm triggers in vSphere. Doing so helps proactively monitor the remaining space and ensures that you're aware of potential issues before they become a crisis. In the vSphere Client, you can at a glance compare the amounts provisioned against the amounts utilized and get an idea of how thinly provisioned your VMs are.

Many common storage arrays now support VMware's vStorage APIs for Array Integration (VAAI). This VAAI support provides several enhancements and additional capabilities, which are explained later in the chapter. But pertinent to the thin-provisioning discussion is the ability of VAAI-capable arrays to allow vSphere to handle thin provisioning more elegantly.

With VAAI arrays, vSphere 5 can also:

- Tell the array to reclaim dead space created when files are deleted from a datastore. Ordinarily, the array wouldn't be aware of VMs you deleted or migrated off a datastore via Storage vMotion (including Storage DRS). VAAI informs the array that those blocks are no longer needed and can be safely reclaimed. This feature must be manually invoked from the command line and is discussed later in the VAAI section.
- Provide better reporting and deal with situations where thin provisioning causes a data store to run out of space. Additional advanced warnings are available by default (when they hit 75 percent full), and VMs are automatically paused when space is no longer available due to overcommitment.

The take-home message is, when planning to use thin provisioning on the SAN, look to see if your storage arrays are VAAI capable. Older arrays may be compatible but require a firmware upgrade to the controllers to make this available. When you're in the market for a new array, you should check to see if this VAAI primitive is available (some arrays offer compatibility with only a subset of the VAAI primitives).

Why do this? At any one time, much of the space allocated to VMs is sitting empty. You can save space, and therefore money, on expensive disks by not providing all the space at once. It's perfectly reasonable to expect disk capacity and performance to increase in the future and become less expensive, so thin provisioning is a good way to hold off purchases as long as possible. As VMs need more capacity, you can add it as required. But doing so needs careful monitoring to prevent problems.

Should You Thin-Provision Your VMs?

Sure, there are very few reasons not to do this, and one big, fat, money-saving reason to do it. As we said earlier, thin provisioning requires careful monitoring to prevent out-of-space issues on the datastores. vCenter has built-in alarms that you can easily configure to alert you of impending problems. The trick is to make sure you'll have enough warning to create more datastores or move VMs around to avert anything untoward. If that means purchasing and fitting more disks, then you'd better set the threshold suitably low.

As we've stated, there are a few reasons not to use vSphere thin provisioning:

- It can cause overcommitment.
- It prevents the use of eager-zeroed thick VMDK disks, which can increase write performance (Chapter 7 explains the types of VM disks in more depth).
- It creates higher levels of consolidation on the array, increasing the I/O demands on the SPs, LUNs, paths, and so on.

- Converting existing VMs to thin-provisioned ones can be time-consuming.
- You prefer to use your storage array's thin provisioning instead.

There are also some situations where it isn't possible to use thin-provisioned VMDK files:

- Fault-tolerant (FT) VMs
- Microsoft clustering shared disks

Does Thin Provisioning Affect the VM's Performance?

vSphere's thin provisioning of VM disks has been shown to make no appreciable difference to their performance, over default VMDK files (zeroed thick). It's also known that thin provisioning has little impact on file fragmentation of either the VMDK files or their contents. The concern primarily focused around the frequent SCSI locking required as the thin disk expanded; but this has been negated through the use of the new Atomic Test & Set (ATS) VAAI primitive, which dramatically reduces the occasions that LUN is locked.

If Your Storage Array Can Thin-Provision, Should You Do It on the Array, in vSphere, or Both?

Both array and vSphere thin provisioning should have similar results, but doing so on the array can be more efficient. Thin provisioning on both is likely to garner little additional saving (a few percent, probably), but you double the management costs by having to babysit two sets of storage pools. By thin-provisioning on both, you expedite the rate at which you can become oversubscribed.

The final decision on where to thin-provision disks often comes down to who manages your vSphere and storage environment. If both are operationally supported by the same team, the choice is normally swayed by the familiarity of the team with both tools. Array thin-provisioning is more mature, and arguably a better place to start; but if your team is predominantly vSphere focused and the majority of your shared storage is used by VMs, then perhaps this is where you should manage it. Who do you trust the most with operational capacity management issues—the management tools and processes of your storage team, or those of your vSphere team?

Data Deduplication

Some midrange and enterprise storage arrays offer what is known as *data deduplication*, often shortened to *dedupe*. This feature looks for common elements that are identical and records one set of them. The duplicates can be safely removed and thus save space. This is roughly analogous to the way VMware removes identical memory blocks in its transparent page sharing (TPS) technique.

The most common types of deduplication are as follows:

File-Level Deduplication The more rudimentary type of array deduplication is known as *file-level* or *single-instance* storage. This looks for identical (not similar—absolutely identical) files spread across the file system and removes duplicate copies. The concept is akin to hard links on a Unix file system, except that each copy of the file is considered a separate entity.

As an example, a company's file server may have 10 copies of the same 5 MB newsletter in 10 separate home folders. File-level deduplication recognizes them as identical files and only needs to store one 5 MB instance instead of the original 50 MB. Microsoft Exchange has been

using this technique from version 5.5 through 2007 on its mailbox stores, to dedupe large email attachments that are replicated many times (interestingly, this functionality has been removed from Exchange 2010 because Microsoft felt it affected performance too much).

In a VMware-centric world, file-level deduplication is usually fairly unhelpful. The arrays look on the file system (NFS in the vSphere context) and find very large VMDK files that despite containing somewhat similar data are almost never exactly identical.

Block-Level Deduplication This more advanced type of deduplication can work at the block level. It has the same idea of finding identical chunks of disk but does so at a much lower level. Block-level deduplication doesn't need to understand the file system, so the technique can usually be applied to block and file storage and can see inside VMDK files.

In a vSphere setup, where VMs are often provisioned from a small set of templates, block-level deduplication can normally provide good space reductions. It isn't uncommon to remove from 25 percent to even 50 percent or more on some datastores. VMs that are very similar—for example, virtual desktops—can benefit greatly from this type of deduplication, with savings of more than 75 percent.

Deduplication can be done inline or post-process. *Inline* means the data is checked for duplicates as it's being written (synchronously). This creates the best levels of space reduction; but because it has a significant impact in I/O performance, it's normally used only in backup and archiving tools. Storage arrays tend to use post-process deduplication, which runs as a scheduled task against the data (asynchronously). Windows Server 2012's built-in deduplication is run as a scheduled task. Even post-process deduplication can tax the arrays' CPUs and affect performance, so you should take care to schedule these jobs only during times of lighter I/O.

It's also worth noting that thin provisioning can negate some of the benefits you see with block-level deduplication, because one of the big wins normally is deduplicating all the empty zeros in a file system. It isn't that you don't see additional benefits from using both together; just don't expect the same savings as you do on a thickly provisioned LUN or VMDK file.

Array Compression

Another technique to get more capacity for less on storage arrays is compression. This involves algorithms that take objects (normally files) and compress them to squash out the repeating patterns. Anyone who has used WinZip or uncompressed a tarball will be familiar with the concept of compression.

Compression can be efficient in capacity reduction, but it does have an impact on an array's CPU usage during compression, and it can affect the disk-read performance depending on the efficiency of the on-the-fly decompression. Traditionally the process doesn't affect the disk writes, because compression is normally done as a post process. Due to the performance cost, the best candidates for compression are usually low I/O sources such as home folders and archives of older files.

With the ever-increasing capabilities of array's CPUs, more efficient compression algorithms, and larger write caches, some new innovative vendors can now compress their data inline. Interestingly, this can improve write performance, because only compressed data is written to the slower tiers of storage. The performance bottleneck on disk writes is usually the point when the data has to be written to disk. By reducing the amount of writes to the spinning disks, the effective efficiency can be increased, as long as the CPUs can keep up with processing the ingress of data.

Downside of Saving Space

There is a downside to saving space and increasing your usable capacity? You may think this is crazy talk; but as with most design decisions, you must always consider the practical impacts. Using these newfangled technological advances will save you precious GB of space, but remember that what you're really doing is consolidating the same data but using fewer spindles to do it. Although that will stave off the need for more capacity, you must realize the potential performance repercussions. Squeezing more and more VMs onto a SAN puts further demands on limited I/O.

Designing for Performance

Often, in a heavily virtualized environment, particularly one that uses some of the space-reduction techniques just discussed, a SAN will hit performance bottlenecks long before it runs out of space. If capacity becomes a problem, then you can attach extra disks and shelves. However, not designing a SAN for the correct performance requirements can be much more difficult to rectify. Upgrades are usually prohibitively expensive, often come with outages, and always create an element of risk. And that is assuming the SAN can be upgraded.

Just as with capacity, performance needs are a critical part of any well-crafted storage design.

Measuring Storage Performance

All the physical components in a storage system plus data characteristics combine to provide the resulting performance. You can use many different metrics to judge the performance levels of a disk and the storage array, but the three most relevant and commonly used are as follows:

Disk Latency Latency is measured in ms and shows the time from the storage request being made to the data then being read or written. A disk's latency is determined by the spin-up time, the seek time, and the rotational latency of the disk. Depending on where the disk latency is measured, this may include the time it takes to get to the disk and back; for example, disk latency as measured on the ESXi host includes everything from the point it leaves the hypervisor to when it returns.

Bandwidth Bandwidth is normally measured as MBps, and it shows the peak rate at which data can move to and from the storage. How quickly data can be read from and written to disk or cache is the most fundamental issue, although storage arrays have a number of optimizations that can significantly increase the numbers.

IOPS IOPS is probably the most-quoted storage performance statistic. In its most basic form, it's a benchmark of how many read and write commands can be executed in a second, although as you'll see, it can be affected by many other factors. Latency, throughput, type of I/O (read versus write, sequential versus random), I/O size, and the rotational speed of the disks all affect the IOPS value. This allows you to predict the performance results of disks and to design the storage accordingly.

How to Calculate a Disk's IOPS

To calculate the potential IOPS from a single disk, use the following equation:

$$\text{IOPS} = 1 \,/\, (\text{rotational latency} + \text{average read/write seek time})$$

For example, suppose a disk has the following characteristics:

Rotational latency: 2 ms

Read latency: 4 ms

Write latency: 5 ms

If you expect the usage to be around 75 percent reads and 25 percent writes, then you can expect the disk to provide an IOPS value of

$$1 / (0.002 + 0.00425) = 160 \text{ IOPS}$$

What Can Affect a Storage Array's IOPS?

Looking at single-disk IOPS is relatively straightforward. However, in a vSphere environment, single disks don't normally provide the performance (or capacity or redundancy) required. So, whether the disks are local DAS storage or part of a NAS/SAN device, they will undoubtedly be aggregated together. Storage performance involves many variables. Understanding all the elements and how they affect the resulting IOPS available should clarify how an entire system will perform.

Disks

The biggest single effect on an array's IOPS performance comes from the disks themselves. They're the slowest component in the mix, with most disks still made from mechanical moving parts. Each disk has its own physical properties, based on the number of platters, the rotational speed (RPMs), the interface, and so on; but disks are predicable, and you can estimate any disk's IOPS.

The sort of IOPS you can expect from a single disk is shown in Table 6.2.

Table 6.2: Average IOPS per disk type

RPM	IOPS
SSD (SLC)	6,000–50,000
SSD (MLC)	1,000+ (modern MLC disks vary widely; check the disk specifications and test yourself)
15 K (normally FC/SAS)	180
10 K (normally FC/SAS)	130
7.2 K (normally SATA)	80
5.4 K (normally SATA)	50

Solid-state drive (SSD) disks, sometimes referred to as flash drives, are viable options in storage arrays. Prices have dropped rapidly, and most vendors provide hybrid solutions that

include them in modern arrays. The IOPS value can vary dramatically based on the generation and underlying technology such as multi-level cell (MLC) or the faster, more reliable single-level cell (SLC). If you're including them in a design, check carefully what sort of IOPS you'll get. The numbers in Table 6.2 highlight the massive differential available.

Despite the fact that flash drives are approximately 10 times the price of regular hard disk drives, they can be around 50 times faster. So, for the correct usage, flash disks can provide increased efficiency with more IOPS/$. Later in this section, we'll explore some innovative solutions using flash drives and these efficiencies.

RAID Configuration

Creating RAID sets not only aggregates the disks' capacity and provides redundancy, but also fundamentally changes their performance characteristics (see Figure 6.2):

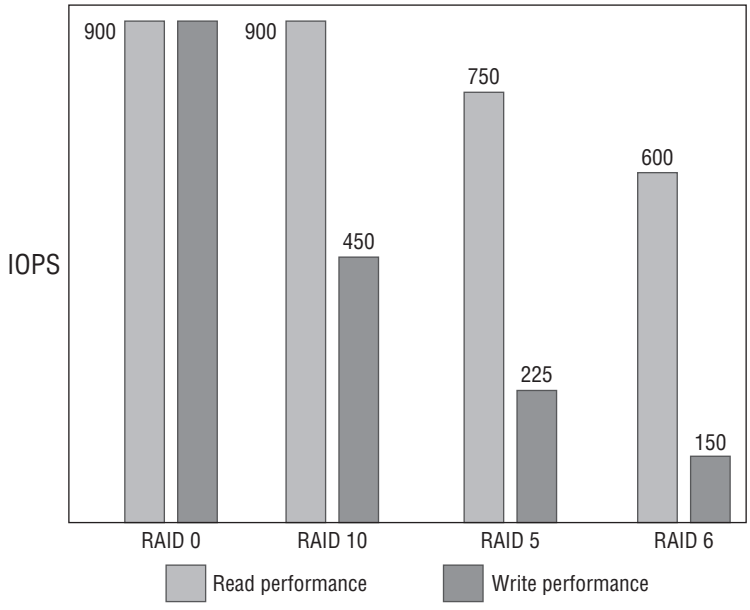


Figure 6.2 Performance examples

RAID 0 The simplest example is for RAID 0. Suppose you take two identical disks, each with an IOPS value of 150, and create a RAID 0 group. (For simplicity, let's assume that the read and write IOPS are the same. Normally, writes are more expensive, so the write IOPS are usually lower.) The result is a disk set that can provide 300 IOPS for both reads and writes. A RAID 0 set of 6 disks gives 900 IOPS ($n \times$ IOPS). But not so fast. Remember, you shouldn't use RAID 0 anywhere near your vSphere environment, because there is no redundancy. If one disk is lost, then so is all the data on all the disks.

RAID 10 RAID 10 provides excellent read performance, which generally goes up in a linear fashion according to the number of disks. With mirrored copies of the data, it can read from the mirrored disks in parallel. With the data striped across the mirrors, you can expect read IOPS approximately equal to the number of disks.

When writing to disk, RAID 10 needs to write to only two disks at a time. The data can still be striped, so you should see performance at around the combined IOPS of half the disks in a set.

For the previous example, if you have a RAID 10 set of 6 disks, it will in theory give you approximately 900 read IOPS ($n \times$ IOPS) but only 450 write IOPS (($n \times$ IOPS) / 2). If you understand your split of reads and writes, you can apportion them appropriately and work out the IOPS value depending on different disk amounts.

RAID 5 RAID 5 is often used because it's a good compromise for the sake of capacity: only one disk is lost to parity. Performance for reads remains good, because all but one of the disks can be simultaneously read from. However, RAID 5 write performance is much lower, because for every write I/O, it needs four actual operations. It needs to read the old data, then read the parity, then write the new data, and then write the new parity. This is known as the *RAID write penalty*.

For example, if you have 150 IOPS disks in a set of 6 disks as RAID 5, you should see read performance of 750 IOPS (($n - 1$) $\times$ IOPS) but write performance of only 225 IOPS (($n \times$ IOPS) / 4).

Additionally, when a disk fails, the set has to run in a degraded mode. Performance tapers off because all the blocks from the failed disk have to be calculated through the parity until the failed disk is replaced and rebuilt. After the failed disk is replaced, all the parity bits must be updated. This explains why failed RAID 5 disks take much longer to replace than failed disks in a mirrored set. Also remember that as the disk sizes increase and the RAID set contains more disks, rebuild times become even longer.

RAID 6 RAID 6 has even less performance than RAID 5 but offers greater protection. In the same example of 6 disks, RAID 6 gives 600 read IOPS (($n - 2$) $\times$ IOPS) but only 150 write IOPS (a penalty of 6 operations for every I/O) (($n \times$ IOPS) / 6). Despite this, RAID 6 is becoming increasingly popular, because it provides reasonably good capacity and provides better protection than RAID 5. With the increased strain that a RAID 5 rebuild puts on the remaining disks, in conjunction with much larger disks (causing even longer rebuild times) and the use of cheaper SATA disks, more and more vendors are recommending RAID 6 as a standard.

Interfaces

The *interface* is the physical connection from the disks. The disks may be connected to a RAID controller in a server, a storage controller, or an enclosure's backplane. Several different types are in use, such as IDE, SATA, SCSI, SAS, and FC, and each has its own standards with different recognized speeds. For example, SATA throughput is 1.5 Gbps, SATA II is backward compatible but qualified for 3 Gbps, and SATA III ups this to 6 Gbps.

Controllers

Controllers sit between the disks and servers, connected via the disk (and enclosure) interfaces on one side and the connectors to the server on the other. Manufacturers may refer to them as *controllers*, but the terms *SPs* and *heads* are often used in SAN hardware. Redundancy is often provided by having two or more controllers in an array.

Controllers are really mini-computers in themselves, running a customized OS. They're responsible for most of the special features available today, such as deduplication, failover, multipathing, snapshots, replication, and so on. Onboard server controllers and SAN controllers present their storage as block storage (raw LUNs), whereas NAS devices present their storage as a usable file system such as NFS. However, the waters become a little murky as vendors build NAS facilities into their SANs and vice versa.

Controllers almost always use an amount of non-volatile memory to cache the data before destaging it to disk. This memory is orders of magnitude faster than disks and can significantly improve IOPS. The cache can be used for writes and reads, although write cache normally has the most significance. Write cache allows the incoming data to be absorbed very quickly and then written to the slower disks in the background. However, the size of the cache limits its usefulness, because it can quickly fill up. At that point, the IOPS are again brought down to the speed of the disks, and the cache needs to wait to write the data out before it can empty itself and be ready for new data.

Controller cache helps to alleviate some of the effect of RAID write penalties mentioned earlier. It can collect large blocks of contiguous data and write them to disk in single operation. The earlier RAID calculations are often changed substantially by controllers; they can have a significant effect on overall performance.

Transport

The term *transport* in this instance describes how data gets from the servers to the arrays. If you're using a DAS solution, this isn't applicable, because the RAID controller is normally mounted directly to the motherboard. For shared storage, however, a wide variety of technologies (and therefore design decisions) are available. Transport includes the protocol, the topology, and the physical cables/connectors and any switching equipment used. The protocol you select determines the physical aspects, and you can use a dizzying array of methods to get ones and zeros from one rack to another.

Later in the chapter in "Choosing a Protocol," we'll examine the types of protocols in more depth, because it's an important factor to consider when you're designing a storage architecture. Each protocol has an impact on how to provide the required redundancy, multipathing options, throughput, latency, and so on. But suffice it to say, the potential storage protocols that are used in a vSphere deployment are Fibre Channel (FC), FCoE, iSCSI, and NFS.

Other Performance Factors to Consider

In addition to the standard storage components we've mentioned, you can customize other aspects to improve performance.

Queuing

Although block storage, array controllers, LUNs, and host bus adapters (HBAs) can queue data, there can still be a bottleneck from outstanding I/O. If the array can't handle the level of IOPS, the queue fills faster than it can drain. This queuing causes latency, and excessive amounts can be very detrimental to overall performance. When the queue is full, the array sends I/O-throttling commands back to the host's HBAs to slow down the traffic. The amount of queuing, or *queue depth*, is usually configurable on devices and can be optimized for your requirements. The QUED column in esxtop shows the queuing levels in real time.

Each LUN gets its own queue, so changes to HBA queue depths can affect multiple LUN queues. If multiple VMs are active on a LUN, you also need to update the `Disk.SchedNumReqOutstanding` value. This is the level of active disk requests being sent to the LUN by the VMs. Normally, that value should equal the queue-depth number. (VMware's Knowledge Base article 1267 explains how to change these values: `http://kb.vmware.com/kb/1267`.)

The default queue-depth settings are sufficient for most use cases. However, if you have a small number of very I/O-intensive VMs, you may benefit from increasing the queue depth. Take care before you decide to change these values; it's a complex area where good intentions can lead to bigger performance issues. Increasing queue depth on the hosts unnecessarily can create more latency than needed. Often, a more balanced design, where VM loads are spread evenly across HBAs, SPs, and LUNs, is a better approach than adjusting queue-depth values. You should check the array and the HBA manufacturer's documentation for their recommendations.

Partition Alignment

Aligning disk partitions can make a substantial difference—up to 30 percent in the performance of some operations. When partitions are aligned properly, it increases the likelihood that the SAN controller can write a full stripe. This reduces the RAID write penalty that costs so much in terms of IOPS.

You need to address partition alignment on vSphere in two areas: the VMFS volume and the guest OS file system. When you create VMFS datastores from within the vSphere client, it aligns them automatically for you. In most cases, local VMFS isn't used for performance-sensitive VMs; but if you're planning to use this storage for such tasks, you should create the partition in the client.

The most likely place where partitions aren't aligned properly is in the guest OSes of the VMs. Chapter 7 will have a more in-depth examination of this topic and how to align or realign a VM's partitions.

Workload

Every environment is different, and planning the storage depends on what workloads are being generated. You can optimize storage for different types of storage needs: the ratio of reads to writes, the size of the I/O, and how sequential or random the I/O is.

Writes always take longer than reads. Individual disks are slower to write data than to read them. But more important, the RAID configurations that have some sort of redundancy always penalize writes. As you've seen, some RAID types suffer from write penalties significantly more than others. If you determine that you have a lot of writes in your workloads, you may attempt to offset this with a larger controller cache. If, however, you have a negligible number of writes, you may choose to place more importance on faster disks or allocate more cache to reads.

The size of I/O requests varies. Generally speaking, larger requests are dealt with more quickly than small ones. You may be able to optimize certain RAID settings on the array or use different file-system properties.

Sequential data can be transferred to disk more quickly than random data because the disk heads don't need to move around as much. If you know certain workloads are very random, you can place them on the faster disks. Alternatively, most controller software attempts to derandomize the data before it's destaged from cache, so your results may vary depending on the vendor's ability to perform this efficiently.

VMs

Another extremely important aspect of your design that impacts your storage performance is the VMs. Not only are they the customers for the storage performance, but they also have a role to play in overall speed.

Naturally, this will be discussed in more depth in Chapter 7, but it's worth noting the effect it can have on your storage design. How you configure a VM affects its storage performance but can also affect the other VMs around it. Particularly I/O-intensive VMs can affect other VMs on the same host, datastore (LUN), path, RAID set, or controller. If you need to avoid IOPS contention for a particular VM, you can isolate it, thus guaranteeing it IOPS. Alternatively, if you wish to reduce the impact of I/O from VMs on others, you can spread the heavy hitters around, balancing the load. Chapter 8, "Datacenter Design," also looks at how disk shares can spread I/O availability.

We've already mentioned guest OS alignment, but you can often tune the guest OS to the environment for your storage array. The VM's hardware and drivers also have an impact on how it utilizes the available storage. How the data is split across VMDKs, whether its swapfile is segregated to a separate VMDK, and how the balance of different SAN drive types and RAIDing are used for different VM disks all affect the overall storage design.

vSphere Storage Performance Enhancements

Later in the chapter, we look at the result of several vSphere technologies that can have an impact on the performance of your VMs. Features such as Storage I/O Control (SIOC), VAAI, and Storage DRS can all improve the VMs' storage performance. Although these don't directly affect the array's performance per se, by optimizing the VMs' use of the array, they provide a more efficient and overall performant system.

Newer Technologies to Increase Effective IOPS

Recently, many SAN vendors have been looking at ways to improve the performance of their arrays. This is becoming important as the density of IOPS required per disk has risen sharply. This jump in the need for IOPS is partly because of the consolidation that vSphere lends itself to, and partly due to advancements in capacity optimizations, such as deduplication.

Write Coalescing

Coalescing is a function of most SPs to improve the effective IOPS. It attempts to take randomized I/O in the write cache and reorganize it quickly into more sequential data. This allows it to be more efficiently striped across the disks and cuts down on write latency. By its very nature, coalescing doesn't help optimize disk reads, so it can only help with certain types of I/O.

Large Cache

Today's controller cache can vary from around 256 MB on a server's RAID controller to hundreds of gigabytes on larger enterprise SANs.

Some SAN vendors have started to sell add-on cards packed with terabytes of nonvolatile memory. These massive cache cards are particularly helpful in situations where the data is being compressed heavily and IOPS/TB are very high. A good example is virtual desktop infrastructure (VDI) workloads such as VMware View deployments.

Another approach is to augment the existing controller cache with one or more flash drives. These aren't as responsive as the onboard memory cache, but they're much less expensive and can still provide speeds that are exponentially (at least 50 times) more than the SAS/SATA disks they're cache for. This relatively economical option means you can add terabytes of cache to SANs.

These very large caches are making massive improvements to storage arrays' IOPS. But these improvements can only be realized in certain circumstances, and it's important that you consider your own workload requirements.

The one criticism of this technique is that it can't preemptively deal with large I/O requests. Large cache needs a period of time to *warm up* when it's empty, because although you don't want to run out of cache, it isn't very useful if it doesn't hold the data you're requesting. After being emptied, it takes time to fill with suitable requested data. So, for example, even though SP failover shouldn't affect availability of your storage, you may find that performance is heavily degraded for several hours afterward as the cache refills.

Cache Prefetch

Some controllers can attempt to prefetch data in their read caches. They look at the blocks that are being requested and try to anticipate what the next set of blocks might be, so they're ready if a host subsequently requests it. Vendors use various algorithms, and cache prefetch relies on the sort of workloads presented to it. Some read the next set of blocks; others do it based on previous reads. This helps to deliver the data directly from the cache instead of having to wait for slower disks, thus potentially improving response time.

Cache Deduplication

Cache deduplication does something very similar to disk deduplication, in that it takes the contents of the cache's data and removes identical blocks. It effectively increases the cache size and allows more things to be held in cache. Because cache is such a critical performance enhancement, this extra cache undoubtedly helps improve the array's performance. Cache deduplication can be particularly effective when very similar requests for data are being made, such as VDI boot storms or desktop recomposes.

Tiering

Another relatively new innovation on midrange and enterprise SANs is the tiering of disks. Until recently, SANs came with 10 K or 15 K drives. This was the only choice, along with whatever RAIDing you wanted to create, to divide the workload and create different levels of performance. However, SATA disks are used increasingly, because they have large capacity and are much less expensive. Add to that the dramatic drop in prices for flash drives, which although smaller provide insane levels of performance, and you have a real spread of options. All of these can be mixed in different quantities to provide both the capacity and the performance required.

Initially, only manual tiering was available: SAN administrators created disk sets for different workloads. This was similar to what they did with drive speeds and different types of RAID. But now you have a much more flexible set of options with diverse characteristics.

Some storage arrays have the ability to automate this tiering, either at the LUN level or down to the block level. They can monitor the different requests and automatically move the more frequently requested data to the faster flash disks and the less requested to the slower but cheaper

SATA disks. You can create rules to ensure that certain VMs are always kept on a certain type of disk, or you can create schedules to be sure VMs that need greater performance at set times are moved into fast areas in advance.

Automatic tiering can be very effective at providing extra IOPS to the VMs that really need it, and only when they need it. Flash disks help to absorb the increase in I/O density caused by capacity-reduction techniques. Flash disks reduce the cost of IOPS, and the SATA disks help bring down the cost of the capacity.

Host-Based Flash Cache

An increasingly popular performance option is host-based caching cards. These are PCIe-based flash storage, which due to the greater throughput available on the PCIe bus are many times faster than SATA- or SAS-based SSD flash drives. At the time of writing, the cards offer hundreds of GBs of storage but are largely read cache. Current market examples of this technology are the Fusion-io cards and EMC's VFCache line.

Host-based flash cache is similar to the large read cache options that are available on many of the mainstream storage arrays, but being host-based the latency is extremely low (measured in microseconds instead of milliseconds). The latency is minimal because once the cache is filled, the requests don't need to traverse the storage network back to the SAN. However, instead of centralizing your large read cache in front of your array, you're dispersing it across multiple servers. This clearly has scalability concerns, so you need to identify the top-tier workloads to run on a high performance cluster of servers. Currently the majority of the PCIe cards sold are only for rack servers; blade-server mezzanine cards aren't generally available, so if an organization is standardizing on blades, it needs to make exceptions to introduce this technology or wait until appropriate cards become available.

Most PCIe flash-based options are focused as read-cache devices. Many offer write-through caching; but because most use nonpersistent storage, it's advisable to only use this write cache for ephemeral data such as OS swap space or temporary files. Even if you trust this for write caching, or the device has nonvolatile storage, it can only ever act as a write buffer to your back-end storage array. This is useful in reducing latency and absorbing peaks but won't help with sustained throughput. Buffered writes eventually need to be drained into the SAN; and if you saturate the write cache, your performance becomes limited by the underlying ingest rate of the storage array.

PCIe flash-based cache is another option in the growing storage tier mix. It has the potential to be very influential if the forthcoming solutions can remain array agnostic. If it's deeply tied to a vendor's own back-end SAN, then it will merely be another tiered option. But if it can be used as a read cache for any array, then this could be a boon for customers who want to add performance at levels normally available only in the biggest, most expensive enterprise arrays. Eventually, PCIe flash read caches are likely to be overtaken by the faster commodity RAM-based software options, but it will be several years before those are large enough to be beneficial for wide-scale uptake. In the meantime, as prices drop, PCIe cards and their driver and integration software will mature, and the write-buffering options will allow them to develop into new market segments.

RAM-Based Storage Cache

A new option for very high I/O requirements is appliances that use server RAM to create a very fast read cache. This is particularly suitable for VDI workloads where the desktop images are

relatively small in size, are good candidates for in-cache deduplication, but generate a lot of I/O. The entire desktop image can be cached in RAM (perhaps only 15 GB worth for a Windows 7 image), and all the read requests can be served directly from this tier. This can be a dedicated server for caching or a virtual appliance that grabs a chunk of the ESXi server's RAM. RAM is orders of magnitude faster than local SAS/SATA SSD or even PCIe flash, so performance is extremely impressive and helps to reduce the high IOPS required of the shared storage. Atlantis's ILIO software is an example of such a product.

With vSphere 5.0, VMware introduced a host capability called content based read cache (CBRC) but opted to keep it disabled by default. When VMware View 5.1 was released about six months later, View had a feature called View Storage Accelerator that enabled and utilized the CBRC. CBRC is VMware's answer to RAM-based storage cache. It keeps a deduplicated read cache in the host server's RAM, helping to deliver a faster storage response and absorbing the peaks associated with VDI workloads.

Server memory cache will only ever provide a read-cache option due to the volatile nature of RAM. Read caches for VDI are useful in reducing peaks, particular in boot-storm type scenarios, but VDI is an inherently write-intensive workload. The argument goes that if you're offloading huge chunks of reads, the back-end storage arrays can concentrate on write workloads. VDI is becoming the poster child of read-cache options, because of the relatively small capacity requirements but high IOPS required. Anyone who has tried to recompose hundreds of desktop VMs on an underscaled array knows how painful a lack of horsepower can be.

Although RAM continues to drop in price and grow in capacity, it will always be an expensive GB/$ proposition compared to other local flash-based storage. It will be interesting to see how valuable RAM cache becomes for more generalized server workloads as it becomes feasible to allocate more sizable amounts of RAM as a read-cache option. The ability of centralized storage arrays to deal more efficiently with heavy write loads will become increasingly crucial.

Measuring Your Existing IOPS Usage

When you know what affects the performance of your storage and how you can improve the design to suit your environment, you should be able to measure your current servers and estimate your requirements.

Various tools exist to measure performance:

- *Iometer* (`www.iometer.org/`) is an open source tool that can generate different workloads on your storage device. It lets you test your existing storage or generate suitable workloads to test new potential solutions.
- To monitor existing VMs, start with the statistics available in the vSphere client. You can also use *esxtop* to look at the following statistics:
  - *DAVG*—Disk latency at the array
  - *KAVG* and *QUED*—Queue-depth statistics showing latency at the VMkernel
- For very in-depth VM monitoring, the *vscsiStats* tool provides a comprehensive toolset for storage analysis.
- Windows VMs and physical servers can monitor IOPS with the *perfmon* tool. Just add the counters Disk Reads/sec and Disk Writes/sec from the Physical Disk performance objects

to view the IOPS in real time. These can then be captured to a CSV file so you can analyze typical loads over time.

- Linux/Unix VMs and physical servers can use a combination of *top* and *iostat* to perform similar recordings of storage usage.

When you're testing VMs, it's worth noting that the hypervisor can create some inaccuracies in guest-based performance tools such as perfmon due to timing issues, especially when the CPU is under strain. Remember to take into account the requirements of nonvirtual servers that may use the same storage, because they may affect the performance of the VMs.

vSphere has added new host and VM performance metrics in both the vSphere Client and in esxtop/resxtop. These additional statistics cover both real-time and trending in vCenter and bring the NFS data on par with the existing block-based support. To make the most of the tools, use the latest host software available.

Local Storage vs. Shared Storage

Shared storage, aka SANs or NAS devices, have become so commonplace in vSphere deployments that local storage is often disregarded as an option. It's certainly true that each new release of VMware's datacenter hypervisor layers on more great functionality that takes advantage of shared storage. But local storage has its place and can offer tangible advantages. Each design is different and needs to be approached with an open mind. Don't dismiss local storage before you identify the real needs of your company.

Local Storage

Local storage, or DAS, can come in several forms. Predominantly, we mean the disks from which you intend to run the VMs, mounted as VMFS datastores. These disks can be physically inside or attached to the host's disk bays. The disks can also be in a separate enclosure connected via a SCSI cable to an external-facing SCSI card's connector. Even if externally mounted, it's logically still local host storage. With local storage, you can mount a reasonable amount of capacity via local SCSI.

You can install vSphere 5 locally on SCSI, SAS, and SATA disks or USB flash drives (including SD cards), although your mileage may vary if the disk controller isn't listed on VMware's approved HCL. Theoretically you can use any of them for local storage for VMs, but clearly USB/SD flash storage was only meant to load the ESXi OS and not to run VMs.

First, let's identify more clearly when you *don't* want to deploy VMs on local storage. Certain features need storage that multiple hosts can access; if these will be part of your solution, you'll need at least some shared storage. Make no mistake, there are definite advantages to using shared storage (hence its overwhelming popularity):

- Local storage can't take advantage of DRS. Although enhancements in vSphere 5.1 mean that shared storage is no longer a requirement for vMotion, DRS still won't move VMs on local storage.
- High availability (HA) hosts need to be able to see the same VMs to recover them when a protected host fails.
- FT hosts need a second host to access the same VM, so it can step in if the first host fails.

- You can manage storage capacity and performance as a pool of resources across hosts, the same way host clusters can pool compute resources. SIOC, Storage DRS, and Policy-Driven Storage features discussed later in the chapter demonstrate many of the ways that storage can be pooled this way.
- RDM disks can't use local storage. In turn, this excludes the use of Microsoft clustering across multiple hosts.
- When you use shared storage, you can recover from host failures far more easily. If you're using local storage and a server fails for whatever reason, the VMs will be offline until the server can be repaired. This often means time-consuming backup restores. With shared storage, even without the use of HA, you can manually restart the VMs on another cluster host.
- Local storage capacity is limited by several factors, including the size of the SCSI enclosures, the number of SCSI connectors, and the number of datastores. Generally, only so many VMs can be run on a single host. As the number of hosts grows, the administrative overhead soon outweighs the cost savings.
- Performance may also be limited by the number of spindles available in constrained local storage. If performance is an issue, then SSD can be adopted locally, but it's expensive to use in more than a handful of servers and capacity becomes a problem again.
- With shared storage, it's possible to have a common store for templates and ISOs. With local storage, each host needs to have a copy of each template.
- It's possible with shared storage to run ESXi from diskless servers and have them boot from SAN. This effectively makes the hosts stateless, further reducing deployment and administrative overheads.

With all that said, local storage has some advantages of its own. If you have situations where these features or benefits aren't a necessity, then you may find that these positives create an interesting new solution:

- Far and away the greatest advantage of local storage is the potential cost savings. Not only are local disks often cheaper, but an entire shared-storage infrastructure is expensive to purchase and maintain. When a business is considering shared storage for a vSphere setup, this may be the first time it has ventured into the area. The company will have the initial outlay for all the pieces that make up a SAN or NAS solution, will probably have training expenses for staff, and may even need additional staff to maintain the equipment. This incipient cost can be prohibitive for many smaller companies.
- Local storage is usually already in place for the ESXi installable OS. This means there is often local space available for VMFS datastores, or it's relatively trivial to add extra disks to the purchase order for new servers.
- The technical bar for implementing local storage is very low in comparison to the challenges of configuring a new SAN fabric or a NAS device/server.
- Local storage can provide good levels of performance for certain applications. Although the controller cache is likely to be very limited in comparison to modern SANs, latency will be extremely low for obvious reasons.

- You can use local storage to provide VMFS space for templates and ISO files. Using local storage does have an impact. Localized templates and ISOs are available only to that host, which means all VMs need to be built on the designated host and then cold-migrated to the intended host.
- You can also use local storage for VM's swap files. This can save space on the more expensive shared storage for VMs. However, this approach can have an effect on DRS and HA if there isn't enough space on the destination hosts to receive migrated VMs.
- Many of the advanced features that make shared storage such an advantage, such as DRS and HA, aren't available on the more basic licensing terms. In a smaller environment, these licenses and hence the features may not be available anyway.
- Local storage can be ideal in test lab situations, or for running development and staging VMs, where redundancy isn't a key requirement. Many companies aren't willing to pay the additional costs for these VMs if they're considered nonessential or don't have SLAs around their availability.

vSphere 5.1's vMotion enhancements, which allow VMs on local disks to be hot migrated, reaffirm that local VMFS storage can be a valid choice in certain circumstances. Now hosts with only local storage can be patched and have scheduled hardware outages with no downtime using vMotion techniques. Local storage still has significant limitations such as no HA or DRS support, but if the budget is small or the requirements are low then this may still be a potential design option.

What about Local Shared Storage?

Another storage possibility is becoming increasing popular in some environments. There are several different incarnations, but they're often referred to as *virtual SANs*, *virtual NAS*, or *virtual storage devices*. They use storage (normally local) and present it as a logical FC, iSCSI, or NFS storage device. Current marketplace solutions include VMware's own VSA (see sidebar), HP's LeftHand, StarWind Software, and NexentaVSA.

VMware VSA

VMware Virtual Storage Appliance (VSA) is VMware's answer to the concept of local shared storage. It creates virtual appliances on the local storage of two or three ESXi hosts, and presents that local storage as NFS datastores for the hosts to store their VMs.

VMware VSA can be configured in two different ways:

Two-Node Cluster Each host has a VSA appliance running on local storage, and it utilizes a software service, the *VSA Cluster Service*, which can run on the vCenter Server to act as a third cluster node. With VSA version 5.1, the VSA Cluster Service can be installed on a different server if the vCenter Server is not in the same physical site as the cluster. This tie-breaker software can be installed on any server; although it's advisable to house it on a non-VSA host. The software is a java package that can be installed as a service on any Windows or Linux server available — it just must be in the same subnet as the two participating hosts. The two-node configuration exports two datastores. The vCenter Server or cluster service server with this third cluster node service doesn't present any storage.

continued

Three-Node Cluster Each of the three hosts runs a VSA appliance on its local storage. Three datastores are exported in this configuration. The vCenter Server doesn't participate as a cluster node.

Each VSA appliance replicates its storage to one other VSA node, so all datastores have one mirrored copy on an alternate host in case a physical server becomes unavailable. Each appliance uses all the available local space; but because the storage is mirrored, only half the local space is available as usable capacity for VMs. Both two- and three-node clusters can only tolerate the loss of a single host.

VSA Appliances

Each VSA appliance VM (one per host) uses one vCPU with a 2 GHz reservation and 1 GB of RAM. The appliance's storage is provisioned across 18 VMDK disks: 2 disks are 4 GB (a mirror of the appliance's guest OS); the remaining 16 VMDKs are used for shared storage and consume the rest of the host's local VMFS storage. The 16 VMDKs are split evenly across two PVSCSI adapters.

The VSA appliance runs SLES 11 Linux and presents its ext4-formatted disks as NFS exports for the hosts to use as shared storage. The appliances do run iSCSI target and initiator software, but this is only used internally to mirror the data between nodes.

VSA Requirements

To run hosts with VSA storage, the feature must be licensed separately. Each host must be running a minimum of ESXi 5.0 installable (embedded or stateless installs aren't supported). The hosts require the following:

- Minimum 6 GB of RAM (24 GB is recommended)
- At least four network adapters (no jumbo frames)

Each VSA needs a vCenter instance, but since VSA version 5.1 that vCenter doesn't have to be in the same location. This version allows each vCenter instance to manage up to 150 separate VSA clusters. VSA 5.1 allows the vCenter server to run on top of a VSA cluster. This is achieved by first installing the vCenter as a VM on local host storage, then migrating the VM onto a VSA provisioned datastore once the service is configured.

Each participating host must have at least four network adapters, because the install creates two vSwitches, each with two uplinks. It dedicates vSwitch0 with its uplinks to front-end connections and vSwitch1 with its two uplinks to back-end connections. The install creates three port groups on the Front End vSwitch: *VSA-Front End*, *VM Network*, and *Management Network*. Two port groups are created on the back-end vSwitch: *VSA-Back End* and *VSA-VMotion*. Despite the vMotion port group residing on the second vSwitch, it must share the same subnet as VSA-Front End and Management Network.

VSA Performance

VSA clusters can provide respectable performance for a small office. The following determining hardware factors dictate how good the performance of the share storage is:

- Speed of the disks
- Number of disks
- RAID type

- Local RAID controller (the size of the onboard write cache is particularly relevant)
- Speed/quality of the replication network (VSA's internode host-mirrored disk replication is synchronous, which means VM disk writes aren't acknowledged until they're written to both copies)

VSA Design Considerations

VSA at its 1.0 release was regarded a somewhat immature solution with a number of key constraints. When it was first introduced, there was an installation requirement that each server's local hard disks had to be configured in RAID 10. This, in combination with the mirroring that VSA does between instances, meant that effectively 75 percent of each disk's capacity was being lost to providing redundancy. Fortunately, within six months of VSA's release, VMware loosened this restriction and now allows RAID 5 and RAID 6 disk sets to participate. VSA 1.0 had no supported way to change the disk capacity. You needed to ensure that you understood the storage requirements before you started. VSA version numbering jumped from 1.0 to 5.1. This helped to reflect VMware's growing product sophistication. Although you still can't add a third node to a two-node cluster, or grow beyond three nodes, at least with 5.1 you can add more disks to existing clusters.

With VSA 5.1 at general release the maximum supported storage configuration options are:

- Eight 3 TB disks in RAID 6 with no hot spare. This provides 18 TB of usable space on a two host cluster or 27 TB across three hosts (each datastore must have room to be mirrored once).
- Twenty-eight 2 TB disks in RAID 6 (sixteen of them in an external enclosure in their own RAID 6 group) with no hot spare. This provides 24 TB of usable space on a two host cluster or 36 TB across three hosts.

These are the *supported* limits. We are not advocating breaking the support limit, but it is worth checking these levels for an update as they are likely to increase between releases as VMware qualifies more configurations.

The VSA appliance on each host uses 1 GB of RAM and an additional 100 MB overhead for each hosted VM. Add memory for the ESXi hypervisor, and 5 GB should be a safe amount to account for all the non-guest VM memory.

When VSA is installed, any other non-VSA hosts in the same vCenter datacenter automatically have the NFS datastores mounted automatically for use as datastores. If you add hosts to the vCenter after the install, the shared VSA datastores can be mounted to those hosts as well. This allows VSA to scale up slightly beyond the three-node storage limit. No other host can join the storage cluster as a participating node, but it can use the shared storage.

In its first incarnation release alongside vSphere 5.0, VMware VSA was probably too limited to be considered by most office locations that had more than a couple of dozen VMs. Its feature set lent itself to an SMB market, where the cost and complexity of shared storage might make an organization contemplate VSA despite its initial shortcomings. Most enterprises regarded VSA as a niche product that has the potential to mature into something more interesting. Since VSA's 5.1 release, its potential market has grown to include Remote Office/Branch Office (ROBO) style deployments. The performance of the three hosts is still too limiting to be considered for more than a handful of VMs, but now that vCenter can remotely manage multiple clusters, enterprises can consider this software array option for certain use-cases.

Virtual arrays allow you to take advantage of many of the benefits of shared-storage devices with increased VMware functionality but without the cost overheads of a full shared-storage environment. Multiple hosts can mount the same LUNs or NFS exports, so the VMs appear on shared storage and can be vMotioned and shared among the hosts. Templates can be seen by all the hosts, even if they're stored locally.

But remember that these solutions normally still suffer from the same single-point-of-failure downsides of local storage. There are products with increasing levels of sophistication that allow you to pool several local-storage sources together and even cluster local LUNs into replica failover copies across multiple locations.

Several storage vendors also produce cut-down versions of their SAN array software installed within virtual appliances, which allow you to use any storage to mimic their paid-for storage devices. These often have restrictions and are principally created so that customers can test and become familiar with a vendor's products. However, they can be very useful for small lab environments, allowing you to save on shared storage but still letting you manage it the same way as your primary storage.

Additionally, it's feasible to use any server storage as shared resources. Most popular OSes can create NFS exports, which can be used for vSphere VMs. In fact, several OSes are designed specifically for this purpose, such as the popular Openfiler project (`www.openfiler.com`) and the FreeNAS project (`http://freenas.org`). These sorts of home-grown shared-storage solutions certainly can't be classed as enterprise-grade solutions, but they may give you an extra option for adding shared features when you have no budget. If your plan includes regular local storage, then some virtualized shared storage can enhance your capabilities, often for little or no extra cost.

Shared Storage

Shared storage provides the cornerstone of most vSphere deployments. Local storage is often still found in small setups, where companies are new to the technologies or lack the budget. To take full advantage of vSphere and all it has to offer, a shared-storage solution is the obvious first choice. Shared storage underlines the primary goals:

Availability Shared storage creates greater redundancy and reliability, and reduces single points of failure.

Performance Shared storage means better I/O performance and scalability. Greater disk spindles, powerful controllers with large read- and write-cache options, and tiers of different storage disk all translate to better performance.

Capacity Shared storage aggregates storage, allows the use of advanced capacity-reduction technologies, and can address large amounts of storage space.

Choosing a Protocol

An increasingly discussed and debated storage topic is which protocol to use. VMware supports several protocols, and with that choice come decisions. With the advent of 10GbE, network-based iSCSI and NFS have become far more competitive against FC-based SANs. Many of the midrange arrays available today come with multiple protocol support included or easily added, so things are much less clear cut than before.

As you'll see, each protocol has its own ups and downs, but each is capable and should be considered carefully. Old assumptions about the protocols can and should be questioned, and preconceptions are often being proven no longer true. It really is time to go back to the requirements and ask why.

As each new release of vSphere becomes available, the support matrix for protocols changes; and the maximum configuration limits regularly increase. In past VMware days, many advanced features or products only worked with certain types of storage. For the most part, this is no longer true: most products work with all supported protocols.

You need to compare the following protocols: FC, iSCSI (using both hardware and software initiators), and NFS exports. A newer addition to the list is Fibre Channel over Ethernet (FCoE); and you should also consider the increasing availability of 10GbE, which is making a big impact on the storage landscape with regard to protocol selection. A few other options are available on the vSphere protocol list, but they're excluded from the rest of this discussion because they aren't considered sufficiently mainstream to be relevant to most readers. These additional options are either vendor specific or used in very specific use cases:

Shared SAS Shared SAS comes in two forms: SAS direct attached and SAS switched. SAS direct attached disks act as local storage for the host to which it's attached. But SAS switched is counted as shared storage. SAS switched arrays usually have only two controllers and can only be connected (shared) between two hosts. SAS switched is supported by VMware as long as the array is on VMware's HCL.

AoE ATA over Ethernet (AoE) eschews IP and uses layer 2 Ethernet to provide a simple storage protocol. It's analogous to FCoE but sends regular ATA disk commands encapsulated over Ethernet. Its commercial implementation is driven by a single vendor, Coraid, but its customers represent only a tiny fraction of the vSphere market. StarWind, popular for its in-guest Windows iSCSI target software, sells an AoE Windows initiator.

InfiniBand InfiniBand is still theoretically a supported protocol for vSphere, although the lack of any available HCL certified drivers for InfiniBand adapter cards makes this a moot point.

For the rest of this chapter, we'll concentrate on the four most common protocols. These are the protocols for which you can expect hardware vendors to provide solutions and that you'll encounter in vSphere environments. Table 6.3 and Table 6.4 summarize the characteristics of each protocol.

TABLE 6.3: Protocol hardware characteristics

	Local	FC	FCoE	iSCSI	NFS
Transfer	Block	Block	Block	Block	File
Transport	Direct SCSI	SCSI encapsulated in FC frames	SCSI encapsulated in FC frames over Ethernet	SCSI encapsulated in TCP/IP	File over TCP/IP

TABLE 6.3: Protocol hardware characteristics *(continued)*

	LOCAL	FC	FCOE	ISCSI	NFS
HOST INTERFACE	SCSI/SAS/ SATA/IDE controller	HBA	1) Hardware initiator: CNA 2) Software initiator: NIC with partial FCoE offload	1) Hardware initiator: iSCSI HBA 2) Software initiator: NIC with VMkernel port or dependent hardware (TOE)	NIC
LINK SPEEDS	Depends on bus speed of controller	Up to 16 Gbps	10GbE	Up to 10GbE	Up to 10GbE
PRIMARY SECURITY CONTROLS	n/a	Zoning LUN masking Some FC switches support VSANs	Zoning LUN masking Some FCoE switches support VSANs	LUN masking CHAP IP security (such as ACLs) VLAN isolation	Export permissions IP security (such as ACLs) VLAN isolation

TABLE 6.4: vSphere feature comparison for each protocol

	LOCAL	FC & FCOE	ISCSI	NFS
RDMS	No	Yes	Yes	No
BOOT ESXI FROM	Yes	Yes*[1]	Yes*[2]	No
IN-GUEST INITIATOR	n/a	FC LUNs can be mapped via NPIV*[3]	Guest software initiator	Guest NFS client
HA	No	Yes	Yes	Yes
VMOTION	Yes*4	Yes	Yes	Yes
DRS	No	Yes	Yes	Yes
STORAGE VMOTION	Yes	Yes	Yes	Yes
STORAGE DRS	No	Yes	Yes	Yes
FT	No	Yes	Yes	Yes

	Local	FC & FCoE	iSCSI	NFS
MSCS clustering	Cluster in a box (CIB) only	Yes	Only with in-guest initiator	Not supported
Thin-provisioned VMDKs	Yes	Yes	Yes	Yes
Datastore file system	VMFS	VMFS	VMFS	NFS
Maximum datastores per host	Practically limited by size of local storage	256	256	256 (default is 8)
Maximum datastore size	≈ 64 TB (but practically limited by size of local storage)	≈ 64 TB	≈ 64 TB	Limited by NAS file system
VMs per volume	2,048	2,048	2,048	Limited by NAS file system

*[*1] vSphere 5.0 can only boot ESXi over FCoE if using a CNA card. vSphere 5.1 can boot ESXi via a software initiator if the NIC card supports FCoE booting. Booting from FC HBAs is fully supported.*

*[*2] For SW initiators, the NIC adaptor must support a feature called iSCSI Boot Firmware Table (iBFT).*

*[*3] NPIV is discussed in Chapter 7.*

*[*4] vMotion is possible with local storage in vSphere 5.1.*

Fibre Channel

FC is the veritable stalwart shared-storage protocol and has been ever since it was first supported by ESX in version 2.0. It's a mature and well-trusted solution in datacenters, and traditionally it's the default solution of many Enterprise SANs. The FC protocol encapsulates all the SCSI commands into FC frames, a lossless transport.

FC fabrics are specialized storage networks made up of server HBAs, FC switches, and SAN SPs. Each connector has a globally unique identifier known as a World Wide Name (WWN). A WWN is further split into a World Wide Port Name (WWPN), which is an individual port, and a World Wide Node Name (WWNN), which is an endpoint device. Ergo, a dual-port HBA will have two WWPNs but only one WWNN.

Hosts can be attached directly to the SAN without the use of a fabric switch, but this restricts the number of hosts to the number of FC SP ports available. FC switches also allow for redundant links from each host to cross-connect to multiple SP controllers.

The FC protocol is a high-bandwidth transport layer with a very low latency. This low latency still sets it apart from other common storage protocols. The FC protocol technically has three different modes, but *switched* (FC-SW) is the only one you're likely to use in a vSphere environment (point-to-point and arbitrated loop are the two legacy modes). The interconnect speeds are set at 1, 2, 4, 8, or the latest, 16 Gbps. vSphere 5.0 requires that 16 Gbps HBAs be throttled back to 8 Gbps, but vSphere 5.1 supports 16Gbps to the FC switch. To get full 16Gbps to the array,

multiple 8Gbps connections from the FC switch to the array need added to the zone. FC fabrics ordinarily use OM2 cables with LC connectors (orange fiber optic cables) these days, although light-blue OM3 cables are becoming more popular with an increase in 8 and 16 Gbps use.

FC storage security is predominantly handled via zoning. *Zoning* is an access-control mechanism set at the FC switch level, restricting which endpoints can communicate. Anything outside the zone isn't visible to the endpoint. Zoning protects devices from other traffic such as registered state-change notification (RSCN) broadcasts and is roughly analogous to VLANing in the Ethernet world. Zoning ensures that hosts that need to see the storage can do so, while those that don't need visibility don't interfere. You can set zones based on specific switch ports (*port zoning* or *hard zoning*) or define them via WWNs (*soft zoning*), which has the advantage of allowing recabling without needing to reconfigure the zoning information. Due to security concerns, some FC switch manufacturers only support hard zoning on their newer switches.

Several zoning topologies are available. The simplest method is to have one large zone with all devices in it. But for vSphere (and most other applications), the recommendation is to use what is called *single initiator zoning*. This means each HBA is in its own zone with the target device. This approach is considerably more secure and prevents initiators from trying to communicate with each other (which they shouldn't be doing in a vSphere setting). An even tighter convention, known as *single initiator/single target zoning*, is to create zones so each single HBA is mapped to a single SP. This takes longer to configure than the other two zoning topology designs; but if you use a sensible naming convention for the zones (for example, HOSTNAME_HBA1_SPA), they can be logical to follow and you can add to them when required.

You can use LUN masking to grant permissions, allowing LUNs to be available to hosts. The LUN masks are set on the hosts themselves or on the SPs. LUN masking is also sometimes referred to as *iGroups*, *access control*, *storage presentation*, or *partitioning*. It effectively gives hosts the ability to disregard LUNs or lets SPs ignore hosts that shouldn't be accessing LUNs.

Zoning or LUN Masking?

When administering a FC storage solution, you should implement both zoning and LUN masking. They're both crucial to maintaining a secure, reliable, scalable, and efficient storage platform.

FC has many advantages when compared to other options:

High speed: Until 10GbE arrived, FC was always the high-speed option.

Lossless with dedicated paths: There is a low risk of oversubscription on the paths.

Low latency: If you have VMs that are sensitive to latency, FC will help prevent issues.

Existing FC equipment: There may already be some FC equipment in the datacenter.

Existing FC skills: Some staff may be familiar with FC.

Security: With dedicated links of fiber optic cables, it's an inherently more secure solution.

Trust: It's a long-trusted, mature storage protocol.

Dedicated network: Normally, the FC fabric is dedicated to storage traffic.

Efficiency: FC frames don't have the TCP/IP overhead that iSCSI and NFS do.

But there are certain potential drawbacks to the FC protocol:

Cost: FC switches, cables, and so on are normally more expensive than equivalent Ethernet equipment.

Initial cost: When you first use FC, a large CAPEX layout is required to get a working fabric.

Unfamiliarity of technology: If your team is new to FC, there is a relatively steep learning curve to implement it.

Fibre Channel over Ethernet

FCoE is a relatively new addition to the protocol list available to vSphere architects. FCoE maps the frame-based FC protocol over Ethernet alongside its IP traffic. Because Ethernet has no built-in flow control, FC needs special enhancements to prevent congestion and packet loss. These enhancements help to deal with the loss and retransmissions in IP-based transport, which is what makes FCoE special. FCoE is designed to run over 10GbE cables.

FCoE can utilize converged network adapters (CNAs), which combine FC HBAs and Ethernet NIC adapters. ESXi often need extra drivers installed for these CNA cards to be recognized. The drivers usually come in two parts: one for the FCoE piece and another for the Ethernet adapter. After the card is installed, it logically appears in the vSphere Client as both an HBA under the storage adapter configuration and as a NIC under the network adapter configuration.

Since vSphere 5.0, the hypervisor offers a software initiator that works with 10GbE NIC cards that include a partial FCoE offload capability. This allows you to access LUNs over FCoE without needing a CNA card or installing third-party CNA drivers. To create the FCoE software adapter on the ESXi host, enable the feature on the NIC adapter. The NIC must be an uplink on a vSwitch that already contains a VMkernel connection. It uses the VMkernel to negotiate the FCoE connection with the physical switch. This connection isn't used for FCoE storage traffic. vSphere 5.0 can boot from a SAN LUN on FCoE if you have a FCoE hardware initiator HBA card, but 5.1 added the ability to boot from FCoE software initiator if the NIC card supports FCoE booting.

FCoE has a great deal of overlap with FC, so if you have an existing FC infrastructure, you should be able to introduce FCoE while avoiding a rip-and-replace style of migration. FCoE is partially important in converged 10GbE infrastructures, such as Cisco's UCS blades, where there is no FC connection to the servers. All traffic, including storage, leave the servers and traverse an Ethernet network initially. In such converged solutions, the physical transport may be fiber or copper based to the northbound switch but is always Ethernet not FC based. To provide connectivity to FC storage arrays, the FC fabric switches cross-connect to the network switches. The FCoE-capable network switches can then relay the FCoE traffic into the FC fabric. If newer SANs have 10GbE connections and can natively use the FCoE protocol, then they can connect directly to the FCoE-capable network switches, which act solely as the fabric with no need for FC switches in the design.

FCoE uses the same zoning techniques as the FC world to regulate access between FCIDs (equivalent of FC WWPNs). FCoE requires Jumbo Frames because the payloads are larger than 1,500 bytes and can't be fragmented.

FCoE shares many of the advantages attributed to FC, along with the following:

Fewer cables: By combining storage and network with high-bandwidth cables, FCoE reduces clutter, increases airflow, and eases management.

Less power: Fewer cables means less power is needed.

CNAs already include 10GbE: If you invest in CNAs but later decide to switch to iSCSI or NFS, then the hardware investment will still be valid.

FCoE switches can interface with FC equipment: You should be able to use existing FC equipment while taking advantage of the converged cabling on the server side.

Low overhead of FC: FCoE has a much lower latency than iSCSI or NFS.

But be mindful of these potential FCoE disadvantages:

Newness of the protocol: FCoE is barely ratified as a standard, and some questions remain about whether it lacks the maturity of the other protocol standards available.

Expense: FCoE is still relatively expensive. Core Ethernet switches that support FCoE are at a premium.

Different hardware standards: The protocol is so young that de facto cables and connectors have yet to emerge. Some first-generation CNA cards can't upgrade to the latest standards.

Little advantage if there is no FC already: Datacenters without FC at the moment are likely to move toward 10GbE iSCSI or NFS, not FCoE.

Lack of experience/knowledge: FCoE is a new, emerging standard, so there is less information relating to it.

iSCSI

iSCSI uses TCP to encapsulate SCSI traffic, allowing block-level storage LUN access across Ethernet cables. Commonly used over 1GbE links, iSCSI has been able to take advantage of 10GbE advances, letting it compete with the traditionally more performant FC protocol.

iSCSI became popular in datacenters predominantly through use by Microsoft servers (as opposed to FC, which was traditionally the focus of Unix servers).

vSphere supports two types of iSCSI initiator:

Hardware Initiator An iSCSI HBA that offloads processing from the host's CPU. The hardware initiator works on independent hardware cards. An independent hardware card offloads both the iSCSI processing and the TCP/IP processing from the host's CPU. This is the classic iSCSI HBA-style card.

Software Initiator Uses VMware's software implementation within the VMkernel, alongside a regular Ethernet NIC adapter. The software initiator can work with two types of NIC adapter cards: dependent hardware cards and regular NIC cards. The dependent hardware cards, or iSCSI TOE cards, offload the TCP/IP processing but still rely on the VMware software initiator for iSCSI processing.

Alternatively, the software initiator can work with pure NIC cards with no inherent offload. With no iSCSI offload capabilities, the NICs need to be uplinks to a vSwitch that has a VMkernel connection.

The hardware initiators have the advantage that they offload some of the CPU processing; but with recent advances in the vSphere software initiator, this has become less of an issue. The current software initiator uses very little CPU (around half a core); and with the increasing

processing power of servers, it's generally thought that the additional cost of the hardware cards is no longer worth the expense. Software initiators have become a much more popular method of connecting to iSCSI targets. Few choose to buy hardware cards for new deployments. Hardware initiators are relatively rare and are used less and less these days.

Although it's possible to run an in-guest iSCSI software initiator to access raw block storage for a VM, it bypasses the ESXi host's storage stack and so is treated like any other VM network traffic. It's unusual for VM traffic to be a bottleneck, but this is the sort of configuration that can saturate VMNICs. This isn't a recommended way to present storage to VMs: it doesn't have the flexibility of regular iSCSI storage, because it can't use Storage vMotion or vSphere snapshots.

NOTE One side case for in-guest iSCSI software initiators is that they can allow you to present very large disks to VMs. VMDK files still have a 2 TB limit, whether they're deployed on VMFS or on NFS. However, with an in-guest iSCSI software initiator, you can theoretically present as large a disk as your array will allow. Needless to say, this isn't a recommended setup. vSphere 5 now allows physical RDMs over the 2 TB limit, so if there is a need to present very large disks to a VM, this is the recommended approach.

vSphere has two methods to discover iSCSI targets:

Dynamic Discovery Also known as SendTargets. The initiator polls the network for targets. Less configuration is required, although removed items can return after a rescan or reboot, or be lost if the target is temporarily unavailable.

Static Discovery You must manually enter the IP addresses of the targets. The target survives rescans, but this method is available only when using hardware initiators.

iSCSI has no FC fabric zoning, although because it's still block-level storage it can use LUN masking to ignore LUNs. Instead of zoning, iSCSI uses Challenge-Handshake Authentication Protocol (CHAP) as a way to provide rudimentary access control for the initiators and targets. CHAP is a three-way handshake algorithm based on a predefined private value, which verifies identity using a hashed transmission. Hardware initiators only allow for the use of one-way CHAP, as opposed to software initiators, which can do mutual CHAP (bidirectional).

Most arrays also let you configure access control based on IP address or initiator name. Make sure your iSCSI traffic is only allowed onto an internal part of your trusted network, because the traffic isn't encrypted in any way. A nonroutable VLAN on a dedicated pair of redundant switches is ideal to segregate and secure iSCSI traffic.

Jumbo frames can be enabled on vSphere hosts and are supported by most iSCSI SANs. They help to increase performance, because the larger packet sizes reduce the overhead of processing the Ethernet packets. Typically, the frames are set to 9,000 maximum transmission units (MTUs). It's important that if you enable jumbo frames, all devices, endpoints (servers and storage), and network devices in between must support and be enabled for this. Enabling jumbo frames on some Cisco switches requires them to be reloaded (which causes a short network outage).

The Ethernet switch ports used for the storage network should have Rapid Spanning Tree Protocol (RSTP) or portfast enabled. This allows an immediate transition if an active link fails.

Chapter 5, "Designing Your Network," discussed various methods to provide suitable network redundancy for Ethernet-based storage. Later in this chapter, the "Multipathing" section will discuss different multipathing techniques, including those covering the iSCSI protocol. But it's worth pointing out at this juncture that your iSCSI design should carefully consider redundancy. The fundamentals involve ensuring that at least two NICs (or HBAs) are configured on

each host for iSCSI traffic. These two NICs should be connected to two separate switches, which in turn are connected to two iSCSI controllers on the SAN.

Dedicated storage switches, which don't handle regular network traffic, make your storage transport more secure. They also help to prevent contention with other IP traffic, improving storage performance. If you don't have access to separate hardware, then you can use layer 2 VLANs to isolate the storage. You should avoid 100 Mbps equipment anywhere in the chain, because it doesn't provide the throughput required to run VMs effectively. Use 1GbE capable switches, NICs, and cables throughout as a minimum.

Ethernet isn't designed for storage, so it can suffer from congestion issues when numerous hosts are attached to a much smaller number of array controllers. This causes oversubscription, which means that packets get dropped and performance degrades. This can be the start of a vicious circle where TCP/IP needs time to see what was dropped and then more time to retransmit. A bad situation gets progressively worse. Using logical separation techniques such as VLANing doesn't help in these cases. If this becomes an issue, you should use dedicated storage switches and, if required, more capable switches with better backplane I/O capacity, which will alleviate the oversubscription.

iSCSI has a number of advantages over the FC and FCoE protocols:

Inexpensive equipment: Compared to FC, the switches and cables are less expensive.

Simplicity: Both the equipment and the protocol itself are well understood. Generally, companies don't need extra training to introduce this equipment. People are accustomed to cheap gray Ethernet cables.

NICs are cheaper than FC HBAs: It's common to use regular Ethernet NICs with software initiators for iSCSI, which are much cheaper than FC HBAs.

Reusable equipment: It may be possible to reuse some existing network equipment and cables.

Windows administrator approval: iSCSI has long been used by Windows administrators, so it's well trusted and understood in most datacenters.

Longer distances: It's possible to connect servers to storage at much greater lengths than with FC.

However, you must also remember a number of disadvantages when considering iSCSI:

1GbE inability to compete with FC/FCoE: Unless you're using 10GbE, then iSCSI will lag behind higher-bandwidth FC/FCoE solutions.

Latency: Even with 10GbE, iSCSI can't provide the low-latency efficiency available with FC.

10GbE expense: Although 1GbE may have felt like free infrastructure if you were reusing old or very cheap equipment, using 10GbE requires expensive switches, NICs, and maybe even upgraded cabling.

Oversubscription: Flooding network links is possible. This is a scaling issue.

TCP/IP overhead: TCP/IP isn't particularly suited to storage. The overhead of TCP/IP to provide for retries, acknowledgments, and flow control reduces efficiency.

Path failovers that can cause long I/O delays compared to FC: To mitigate this risk, you may need to increase the SCSI timeout in every guest OS.

Lack of support for Microsoft clustering: Again, due to potential long I/O delays during failover, Microsoft clustering isn't supported using iSCSI VMFS datastores.

NFS

NFS is a very mature file-sharing protocol that allows several clients to connect at the same time. NFS file shares are known as *exports*. vSphere requires that NFS exports use version 3 of the protocol, even though version 4 has been available and ratified for many years.

NFS is fundamentally different from FC, FCoE, and iSCSI in that it isn't block-level storage, but file-level. It's common to refer to the block-level arrays as *SAN devices*, but refer to NFS as *NAS devices*, even though many SANs can now provide NFS exports. Block devices provision their disks as LUNs, which can be used as VMFS volumes or RDMs in vSphere. But NFS exports are used as a remote file system, and VMs are placed directly on them.

VMFS

VMware's Virtual Machine File System (VMFS) is the default file system to store VMs in vSphere. It's a highly optimized, clustered file system that can efficiently store very large disk files and present them across multiple hosts.

Traditionally, clustered file systems have been very complicated to set up and configure, but VMFS is simple to use. VMFS can enable advanced vSphere features, such as DRS and HA, which rely on multiple hosts accessing the same VMs.

VMFS allows up to 64 hosts to connect to the same volume and is responsible for all the required file-locking operations. A VMFS volume on a single LUN can be dynamically grown up to a 64 TB limit (if the storage array supports dynamical growing LUNs) or concatenated with additional extents (LUNs) up to the same 64 TB limit.

VMFS can recognize SAN snapshot copies and mount them. A signature is written to each VMFS volume, and this can be resignatured to allow the snapshot copies to be used alongside the originals.

VMFS volumes use block LUNs from local, FC, or iSCSI arrays, as opposed to NFS file exports. RDMs are a special type of disk format that uses a mapping file on a VMFS volume to point to a separate raw LUN (RDMs are discussed in more depth in Chapter 7).

Traditionally, block storage (particularly FC) had better support for all the latest features. But these days, almost all premier features are available for NFS. In fact, some newer VMware View options have been released for NFS before their block-based alternatives.

NFS has historically been criticized for its performance versus FC and iSCSI. This was due in large part to cheaper NAS devices not being able to stand up against enterprise-class SANs, rather than to a deficiency in the protocol itself. For the vast majority of workloads, NFS is more than capable; and coupled with 10GbE, performance can be comparable to FC 8 Gbps.

Bandwidth is closely related to the physical transport, and there isn't much difference between 8 Gbps FC and 10GbE NFS. IOPS tends to come down to cache and disk spindles/

speed, so even 16 Gbps FC connections might not necessarily provide much better performance than 10GbE NFS (or iSCSI or FCoE, for that matter). The primary differences between FC and NFS are latency, failover times, and multipathing mechanisms.

NFS is easy to plan and configure, and it's normally far less costly than FC to set up and maintain. For this reason, it's very popular for small to medium companies and is often the default choice for VDI deployments.

By default, the number of NFS exports that any host can mount is only 8, but an advanced setting allows you to increase this to 256. Even if you think you'll never grow beyond the eight-datastore limit, it's a good idea to increase this number before provisioning the first storage, because an increase in the future requires host reboots.

NFS exports can be mounted on hosts via IP addresses or hostname, but IP address is the recommended choice. If local procedures require you to use hostnames, check to see whether the name servers are virtual. If so, it's advisable to either make an exception and use IP addresses when mounting them, or create entries in the `/etc/hosts` file of each host. Otherwise, it's possible to get stuck in a chicken-and-egg situation where the hosts can't resolve the NFS exports, because all the name servers are turned off (because they live on the NFS exports). Name resolution is so important to other services that you should plan carefully if all DNS (or WINS) servers are virtual.

As with iSCSI, the network traffic isn't encrypted. And NFS doesn't use CHAP to authenticate initiators and targets, so it's even more important to only span a trusted network. Most NAS devices can isolate their traffic to specific IP hosts, but this is easy to spoof if the network isn't suitably isolated. Unfortunately, the vSphere hosts must mount the exports with root access, which is a security concern in itself. For this reason, dedicated isolated storage switches are highly recommended if security is an especially important design consideration.

You can adjust a number of advanced NFS settings to fine-tune the hosts to the particular NAS unit you're using. You should consult the storage vendor's documentation to ensure that you implement its best practices.

Much of the advice given in the previous section for iSCSI network configurations is just as applicable to NFS. If possible, do the following:

- Separate network devices to isolate the storage traffic.
- Use nonroutable VLANs.
- Use redundant network links.
- Use jumbo frames.
- Enable RSTP or portfast on the switch ports.
- Use switches with sufficient port buffers.

NFS can offer the following advantages (again, many are common with iSCSI because they share the same physical transport layer):

Inexpensive equipment: In comparison to FC, the switches and cables are less expensive.

Simplicity: Both the equipment and the protocol itself are well understood. Generally, companies don't need extra training to introduce this equipment.

Reusable equipment: It may be possible to reuse some existing network equipment and cables.

Longer distances: It's possible to connect servers to storage at much greater lengths than with FC.

And here are some NFS-specific advantages:

Trust: NFS is a well-trusted, mature protocol, particularly among Unix administrators.

Inexpensive NAS units: NAS-only devices are often much more affordable than a SAN, although these are more suited to smaller environments.

Ease of setup: NFS is extremely easy to set up on vSphere hosts.

Scalability: Datastores can be much larger and contain many more VMs than VMFS on block storage.

Thin provisioning: The disk format is set by the NFS server, and by default the VMDK files are thin-provisioned automatically.

Additional VM options: NFS arrays often have more integrated snapshot, backup, and replication options, because the array understands the file system and can touch the files directly.

NFS has the following disadvantages in common with iSCSI:

1GbE inability to compete with FC/FCoE: Unless you're using 10GbE, then NFS will lag behind higher-bandwidth FC/FCoE solutions.

Latency: Even with 10GbE, NFS can't provide the low-latency efficiency available with FC.

Expense of 10GbE: Although 1GbE may have felt like free infrastructure if you were reusing old or very cheap equipment, using 10GbE requires expensive switches and NICs.

TCP/IP overhead: The protocol isn't suited to storage. The overhead of TCP/IP to provide for retries, acknowledgments, and flow control reduces efficiency.

Path failovers can cause long I/O delays: To mitigate this risk, you need to increase the SCSI timeout in every guest OS.

Lack of support for Microsoft clustering: Again, due to potential long I/O delays during failover, Microsoft clustering isn't supported using NFS.

No RDM support: In turn, no file can exceed 2 TB.

And these are NFS-specific limitations:

- Can't aggregate bandwidth across multiple Ethernet cables (see the "Multipathing" section).
- Not ideally suited to very high I/O VMs that would normally get dedicated datastores/LUNs.
- Uses some additional CPU processing.
- No boot from SAN; you can't boot from an NFS export.
- NFS is particularly susceptible to oversubscription because of the very high VM density possible on each datastore.

Protocol Choice

After carefully looking at the protocols, their constraints, and their impacts, a number of key factors tend to decide which is best suited to a design.

Companies always favor sticking to an existing implementation, and for good reason. You're likely to already have several pieces, and you probably want to avoid a complete rip-and-replace strategy. The ability to carefully transition to a new protocol, especially regarding something as critical as primary storage, is an important consideration. If this is a trusted proven solution that you're merely hoping to upgrade, then existing skills and experience are tangible assets.

Performance is a factor that may influence your decision. In most general situations, FC or 10GbE with iSCSI or NFS is likely to be more than sufficient for 99 percent of your bandwidth needs. The VMs' IOPS come down to several things, but ultimately it's the SP cache, any SP "secret sauce" such as efficient write coalescing, and the number and speed of the underlying disks. The protocol has very little impact in a properly designed environment. However, one key area where performance may influence the protocol choice is latency. If the design requires the potential for very-low-latency VMs (perhaps a real-time database), then FC is your friend (unless you can deal with the limitations of DAS).

NFS grew to be a popular Ethernet alternative to iSCSI during the vSphere 3 and 4 releases because of the larger datastores possible (iSCSI datastores were limited to 2 TB) and SCSI locking issues that could restrict the number of VMs on its datastores. NFS proved to be far more flexible, allowing for large commodity datastores. With vSphere's VMFS-5 datastores up to a maximum of 64 TB, its new ATS locking allowing greater density on iSCSI LUNS, and simplified iSCSI port binding, we think iSCSI is likely to see a resurgence in popularity. iSCSI's significantly better multipathing support provides a serious advantage over NFSv3 in larger environments. Additionally, monitoring and troubleshooting iSCSI as a block-based protocol is arguably better supported on vSphere than NFS. The ease of administering a file-based array will always appeal to the SMB market, whereas larger organizations are better suited to the moderately more complex iSCSI.

Costs can influence the protocol used. Often, NAS devices are cheaper than SANs, and iSCSI SANs are cheaper than FC ones. But many of the latest midrange storage offerings give you the flexibility to pick and mix several of the protocols (if not all of them). FC has always been regarded as the more expensive option, because it uses its own dedicated switches and cables; but if you're trying to compare FC to protocols using 10GbE, and you need new hardware, then both are comparatively priced.

10GbE has the added advantage of potential cable consolidation with your host's networking needs. A 10GbE NIC with partial FCoE offloading is arguably the best of all worlds, because it gives you the greatest flexibility. They can connect to a FC fabric, can provide access to iSCSI or NFS, and act as the host's networking NICs. FCoE CNA hardware is still in a state of flux; and as we've seen with the demise of iSCSI HBAs, now that FCoE software initiators are available in vSphere, the CNA cards are likely to be used less and less. Cisco is pushing forward with its Twinax cables with SPF+ connectors, which have so far become the de facto standard; and Intel is pushing 10GbE-capable adapters onto server motherboards.

An interesting design that's becoming increasingly popular is to not plump for a single protocol, but use several. Most arrays can handle FC and Ethernet connections; so some companies are using NFS for general VM usage with their large datastores, for more flexibility for growth and array-based utilities; and then presenting LUNs on the same storage via FC or iSCSI for the VMs more sensitive to I/O demands. It's the ultimate multipathing option.

Finally, remember that DAS can be a viable option in certain, albeit limited circumstances. If you're deploying a single host in a site, such as a branch office, then introducing an additional storage device only introduces another single point of failure. In that situation, shared storage would be more expensive, would probably be less performant, and would offer no extra redundancy.

Multipathing

vSphere hosts use their HBAs/NICs, potentially through fabric switches, to connect to the storage array's SP ports. By using multiple devices for redundancy, more than one path is created to the LUNs. The hosts use a technique called *multipathing* to make the path-selection decisions.

Multipathing can use redundant paths to provide several features such as load balancing, path management (failover), and aggregated bandwidth. Unfortunately, natively vSphere only allows a single datastore to use a single path for active I/O at any one time, so you can't aggregate bandwidth across links.

SAN Multipathing

VMware categorize SANs into two groups:

Active/Active Active/active arrays are those that can accept I/O to all LUNs on all of their SPs simultaneously, without degrading performance (that is, across an SP inter-connect). Every path is active.

Active/Passive Active/passive arrays allow only one SP to accept I/O for each LUN, using other SPs for failover. SPs can be active for some LUNs while being standbys for others—thus all SPs can be active simultaneously, but not for the same datastore. Effectively, a LUN is owned by a particular SP.

Confusingly, storage vendors often refer to their active/passive arrays as active/active if the SPs are both online in this active/standby-standby/active style, to differentiate themselves from arrays that have only one SP active while the other SPs are ready to accept a failover.

vSphere hosts by default can use only one path per I/O, regardless of available active paths. With active/active arrays, you pick the active path to use on a LUN-by-LUN basis (fixed). For active/passive arrays, the hosts discover the active path themselves (MRU).

Native Multipathing Plugin

vSphere 4 introduced a redesigned storage layer. VMware called this its Pluggable Storage Architecture (PSA); and along with a preponderance of Three Letter Acronyms, gave vSphere hosts the ability to use third-party multipathing software—Multipathing Plugins (MPPs).

Without any third-party solutions, hosts use what is called the Native Multipathing Plugin (NMP). The terminology isn't that important, but the NMP's capabilities are, because they dictate the multipathing functionality for the vSphere hosts. To further categorize what native multipathing can do, VMware split it into two separate modules:

Storage Array Type Plugin (SATP): Path failover

Path Selection Plugin (PSP): Load balancing and path selection

SATP

The host identifies the type of array and associates the SATP based on its make and model. The array's details are checked against the host's `/etc/vmware/esx.conf` file, which lists all the HCL-certified storage arrays. This dictates whether the array is classified as active/active or active/passive. It uses this information for each array and sets the pathing policy for each LUN.

PSP

The native PSP has three types of pathing policies. The policy is automatically selected on a per-LUN basis based on the SATP. However, as you can see in Figure 6.3, you can override this setting manually:

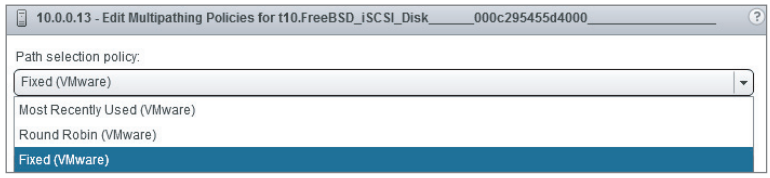


FIGURE 6.3 Path Selection drop-down

Fixed The default policy for active/active array LUNs. It allows you to set a preferred path, which the host uses unless the path has failed. If the preferred path fails and then become available again, the path automatically returns to the preferred one. With a fixed policy, you set the HBA to LUN mappings, providing basic load-balancing to maximize the bandwidth usage across the host's HBAs. Active/passive arrays can suffer from path thrashing if this policy is used.

Most Recently Used (MRU) The default policy for active/passive array LUNs. The MRU policy takes the first working path it finds during bootup. If this path fails, the host moves to another working path and continues to use it. It doesn't fail back to the original path. No manual load-balancing can be performed because MRU doesn't have preferred paths. No configuration is required to use MRU.

Round Robin (RR) RR rotates through all the available optimized paths, providing automated load balancing. This policy can safely be used by all arrays, but active/active arrays with their all-active paths can queue I/O across every path. Microsoft-clustered VMs can't use RR-based LUNs.

MULTIPATHING PLUGIN

Array manufacturers can provide extra software plug-ins to install on ESXi hosts to augment the NMP algorithms provided by VMware. This software can then optimize load-balancing and failover for that particular device. This should allow for greater performance because the paths are used more effectively, and potentially enable quicker failover times. EMC and Dell are examples of storage vendors that have a MPP available.

ALUA

Asymmetric arrays can process I/O requests via both controllers at the same time, but each individual LUN is owned/managed by a particular controller. If I/O is received for a LUN via

a controller other than its managing controller, the traffic is proxied to it. This proxying adds additional load on the controllers and can increase latency.

Asymmetric logical unit access (ALUA), part of the SPC-3 standard from 2005, is the technology that enables an array to use the controllers' interconnects to service I/O. When the link is used, performance is degraded (asymmetric), and therefore without the use of the appropriate ALUA SATP plugin, vSphere treats it as an active/passive array. When a host is connected to an ALUA-capable array, the array can take advantage of the host knowing it has multiple SPs and which paths are direct. This allows the hosts to make better failover and load-balancing decisions. ALUA also helps to prevent the classic path-thrashing problem that is possible with active/passive arrays.

Both RR and MRU policies are ALUA aware and will attempt to schedule I/O via the LUN's Active-Optimized path. RR is considered a better choice for most Active/Passive arrays, although not all arrays support or recommend this pathing policy so check with your vendor. There are two ALUA transition modes that an array can advertise:

Implicit: The array can assign and change the managing controller for each LUN.

Explicit: A host can change the LUN's managing controller.

An ALUA array can use either or both modes. vSphere supports all combinations of modes. The controllers' ports are treated collectively via a target portal group (TPG). The TPG advertises the following possible active states to the hosts:

- Active Optimized
- Active Non-Optimized
- Standby
- Unavailable
- In Transition

Paths can be given a ranking via an `esxcli` command that give administrators some control over the pathing decisions. However, active optimized paths are always picked over active non-optimized paths, even if their set rank is lower.

VMware licenses the ability to use both third-party MPPs and ALUA. To use either of these functions, you need to purchase vSphere Enterprise licenses.

Additional iSCSI Considerations

iSCSI has some additional SAN multipathing requirements that differ depending on the type of initiator used.

Hardware iSCSI Initiators

When you're using hardware initiators for iSCSI arrays, vSphere multipathing works effectively the same as it does for FC connections. The hosts recognize the HBAs as storage adapters and use the NMP with SATP selection and PSP pathing.

Some iSCSI arrays use only one target, which switches to an alternate portal during failover. Hosts detect only one path in these instances.

Software iSCSI Initiators

Software iSCSI initiators require additional configuration steps to use vSphere's storage MPIO stack. By default, software iSCSI uses the multipathing capabilities of the IP network. The host can use NIC teaming to provide failover, but the initiator presents a single endpoint so no load-balancing is available.

To use the vSphere storage NMP and enable load-balancing across NICs, you must use a technique known as *port binding*. Don't use network link aggregation, because you want to define separate end-to-end paths. Follow these steps to enable port-binding for two NICs for iSCSI:

1. Create a vSwitch, add both NICs, and create two VMkernel ports, each with a separate IP address (each NIC needs a one-to-one mapping with a vmk port), as shown in Figure 6.4.

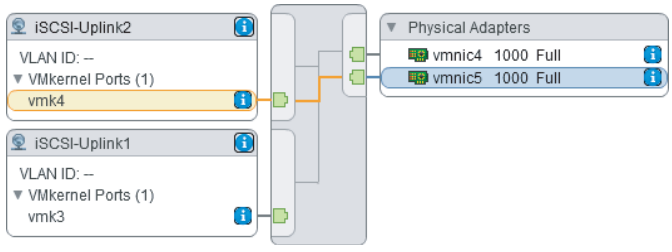


FIGURE 6.4
iSCSI multipathing vSwitch

2. Bind the VMkernel ports to their own NICs. Each VMkernel port should have only one NIC set as Active; all other NICs on the vSwitch must be set to Unused.
3. Enable the software initiator.
4. In the properties of the software initiator, select the Network Configuration tab, and add each port/vmk pair in turn to the iSCSI initiator.
5. Add the targets. If the targets have already been added, you must reestablish the sessions via the `esxcli iscsi session` command (or reboot the host). See Figure 6.5.

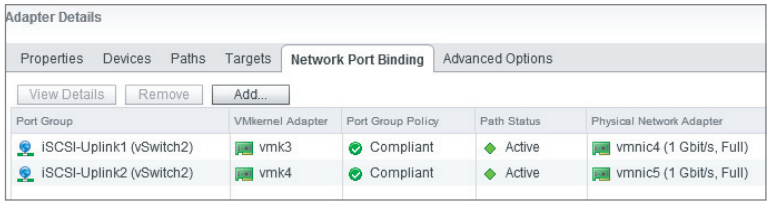


FIGURE 6.5
iSCSI multipathing port binding

NOTE You must place the software initiator in the same subnet as the iSCSI array's target, because port binding in vSphere can't route traffic.

NAS Multipathing

NAS multipathing is fundamentally different than SAN multipathing in vSphere because it relies entirely on the networking stack. MPIO storage tools such as SATPs and PSP aren't available, so IP-based redundancy and routing is used.

For each NFS export mounted by the host, only one physical NIC is used for traffic, despite any link-aggregation techniques used to connect multiple NICs together. NIC teaming provides failover redundancy but can't load-balance an export. But by creating multiple exports along with multiple connections on different subnets, you can statically load-spread datastore traffic.

Chapter 5 looked at designs to provide network redundancy. As an outline, you can use two different methods to create NIC failover for an NFS mount:

- Create two (or more) vSwitches, each with a VMkernel interface, or a single vSwitch with multiple VMkernel uplinks. Each uplink connects to a separate redundant physical switch. The VMkernel interfaces and NFS interfaces are split across different subnets.
- With physical switches that can cross-stack, you need only one VMkernel interface (and so only one IP address). The NAS device still needs multiple IP address targets. The vSwitch needs at least two NICs, which are split across the two cross-stacked switches. The VMkernel's vSwitch has its load-balancing NIC teaming policy set to "Route based on IP hash." You need to aggregate the physical switch ports into an 802.3ad EtherChannel in static mode, or if you are using a v5.1 vDS you can use its dynamic Link Aggregation Control Protocol [LACP] support to configure the EtherChannel. The vDS 5.1 dynamic LACP support is only for the link aggregation setup, it doesn't change the way the actual network load balancing is done — this is still the IP hash algorithm. vSphere 5.0 introduced a very basic method to load spread. By mounting multiple NFS exports on multiple hosts via FQDNs instead of IP addresses, you can use round-robin DNS to mount the targets under different IP addresses.

The load-based teaming (LBT) algorithm available with the Enterprise Plus license is one technology that can load-balance saturated links if there are multiple VMkernel connections to each uplink, split across multiple subnets pointing to multiple target IPs.

NFS can densely pack VMs onto a single connection point with its large datastores, native thin-provisioning, and NAS-based file locking. Sharing the load across several NICs is particularly important when you're using 1GbE NICs as opposed to 10GbE connections.

It's difficult to achieve good load balancing with NFS datastores, and in practice NFS multipathing tends to be limited to failover. As an environment scales up, load-balancing becomes a more important consideration, and often iSCSI may prove to be more suitable if your array supports it.

Finally, give special consideration to the longer timeouts associated with NFS. FC and even iSCSI fail over much more quickly, but NFS can take long enough that you should adjust the VM's guest OSes to prepare them for the possibility that their disk may be unresponsive for a longer time.

vSphere Storage Features

vSphere version 5 was known as *the storage release* for good reason. The number of enhancements, performance improvements, and added functionality meant that storage became a

first-class citizen in vSphere resource management. Many of the things that administrators have become accustomed to for host management, such as shares, limits, DRS, vMotion, and affinity rules, have storage equivalents now.

Closer communications between vCenter and storage arrays allow much of the heavy lifting to be accomplished by the storage, reducing host involvement. The storage arrays are highly efficient at moving data between disks, so offloading storage tasks helps to improve overall operational efficacy.

More information is being accepted back into vCenter from the arrays. vCenter collates and presents the information in terms of datastores and VMs, allowing administrators to make more informed decisions about VM placement and create policy-based rules to manage the storage.

vSphere Storage APIs

As vSphere has matured as a product, VMware has deliberately focused on creating a great set of partner accessible application programming interfaces (APIs). These APIs provide a consistent experience to write supporting applications and tools against vSphere. APIs are the common methods to interact with vSphere, and they expose features to external developers. They help to ensure that between version upgrades of vSphere, minimal changes are necessary to keep associated applications compatible.

As new features are added to vSphere, APIs are introduced or existing ones augmented to reveal the new functions and create a common way of executing against them. Therefore, software and hardware vendors often have to update their applications and firmware to take advantage of the new features.

Several sets of important storage-related APIs are available. You should check with your vendors to see whether their products support the latest vSphere 5 APIs and if you need to update anything to take advantage of improved compatibility or functionality:

vSphere APIs for Data Protection (VADP) Data-protection calls include the change block tracking (CBT) tools. CBT replaced the VMware Consolidated Backup (VCB) technique that was used previously. Many third-party backup tools use VADP to snapshot VMs and quiesce datastores ready to grab off-LAN backups. VADP integration can remove the need for in-guest backup agents.

vSphere APIs for Multipathing (VAMP) Set of APIs to control I/O path selection from hosts to storage devices. Comprises a number of multipathing plugins under the guise of the PSA, such as NMP, MPP, SATP, and PSP, which were all explained in the "Multipathing" section earlier in this chapter. VAMP also allows storage partners to certify their arrays against ESXi and provide enhanced drivers and multipathing tools.

vSphere APIs for Array Integration (VAAI) Allows storage-related tasks to be offloaded from the hypervisor to the storage device. This can significantly reduce the ESXi server's overhead and minimize storage traffic across the SAN fabric.

VAAI is explained in much more depth in the next section.

vSphere APIs for Storage Awareness (VASA) VASA allows vCenter to gain an awareness of a storage device's capabilities. It usually requires a vCenter plugin or some piece of software from the storage vendor for the particular make and model. vCenter can match

these capabilities to the datastores, providing vSphere administrators more detail so they can make more informed storage decisions.

VASA is explained in more depth later in the chapter.

VAAI

vSphere API for Array Integration (VAAI) is a set of storage APIs that VMware initially introduced in vSphere 4.1. These VAAI capabilities are classified into what VMware terms *primitives*. To use the primitives, the storage array must include appropriate support, and each array may only provide a subset of support with some of the primitives. NFS support wasn't available with the 4.1 release, but 5.0 included some equivalent matching features.

VAAI integration with an array means you can offload storage tasks that are normally performed by the host directly to the array. Doing so reduces host CPU, memory, network, and fabric loads and performs operations more efficiently and quickly. VAAI support is divided between block storage and file storage. Because the underlying storage is different, the implementation of these optimizations differs. Many of the primitives are comparable, but they're treated separately because in practice they're executed differently.

VAAI for Block-Based Datastores

VAAI has the following primitives for block-based datastores:

Full Copy Full Copy (also known as Clone Blocks or XCOPY) uses SAN technologies to perform data copies without requiring the hosts to read and write everything. It reduces the time and overhead of Storage vMotions, template deployments, and VM clones. This offloading reduces the hosts' CPU expenditure and reduces fabric I/O.

Block Zeroing Block Zeroing (also known as Write Same) instructs the SAN to repeat certain commands, allowing it to easily zero out disks for eager-zeroed thick disks. This significantly reduces the time required to create these disks, which are used by FT or MSCS-enabled VMs and advised for use with VMs that need high disk performance or added security.

Hardware-Assisted Locking Locking is required as VMFS is a clustered file system where multiple hosts read and write to the same LUN. Locking allows the host to understand when it has control, or another host does, and how to seize control if a host goes offline unexpectedly.

Hardware Assisted Locking, also known as Atomic Test & Set (ATS) locking, provides a better locking mechanism than the existing SCSI reservations traditionally used. This can help improve datastore access efficiency and scalability. VMFS-5 in vSphere 5.0 enhanced the hardware-assisted locking capabilities further.

Thin Provisioning Stuns and Reclamation vSphere 5.0 added support for thin-provisioning stuns. This had been earmarked for release in 4.1 as a fourth primitive but was withdrawn shortly before 4.1 was made available. If a thinly provisioned datastore runs out of space, any VMs with outstanding disk-write I/O are paused. The remaining VMs continue to run until additional datastore space is freed up or added, or those VMs want to write to disk and are paused. This stun feature prevents VMs from crashing and provides a gentler approach to dealing with out-of-space datastores as a result of thin provisioning. vSphere 5 also added in a

new default vCenter alarm to warn users when a thinly provisioned datastore goes over 75 percent usage.

Thin provisioning dead-space reclamation in vSphere 5 (or SCSI UNMAP, as it's also known) tells a storage array when a previously used block of space is no longer used. This is particularly useful in conjunction with Storage DRS, because the latter is likely to cause far more Storage vMotions to occur. There was a performance issue with this feature in the original 5.0 release. 5.0 Patch 2 disabled it as a result; and 5.0 Update 1 fixed the problem but left it globally disabled by default, resulting in the need to invoke with this manually from the command line with:

```
vmkfstools -y
```

VAAI for File-Based Datastores (NFS)

vSphere 5.0 added the NFS VAAI equivalents. Vendor-specific vCenter plugins are required for the NFS VAAI primitives:

Full File Clone This is similar to the block-based Full Copy command, by offloading cloning to the array directly.

Reserve Space The Reserve Space function is similar to the Block Zeroing VAAI primitive. It allows NFS datastores to now create lazy-zeroed and eager-zeroed disks.

Native Snapshots VM snapshots can now be offloaded to the array with the Native Snapshots primitive. This was introduced in vSphere 5.1 and requires that the VMs are at hardware version 9.

Extended Statistics Extended statistics provide insight into the NAS datastores. This helps to prevent thin provisioning out-of-space issues on NFS storage.

Use of VAAI is license dependent, so the hosts must have a minimum of an Enterprise-level license to take advantage of the hardware acceleration. The primitives help to remove bottlenecks and offload storage tasks that are expensive for the hosts to perform. This not only improves host efficiency, but also increases scalability and performance.

Check the VMware storage HCL for compatibility with different arrays: some arrays may require a firmware upgrade to support the VAAI primitives, and only a subset of primitives may be available.

VASA

VASA is the set of standardized API connections that provide vCenter with insight into the capabilities of a storage array. If the array supports the API, then it can advertise three primary information sets:

Storage topology: How the array is configured. For example, RAID sets, replication schedules, LUN ownership, and thin provisioning.

Functional capabilities: What the array is capable of.

Current state: Can include health, configuration changes, events, and alarms.

For vCenter to support the array, two things are normally required. First, the array must support the API. Often this means a firmware upgrade to the SPs to add support. VMware's HCL

has details on which arrays support VASA and their required array software level. Second, the array vendor will provide a plugin for vCenter or additional software that needs to be installed. This allows vCenter to correctly interpret the data being provided by the array. The information provided may differ between storage vendors.

VASA helps in the planning, configuration, and troubleshooting of vSphere storage. It should reduce the burden normally associated with managing SANs and lessen the number of spreadsheets needed to keep track of LUNs. As you'll see later in this chapter, the administration of several vSphere tools such as profile-driven storage and datastore clusters can benefit from the additional visibility that VASA information can provide. Storage tasks can become more automated, and users have an increased situational awareness with on-hand insight to the array.

Performance and Capacity

Several vSphere storage features are centered around maximizing the performance and capacity of vSphere's storage. Many of the technologies layer on top of each other, resulting in Storage DRS. Others push the limits of the file system or bring some of the innovation found in the storage arrays to use in vCenter operations.

VMFS-5

As of vSphere 5.0, the default VMFS volumes created are VMFS-5. VMFS-3 datastores created by legacy hosts are still supported and fully functional under vSphere 5. A number of enhancements in VMFS-5 provides additional opportunities from a design perspective:

- VMFS-5 datastores can be up to 64 TB in size without having to combine multiple 2 TB LUNs. This reduces the management overhead previously associated with such large datastores. (The extents feature is still available to concatenate additional LUNs, if you need to grow existing volumes with extra space.)
- Very large volumes and file sizes are now possible with 64 TB physical RDMs. VMDKs and virtual RDMs are still limited to 2 TB minus 512 KB.
- A single 1 MB block size is used for all newly created VMFS-5 volumes. These 1 MB block volumes no longer restrict the file size.
- Numerous performance and scalability improvements have been made, such as ATS file locking, an improved subblock mechanism, and small file support. These are perhaps not immediately relevant to design per se, but the functionality they provide, such as expanding the number of VMs per datastore, is important.

What to Do If You Have VMFS-3 Volumes

Older VMFS-3 volumes are still supported by vSphere 5 hosts as valid datastores, so you don't have to upgrade immediately. But you can't take advantage of the improvements of VMFS-5 until you do. Clearly, a newly designed and implemented vSphere 5 environment won't have VMFS-3 datastores to worry about. But if you've gotten to vSphere 5 via an upgrade, or you've joined legacy hosts to your vCenter, then you'll want to migrate to VMFS-5 at some point.

Before you consider how you want to migrate that datastore, you'll need to ensure that all the hosts that need to connect to it are ESXi 5.0 at minimum. Legacy hosts can't mount VMFS-5 volumes.

continued

You have two options for the migration. First, you can create a fresh datastore formatted with VMFS-5 and migrate your VMs, probably with Storage vMotion. This requires a bit of planning, and you'll need a spare LUN (or at least sufficient space in your existing datastores). Preferably, this LUN is at least as large as the biggest datastore you want to replace. Alternatively, you can perform an in-place upgrade. Such an upgrade is nondestructive and nondisruptive—the VMs can stay where they are and don't even need to be powered off. Obviously this is much less onerous than the migration strategy.

So why consider creating new VMFS-5 datastores rather than simply running in-place upgrades? Because when you merely upgrade a datastore, the following adverse results occur:

- The old variable block sizes are carried over. This can affect the performance of subsequent Storage vMotions.
- An upgraded VMFS-5 volume doesn't have all the scalability enhancements found in native VMFS-5 datastores.
- There may be inconsistencies across your datastores. In addition to the potentially different block sizes, the starting sectors will be different, and datastores under 2 TB will have MBR partitioning (as opposed to GUID partitioning on native VMFS-5).

If a full migration strategy is too difficult to consider initially, you can plan two phases. Apply the upgrade to take immediate advantage of most of the VMFS-5 benefits, and then revisit the datastores with a migration to ensure a consistent outcome with all the features. Just be sure to appropriately record (perhaps with a suffix on the datastores' name) which datastores have been upgraded and are awaiting a clean reformat.

Rebuilding the datastores not only provides the full spectrum of VMFS-5 features, but also gives you the opportunity to redesign your storage layout more appropriately for vSphere 5. There is a good chance that if the datastores are old, they were sized and built with an older architecture in mind.

Storage I/O Control

Storage I/O Control (SIOC) is a feature that was introduced in vSphere 4.1 to improve the spread of I/O from VMs across a datastore. It provides a degree of quality of service by enforcing I/O shares and limits regardless of which host is accessing them. SIOC works by monitoring latency statistics for a datastore; when a predetermined level is reached, SIOC scales back I/O via allotted shares. This prevents any one VM from saturating the I/O channel and allows other VMs on the datastore their fair share of throughput.

Just as CPU and memory shares only apply during contention, SIOC will only balance the I/O spread when the latency figures rise above the predefined levels. SIOC can enforce I/O with set IOPS limits for each VM disk and distributes load depending on the datastore's total shares. Each host with VMs on the datastore uses an I/O queue slot relative to the VM's shares, which ensures that high-priority VMs receive greater throughput than lower-priority ones.

In vSphere 5.0, this feature has been extended to NFS datastores (previously only VMFS volumes were supported). RDM disks still aren't supported.

To configure SIOC, do the following:

1. Enable the SIOC feature in the datastore's properties.
2. Set the shares and IOPS limit for each VM disk on the datastore (optional).

By just enabling SIOC on the datastore, you're automatically protecting all the VMs from a VM that is trying to hog the I/O to a datastore. Without any adjustment in the second step, all the disks will be treated equally; so unless you need to prioritize particular VMs, enabling it on each datastore is all that's required. If you're worried about a specific VM being a bully and stealing excessive I/O, then a limit on that one VM is all that's required. However, just as with CPU and memory limits, be careful when applying limits here because they artificially block the performance of the VM's disks and apply even when there is no contention on that datastore's I/O. Shares are the fairest method to use and the least likely to cause unexpected side effects.

SIOC only works if it knows about all the workloads on a particular datastore. If the underlying disk spindles are also assigned to other LUNs, then SIOC will have problems protecting and balancing I/O for the VMs, and a vCenter alarm will trigger. You should set the same share values across any datastores that use the same underlying storage resources. SIOC requires an Enterprise Plus license for every host that has the datastore mounted.

It's possible to adjust the threshold value set on each datastore. By default in vSphere 5.0, it's set to 30 ms, but you can use any value from 10 ms up to 100 ms. The default value is appropriate in most circumstances; but if you want to fine-tune it to a specific disk type, then SSD datastores can be set lower at around 10–15 ms, FC and SAS disks at 20–30 ms, and SATA disks at 30–50 ms. Setting the value too high reduces the likelihood that SIOC will kick in to adjust the I/O queues. Setting it too low means shares are enforced more frequently, which can unnecessarily create a negative impact on the VMs with lower shares. vSphere 5.1 automatically determines the best latency threshold value to use for each datastore. It tests the datastore's maximum throughput and sets the threshold to 90% of the peak.

Whereas SIOC prevents bottlenecks in I/O on the datastore, NIOC prevents bottlenecks on individual network links. On converged networks where IP-based storage traffic is less likely to have dedicated NICs, then NIOC can complement SIOC and further protect the storage traffic.

Datastore Clusters

A datastore cluster is a new vCenter object that aggregates datastores into a single entity of storage resources. It's analogous to the way ESXi hosts have their CPU and memory resources grouped into a host cluster. A datastore cluster can contain a maximum of 32 datastores and you're limited to 256 datastore clusters per vCenter instance.

You can keep datastores of different capacities and with different levels of performance in the same clusters. But datastore clusters are the basis for Storage DRS, and as will become apparent in the following sections, you should try to group datastores with similar characteristics in the same datastore cluster. In the same vein, datastores located on different arrays, unless identical and identically configured, aren't good candidates to cohabit a cluster. Consider the number of disks in the RAID sets, type of RAID, type of disks, and manufacturers or models with different controller capabilities and performance. For example, imagine you try to mix some small, fast SSD-based datastores with some larger, slow SATA-based datastores in the same datastore cluster. The I/O and space balancing will inherently work against each other because the Storage DRS will favor the SSD datastores for speed, but the SATA datastores for their capacity. Having similar disk performance provides a stable and predictable environment in which Storage DRS can work well. If you have datastores with very different characteristics, then you should consider splitting the datastore cluster into smaller but more balanced clusters.

Datastore clusters can contain VMFS or NFS datastores; but as a hard rule, you can't have NFS and VMFS together in the same datastore cluster. Additionally, you shouldn't have replicated and nonreplicated datastores together in the same datastore cluster. You can put VMFS-3,

upgraded VMFS-5, and natively built VMFS-5 volumes in the same datastore cluster; but given an understanding of the differences between them and the impact that this can have on Storage vMotion, capacity limits, and locking mechanisms, it isn't something we recommend. If you have a mixture of VMFS volumes, you should ideally rebuild them all to VMFS-5. If you can't rebuild them to VMFS-5, then you should consider splitting them into multiple clusters until they can be rebuilt. If there are enough VMFS-3 or upgraded VMFS-5 datastores with disparate block sizes, it would be advantageous to group them by their type.

Storage DRS

Just as datastore clusters are comparable to host clusters, Storage DRS is commensurate to host DRS. Storage DRS attempts to fairly balance VMs across datastores in the same datastore cluster. It looks at capacity and performance metrics to store VMs in the most appropriate location. It takes a datastore cluster as its boundary object and uses Storage vMotion to relocate VMs when required.

Storage vMotion in vSphere 5

Storage vMotion is a vSphere feature that allows running VMs to migrate nondisruptively from one datastore to another. During the Storage vMotion, the VM is always running on the same host server. The VM can move to any datastore that the host has mounted, including moving from local to shared volumes and back, or NFS to block-based storage. Additionally, Storage vMotion can be used to transform the disks from thin-provisioned to thick and back again, if the destination datastores support it.

Unlike previous versions, the Storage vMotion included in vSphere 5 uses a method known as Mirror Mode, which improves efficiency. It's called Mirror Mode because it mirrors I/O to both source and destination if it knows it has already copied that block. This means the Storage vMotion operation is conducted in a single pass over the disk and no longer needs to repeatedly copy delta snapshots. This not only improves the speed of the process but also makes the duration more predictable and allows VMs with snapshots and linked clones to be moved.

In vSphere 5, the snapshot delta disks are stored in the same directories as the parent disk, as opposed to previous versions that kept them all in the VM's home directory. This home directory is still set with the parameter `workingDir`, but it's now only used by snapshots to store the data file (`.vmsn`) aside the VM's other home files. This means all the delta disks share the same performance characteristic as the parent disk, and growth of the delta disks isn't forced into the parent's home directory.

If the source and destination are on the same array, and the array supports VAAI hardware acceleration, then the Storage vMotion should be offloaded as an in-band array operation. This can significantly increase the speed of your Storage vMotions. vSphere 5.1 Storage vMotion can run up to four parallel VMDK disk migrations per VM, instead of running each disk serially. This can speed up the overall Storage vMotion especially if the disks are spread across datastores backed by different spindles.

In addition to using datacenter cluster constructs and the Storage vMotion process, Storage DRS also uses SIOC to gather I/O metrics and information about capabilities of each datastore.

When you enable Storage DRS, SIOC is automatically turned on as long as all the hosts connected to the datastores are at least ESXi 5.0.

It's worth noting that SIOC and Storage DRS are largely independent technologies, which can complement each other in a final design solution. SIOC provides immediate protection on the I/O path for VM performance: it's a reactive, short-term mechanism. In comparison, Storage DRS is measured over several hours, attempting to preemptively prevent issues and solving not only performance bottlenecks but also capacity ones.

Storage DRS will work with both VMFS- and NFS-based storage (although as we've said, you shouldn't mix them in the same datastore cluster).

TIP Storage DRS is both a performance and a capacity feature.

Performance and Capacity Thresholds

When a datastore cluster is created, the wizard allows you to adjust the capacity and I/O thresholds that trigger or recommend migrations. Figure 6.6 shows the basic and advanced settings available:

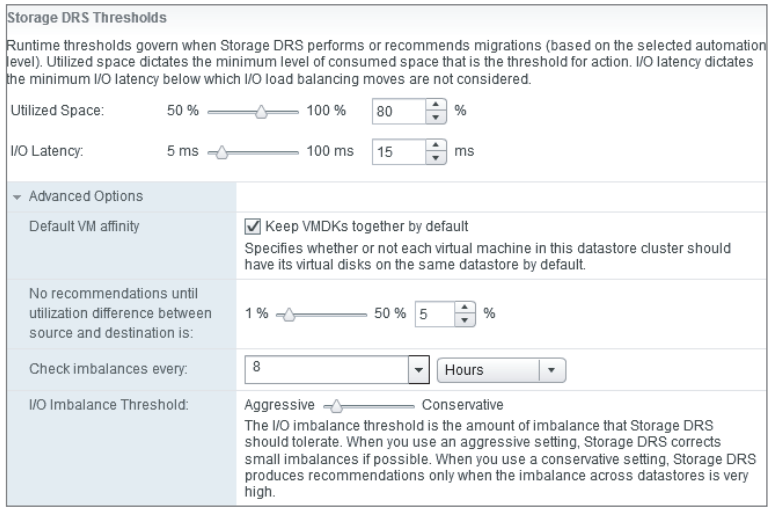


FIGURE 6.6 Storage DRS threshold settings

Utilized Space The Utilized Space slider lets you specify how much used space should be the upper limit for each datastore. This can be a value between 50 percent and 100 percent. Setting it as high as 100 percent effectively tells Storage DRS to ignore datastore capacity as a threshold.

This metric will help to avoid an out-of-space datastore by recommending migrations or automatically remediating space imbalances. The setting isn't a warning limit or an upper level of how full the organization wants to see its datastores; it's the point below which you don't care if the datastores are unbalanced. Finding the right setting for the datastore cluster may depend on how much space is currently being used (assuming these aren't new

datastores), how quickly an increase of space is consumed and recouped, how quickly the array will take to Storage vMotion disks, and how risk-averse the environment is.

Storage DRS won't move or recommend a move until one of the datastores hits the threshold. If the slider is set too low, too many migrations may be generated. Each Storage vMotion creates work for the host and the array when disks are moved, so unnecessary migrations consume resources that could be better used. Setting the slider too high could mean that the datastores become excessively unbalanced and a datastore fills up before Storage DRS has had the chance to move the existing disks around.

Thin-provisioned disks and datastores are accommodated in two ways with Storage DRS. When the disks are thin-provisioned by vSphere, Storage DRS monitors the growth of the VMs as well as the datastores, and it looks at allocated space when considering potential migrations. If the datastores are on array-based thinly provisioned datastores, then the VAAI thin-provision primitive is important because it ensures that the space created after a Storage vMotion occurs is freed on the datastore.

I/O Latency The I/O Latency threshold is set to prevent longer-term performance imbalance across the datastores in the cluster (remember, SIOC is used to prevent I/O bottlenecks in near real time). It evaluates the I/O latency, measured in ms, over a day and recommends Storage vMotions to equal out the I/O loads between VMDK disks across the datastores.

In the SIOC section, we recommended I/O latency levels for different disk types. By default in vSphere 5.0, the SIOC I/O latency is set to 30 ms. The Storage DRS latency level should always be set lower than or equal to the SIOC level. This ensures that a longer-term proactive remediation is used in preference to SIOC throttling when the latency is more chronic and can be resolved easily with spare I/O capacity on other datastores. Storage DRS uses the 90th percentile of latency so it doesn't pay attention to the peaks and surges of activity that could throw off the calculations and are arguably better dealt with by SIOC shares. vSphere 5.1 uses the same 90% default for SIOC, but if you choose to override the latency default you should ensure that the Storage DRS value isn't greater than the SIOC value you set.

Advanced Options You can also access three advanced options via a drop-down menu on the same page, as shown in Figure 6.6:

Space Utilization Difference This setting is used to prevent excessive rebalancing when all the datastores are close to the threshold levels. It guarantees that a Storage vMotion will only choose a destination if there is enough of a difference from the source datastore. By default, if there isn't a difference of at least 5 percent between the source and any potential destination datastores, then no migrations will occur or be recommended.

I/O Load Balancing Invocation Interval The interval, eight hours by default, is the frequency at which I/O load balancing recommendations are made. If this value is set to 0, then I/O load balancing is disabled. Initially, you'll need to wait 16 hours for the first recommendations to be made. Recommendations are made every 8 hours, but they're based on the previous 24 hours' worth of data.

I/O Imbalance Threshold You can set Storage DRS to be more or less aggressive, similar to the way it can be set for host DRS clusters.

Generally, the settings in the advanced options can be left at their default values. You'll only change one or more of them if you're experiencing problems with the defaults or have an unusual requirement.

One of the important things to remember about the performance and capacity thresholds is that if no datastore is considered to be running out of space, or if performance is not degrading according to the threshold levels you set, then Storage vMotion won't recommend any migrations. Unlike host DRS, where the vMotion migration is simply set with the aggressive/conservative slider, Storage DRS has set limits to reach—percent of space or I/O latency counters—before it even considers differences and aggressiveness. This gives you considerably more control in your environment and also reflects the additional resource cost of Storage vMotions over host vMotions.

Additionally, not only do thresholds have to be reached, but there has to be sufficient imbalance before any recommendations are made. It isn't enough to exceed a space or I/O ceiling; there must be a suitable destination datastore that hasn't reached the threshold or that isn't approaching the same levels of overuse.

Initial Placement and Regular Balancing

Storage DRS is invoked during the initial placement of disks and on a regular basis at frequent intervals. Whenever a VM is created, cloned, cold-migrated, or Storage vMotioned, Storage DRS attempts to best-place the disks to balance the space and the I/O across the datastore cluster. This mean VMs are balanced across datastores from the outset, and it can simplify and expedite the provisioning process because you know vCenter has already calculated the best-fit for the VM without the need for manual calculations. Instead of specifying a datastore, you select the appropriate datastore cluster, and Storage DRS intelligently decides the best home for the VM. Even if I/O balancing has been disabled for regular balancing, the I/O levels are still considered during the initial placement to ensure that a new VM isn't placed on a heavily loaded datastore.

The frequent and ongoing balancing of the datastores after the initial placements ensures that as disks grow and new I/O loads are added, any imbalances that result are dealt with. The I/O latency is evaluated every eight hours with recommendations made every day, and the space usage is checked every two hours. Even if there isn't enough space or I/O capacity in a destination datastore, Storage DRS can move smaller VMs around to create suitable space. Ongoing balancing also helps when additional datastores are added to the datastore cluster, so that the additional capacity and potential I/O use are absorbed by the cluster and VMs can quickly start to take advantage of it.

Storage DRS's initial placement is arguably the most immediate and beneficial feature, even if doesn't sound as compelling as automated load-balancing. It makes the work of configuring datastore clusters instantly apparent, without the onerous testing that most organizations will run before being comfortable with automated load-balancing. Initial placement helps prevent most storage bottlenecks from the outset. It provides a more scalable, manageable design.

Automation level

There are two automation levels for Storage DRS in a datastore cluster:

- No Automation (Manual Mode)
- Fully Automated

Unlike host DRS clusters, there is no halfway Partially Automated setting, because a data store cluster's automation level doesn't affect the initial placement. The initial placement is always a manual decision, although it's simplified with the aggregated datastore cluster object and best-fit recommendation.

Fully Automated means all ongoing recommendations are automatically actioned. Manual Mode is safe to use in all cases, and this is where most organizations should start. It allows you to see what would happen if you enabled Fully Automated mode, without any of the changes actually occurring. It's advisable to run under Manual Mode until you understand the impacts and are happy with the recommendations Storage DRS is making. After checking and applying the recommendations, you can turn on Fully Automated mode if it's deemed suitable.

If there are concerns regarding a Fully Automated mode implementation impacting the performance of the hosts or storage arrays during business hours, it's possible to create scheduled tasks to change the automation level and aggressiveness. This allows you to enable the Fully Automated mode out-of-hours and on weekends, to increase the likelihood that migrations happen during those times and to reduce the risk of clashing with performance-sensitive user workloads.

Individual VMs can also be set to override the cluster-wide automation mode. This gives two possibilities: set the cluster to Manual Mode and automate a selection of VMs (and potentially disable some); or, alternatively, set the cluster to Fully Automated but exclude some VMs by setting them to Manual or Disabled. When you disable a VM, its capacity and I/O load are still considered by the cluster when making calculations, but the VM is never a candidate to be moved. This allows a granular enough design for applications that should only be moved under guidance or never at all.

Manual Mode lets an organization become comfortable with the sort of moves that Storage DRS might make should it be set to the Fully Automated setting, and it also lets you test the different threshold levels. Before you move from Manual to Fully Automated, take the time to adjust the threshold levels and monitor how the recommendations change. You should be able to draw the levels down so Storage DRS makes useful recommendations without being so aggressive as to affect performance with overly frequent moves.

Maintenance Mode

Storage DRS has a Maintenance Mode, again mirroring the host DRS functionality. Maintenance Mode evacuates all VMs from a datastore by Storage vMotioning them to other datastores in the cluster while following Storage DRS recommendations. Storage DRS ensures that the additional load is suitably spread across the remaining datastores. Just as Storage DRS assisted with load-balancing capacity and performance when a new datastore was being added, Maintenance Mode helps with load-balancing when a datastore is being removed.

Maintenance Mode is useful when entire LUNs need to be removed for storage array maintenance, and it's of particular assistance during a planned rebuild of VMFS datastores to version 5. You can create an initial datastore cluster of VMFS-3 volumes; then, datastore by datastore, you can clear them of VMs with Maintenance Mode, reformat them fresh with VMFS-5, and join them to a VMFS-5–only datastore cluster. As space on the new cluster is created, VMs can be migrated across. This is also an excellent time to upgrade the VMs to the latest VMware tools and VM hardware, replace network and SCSI hardware with VMXNET3 and PVSCSI, and perform any appropriate additional vSphere 5 upgrades to the VMs. This cut-over provides a clear delineation of upgraded VMs residing on the new datastore cluster.

Affinity Rules

Affinity rules in datastore clusters are similar to the rules that can be created in host clusters, except they're used to keep together or separate a VM's disks. Datastore cluster affinity rules

allow control of a VM's disks or individual disks. Affinity rules are enforced by Storage DRS during initial placement and subsequent migrations, but they can be broken if a user initiates a Storage vMotion.

By default, an inherent storage affinity rule is in place for all VMs registered in a cluster, which means a VM's disks and associated files will stay together in the same datastore unless you manually split them or create an anti-affinity rule. This makes troubleshooting easier and is in keeping with what vSphere administrators expect to happen. However, three sets of affinity rules are available in datastore clusters, should your design require them:

VM Anti-Affinity A VM anti-affinity rule keeps all the disks from two or more VMs on separate datastores within the same cluster. This is useful if you have a scaled-out set of VMs, such as a farm of web servers, that you'd like to keep on different datastores to prevent any single points of failure.

VMDK Affinity VMDK affinity ensures that disks from the same VM are kept on the same datastore. This maximizes VM availability when all the disks from a VM are required for the application's uptime. Spreading disks like that only increases your risk by multiplying the failure points.

VMDK Anti-Affinity Selecting VMDK anti-affinity for some of a VM's disks ensures that they're separated across multiple datastores. This can be useful if the guest OS is using any disk mirroring or RAID-type software in which you want the disks split for redundancy's sake. You can also use VMDK anti-affinity if you know that certain disks are extremely I/O intensive or unusually large, and you always want to keep them apart to manually spread the load—for example, two data disks of a database server.

Splitting a VM's Disk Files by Type

Although you can create multiple VMDK anti-affinity rules for all your VMs to separate their common disk types (for example, OS, swap, and data), we don't recommend it. Doing so doesn't keep all the like disks together, and you'll end up with so many rules that Storage DRS will grind to halt and no sensible recommendation will be made. A better approach is to use Storage Profiles, which are explained later in this chapter. Using Storage Profiles, you can identify datastores and datastore clusters for specific use cases and mark individual disks to fit profile types.

Although the cluster default is to keep all of a VM's disks together, this restricts Storage DRS's options when it tries to balance the disks as much as possible. If you want the best possible balance of performance and capacity, you can remove the cluster's inherent VMDK affinity or enable it on a per-VM basis. Just be aware that you may increase your risk by spreading a VM across multiple datastores, with a single failure of a datastore likely to affect far more VMs.

If you set the host cluster option to keep all of a VM's swap files on a host's local disk or a specified datastore, then Storage DRS is automatically disabled on those disks.

Storage DRS Impacts

There are certainly circumstances in which you should be wary of enabling all the Storage DRS features. However, as a general rule, you should attempt to aggregate your storage into datastore

clusters and set Storage DRS to Manual Mode. This can safely be enabled for workloads, and you immediately take advantage of reduced points of management through the grouping of datastores, the initial placement recommendations, the ability to create affinity rules, and the ongoing recommendations for capacity management.

Although Storage DRS is aware of thinly provisioned disks created by vSphere, it can't recognize array-based thin-provisioned LUNs by default. This doesn't create a problem for vSphere, but could cause over-provisioning issues for the array if it migrated VMs onto disks that weren't appropriately backed by enough storage. One of the VAAI primitives, if it's available with your array, can warn you about the issue and create a vCenter alarm when the array's LUN is 75 percent full.

When a datastore's underlying disk is deduplicated or compressed on the array, Storage DRS is unaware and won't factor this into the migration calculations. When a VM is moved via Storage DRS, it's effectively inflated on the array, even though the space balancing will appear to have been successful to vCenter. The amount of space recovered by a move may not be as much as expected, but Storage DRS will continue to recommend further moves until the required balance is found. This shouldn't cause any issues, but the space won't be truly balanced on the back end until the dedupe or compression job is run again. To lessen the effect, you can plan to apply the Storage DRS recommendations shortly before the array's space-recovery job is next due to commence.

You should be aware of the array's scheduled tasks if you run regular snapshot jobs. After VMs have been moved between LUNs, you should rerun any snapshot jobs to make sure the new layout is incorporated. If you use VMware-aware backup software, check with the vendor to be sure it's Storage vMotion and Storage DRS aware.

In a couple of cases, you should consider turning off some of Storage DRS's features. With the release of vSphere 5.0 and SRM 5.0, VMware doesn't support the combination of SRM-protected VMs being automatically migrated around with Storage DRS. It can leave some VMs unprotected before SRM realizes the disk files have moved. Additionally, Storage vMotion, and by extension Storage DRS, isn't supported by SRM's vSphere Replication (VR) feature.

Use caution when combining Storage DRS's I/O load-balancing with any underlying storage array that uses automatic disk tiering, because the I/O balancing may not work as you expect. Storage DRS finds it hard to categorize the underlying disks' performance because it may be hitting one or more tiers of very different disks. Also, some tiering software works as a scheduled task on the array, which is unlikely to be aligned perfectly with Storage DRS's runs. This again will tend to cause spurious results, which could create non-optimal I/O recommendations.

As a general rule, you should enable Storage DRS for out-of-space avoidance and initial placement whenever possible. However, you should seek advice from your storage vendor with regard to any capacity-reduction or performance-enhancing features and their compatibility with Storage DRS.

Storage Management

In addition to all the performance and capacity enhancements, vSphere's ability to manage storage has grown enormously. These storage-management features have become possible largely due to the adoption by many of the storage vendors of the new storage APIs discussed earlier. The information and efficiencies provided by VAAI and VASA allow for greatly enriched storage-management capabilities.

Datastore Clusters

We've already explained the functionality associated with datastore clusters under the guise of the improved capacity and performance possible with Storage DRS. However, it's worth remembering that datastore clusters also provide a substantial improvement in storage management in vSphere. The ability to reference one cluster object backed by multiple datastores is a significant step in scalable management.

vSphere administrators have grown accustomed to dealing with host resources as a cluster of compute power instead of lots of individual servers. Now datastore clusters provide a similar analogy to vSphere storage.

Profile-Driven Storage

Profile-driven storage, or Storage Profiles as it's commonly called, is a feature that defines tiers of storage grouped by their underlying capabilities. This grouping of storage lets you apply policies and run ongoing compliance checks throughout the VM's lifecycle, allowing for greater levels of automation, scalability, and discoverability. During VM provisioning, datastores or datastore clusters can be chosen more appropriately when you have a better understanding of the capabilities of each.

Datastore Capabilities

The profile-driven storage tiers can be designated in one of two ways:

VASA VASA allows you to see in vCenter the characteristics the array makes available. This depends on the array being a suitable provider and normally requires the installation of a vendor specific plugin on the vCenter Server.

User-Defined A vSphere administrator can define Storage Profiles manually. They allow you to tag datastores with known capabilities, so those capabilities can be grouped and datastores treated collectively when a desired capability is required.

If VASA information is available to vSphere, then the process of tagging datastores is automatically provided for you. However, at the time of writing, there is little out-of-the-box support for this, and each vendor implements the capabilities in its own way. If all your storage is provided by one vendor and one model series, then the VASA information can be an invaluable timesaver. VASA not only provides the information but also associates it with the appropriate datastores. You still have to create the Storage Profiles to assign the tagged datastores to the VMs.

If you're dealing with a very mixed environment, or a VASA plugin is unavailable or doesn't provide the required detail, then you can define your own classifications to suit your individual needs. You'll need to create the profiles and then manually associate them with each applicable datastore. The ability to define your own storage tiers means you can use VASA-capable arrays to support this feature, and any array on VMware's HCL can be defined manually.

VASA information describes capabilities such as RAID types, replication, thin provisioning, compression, and deduplication. User-defined capabilities provide the capacity to include data store tagging based on non-array-specific datastore implementations in an organization, such as backup levels or DR protection and replication levels.

It's common to label such tiers with monikers like Gold, Silver, and Bronze. You should be cautious with naming your datastore-capable tiers generically, because they often have subtly

different connotations between teams and between functional use cases. If you can drive a disciplined strategy across the business, where the same VMs will be Silver for DR replication, VM performance, backup RPO/RTO, disk capacity, array LUNs, network QoS, and support SLAs, then basing your vSphere storage tiers on this structure makes obvious sense. However, if you're commonly faced with application exceptions here, there, and everywhere, we recommend that each objective have different naming conventions. The meaning of Gold, Silver, or Bronze is often different for everyone.

VM Storage Profiles

The VM Storage Profiles depend on how you intend to group your VMs and what you intend to use profile-driven storage for. There are countless ways to classify your VMs. Again, and definitely confusingly, a common approach is to group and label them as Gold, Silver, and Bronze. For the same reasons expressed in the previous section, this is rarely a useful naming strategy. Defining the Storage Profiles with clear descriptions is more practical and user-friendly. How you group the VMs is a more interesting design choice.

Some typical VM Storage Profile groupings you can create are based on the following:

- Application
- Business criticality
- Department
- Performance demands
- VMDK-specific categorizations

Once the VM Storage Profiles are created, they can be mapped to datastore capabilities. You can map Storage Profiles one-to-one with a datastore capability, with one-to-many meaning a single capability that can stretch multiple profile use cases (for example, all RAID 1/0 datastores can be used by four different application groups); or, less likely, you can use a many-to-one case (for example, both RAID 5 or RAID 6 storage can be used for a particular department's VMs).

When the VM Storage Profiles have been created and the datastores have been classified, the profiles are automatically attached during a new VM's deployment. The profiles are only a guide; the destination is still a user's choice, and as such incompatible datastores or datastore clusters can be selected. VM Storage Profiles aren't automatically and retrospectively fitted to existing VMs. Those VMs that already exist must be classified manually, to ensure that they're on the correct type of storage and their future compliance can be checked.

Compliance

Storage Profiles are useful in easing the provisioning of new VM, and they allow compliance checking at any stage in the future. The compliance reports identify any VMs and any VMDK disks that aren't stored on an appropriate datastore.

Storage Profile Benefits

Storage Profiles bring a profile-based system to the storage decisions at the time of VM creation. This minimizes the per-VM planning required and increases the likelihood that the VM will be placed on the right type of storage from the outset. Whereas datastore clusters ensure that in a

collection of similar datastores, the best datastore is chosen for I/O and capacity reasons, Storage Profiles help you pick the right type of datastore across differently backed options. As a simple example, when you deploy a VM, the Storage Profile helps you choose between the datastores (or datastore clusters) that are replicated and those that aren't; but datastore clusters let you choose the best replicated datastore.

Profile-driven storage can assist when datastore clusters are created. Datastore clusters work most efficiently as vCenter objects when they group similar datastores together. Profile-driven storage tiers help to identify those datastores that are backed by the most similar LUNs. When a VM is matched to a Storage Profile, a datastore cluster can be presented instead of individual datastores.

Storing different VMDK disk types on differently backed datastores is made much less complex with Storage Profiles. It's possible to split each VM's disk into categories; you can do this by applying profiles down to the VMDK level. It allows for subsequent checking and remediation to ensure ongoing compliance. It's now feasible to split guest OS disks onto different datastores in a way that is manageable at scale. Be aware that Storage DRS balancing is disabled for VMs split across multiple datastore clusters. As an example, consider these datastore clusters:

DS cluster—OS disks: Datastores with RAID 5 backed LUNs that are replicated daily to the DR site

DS cluster—swap disks: Small datastores with RAID 1/0 backed LUNs that aren't replicated and are never backed up

DS cluster—data disks: Datastores with RAID 6 backed LUNs that are replicated hourly to the DR site

Clearly, applying this level of definition would be time-consuming; but if your storage needs to be managed in such a granular fashion, then profile-driven storage is an invaluable toolset.

Datastore and Host Cluster Designs

Prior to the emergence of datastore clusters in vSphere 5, the single aspect of host cluster sizing was relatively straightforward. Chapter 8 looks at the classic discussion of cluster sizing, i.e. one large cluster of many hosts, or several smaller clusters with fewer hosts in each. There are advantages and disadvantages of each approach, and different circumstances (the functional requirements and constraints) call for different designs. Before vSphere 5, each host cluster would normally be connected to multiple datastores, and the recommended practice of ensuring every host in the cluster was connected to every datastore meant DRS was as efficient as it could be.

As we've just seen, datastore clusters have a similar premise. You can have one datastore cluster of many datastores, or several datastores clusters each containing fewer datastores. Taken in isolation, host clusters and datastore clusters each present a complex set of design decisions. But the two-dimensional aspect of matching datastore clusters to host clusters can make the design exponentially more convoluted. For example, it is entirely feasible to have one host cluster connected to multiple datastore clusters. Alternatively, many host clusters could be attached to only one datastore cluster.

Add in the complexity of multiple arrays: potentially one datastore cluster backed by multiple storage arrays, or conversely a single array supporting multiple datastore clusters. Also, consider the conundrum that datastore clusters aggregate datastores, so looking another layer down,

for each datastore cluster do you have a few large datastores or many more datastores that are smaller in size? Where do vDS boundaries align to each of these? Clearly, there are so many factors to consider such as the size of the VMs (storage capacity, storage performance, vCPU, vRAM, vNIC connectivity), the storage arrays (performance, capacity, protocols, functionality such as VAAI primitives, connectivity), and hosts (CPUs, cores, RAM, storage and network connectivity) that each design will need very careful analysis; there is never one design that will fit all situations.

So how do we align datastore and host clusters? There are two crucial aspects to consider which help clarify these choices. There are multiple layers spanning out from the VMs. Each VM needs host resources (vCPUs and vRAM) and storage resources (VMDKs). From the VM, they must run on a host which in turn runs in a host cluster which is contained in a datastore object. That VM's VMDK disks are stored in a datastore, which can be part of a datastore cluster which is contained within the same datastore object. So to understand the datastore and host cluster requirements it is critical to look carefully at the VMs in the context of the datacenter object. Those are the two foundational elements that will have the strongest influence. A holistic overview of all the VMs in a datacenter will drive the architecture of the interim layers. From this point the datastore clusters and host clusters can be designed on their own merit. But remember that the most efficient solution is likely to be one that aligns both cluster types, whether that is one host cluster and one datastore cluster in the datacenter, or aligned multiple clusters thereof, if cluster scaling become an issue.

There are hard limits on these logical constructs that will shape and potentially restrict the optimal configuration. For example, in vSphere 5.1 there is a maximum of 32 datastores in each datastore cluster, a maximum of 32 hosts per host cluster, and no more than 64 hosts to each datastore. There are also numerous limits to the VMs and their components against host and storage resources.

If there are overriding reasons to segregate VMs into one or both cluster types, then maximal cross-connectivity will lessen any design restrictions. For example, if you decide you need two host clusters (one full of scaled up hosts, the other scaled out), and three datastore clusters (one from your legacy SAN, one from fast SSD datastores, and one from the SATA pool), then try to have all hosts from both host clusters connected to all datastores in all datastore clusters. This provides the most technically efficient scenario for DRS, Storage DRS, HA, DPM, etc; considering the cluster divisions you mandated.

vSphere Replication

VMware's Site Recovery Manager (SRM) version 5.0 that was released in 2011 introduced in-built replication to asynchronously copy VMs to a recovery site. This removed the dependence on storage array replication that had been a prerequisite and allowed the hosts to handle the disk copying.

vSphere 5.1 includes this replication as a native feature without the need for SRM. This allows the hosts to provide basic failover protection for VMs without any inherent features in the storage arrays. This provides a basic but cost effective method to copy VMs offsite for DR purposes without involving complex matching of arrays across sites (or intra-site). vSphere replication is included with Essentials Plus and above licenses, and so provides a limited solution that even SMB customers can utilize. vSphere replication doesn't include SRM capabilities like automation, orchestration, multi-VM recovery, reporting, and so on.

The vSphere replication is configured on a per VM basis, and includes Windows guest OS and application quiescing via the VM's VMware tools support for Microsoft VSS. The replication

can take place between any type of vSphere supported storage; VMFS, NFS, local storage, except Physical RDMs. Unlike most array based storage replication, there are no requirements to change the storage layout or configuration. Enabling this on each VM is non-disruptive and only delta disk changes are transferred. Because it uses a special vSCSI filter agent it doesn't prevent replicating VMs with snapshots or interfere with VADP CBT backup-type applications. Any snapshotted disks are committed on the destination so no rollback is possible. Only powered-on VMs get replicated and FT and linked clone VMs can't be protected. VMs must be at least hardware version 7.

Although this feature was made available with vCenter 5.1, the replication agent required has been included since ESXi 5.0, so any vSphere 5 hosts are ready. The minimum RPO time possible for each VM is 15 minutes depending on the bandwidth available and the rate of data change. Although multiple sites can be protected, each vCenter can only have one vSphere replication appliance which limits you to one recovery site per vCenter instance. A maximum of 500 VMs can be protected in this way. vSphere replication is compatible with vMotion, HA, DRS, and DPM, but not with Storage vMotion or Storage DRS.

Summary

Now that you understand the elements that make up the storage landscape, your design should consider all four primary factors (availability, performance, capacity, and cost) and reflect the importance of each.

Availability is likely to be very important to the solution unless you're designing storage for a noncritical element such as a test lab. Even what may be considered secondary nodes, such as DR sites, need appropriate redundancy for high availability.

Performance is probably the key to any good storage design these days (high availability is almost taken as a given and doesn't need as much consideration—just do it). With advances in storage devices, it's easy to pack a lot of data onto a relatively small number of disks. You must decide how many IOPS your VMs need (and will need going forward) and use that number to design the solution. You can rely on spindles or consider some of the vendor's new technologies to ensure that performance will meet your requirements. Centralized company datacenters, headquarter buildings, and anywhere with large VDI implementations or intensive database servers will doubtlessly be avid consumers of these performance enhancements.

Capacity must always be considered, so the ability to assimilate your data requirements and understand future growth is very important. An appreciation of this will guide you; but unlike with performance, which can be difficult to upgrade, your design should include strategies to evolve capacity with the business's needs. Disks grow in size and drop in price constantly, so there is scope to take advantage of the improvements over time and not overestimate growth. Capacity can be very important—for example, smaller offices and remote branch offices may consider capacity a crucial element, even more important than performance, if they're only driving large file servers.

Cost will always dictate what you can do. Your budget may not be just for storage, in which case you have to balance it against the need for compute power, licensing, networking infrastructure, and so on. Most likely the funds are nonnegotiable, and you must equate the factors and decide what the best choices are. You may have no budget at all and be looking to optimize an existing solution or to design something from hand-me-downs. Remember in-place upgrades and buy-back deals. There is always a chance to do more for less.

In addition to the fundamentals, other design aspects are worth considering. For example, before you purchase your next SAN, you may ask yourself these questions:

- How easy is this solution to roll out to a site? How easy is it to configure remotely? Is any of it scriptable?
- Physically, how large is it? Do you have the space, the HVAC, the power, the UPS, and so on?
- How is it managed? What are the command-line and GUI tools like? Can multiple arrays be managed together, and managed with policies? How granular is the security?
- What are the reporting features like?
- How easy is it to troubleshoot, upgrade firmware/OS, and add extra disk enclosures?
- Is there any vCenter integration? Are there any special plug-ins?

Extra array functionality may be required, but that's out of scope of this chapter. For example, things like SAN replication and LUN snapshots can play a part in other designs such as backups, DR, application tiering, and so on. Every situation is different.

Planning for the future is normally part of an overall design: you must prepare for how storage will grow with the business. Think about how modular the components are (controllers, cache, and so on), what the warranty covers, how long it lasts, and what support levels are available.

Incorporate the new storage functionalities in vSphere 5 to take advantage of hardware offloading as much as possible. Use the enhanced manageability through datastore clusters with their Storage DRS automation, and make smarter administrative choices with profile-driven storage policies and compliance checking.

Finally, take as much time as possible to pilot gear from different vendors and try all their wares. Use the equipment for a proof of concept, and test each part of your design: protocols, disks, RAID groups, tiering, and so forth. You may be able to clone all or at least some of your production VMs and drop them onto the arrays. What better way to validate your design?

Chapter 7

Virtual Machines

Virtual machines (VMs) are central to any vSphere design. After all, isn't that why we try so hard to optimize all the other pieces of the puzzle? Many organizations spend considerable time and resources ensuring that the network, servers, and storage are suitably redundant, efficient, and capacious. However, often the design of the VMs is paid lip service. You can gain tremendous benefits by giving a little thought to how the VMs are designed and configured.

This chapter will explore what makes up each VM, to help you understand how to take advantage of the different options and realize the impact of these decisions on the rest of your vSphere design. The guest operating system (OS) within the VM can also affect overall performance, along with how each instance is deployed and managed. Interesting techniques exist to minimize the management required and improve standardization in an environment. Finally, we'll look at various strategies you can use to mitigate the effects of host failures on VMs.

VM design is a balancing act between ensuring that each VM has the resources and performance characteristics it needs, and preventing waste. If a VM is overprovisioned in some way, it's unlikely to benefit and will eventually penalize the VMs around it.

Specifically, this chapter looks at the following:

- Hardware components, options, and resource allocation for each VM, with emphasis on the network and storage
- How the guest OS and applications affect VMs
- Using clones and templates to more efficiently deploy VMs
- How to protect the VM's availability
- Understanding VM interrelationships with VMware's Infrastructure Navigator tool

Components of a Virtual Machine

A VM is a construct of virtual hardware, presented to the guest OS. The guest sees the hardware as if it were a regular physical computer. For all intents and purposes, the guest OS is unaware that it's potentially sharing the hardware with other VMs.

Virtual Machine Interface (VMI)

Previously, some Linux guests could take advantage of the Virtual Machine Interface (VMI) paravirtualized feature, which allowed certain guests to be aware of its virtualized status. The feature has been retired and is no longer available to VMs in vSphere 5.

VMware presents very generic hardware to the guest, allowing the greatest compatibility for the widest range of OSes. Most modern OSes can detect and run on a vSphere VM without the installation of extra drivers. When you're creating a new VM, if the OS is listed in the wizard as a supported guest OS, then you can change the base hardware to be more appropriate. Different hardware options are available for some items, and VMware provides OS-specific drivers where appropriate. VMware has additional optimized drivers that can improve on the more generic ones found in the OSes.

vSphere 5.0's new Web Client GUI interface exposed the ability to create and reconfigure VMs. The 5.1 version of the Web Client added additional VM functionality, particularly in the new *Manage* tab where Alarms, Tags, Permissions, Storage Profiles, Scheduled Tasks, and vServices can be set. The Summary tab in the new Web Client has also been redesigned (see Figure 7.1). Although the Windows-only client remains and is fully supported with vSphere 5, from 5.1 all new functionality is being added only to the Web Client. For example, the ability to create VMs *compatible with vSphere 5.1 and above* (VM hardware version 9) can only be completed in the Web Client.

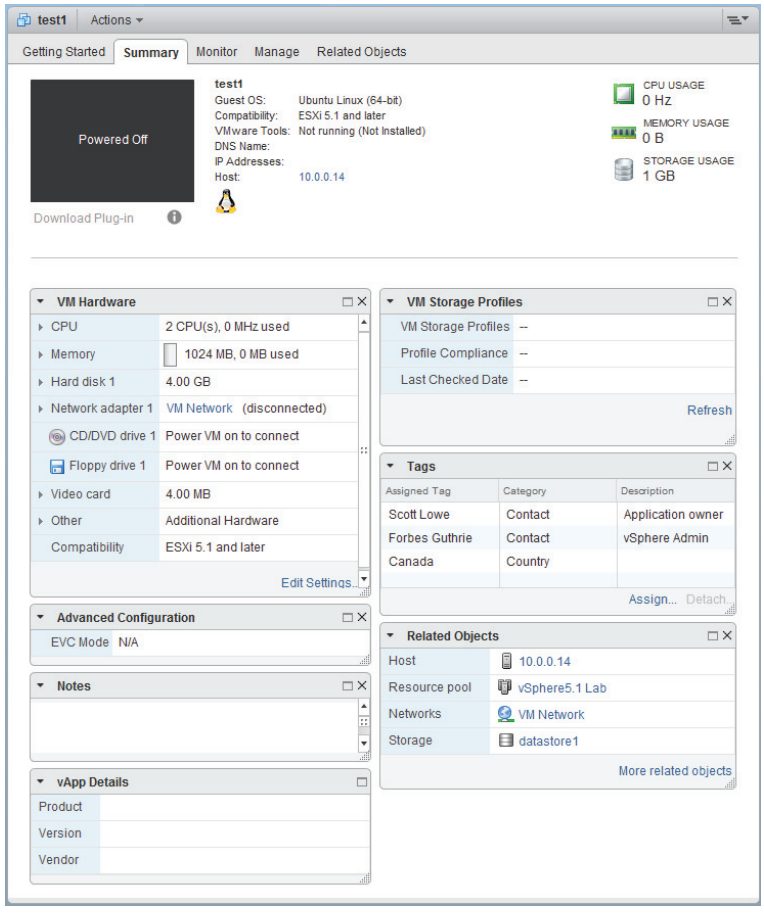


FIGURE 7.1
VM Summary tab in the vSphere 5.1 Web Client

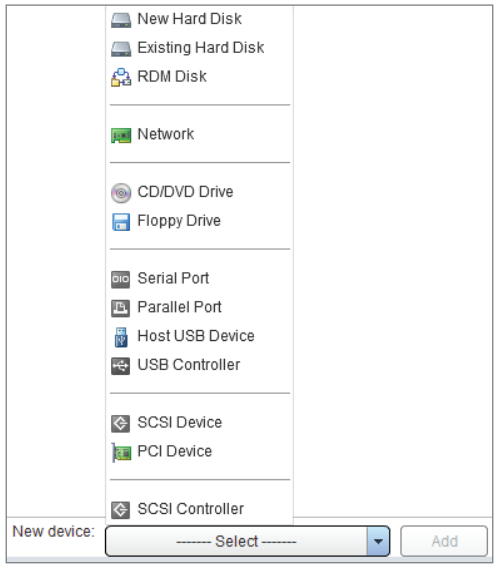
Base Virtual Machine Hardware

As a base, all vSphere 5 VMs are created with the following hardware, regardless of the OS chosen, the underlying hardware, or any options you select in the New Virtual Machine wizard:

- Phoenix BIOS or EFI-based firmware
- Intel 440BX-based motherboard
- Intel PCI IDE controller
- IDE CD-ROM drive
- IDE floppy drive
- SVGA video adapter

In addition, CPUs and RAM are always added. However, they're limited to the underlying hardware: you can't allocate more CPUs or memory than the physical host has installed. The number of CPUs and the amount of RAM are allocated by default based on the guest chosen; you can manually adjust them later in the creation wizard. Other hardware options are available, either during the VM's initial creation or as additions later as shown in Figure 7.2.

FIGURE 7.2
Adding devices to a VM



Hardware Versions

As VMware's hypervisors have evolved, so has the VM hardware shell that is presented to guests. The hardware versioning basically determines what functionality the hypervisor should expose to the VM. In vSphere 5.0 the native hardware is version 8, and in 5.1 this goes up to version 9.

Although vSphere 5.0 can happily run versions 4, 7, and 8 VMs alongside each other, older vSphere ESXi hosts can't run VM hardware meant for newer hosts. It's relatively straightforward to upgrade VMs from one version, but you should ensure that the VMware Tools are upgraded first so drivers capable of understanding the new hardware are in place. After you've upgraded the tools, power off the VM; then you can upgrade the hardware to the latest version. vSphere 5, in particular 5.1, has relaxed the hard requirement to upgrade tools first, before the virtual hardware. As long as a recent version (tools from vSphere 4 or 5) is installed, then upgrading to the hardware compatibility version that comes native in 5.1 should be successful. However, upgrading hardware always requires a shutdown, so this is a great opportunity to play things safe and upgrade the VMware Tools.

vSphere 5.0 can create hardware version 4 VMs, which was the default for VI3 (ESX/ESXi 3.x), and hardware version 7, which was the default in vSphere 4, in addition to its native version 8. The ability to create older versions of VM hardware aids backward compatibility when you have a mixed environment containing legacy hosts, common during an upgrade. Also, VMs from third-party software vendors tend to come in older versions, such as 4 or 7 packaging, to maintain the greatest compatibility possible for customers.

Missing Hardware Versions?

In case you're wondering where hardware versions 5 and 6 went, VMware shares the VM hardware versioning with its hosted hypervisor products (Workstation, Fusion, Server, and Player). If you're curious, this is where versions 5 and 6 fitted in. After hardware version 4, which was used by ACE 2.x, ESX 3.x, Fusion 1.x, Player 2.x, Server 1.x and Workstation 4/5, VMware choose hardware version 6 for its release of Workstation 6.0. Hardware version 5 was skipped entirely. The remaining products including ESX/ESXi rejoined at version 7.

vSphere 5.1 has taken a new approach to VM hardware and has changed the terminology. VMware wanted to reduce the upgrade burden and the perception that upgrading virtual hardware was a necessity after each host upgrade. This particularly helps software vendors that produce prepackaged appliances, which don't necessarily need to take advantage of later features. The new monikers for each hardware version are listed in Table 7.1.

Table 7.1: VM hardware compatibility

vSphere 5.1 Compatibility Description	Hardware Version
ESXi 5.1 and later	9
ESXi 5.0 and later	8
ESX/ESXi 4.x and later	7
ESX/ESXi 3.x and later	4

Although VMs can be upgraded, downgrading them is considerably trickier. One approach is to snapshot each VM before you upgrade it. Doing so provides a temporary roll-back point;

however, it isn't feasible to keep these snapshots long-term, and reverting changes to get back to a previous hardware version also destroys any changed data. Snapshots can only be used to guard against problems during the upgrade itself.

VMware Converter is a tool primarily aimed at virtualizing physical machines in a process most commonly known as *P2Ving* (physical to virtual). However, Converter can also be used to downgrade to earlier versions. It's a freely downloadable tool, and it works seamlessly with vCenter.

During a host upgrade project, all the emphasis is placed on the hosts. Often the VMs are missed, because upgrading isn't a definitive requirement, and this is the one step that in a multihost, shared-storage environment actually needs VM downtime. Upgrading the VMs not only brings proper host/VM compatibility but also introduces a number of improved features. Some recent hardware benefits include access to larger hardware maximums, new storage devices and new network-card options, hot-plug support for CPUs and memory, passthrough to Peripheral Component Interconnect (PCI) devices, access to hardware-accelerated 3D graphics cards, and virtual CPU performance counters. An upgrade project should not only feature upgrading the VMs but also take advantage of the devices.

The best time to upgrade the VMs is when all the hosts in a cluster have been upgraded. If some VMs are upgraded before all the hosts are, then this can affect vMotion choices, distributed resource scheduling (DRS), high availability (HA), and distributed power management (DPM) effectiveness.

It's important to remember to convert the templates to a VM and upgrade them at this time as well, to ensure that all new VMs are deployed at the latest version. Be careful if you maintain one set of templates for several clusters and sites, because you'll need to keep two versions until all the hosts are upgraded.

Virtual Machine Maximums

The VM hardware version dictates the features available to, and the scalability of, a VM. Ultimately, you can only add hardware to a VM if it's available on the host: for example, you can't add more virtual CPUs (vCPUs) to a single VM than there are logical processors in the host server. Table 7.2 shows the maximum amount of each hardware component that you can add to a VM at each of the recent versions.

Hardware Choices

Once a VM has been created, you can alter the default hardware. The following sections discuss each element and the options available to customize the VM. Figure 7.3 shows the basic VM hardware choices available.

CPU

Each VM is created with the minimum number of vCPUs that the selected guest OS can support (usually 1, but you'll notice, for example, if you create a nested ESXi VM that the creation wizard will automatically select 2). The number of vCPUs can be increased up to 64 in vSphere 5.1 (32 in vSphere 5.0), or the maximum number of logical processors that the server hardware contains, whichever is lower. Many other vCPU options exist and can be configured via the drop-down menu. These extra settings and their impact on the VM's design are discussed later in the chapter in the section "Virtual Machine CPU Design."

TABLE 7.2: Virtual machine hardware maximums

HARDWARE COMPONENT	ESX/ESXI 4.X	ESXI 5.0	ESXI 5.1
VM hardware version	7	8	9
vCPU	8	32	64
RAM	255 GB	1011 GB	1011 GB
SCSI adapters	4	4	4
SCSI devices	60 (15 per adapter)	60 (15 per adapter)	60 (15 per adapter)
IDE devices (hard disks or CD-ROMs)	4	4	4
Network adapters	10	10	10
Video memory	128 MB	128 MB	512 MB
Video 3D acceleration	No	Software only	Hardware and software

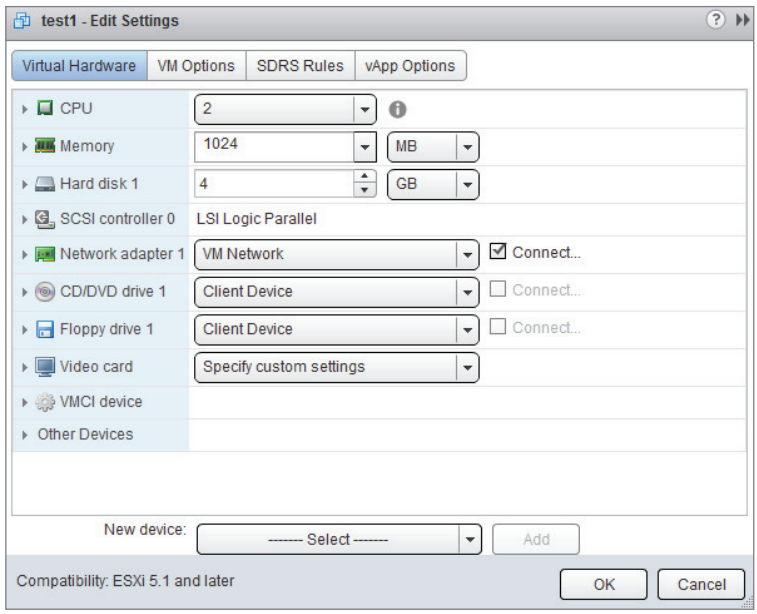


FIGURE 7.3
Virtual machine hardware

Memory

In a parallel to a VM's CPUs, a base minimum RAM is allocated to a VM according to the recognized guest OS's preconfigured safe minimum. Again, more can be added, up to the amount fitted to the physical server or the vSphere 5 1TB limit (1011 GB to be exact). If the compatibility level on the VM (that is, the VM hardware level) has been set to 4.x or later, then this is limited to 255 GB. VM compatibility of ESXi 3.x or later reduces this to just below 64 GB. The advanced memory options configurable are discussed in the section "Virtual Machine Memory Design" later in this chapter.

Disks

In a VM's settings, the primary disk option is to increase its size via a spinner control and drop-down menu. Despite the presence of the spinner, you can't decrease the size of a disk once the VM has been started. During the VM's creation, and when adding new disks, options are available to select an existing disk instead of a newly forged one. Additionally, instead of creating standard Virtual Machine Disk Format (VMDK) disks, you can create raw device mapping (RDM) files that map directly to a storage area network (SAN) logical unit number (LUN). These options and more will be discussed in further depth in the section "Virtual Machine Storage Design" later in this chapter. It's worth noting at this stage, though, that each VM is limited to 4 IDE disks and potentially a total of 60 SCSI disks.

SCSI Controllers

When a SCSI hard disk is attached to a VM, a new SCSI controller is also added. Each SCSI adapter can have 15 disks connected to it, and you can add up to 4 SCSI controllers.

A VM can select from four different types of SCSI controller, which we'll discuss later in the section "Virtual Machine Storage Design." They're added automatically to best suit the guest OS; you have to manually change them if you have different needs.

Network Adapter

The VM network adapter settings allow you to change several hardware options; but from a VM design perspective, they let you select different adapter types and manually set MAC addresses. Both of these options are discussed later, in the "Virtual Machine Network Design" section. The base configuration merely allows you to select the designated port group (subnet) and whether the adapter is in a connected or disconnected state. You can add up to 10 network cards to a VM, each with its own individual settings.

CD/DVD Drive

The CD/DVD drive allows you to connect the VM to your client workstation's local drive (using the Passthrough IDE mode), the host's optical drive (with the Emulate IDE mode), or an ISO image on a datastore. Figure 7.4 displays the typical options available.

If you're attaching a host USB optical drive, then it must be attached as a SCSI device that doesn't support hot adding or removing as an IDE drive does. Remember that attaching a host CD/DVD drive to a VM should only be a temporary action to install software or copy data into the guest. If possible, disconnect the drive after it has served its purpose: leaving it connected will prevent vMotions, which in turn will affect automated operations such as DRS balancing.

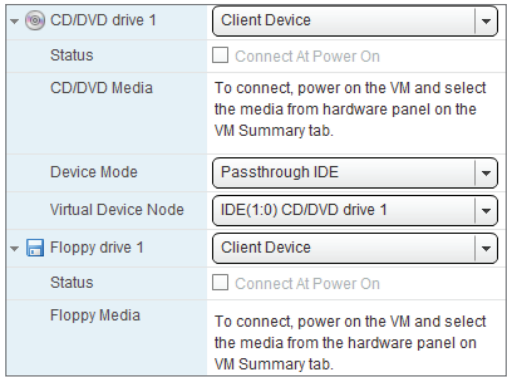


FIGURE 7.4
CD/DVD drive hardware options

FLOPPY DRIVE

The floppy drive is treated very similarly to an optical drive, although the settings also let you create new `.flp` images. You can have a maximum of two floppy drives per VM. Floppies can be either image files or drives connected to the client computer. Physical floppy drives on the host can't be passed through to a VM.

VIDEO CARD

You can't add a video card to or remove it from a VM; it comes as standard. Figure 7.5 shows how you can adjust the number of displays from one up to a maximum of four, allocate more video memory, and enable 3D support.

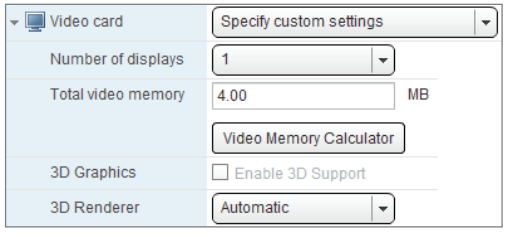


FIGURE 7.5
Video card hardware options

The default memory is 4 MB, which is enough for one screen with a resolution of 1176 × 885. Increase the memory if you need more screens or a higher resolution. This setting is most important for virtual desktop infrastructure (VDI) designs.

vSphere 5.0 included support for a software emulation of a 3D graphics card in VMs. This, in combination with the Windows Display Driver Model (WDDM) guest driver, is capable of driving Windows' Aero-style GUI elements. Basic support for OpenGL 2.0 features is available; depending on the specific applications and use cases, this may be sufficient to provide users with appropriate 3D capabilities. vSphere 5.0 is also capable of supporting PCI passthrough, so it's theoretically possible to install a card in the server, pass it directly through to a single VM,

install the necessary drivers in the guest, and provide hardware 3D graphics support this way. However, the one-to-one mapping of card to VM means this solution isn't scalable for a VDI environment and is unlikely to be useful in anything more than edge-case scenarios.

vSphere 5.1 introduced support in desktops for one-to-many hardware graphics. Certain NVIDIA multi-GB video cards can be presented through to VM, allowing you to allocate a slice of the video card's memory to each VM.

VMCI Device

VM Communication Interface (VMCI) was a communication method that could be used between VMs on a single host or between a VM and the host itself. VMCI was introduced in vSphere 4. It aimed to minimize the overhead associated with traditional networking stacks. Because it didn't use the guest or VMkernel networking stack, it had relatively high performance compared to TCP/IP sockets.

Applications needed to be written specifically to use VMCI sockets, and drivers were included in the Windows and Linux versions of VMware Tools.

VMCI guest-to-guest support has since been retired with vSphere 5.1. It can still be enabled in 5.0, but in the 5.1 GUI even this option has been disabled. It remains available on VMs already configured for VMCI, but it can no longer be enabled on 5.1 hosts.

Additional Devices

A number of additional devices can be added to a VM, even though they don't come as standard. Figure 7.2 showed the full listing. Augmenting the devices already discussed, the following ancillaries are also available:

Serial Port and Parallel Port You can connect both serial ports (COM) and parallel ports (LPT) to a VM. They can be useful when an application requires either type of port for licensing purposes, like an old software dongle, or if there is a requirement to support old hardware, such as an old facsimile modem. Often a better alternative is to attach these devices via a special Ethernet adapter, thus avoiding this hardware condition.

Figure 7.6 shows how a VM's serial/parallel port can pass through to the host's hardware, output the data to a file on a host's datastore, or present it as a named pipe to the guest OS. You can also redirect the serial port over a network link via telnet or SSH, which enables the use of third-party serial concentrators.

Each VM can have up to three parallel ports and four serial ports assigned.

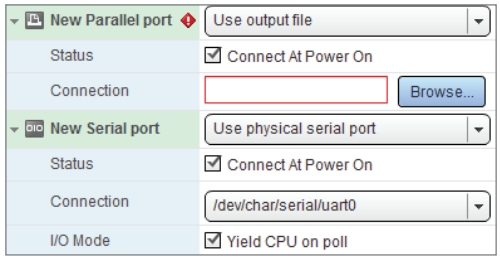


Figure 7.6
Serial port hardware options

vSphere 4.1 was the first release to encompass USB support into the VMs. The initial USB implementation relied on USB devices attached to the ESX/ESXi hosts. vSphere 5.0 brought support for client-connected devices and USB 3.0.

USB Controller To connect a host or a client-attached USB device, the VM must first have a USB controller device attached to it. Two types of controllers exist:

- The EHCI+UHCI controller is the standard USB 1.1/2.0 device controller and is available to VMs from hardware version 7 and above.
- The xHCI controller is a new USB 3.0 controller that needs at least VM hardware version 8. Using this controller requires that an xHCI driver be installed in the guest OS; currently only Linux drivers exist. Windows 8 and Server 2012 are likely to include suitable drivers on their release.

A VM can use only one type of controller, so you must choose which of the two will connect all the potential USB devices. Each VM can have up to 20 USB devices attached to that single controller, but each device can be connected to only a single VM at a time. USB devices can't act as a VM's boot device.

Host-Connected USB Device The original host-connected USB option is a good fit for licensing or security dongles and permanently connected server devices. Each USB device is plugged into a server passed through to a specific VM. The server can host only 15 USB controllers and can't serve the newer xHCI-type controllers (USB 3.0).

To attach a device through to a VM, the VM must be registered on that host for the initial connection. At that time, you can explicitly enable vMotion, and then the VM can be vMotioned to another host while still attached to the USB device. By extension, DRS is supported, but neither DPM nor FT are compatible with USB devices. Hot-adding memory, CPUs, and PCI devices temporarily disconnects any USB device.

Client-Connected USB Device Client-connected USB device support works well in VDI environments and allows users to connect temporary devices such as mass-storage devices and user-specific devices like smart-card readers. This feature has been available since vSphere 5.0. It permits USB devices attached to a client's workstation to be connected to a VM. Connecting USB devices via their client requires at least vCenter 5.0 and a 5.0 client (Web or Windows client). You can successfully attach these devices to VMs running on ESX/ESXi 4.1 hosts, as long as the vCenter and client are at version 5.0.

vMotion is supported for all client-connected USB devices. There is no need to definitively enable this as is the case with host-connected devices.

PCI Device A VM can attach to a host PCI or PCIe device directly. To enable this passthrough, the host must first be enabled for DirectPath I/O; this requires a reboot and is only supported with Intel VT-d (Nehalem) or AMD IOMMU (experimental) capable host CPUs, which must be enabled in the BIOS.

Numerous limitations are imposed when you choose this as part of a design, because when a VM is configured with a passthrough PCI device, it's tied to the hardware: no FT, HA, hot-plugging, suspending, or record/replay operations are permitted. vMotion (and therefore DRS) has been supported since vSphere 4.1 with DirectPath I/O.

Each host can have four DirectPath I/O devices. However, a device that's configured for passthrough can't then be accessed by the VMkernel and subsequently used by other VMs.

PCI passthrough devices aren't often used, because users recall the poor I/O performance in previous ESX versions. The paravirtual SCSI (PVSCSI) storage adapters and VMXNET3 network adapters give VMs excellent near-native I/O results.

Considering the substantial feature restrictions when using this option, it's hard to recommend. Be cautious about including it in your design.

SCSI Device The option to add SCSI devices directly to a VM allows a passthrough of physical SCSI devices connected to the host server. For example, a SCSI-attached tape unit could be patched through to a backup application running in a VM. Just as adding PCI devices to a VM creates a number of limitations on your configuration, so does adding SCSI devices.

Removing or Disabling Unused Hardware

Each VM should be presented with only the hardware it requires. Like other vSphere resourcing, a VM should have what it needs but not waste host resources on what the VM doesn't require.

VM hardware that isn't needed can be removed, disconnected from the host hardware, disabled in the VM's BIOS, or even disabled in the guest OS. Typically, floppy and CD drives, USB controllers, LPT and COM ports, and unused NICs and storage controllers are likely candidates.

A guest floppy drive is rarely used in a VM. The only common use case is the presentation of driver FLP files during OS installations. You should be safe removing this device from your VMs.

Excess hardware unnecessarily uses interrupt resources. OSes poll devices on a regular basis, which requires CPU cycles to monitor. Other devices reserve memory that could be used by other VMs. Even excessive vCPUs use more interrupts than uniprocessor VMs. You can tweak the number of timer interrupts in Linux VMs; however, most modern Linux kernels use a *tickless timer*, which varies the timer interrupt rate to reduce the number of wake-ups (introduced in the mainline 2.6.21 kernel). Older Linux VMs may benefit from a reduction in their timer interrupt settings.

You can disconnect optical drives and COM and LPT ports from the VM, or at least connect them to files instead of physical devices, when they aren't being used. Again, COM and LPT port are rarely used, so you should consider removing them from a VM altogether.

Some hardware can also restrict other features. For example, FT won't work while serial or parallel ports are connected or while CDs, floppies, USB passthrough devices, Fibre Channel N-Port ID virtualization (NPIV) ports, or any hot-plug features are enabled. If you don't need the hardware, disable or disconnect it. If you'll never need it, remove it.

Later in the chapter, in the section for "Clones, Templates, and vApps," we discuss how removing unnecessary hardware and right-sizing the images with a minimal approach is a great practice to encompass. Eliminating unnecessary resources from each VM at the time of its inception will significantly reduce overhead across the future infrastructure.

Virtual Machine Options

In addition to the configurable hardware options, there are more option choices for each VM. The VM options tab in the vSphere Web Client is split into the following drop-down sections.

General Options

The General Options displayed in Figure 7.7 provide basic information about the VM such as its name, the location of its configuration file (the `.vmx` file), and which guest OS it's configured for. The location of the VM's working directory is by default alongside the configuration file. In versions of vSphere prior to 5.0, the working directory stored all of its disks' snapshot files. This is no longer the case, because each disk's snapshot is now stored alongside the parent disk, but the working directory still contains several other volatile files such as the VM's suspend file and swap file by default. You may wish to change the working directory for a VM's design, so you can dictate where these variable files reside.

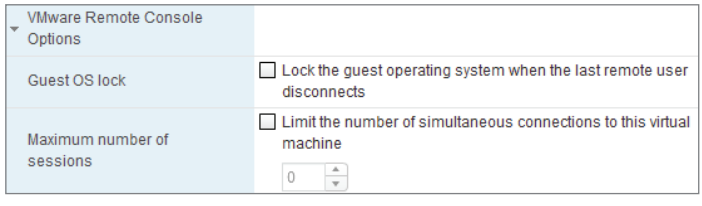
Figure 7.7
General options

The guest OS and version are set when the VM is created. Many of the default choices about the hardware and its setting are derived from this setting. If you run the guest OS through an in-place upgrade, remember to power off the VM and change this setting accordingly. This won't change the preconfigured hardware, but it ensures that any extra hardware added afterward will default to the most appropriate choice.

Remote Console Options

Two remote console settings can be configured, as shown in Figure 7.8. The first check box ensures that the console is locked when no users remain connected. The second option allows you to limit the number of remote consoles with an enabling check box and then set the number of users.

Figure 7.8
Remote Console Options

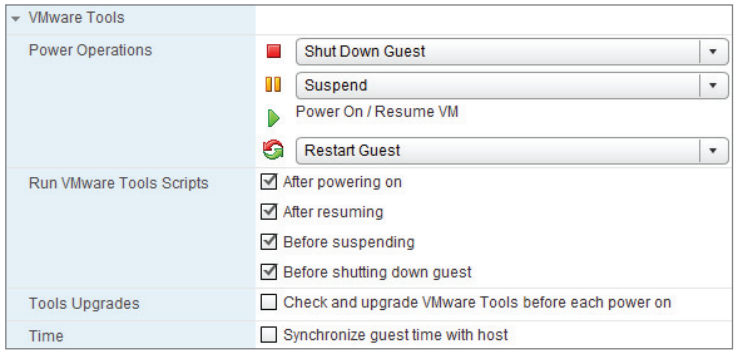


Both options are disabled by default, but enabling them is a sensible security measure if access to the guest OS, the data within, or the guest's ability to act as a springboard to other targets is of a particular concern. Remember, even without these settings, a user still must explicitly have at least VM User permissions on the VM, or an object hierarchically above it, to access the console.

VMware Tools

The various tool settings shown in Figure 7.9 determine how the power buttons should react and allow scripts to run inside the guest OS. Usually the default options are suitable. If the guest is supported and has VMware Tools installed, then the power option defaults resolve to "soft" operations.

Figure 7.9
VMware Tools options



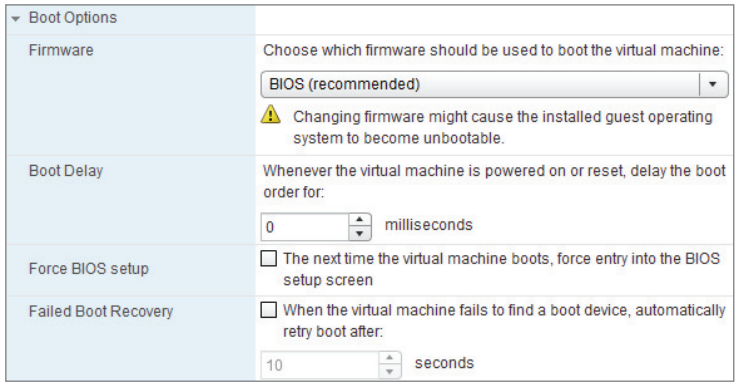
Power Management

Power management corresponds to how the VM reacts when the guest OS is put in standby. You can leave the VM turned on (with the Wake-On-LAN option) or suspend the VM. This design decision is determined by balancing how long you're willing to wait for the VM to wake up, against saving host resources by suspending it.

Boot Options

The Boot Options shown in Figure 7.10 give you control over the BIOS delay and let you access the boot order. These are normally changed only for a specific event; but for VM design, they can be important if you wish to disable options in the BIOS. vSphere 4.1 added the ability to automatically reboot the VM if no boot device is found. vSphere 5.0 brought the option to use an EFI firmware interface in VMs. This is defined when the VM is created based on best fit for the guest OS. Once the guest OS is installed, switching between the two will usually result in an unbootable OS.

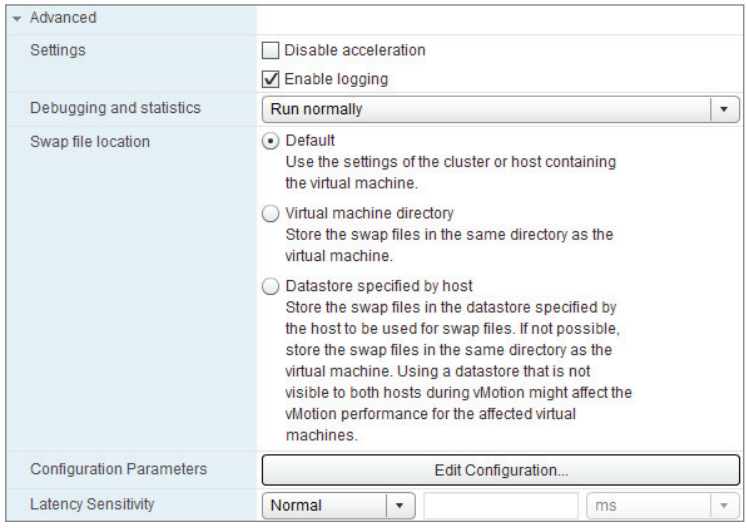
Figure 7.10
Boot Options



Advanced

On the VM Options tab, you can change a number of advanced options shown in Figure 7.11.

Figure 7.11
Advanced options



Settings Under Advanced Settings, you can choose to disable acceleration and enable logging. Ordinarily, these options are used only in remedial circumstances, when you're experiencing an issue with a VM.

Debugging and Statistics Again, this option is only used when you're troubleshooting a particular problem and VMware's technical support has asked for additional detail so they can investigate.

Swap File Location Each VM has a swapfile (in addition to the guest's swap/pagefile settings) that the host can forcibly push memory pages to if it has to. The swapfile by default is stored in the same datastore folder as the VM's configuration file. However, this can be overridden by a host or cluster setting that determines the VM's default. You can override these host and cluster defaults in this VM option, setting to store along with the VM or to store on the host's chosen location.

Storing a swapfile on the host's datastore has the obvious advantage of saving valuable SAN input/output operations per second (IOPS) and avoiding SAN replication of transient data. However, before you decide to move all VM swapfiles off shared storage, be aware that negative effects are associated with this choice. Enough local datastore space must exist to accommodate the swapfiles. A VM's swapfile is equal to its configured RAM minus any memory reservation. But when a host is a member of a DRS or an HA cluster, the total space required by a host is unpredictable while VMs move around. vMotions slow down significantly, because the swapfile must be copied from local disk to local disk before the transfer is complete. In most situations, it's undesirable to save swapfiles on local storage, due to the way this can affect DRS effectiveness and the fact that HA's ability to power-on the VMs may be compromised. If SAN performance or replication traffic is a concern, then a separate nonreplicated, lower-tier SAN LUN just for swapfiles is often a better solution.

Configuration Parameters The Configuration Parameters button lets you add extra settings, which are inserted into the VM's configuration file. Nothing here is part of a standard VM design. Nonstandard parameters may be requested by VMware's technical support or a VMware Knowledge Base (KB) article to resolve a known issue.

Latency Sensitivity The Latency Sensitivity setting, which can be set to Low, Normal, Medium, High, or Custom, is a preference setting for the CPU scheduler. The Custom setting allows you to define a level based on milliseconds. This latency setting attempts to prioritize VMs on the CPU scheduler so that VMs that are particularly sensitive to any latency can be prioritized and, ideally, have latency reduced. It specifies the scheduler's delay tolerance per VM. This setting doesn't provide any guarantee, as a reservation would, but instead acts in a way similar to a resource share.

Fibre Channel NPIV If a host server is Fibre Channel (FC) attached with an adapter that supports NPIV, this option can set the VM with its own World Wide Name (WWN). Using NPIV can arguably provide greater control over a SAN LUN's usage. It can allow a SAN administrator to monitor a VM's traffic more closely, tie specific security zoning around it, and configure special quality of service (QoS) for it. To use this, the VM must have a RDM disk already added.

SDRS Rules

A VM's settings dialog has a third tab, SDRS Rules, which permits the addition, editing, and deletion of Storage DRS rules. Storage DRS was discussed extensively in Chapter 6, "Storage," but in summary it allows rules to affect how a VM's disks should react when they're stored in a datastore cluster. When Storage DRS is enabled, it can recommend or automatically move a VM's disk to balance capacity and/or performance across the datastores. The SDRS rules dictate the affinity of VMDK disks and VMs in the datastore clusters.

vApp Options

The vApp options let you set or change virtual appliance settings such as product descriptions, IP allocations, and Open VM Format (OVF) environmentals. It's unlikely that you'll need to modify these unless you plan to distribute a VM as a virtual appliance. If the options here aren't sufficient for your packaging needs, VMware Studio is a freely downloadable tool that is designed specifically to package vApps.

vServices

A VM's vServices options show which appliance applications have been registered in vCenter. Administrators wouldn't typically set these themselves, but they can be delivered as part of an application or a vApp package. For example, vSphere Update Manager (VUM) registers itself with the vServices tool, alerting users to problems with the VUM service centrally from within vCenter. An application needs to register itself via a vCenter extension to be monitored this way. vServices can be useful not only in monitoring, but also in basic troubleshooting and checking on service dependencies.

Naming Virtual Machines

A VM's name is chosen during its creation or deployment from a template and sets the default moniker for its location folder and files. This name is transparent to the guest, but matching it

to the VM's hostname avoids confusion. If the VM name is changed, the VM retains the original naming against its files on the datastore. However, migrating to another host via cold migration or Storage vMotion renames these files (although not during a regular vMotion). This helps ensure consistency; otherwise, troubleshooting issues can become more complicated if the names no longer match.

It's always advisable to avoid spaces and special characters in a VM's name. This is also beneficial because it makes working with the VM at the command line or with scripts much easier. Additionally, keeping all the letters lowercase makes for a more palatable keyboard experience.

VMware Tools

The VMware Tools are a software package that you should install in each VM. The tools include optimized drivers, a tools service, a tools control panel applet, and a specialized memory driver. These tools improve the VM's performance and aid management. They facilitate the VM heartbeat, which hosts use to tell that the VM is responding. They can also enable time synchronization between the host and the VM, which we'll examine later in the chapter. You can use the tools to quiesce the file system, copy and paste operations to the console, and send shutdown commands.

All versions of VMware Tools included since vSphere 4.0 are supported in VMs running on vSphere 5. Upgrading the tools to the latest version isn't mandatory, but it's advisable because newer versions will include support for more recent versions of the VM hardware level. A new feature of the VMware Tools packaged in vSphere 5.1 is that once installed, subsequent updates to Windows guests (Vista and later) shouldn't require a reboot in the majority of cases. The VMware Tools update process can be run without causing disruption.

Your VM design should include the installation of the VMware Tools, because they provide important functionality. Every supported guest OS has a set of tools available.

Notes, Custom Attributes, and Tagging

vCenter has always provided a way to store user-generated metadata about VMs. The *notes* field was the primary mechanism for VM object descriptions. It was an open text area to record anything useful about the VM in free-form. Some third-party tools like backup applications had a nasty habit of hijacking this field and overwriting it with their own content. Better data structures could be created by using vCenter *custom attributes*, which allowed multiple entries per VM and a descriptive name for each field. However, custom attributes were never particularly discoverable, so even once they had been created by one user, other users need to know about their existence and manually reveal the columns to see the stored attributes.

vCenter 5.1 introduces the concept of *tagging* for a wide variety of object levels. Nowhere are they more useful than at the VM level. vCenter tags are similar to custom attributes but infinitely more visible and searchable, and they make grouping VMs a breeze. Tags are so analogous to the custom attributes of yore that the upgrade process offers to convert them automatically. The basic premise remains: you create a category label that will be available to all instances of that object, and then you tag individual objects with a corresponding entry. The categories can be defined as allowing a single tag or multiple tags per object. For example, if you create a category titled Country, then each object can have only one tag associated with that category—that is, the VM can reside in only one country at a time. Alternatively, a category called Applications to record which applications are installed on each VM needs to allow multiple tags, because several applications may be installed on one VM.

Sizing Virtual Machines

Appropriate sizing is an important part of the overall VM design, because you need to give special consideration to each of the components we've just reviewed. As your vSphere environment grows, the initial sizing of VMs will become increasingly important. If you provide insufficient resources to users from the outset, this can leave a bad impression. Those first impressions of a virtualized alternative to the status quo can often be hard to overcome. However, wasteful allocation can grow to become a serious drain on your resources.

We already discussed disabling or removing unneeded hardware, but remember that there are real benefits to pruning your VMs. As a general rule, it's easier to add more hardware, capacity, or performance than it is to take it away. Often, resources can be hot-added to VMs, and OSes can normally cope with new hardware being presented. But removing hardware can be difficult—it nearly always requires at least a reboot, and often a complete rebuild is needed.

One of the difficulties with keeping VMs small is the attitude of software vendors and users. First, vendors regularly quote application requirements based on the minimum physical server available. Even though the application may be very small and have minimal CPU, memory, and disk requirements, the vendor will quote the latest quad-core CPU with 8 GB RAM and 146 GB mirrored disks. The other common error is that a physical server is always bought with the intention that initial scaling must provide all the performance and capacity the server will need for its entire lifetime. That's at least 3 years, and more normally 5 years. This doesn't take into account the dynamic growth that is possible with the abstracted vSphere layer that sits under each VM.

To best control VM sizing, you should start with sensible initial standards. A hardware tiering scheme is often helpful. For example, create a set size of VMs for small, medium, and large requirements, where each has a list of predefined hardware pieces. As each VM request comes through, fit the applications requirement to one of these standards. Always try to avoid pressure to unnecessarily bump up to the next level if you think it isn't warranted, but remain flexible enough to add extra resources as required. Chargeback, or information-only *showback* schemes, can be employed internally to help curb excessive business unit demands on new VMs.

To identify existing over- or under-provisioned VMs, several monitoring and capacity-management tools are available, such as VMware's own vCenter Operations Manager. Workloads change over time, applications are upgraded, and user bases fluctuate, so it's important to periodically seek out VMs that are starved or gluttonous. The hypervisor's primary purpose is to balance resources among VMs. If VMs aren't sized appropriately, resource-management techniques such as DRS initial placement, non-uniform memory architecture (NUMA) memory locality, Storage DRS disk-size balancing, and vCPU/vRAM shares won't be as effective or efficient.

Remember, one of the primary reasons for virtualization is to recoup the overprovisioning associated with nonvirtualized hardware. Don't just provide the same hardware in a VM that's present in the physical server you hope to replace. Design your vSphere servers and VMs to fit your purpose, so that all the workloads can function as required. Wasted resources in one VM will eventually hurt other VMs.

Virtual Machine CPU Design

Since vSphere 5.1, a VM can have from 1 to 64 vCPUs. The most vCPUs that can be allocated to a VM depends first on the number of logical cores the physical hardware has. This includes not

only the number of filled CPU sockets, but also cores and HyperThreading (HT) cores enabled. Second, the VM hardware version, or compatibility, affects how many vCPUs can be allocated. Hardware version 9 (ESXi 5.1 and later) allows up to 64, but version 8 (ESXi 5.0 and later) VMs can only have up to 32, and version 7 (ESX/ESXi 4.x and later) up to 8. Last, current vSphere licensing limits a VM's vCPUs to 8, unless the host has an Enterprise Plus license, which allows the full amount to be allocated.

Converting a VM from a single vCPU to multiple vCPUs requires that the guest OS be able to handle more than one CPU—this is known as a symmetric multiprocessor (SMP). Some older OSes may not be ready for this; for example, an old P2Ved Linux VM may need its kernel recompiled for SMP. Some OSes can convert their kernels to SMP but have difficulty dropping back to uniprocessor hardware; for example, Windows 2000 can drop back to a single CPU without issue, but converting a Windows 2003 VM requires manual intervention. Adding a second vCPU to a VM therefore shouldn't be taken lightly.

Generally, it's considered prudent to start all VMs with one vCPU until you have a clearly identified reason to provide more. You should check that the applications in the VM can take advantage of the extra vCPUs and can use multiple threads sufficiently before adding them. Too many vCPUs only penalizes the hosts, often without benefiting the applications on the VM. If a VM can't take advantage of the extra vCPUs, it can have a detrimental effect on other VMs' performance. Additionally, vSphere's fault tolerance (FT) feature only works with single-CPU VMs, so any design considering FT should have only one vCPU for the VM to be protected.

Too Many vCPUs?

If you suspect that a VM has too many vCPUs allocated to it, you can test the theory by recording the effect when you drop it down. The vCenter Advanced performance graphs have a CPU metric called *co-stop* that shows how much the VM was delayed because it was waiting for the CPU scheduler. This can indicate that too many vCPUs are allocated and that reducing them would help the scheduler work more efficiently. Comparing co-stop and ready times, before and after a vCPU change, should highlight any overprovisioning. In esxtop, co-stop is named %CSTP.

Adding extra vCPUs to your VMs has an impact not just on the VMs themselves but also on other VMs on the same host, and even on other cluster members. More vCPUs change the reporting metrics, the HA slot size for all VMs in the cluster, and the ratio of vCPUs in a cluster. If there is any CPU pressure on the host, wasted non-used vCPUs will compromise the performance of all the host's VMs. vCPU allocation is a fine balance between an individual VM's requirement for performance versus the needs of the rest of the host/cluster.

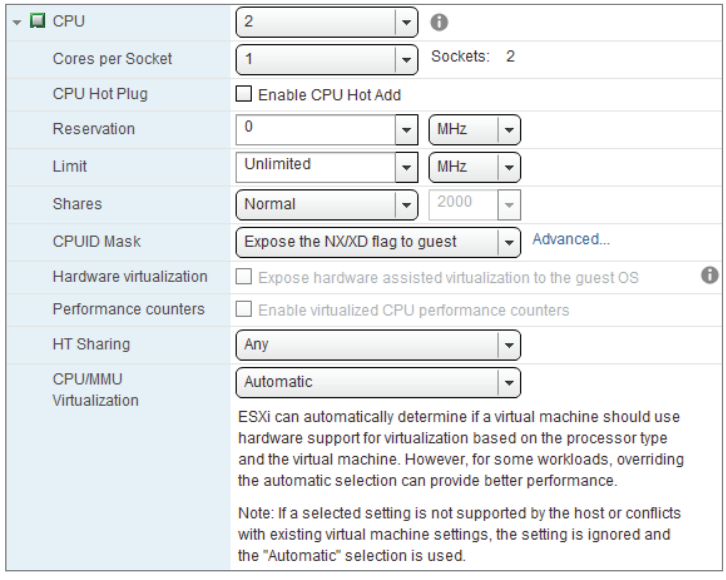
In Chapter 4, "Server Hardware," we discussed NUMA. However, it's worth considering how multiple vCPUs can impact NUMA hosts specifically. VMs with more vCPUs than there are cores in a host's NUMA node can see performance issues on NUMA-enabled systems. The VM is forced to use memory from a remote memory node, which in turn increases latency. vSphere can recognize this and has additional algorithms that improve the vCPU and memory allocation, known as Wide VM NUMA, for those relatively large VMs. If you're running VMs that fit this profile, then you may see benefit from ensuring that your hosts are running at least vSphere 4.1.

Virtual NUMA (vNUMA) was introduced in vSphere 5.0. It reveals the underlying host NUMA topology to guest OSes that are NUMA-aware. This means the guests can schedule

themselves in the most efficient way for the underlying hardware. vNUMA is by default only enabled on VMs that have more than eight vCPUs. The vNUMA settings are configured when it's first powered-on, so avoid moving a vNUMA VM to a host with a different physical NUMA topology. This is another good reason to keep clusters hardware consistent.

Figure 7.12 shows the layout for the CPU section of a VM's settings. Each of the settings are discussed in the following sections.

FIGURE 7.12
vCPU settings



Cores per Socket

The Cores per Socket setting allows you to allocate vCPUs to VMs as virtual cores instead of sockets. This doesn't affect the VM from a host perspective, but purely determines how the guest OS interprets its virtual hardware. As far as the hypervisor and its resource allocation are concerned, allocating one socket with two cores is the same as two sockets each with one core. The benefit is realized in guest OSes where there is a restriction on the number of sockets to which they can allocate resources. For example, Windows 2008 standard edition will only use the first four sockets, but it can take advantage of more cores. Unless there is a good reason to change the default of 1 core per socket, you should scale the vCPUs with the virtual socket value. vNUMA calculations in the guest OS can be detrimentally affected by dividing up virtual sockets among virtual cores. This could lead to a less efficient vNUMA load placement.

CPU Hot Plug

The CPU hot-plugging feature is visible only if the guest OS set in the general options is recognized as capable of supporting it. CPUs can be hot-added or removed, whereas memory can only be hot-added. VMware Tools must be installed to use this feature; and despite the fact that you can hot-plug these devices, the VM must initially be turned off to enable the feature. So,

you must plan ahead to have it dynamically available. If this is something you'll use, you must consider and test the guest OS, kernel SMP support, applications, and so on, because support depends on several guest factors.

When using CPU hot plugging, VM hardware should be at least version 8 (compatible with ESXi 5.0 and later), the first version to allow hot-adding with multicore VMs. Enabling hot plugging adds a small amount of guest resource overhead, prevents the use of vNUMA, and temporarily disconnects any USB passthrough devices when you make a change. For these reasons, hot plugging probably isn't something you want to enable wholesale; reserve it for particularly dynamic VMs that can't afford the downtime associated with adding CPUs.

Resources

Each VM has a number of resource attributes you can use to fine-tune its resource allocation against other VMs.

The CPU resources panel allows you to set shares, a reservation, and a limit for the VM, as shown in Figure 7.12. You can set these resource-allocation options for the VM, at the resource pool level in a DRS cluster, or both. If the resources are set at both, they're first carved up at the resource pool level; then the individual VM settings apply in the resource pool.

Generally, the vSphere hypervisor provides excellent scheduling. And normally, if hosts are sufficiently resourced, you can leave the default settings as they are. If you wish to control which VMs receive more priority or resources, it's fairer, more effective, and less prone to error to allocate these at a resource pool level.

The next chapter looks carefully at resource pool design, and we'll leave some of the discussion until then. However, because resources can be allocated here at the VM level, you need to understand the impact of doing so. A design should try implementing any scheduling at the resource pool level whenever possible.

CPU Shares

You can set CPU shares to a low (500 per vCPU), normal (1000), high (2000), or custom level. Shares only take effect during periods of contention, so as long as the host has enough resources to meet all the demand, the shares are never used. This allows all the resources to be used when a VM needs them, if they're available, and prevents the waste associated with limits.

But shares depend entirely on what all the other VMs on the host are set to. CPU resources aren't guaranteed with them, and their effective allocations will change as other VMs are added, change, or disappear.

The CPU shares set on a VM will impact the CPU allocation on all the other VMs on the host if there aren't enough CPU cycles to meet demand. Be careful of multi-vCPU VMs, because they receive shares in proportion to the number of vCPUs. Think of a VM that has four vCPUs. It's considered a more important application and so is given shares at the high level; this means it ends up with eight times the shares of a normal, single-vCPU VM.

CPU Reservation

The CPU reservation is set to zero by default. If you increase this value, it guarantees that amount of CPU cycles regardless of any shares set. They're reserved as soon as you power on the VM and can then affect the ability of other VMs to reserve CPU resources.

The VM can use more or less than the reservation set. If it isn't being used by the VM that's reserving it, other VMs can use the idle resources, at least until the reserving VM requires them. The reservation prevents other VMs from draining resources, to a certain point. However, the more you reserve for one VM, the less is available to be reserved by others. Excessive CPU reservations also negatively impact HA slot sizes. Although setting a reservation may prevent problems in one VM, it negatively affects the VMs around it.

CPU Limit

A CPU limit prevents a VM from using too many resources. The goal is to reduce VM performance! Think seriously before you set a limit anywhere, because doing so is rarely justified and is almost never a good idea.

A limit always restricts the VM, even when there is no contention. It's always applied. You can set a limit if a VM regularly runs out of control and impacts other VMs negatively. The textbook reason to impose limits is to prepare users for degraded performance as more VMs are added to a host. This creates a level of end-user consistency and cripples those VMs artificially. Frankly, that's a waste of resources. Generally, if you think you need limits, use shares and reservations instead.

Additional CPU Settings

In addition to the base CPU settings and resource controls available for a VM, a number of advanced CPU options are available.

CPUID Mask

vMotion compatibility is discussed in Chapter 4 and is normally configured at a cluster level, but a VM can change its CPU mask on an individual basis as shown in Figure 7.12. This allows you to hide certain CPU features from the VM and let it vMotion across noncompatible hosts.

The only VMware-supported mask is the NX/XD execute bit, but clicking the Advanced link lets you mask more flags. This can be useful in a test or lab environment where support is less important and where you may have an eclectic mix of old hardware.

Generally, it's easier to enable Enhanced vMotion Compatibility (EVC) across the entire cluster, as we'll discuss in the next chapter.

Hardware Virtualization

In ESXi 5.1, you can expose full CPU virtualization down to the guest by selecting the virtualized hardware virtualization (VHV) check box. This allows hypervisors to run as *nested* guests. This is very useful in test and lab environments where you need to run several ESXi hypervisors but are limited by physical hardware. It also permits the morally questionable practice of testing non-VMware hypervisors.

CPU Performance Counters

New to vSphere 5.1, and therefore requiring hardware version 9, is the ability to enable virtual CPU performance counters inside guest OSes. Unfortunately, these additional counters can't be enabled if the host is in an EVC cluster, so this may preclude the use of EVC and even force you to split the cluster if the hosts are sufficiently different. These counters are likely to be used by software developers only during debugging.

HT Sharing and Scheduling Affinity

The HT Sharing resource option sets individual HT modes for a VM. Ordinarily, the vSphere hypervisor deals with HT very well with its optimized CPU scheduler. However, some software recommends that HT be disabled, because it can conflict with its own CPU multiprocessing techniques. This setting allows you to keep HT enabled on the hosts but change it for a particular VM if you wish.

Normally this option is set to Any, which lets the VM's vCPU share cores with its other vCPUs or another VM's vCPUs. The second mode is None, which means the vCPU doesn't share the core with anything, and the hyperthread is stopped while the vCPU is using it. Finally, if you choose Internal and the VM has two vCPUs, the core is only shared with itself; otherwise, any other number of vCPUs will revert to no sharing. You can set this option regardless of whether the VM is turned on or off.

You can set the scheduling affinity to fix which cores are used. Setting the CPU affinity doesn't isolate and dedicate a physical CPU to a VM; it only restricts the movement of that particular VM. This is only used if you determine that a VM's workload has significant inter-vCPU communications, such as graphics intensive applications. You should avoid this unless necessary, because it creates a limitation on vMotion and degrades the host's ability to balance other workloads as efficiently; it's normally better to use other CPU resource settings.

These settings aren't visible if the VM is in a DRS cluster set to Fully Automatic or if the host doesn't have the hardware to support it.

CPU/MMU Virtualization

Hardware virtualization offload support is automatically handled for VMs, but you can individually configure it for special use cases. More details on CPU and MMU hardware assisted virtualization can be found in Chapter 4.

Virtual Machine Memory Design

In vSphere 5, you can apportion RAM to a VM in multiples of 4 MB, with a minimum of 4 MB (VMs using EFI firmware must have a minimum of 96 MB to power-on) and maximum of 1011 GB (assuming the host has that much to give a single VM). Although 4 MB seems like a ridiculously small amount to give a VM, it's occasionally found when administrators want to prevent vSphere users from turning on their VMs. vSphere can allocate such large amounts of memory that a VM's RAM tends to be limited only by the physical host. Figure 7.13 shows the memory settings available for each VM.

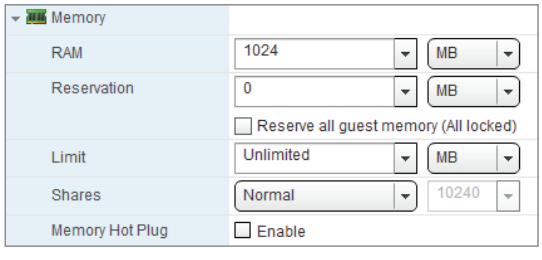


Figure 7.13
Memory hardware options

The advanced memory techniques discussed in Chapter 4 mean the VM always sees the amount of memory you've allocated it, even though it may not physically have access to that much RAM. This can be because the host is reclaiming idle pages, sharing pages with other VMs (TPS), compressing them, swapping to host cache (SSD) if available, or adhering to memory limits that have been set. In extreme cases, the VM's memory may not even be from physical RAM but is being forcibly swapped to disk by the hypervisor. We covered this in much more depth in Chapter 4.

A VM's memory allocation has an effect if you don't assign it at the right level. Not enough memory, and the VM may be forced to swap with its own paging file, even if the host has ample amounts. Too much memory, and too much overhead is reserved, preventing other VMs from reserving it. Each VM should be allocated just a little more than the average memory usage, to allow for small spikes.

VMs running on NUMA-enabled hosts can be affected if they have more memory allocated to them than the lowest configured NUMA node. Memory is split across NUMA nodes depending on physical placement in the server. If you're running very large memory VMs on a NUMA host, you should check that the RAM is set properly in the DIMM slots, so VMs aren't forced to use nonlocal memory.

Resources

Similar to CPU options, memory can be allocated at the VM and resource pool levels. Where possible, designs should aim to set these at the resource pool level. Memory shares, reservations, and limits operate like their CPU counterparts but differ in a few crucial ways. We'll look at how they differ next.

Memory Shares

Memory shares work just like CPU shares and are used only during memory contention. They entitle a VM to a certain slice of memory, in line with the other VMs' shares, subject to any reservations and limits set. To prevent wastage in VMs that have a high proportion of shares but unused memory, an *idle tax* is factored in to the calculations. This reclaims more memory from VMs that aren't using their allocated share. Memory shares shouldn't be changed unnecessarily, but they're preferable to reservations and limits.

Memory Reservations

A memory reservation is different from a CPU reservation because it's selfish and doesn't release idle resources back to other VMs the same way. Until a VM uses the memory, others can use the reserved memory; but as soon as it's used, it's not released until the VM is powered off. It's never reclaimed. Unfortunately, Windows addresses all of its memory when it boots up, so the entire memory reservation is held. Linux only touches the memory when it needs to, thus minimizing the impact.

Like a CPU reservation, a memory reservation may have a positive effect on the VM but can negatively affect its surroundings by reducing available memory and changing HA slot sizes.

Memory Limits

Just like CPU limits, memory limits are generally a bad idea. Memory limits are probably even worse, and they're easily avoided because you can set a VM's memory level far more effectively by reducing its RAM allocation.

When a VM boots up and applications start, they make memory-management decisions based on the amount of RAM they think they have. Setting a memory limit doesn't change this behavior: the VM still believes it has the full allocation, which it's likely to try to use. With a limit set, every request over the limit is forced into VM swap, seriously degrading performance. However, if you reduce the RAM setting, the guest is far less likely to swap as much, because it knows where the limit really is. Avoid memory limits if possible.

Additional Memory Settings

A number of advanced memory options are also available, depending on the VM's guest OS and the underlying physical hardware.

Memory Hot Plug

This feature is visible only if the guest OS set in the general options is recognized as capable of supporting it. CPUs can be hot-plugged or removed, but memory can only be hot-added. VMware Tools must be installed to use this feature; although you can hot-plug these devices, the VM must initially be turned off to enable the feature. So, you must plan ahead to have it dynamically available. Similar to CPU hot plugging, enabling memory hot plugging consumes additional resource overhead, so don't enable it on a VM unless you're likely to use it.

NUMA Memory Affinity

NUMA memory affinity settings are available only if the host supports them and isn't a member of a fully automatic DRS cluster. The settings, shown in Figure 7.13, allow you to select the NUMA node affinity. This forces the VM to use memory from certain nodes and not others. Because this is host-specific, the affinity settings are cleared when the VM moves to another host; and memory affinity only works effectively if you also specify the CPU affinity.

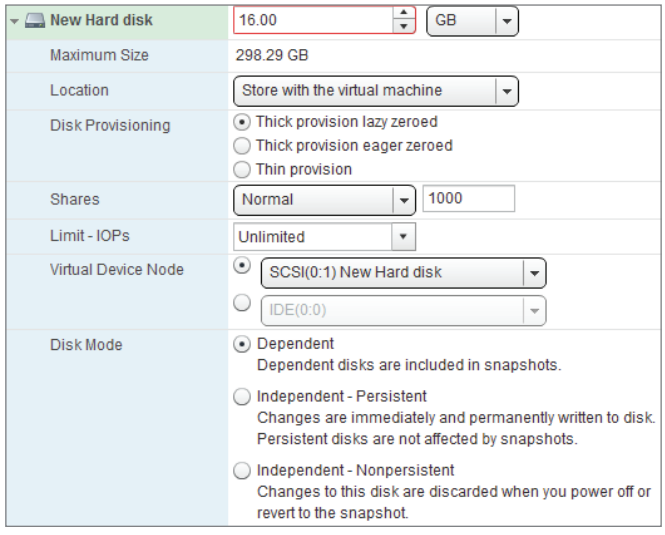
Applying specific NUMA CPU and memory settings can be useful if you have a very static environment with a smaller number of VMs. VMs with large, memory-intensive workloads can benefit from static NUMA mappings, but ordinarily such fine-grained adjustments aren't required.

Virtual Machine Storage Design

One of the crucial design factors for VMs is its storage. vSphere provides a great deal of flexibility for storing VM data, and that gives rise to numerous decisions. Chapter 6 details vSphere storage design and how ESXi hosts connect, but each VM has various storage options

determining how its disks are presented. Figure 7.14 shows how these options are laid out in the vSphere 5.1 Web Client.

FIGURE 7.14
Disk options



Disks

Although this isn't necessarily vSphere specific, you should consider the number and size of disks to present to each VM. The disks that are presented to VMs have several layers of abstraction. A physical server or workstation with one or more disks inside, and perhaps hardware or software RAID, usually sees one large disk, which it splits into partitions to use. However, VMs are much freer to split their storage to exactly what is needed.

With this freedom, VMs are normally divided into several smaller disks with a single partition on each. These disks are easily grown (assuming spare space exists in the datastores), but contiguous partitions make growing all but the last one on the disk more difficult. For that reason, it's advisable to create VMs with only one partition per disk.

The ease with which you can add disks of any size gives rise to more options when splitting up OS space. On Windows guests, it's common practice to split off the OS C drive and have separate disks for user data, program files, logs, swapfiles, and so on. For example, a Windows SQL VM may have a separate disk for the OS, the databases, the SQL logs, the swapfile, and a backup drive. A Linux VM can have separate disks for each slice of its file system, so a typical setup may have individual disks for `/`, `/boot`, `/usr`, `/opt`, `/home`, `/etc`, `/tmp`, `/var`, `/var/log`, and so on; the list can be endless to suit your requirements.

Splitting a VM's Partitions onto Separate Disks

If you're P2Ving an existing physical server with VMware Converter, and it has more than one partition on a disk, watch for the advanced disk options to select a customized target disk layout. This lets you split each partition out onto its own VMDK disk file during the P2V. And if you have a VM with multiple partitions on a single virtual disk, VMware Converter's V2V is an easy way to fix it.

Another advantage of splitting out each piece of file system is the granularity it gives you to select different performance profiles for different areas. The underlying storage from which the disks are carved can be on different RAID sets, with different disk speeds and different spindle counts, all providing different levels of performance. So for example, a database server can run its OS off an inexpensive RAID 6–based disk; its swap can sit on a fast but vulnerable RAID 0–based disk; and the transaction log and DB files can be on a high-performing, redundant but costly RAID 10–based disk.

This physical separation of logical components also allows you to split off areas of a VM that you want to treat differently. For example, you may want to avoid using SAN replication on certain transient data like swap space. Or you may wish to provide greater protection to some data on the underlying storage. Backups can be simplified by avoiding entire disks that don't ordinarily need to be backed up.

Each VM can have a total of 60 VMDK disks attached to it, each of which can be close to 2 TB, giving you the sort of scalability to quench even the most insatiable of capacity appetites. However, one of the advantages of virtual disks is the ability to make them smaller than usual. On a physical standalone server, you may as well use all of the disk capacity from the outset. With VMs, you should start the disks small and grow them as required. For each VM, you should consider the size of the OS, the applications, the user data in combination with the number of users, the swap, the logs and spooling data, with some room for growth.

Disk Types

vSphere VMDK disks come in three different types:

Thick Provision Lazy Zeroed All the space is allocated on the datastore at the time of creation. It isn't pre-zeroed, so it's quick to create. As the disk is written to, the space is zeroed as the I/O is committed. Zeroing the disk ensures that no old data from the underlying storage is found in the new disk.

Thick Provision Eager Zeroed Again, all the space is preallocated to the disk on the data store when the disk is first created. However, with eager-zeroed thick disks, the entire space is zeroed out at this time. These disks can take a significant time to create, but when they're ready they exhibit a marked performance improvement over new zeroed thick disks. For this reason, ensure that any I/O-intensive disks are prepared this way or converted to this format if already provisioned. Storage arrays capable of the write same/block zero vStorage APIs for Array Integration (VAAI) primitive will reduce the time to create these disk.

Thin Provision Similar to thin provisioning on a storage array, VMDK thin disks are only allocated space as they grow from disk I/O. The disk starts small and is grown as the space is zeroed, ready for disk I/O. It won't grow beyond its allowed size. Despite speculation to the contrary, thin provisioning doesn't impact performance but performs extremely closely to that

of zeroed thick disks. The main advantage of thin disks is the space you save by not allocating everything up front. However, some guest disk operations, such as defragmentation, cause thin disks to inflate. You must take care not to overcommit a Virtual Machine File System (VMFS) volume with a thin disk in it. You can use appropriate vCenter alarms to mitigate the likelihood of this.

SE sparce Disks

vSphere 5.1 introduced a new disk type called *SE sparce* (Space Efficient) disks, although also referred to as *FlexSE* in some places. Currently these disks are not for general purpose vSphere use but limited to VMware View desktops. SE sparce disks are similar to thin provisioned disks in that they will grow over time as data is written to them, but they can also be manually shrunk after data is deleted.

The different types are best seen when you add a new disk to a VM. You're given the option of which type of disk format you'd like, as shown in Figure 7.14.

When you create a new VM or add a new disk, the default format is thick provision lazy zeroed. If the underlying storage isn't a VMFS volume but a Network File System (NFS) datastore, then the VMDK type can be dictated by the NAS device, and the disks are thin provisioned. If the NAS device has support for the appropriate VAAI primitive, then it may be able offer thick-provisioned disks. See Chapter 6 for details about the VAAI primitives.

Fault tolerance requires thick provision eager zeroed disks, and Microsoft clustering needs it for its quorum and shared disks. You can't simply switch between formats; but when you Storage vMotion a VM from one datastore to another, you're given the choice of what the destination format should be. This gives you a straightforward method to convert your disks to the format you need.

Disk Shares and IOPS Limits

A VM can set shares on a per-disk basis, as shown in Figure 7.14. This VM disk share is only apportioned across a host, not at a resource pool level, and it allows for a very rudimentary level of control. As a share, it applies only during I/O contention.

A feature known as Storage I/O Control (SIOC), introduced in vSphere 4.1, lets shares apply at the datastore level after certain latency thresholds are met. At the VM level, you can enforce IOPS levels on VMs, again shown in Figure 7.14, and prevent one VM from heavily affecting others.

Disk Modes

Regular VM disks are created as VMDK files. You can create these disk files on either block-based datastores or NFS exports. There are three disk modes to select from, as you saw earlier in Figure 7.14.

Dependent

The default disk mode, the one in which all disks are initially created as, is *dependent* mode (some older documentation refers to this as *snapshot* or *normal* mode). Unsurprisingly, the discernible differentiator is that dependent mode VMDK disks can use the snapshot feature.

vSphere VMDK snapshots aren't like SAN snapshots; they aren't copies, but change deltas. It's just a point in time, where the disk I/O is redirected to another disk file. You can then choose to either incorporate those changes back into the main disk (committing) or discard the changes to revert to the original snapshot (deleting). The most important thing to remember from a design point of view is that they're only intended for short-term use. If you want to keep whole copies of VMs at particular times, use vSphere cloning or a suitable backup solution.

Thick-provisioned disks (the default on block-based datastores) without snapshots are static-sized files. However, as soon as you take a snapshot, the static file remains, and new changes are written out to new space. This means that when a snapshot is taken, you can unexpectedly fill your datastore without provisioning any new VMs. Creating snapshots also places restrictions on the VM, such as no longer being able to Storage vMotion the VM.

Finally, as each snapshot is taken, it creates a chain of deltas. As the number of delta files increases, the chaining becomes more complicated and more prone to issues. A break in the chain at any point can lead to data loss. The snapshots can also have an associated performance overhead, so if eking out every ounce of disk I/O is important, you should avoid leaving snapshots in place.

Don't plan to keep your snapshots too long. They're good for short-term use, such as when you're patching or testing changes. Thankfully, the snapshot algorithms keep getting better. vSphere 4.0 came with a much-improved way of dealing with snapshots, making them considerably more reliable. More patches were included in 4.1, reducing the space required to commit snapshots. vSphere 5 now stores snapshot delta files in the same directory as the parent disk, instead of the home folder where previous versions kept them. This ensures that the delta disks can expect the same performance characteristics as the parent, whereas previously the home directory might have had different underlying storage. The best advice is to keep your hosts at the latest version.

Independent Persistent

Independent persistent disks operate like regular hard drives. All changes are immediate, and there is no potential performance degradation as is associated with snapshot mode. However, with the lack of snapshot functionality, certain vSphere features are unavailable, such as VMware Data Protection (VDP), and lots of third-party backup tools.

Independent Nonpersistent

Independent nonpersistent disks differ in that all changes are lost when the VM is powered off (but not when it's rebooted). This returns the disk to its original state, losing all subsequent changes. This disk mode is useful for environments where you want to keep a consistent running image. A good example use case of nonpersistent disks is a kiosk-style terminal, a teaching lab, or a VM used to create ThinApp packages, where you want to return to exactly the same configuration on a regular basis.

SCSI Controllers

vSphere supports four types of SCSI controller:

BusLogic Parallel The BusLogic controller provides support for older OSes and is the default for Windows 2000 guests.

LSI Logic Parallel The LSI Logic Parallel controller is supported for newer OSes with built-in support on Windows 2003 and later. VMware also recommends that you use the LSI-based controller for Red Hat installs.

Both default controllers should have identical I/O performance, and they differ only slightly in their hardware presentation. If the guest doesn't have either driver installed, pick whichever is easiest to install.

LSI Logic SAS The LSI Logic SAS controller has built-in support for clusters on Windows 2008. It provides a small performance boost over the two legacy controllers. However, this controller is only available for VMs whose hardware is at least version 7.

PVSCSI The PVSCSI adapter is VMware's own paravirtualized controller for high-performance VMs. It can provide an increase in throughput while reducing CPU overhead. However, its use should be reserved for high-I/O VMs, because it can potentially have a higher latency than other controllers if I/O rates are lower. The PVSCSI driver coalesces interrupts to reduce the amount of CPU processing required. If the I/O is too low, then all it does is introduce delay.

You can use the PVSCSI driver on version 7 and above VMs, and it supports Windows 2003/2008 and Red Hat Enterprise Linux 5 (RHEL 5). With the initial introduction of PVSCSI controllers, there was no support for boot disks, but this was resolved for Windows guests in 4.0 Update 1. It's common to use a default controller for boot/OS disks for ease of install and then add a second controller for the higher-workload disks.

Depending on the guest you select, the default used is either the BusLogic or the LSI Logic controller, unless it's a Windows XP VM. Windows XP guests don't have the necessary drivers for these SCSI controllers and default to using IDE disks. Each VM can have up to 4 different controllers and up to 15 devices per controller.

Adding more SCSI controllers is also an effective way to distribute storage processing in a VM and can significantly increase storage I/O. For VMs that need the best storage throughput, such as a critical database server, reserve the first SCSI controller for the OS and swap disks, and add up to the maximum of three PVSCSI controllers to spread additional high I/O disks. One can be used for DB disks, one for log disks, and another for TempDB disks. This will marginally increase the guest's CPU but is usually a small price to pay for the gains in storage throughput.

SCSI Bus Sharing

The SCSI bus-sharing policy is set for each controller. By default, it's set to None, meaning that only that particular VM can lock the disk file. However, if guest clustering software running on multiple VMs needs concurrent access to the disk, you can set this to Virtual for cluster in a box (CIB) or Physical for cluster across boxes (CAB).

RDMs

RDM disks are an alternative to normal VMDK disks. RDMs are small mapping files to raw LUNs. They allow the ESX hosts to address the LUN as if it was a VMDK, but this means the VM can have direct access to the entire LUN.

The RDM file contains all the metadata required to manage and proxy the disk access, instructing the VMkernel where to send disk instructions. VMware recommends that you use RDMs only when justified, because the preferred disk format is regular VMDK virtual disks.

Two types of RDM exist—virtual and physical—and your use case will dictate which one is appropriate. Note that both types of RDM support vMotion, despite a common perception that this is available only on virtual RDMs.

Virtual Compatibility Mode RDM

Virtual RDMs act just like regular VMDK files. They virtualize the mapped device so that they appear to the guest OS to be disks from a VMDK datastore. This allows the use of snapshots; and because the RDMs hide the underlying hardware, the LUN is potentially more portable when moving to new SAN equipment. Virtual RDMs are used for CAB-style Microsoft clustering. CIB can also use virtual RDMs, but VMware recommends using VMFS-based VMDKs unless you're likely to reconfigure them to a CAB cluster eventually.

Physical Compatibility Mode RDM

Physical RDMs have almost complete direct access to the SCSI device, which gives you control at much lower levels. However, this means you can't use the snapshot feature. Physical RDM mode is useful for SAN management agents that require access to hardware-specific commands. Physical RDMs are also used for physical to virtual (n+1) Microsoft clustering.

RDM Usage

RDMs are used for a variety of reasons:

Application Requirements Some applications need to make direct calls to the block table. The common example of this, and one just discussed, is Microsoft clustering, which needs access to RDMs to cluster across vSphere hosts or from a vSphere host to a physical windows server.

SAN Technology Some older SAN technologies like replication, deduplication, and snapshots may not work with VMFS volumes. SAN management and storage resource management (SRM) applications may need lower-level access.

NPIV NPIV only works with RDM disks. It allows a VM to claim a virtual port from the host's HBA, and it enables finer control of things such as security (via per port zoning), bandwidth priorities, and QoS.

Migrating Data to Virtual If a very large LUN is attached to a physical server, then it's possible to attach the LUN directly to the replacement VM. You should always try to move the data over to a VMFS volume, but sometimes an RDM can provide a good transitional platform.

Flexibility If you think you may need to move an application from a VM back up to a physical server, perhaps to promote a staging server to a production physical server, then making it a physical RDM from the outset can make the migration much easier.

Large Disks As of vSphere 5.0, physical RDMs allow individual disk sizes up to 64 TB, whereas regular VMDKs (regardless of being on VMFS or NFS datastores) and virtual RDMs are still capped at 2 TB. If you have very large file requirements and can't use in-guest mount points or guest volume management (such as dynamic disks or LVM) to concatenate several disks, then physical RDMs can be used.

Misinformation RDMs were considered by some to be the best solution for very high-performance I/O requirements. This belief isn't justified, and the performance differential is negligible, but the myth is still perpetuated in many quarters.

RDMs have several drawbacks. They're inherently less manageable, and they lack the portability that makes regular VMDK disk files the default choice. They also require the entire LUN to be dedicated to only one VM disk. This can cause serious scalability issues, because each host can have a maximum of only 256 LUNs. This may sound like a lot, but if they're being used by VMs with several disks, this ceiling can have an effect. Heavy RDM use also has other problems scaling, because the workload on the storage team grows when they have to create and manage that many more LUNs.

RDMs shouldn't be anyone's first choice, but they're indispensible in certain circumstances. They're a useful tool that you can call on in your design if necessary; but try to avoid them if possible, because they're limiting.

Storage vMotion

Storage vMotion is an interesting capability from a design perspective. It allows for zero-downtime datastore migrations. But you should note a couple of things while creating solutions around it. First, Storage vMotion requires at least a Standard level license; this isn't available for Essential and Essential Plus customers.

Second, you should be aware of the impact of Storage vMotion migrations. They're disk-intensive operations. The VM has to continue to read and write to the array while it's reading from the source and writing to the destination. Both source and destination are likely to have other VM traffic I/O, which can be affected by the Storage vMotion. So, if you're using Storage vMotion to migrate a large amount of data, although there may be no downtime, it can have a significant effect on overall SAN performance.

The Storage vMotion can up large chunks of space on the source datastore if the VMs being moved have very large disks or are very disk intensive. Make sure you have sufficient room; otherwise, the datastore will quickly fill and cause issues for all the VMs sharing the datastore.

VAAI offloading capabilities, which we discussed in the last chapter, can reduce this impact significantly if used in conjunction with a compatible storage array.

Moving VMs between VMFS-5 volumes that have been upgraded from VMFS-3 with different block sizes will suffer from degraded transfer performance. Consider rebuilding your upgraded datastores if there isn't a consistent block size.

Cross-Host vMotion

vSphere 5.1 introduces a new feature to the regular migration wizard: Cross-Host vMotion. Essentially, a vMotion and a Storage vMotion are combined in one operation. However, this feature doesn't require the usual Storage vMotion prerequisite of both hosts being able to see the same shared storage. A VM located on a host's local disks can be Cross-Host vMotioned to another host's local disks or between any 2 hosts that don't see the same datastores.

Clearly, the advantage of shared storage is that the vMotion is quick; a Cross-Host vMotion requires that the entire VM be copied across the wire, which takes considerably longer. However, in circumstances where shared storage isn't available or two hosts can't have the same shared storage presented to them, this is a valuable feature that prevents the VM outage normally associated with a cold migration.

Cross-Host vMotions will complete significantly faster if both source and destination VMFS volumes have a 1 MB block size. If there isn't any shared storage to enable a Storage vMotion to

take place, the disks are transferred over the vMotion network (although snapshots are transferred over the Management network). Therefore multiple vMotion vmknics will speed the VM's movement as the traffic is load balanced across all available connections.

VM Storage Profile

The storage profile feature is explained in depth in Chapter 6 and is evident in a VM's configuration. When a VM is created, if any storage profiles exist in the datacenter that is selected as a target, you can choose one. The VM's summary page, shown in Figure 7.1, details the storage profile compliance details. See Chapter 6 for further details.

Virtual Machine Network Design

vNICs are the network adapters presented to VMs. Note that they're different from VMNICs, which is the name hosts give to physical network adapters and vmknics which are VMkernel interfaces.

> **Watch for Terminology Confusion**
>
> VMware's vNICs are the network adapters on a VM. However, not all vendors use this term in the same way, which can lead to confusion. For example, the popularity a few years ago of everything being "virtual" has meant that several server vendors often call their network adapters vNICs if there is any level of abstraction involved. For example, Cisco UCS blades are presented "vNIC"s for their northbound connectivity through the I/O modules to the fabric interconnects. These vNICs are what a VMware administrator thinks of as a VMNIC.

Ordinarily, each VM has only one vNIC. Unlike a physical server, you gain no benefit such as increased bandwidth or additional redundancy by adding a second vNIC. Figure 7.15 shows a VM's vNIC options.

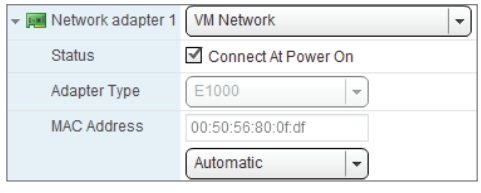


Figure 7.15
Disk provisioning types

There are a couple of reasons you might add a second vNIC. For example, you may wish to bridge two networks. Take care not to create the network loops that vSwitches help avoid. Firewall-appliance VMs need a second vNIC to bridge both sides of the DMZ. Also, you may want your VM to access more than one subnet, because in your environment different types of data are segregated onto separate VLANs—for example, if all backup agent traffic is forced onto its own subnet. Software-clustering solutions often require access to an extra heartbeat link, and a second vNIC makes it convenient to access this.

vNIC Drivers

Each VM can select from four different vNIC adapter types. Normally, a VM is deployed with one network adapter, and the type is automatically selected depending on the VM's hardware version, the host's version, and the guest OS. However, you can change this choice—for example, if the VM's hardware is upgraded after the VM has been built. It's often a great way to take advantage of new features or more optimized drivers.

Flexible

Flexible is the default vNIC type for 32-bit guests used for VMs that were originally deployed on ESX 3.0 hosts. It will function as a vlance adapter if VMware Tools aren't installed but as a VMXNET device if the VMware Tools are detected:

Vlance A vlance adapter emulates an AMD PCnet32 LANCE network card, an old 10 Mbps NIC. It's used for maximum compatibility, because this driver is included in the default install of most modern OSes. Practically all Linux distributions include drivers for this adapter. But support is starting to be dropped—it's no longer included from Windows Vista onward.

VMXNET VMXNET was the first paravirtualized VMware driver, meaning that it was designed for the virtualized environment to minimize I/O overhead while passing traffic to the physical interfaces. There is no hardware equivalent, so the VMware Tools must be installed.

E1000

An E1000 vNIC emulates an Intel E1000 network adapter. It's primarily the default for 64-bit guests.

VMXNET 2 (Enhanced)

VMXNET 2 (Enhanced) is an upgrade to the first paravirtualized driver for ESX hosts. It includes support for performance features such as jumbo frames and certain hardware offloading.

Again, like the VMXNET vNIC, it requires the VMware Tools to be installed, because there is no physical hardware equivalent. VMXNET 2 has been available since version 3.5, but the set of supported guest OSes is limited.

VMXNET 3

VMXNET 3 is the latest paravirtualized driver, introduced with vSphere 4.0. It was completely rewritten and didn't come from the VMXNET lineage. It supports lots of new performance features to improve network scalability and makes the most of IPv6 and newer 10GbE network cards.

This vNIC is supported only on VMs with hardware version 7 or later, and guests must have the VMware Tools installed. Check compatibility with your guest OS, because support for VMXNET 3 is the most limited.

DirectPath I/O

DirectPath I/O isn't a vNIC adapter in the traditional sense, but it uses PCI passthrough, allowing a VM to bypass the hypervisor network stack and giving it direct access to the physical NIC hardware. DirectPath I/O may provide a minor increase in throughput over vNICs, but arguably it's most useful because it can reduce the CPU load for network-intensive VMs.

However, the use of DirectPath I/O as a feature has a number of restrictions that severely limit its suitability in most circumstances:

- Can't vMotion (therefore limiting features such as DRS)
- Locks the physical NIC to that particular VM; no other VM can use it as an uplink
- No snapshots
- No suspend/resume
- No FT
- No NIOC
- No memory overcommit
- Can't use with VMsafe tools (vShield Endpoint solutions)

With vSphere 5.0, Cisco's UCS platform isn't restricted by the first four limitations listed, but it still can't use FT, NIOC, memory overcommit, or VMsafe.

SR-IOV

Support for single root I/O virtualization (SR-IOV) was added to vSphere in version 5.1. SR-IOV is analogous to DirectPath I/O but crucially allows multiple VMs to address the same PCI card. It has similarly restrictive impacts on the VMs and additional requirements over and above those for DirectPath I/O. Chapter 4 described SR-IOV in greater depth.

Guests Reporting Incorrect NIC Speeds

The speed that the drivers report in the guest OS isn't necessarily the actual speed of the network traffic. The drivers report what they believe they're capable of, but their actual speed depends on the underlying physical network adapter. Some drivers think they're capable of only 10 Mbps or 100 Mbps; but if the host is fitted with 1 Gbps NICs, then the VMs aren't limited by the drivers.

Table 7.3 describes the features available in each version of the vNICs in vSphere 5.

TSO TCP segmentation offload (TSO) reduces the CPU overhead associated with network traffic, to improve I/O performance. TSO-enabled NIC hardware can be used, but it isn't necessary to take advantage of TSO performance gains. It's supported only in certain OSes.

Jumbo Frames Jumbo frames are any Ethernet frames larger than the standard 1,500 bytes. This feature allows VMs to send frames up to 9,000 bytes, which reduces the I/O overhead incurred on each Ethernet frame. Each network device must be enabled for jumbo frames, end to end. To enable this for VMs, you must configure the vSwitch's maximum transmission unit (MTU), which changes this setting for all uplinks attached. Then, you must configure the NIC in the guest OS for jumbo frames.

SplitRx SplitRx allows the ESXi host to use more than one physical CPU to process packets received from one queue. When there is intrahost VM traffic, SplitRx helps to increase the throughput. If several VMs on a single host are all receiving the same multicast traffic, then SplitRx can increase the throughput and reduce the CPU load. vSphere 5.1 will automatically enable SplitRx mode on vmxnet3 adapters if inbound external traffic is destined for at least 8 VMs or vmknics. SplitRx can be manually enabled for an entire ESXi host or on a single vNIC.

MSI/MSI-X Message signal interrupts (MSI) is supported by VMXNET 3 drivers with three levels of interrupt mode: MSI-X, MSI, and INTx. It allows the guest driver to optimize the interrupt method, depending on the guest's kernel support. MSI uses an in-band PCI memory-space message instead of an out-band PCI INTx pin. This can lower overall interrupt latency.

Ring Size With each newer vNIC, the receive and transmit buffers have increased. A larger ring size creates more buffer, which can deal with sudden traffic bursts. There is a small impact on CPU overhead as the ring size increases, but this may be justified if your network traffic has bursty throughput. You can alter the buffer size in the VM's VMX configuration file.

RSS Receive-side scaling (RSS) can be used by some new Windows guest VMs. It distributes traffic processing across multicore processors to aid scalability and reduces the impact of CPU bottlenecks with 10GbE network cards. RSS must be enabled in the guest's NIC driver settings.

NAPI New API (NAPI) is a feature for Linux-based guests to improve network performance by reducing the overhead of packet receiving. It defers incoming message handling to process messages in bundles. This allows for greater CPU efficiency and better load handling.

LRO Large receive offload (LRO) is another Linux guest technology, which increases inbound throughput by aggregating packets into a larger buffer before processing. This reduces the number of packets and therefore reduces CPU overhead. LRO isn't suitable for extremely latency-sensitive TCP-dependent VMs, because the traffic aggregation adds a small amount of latency.

TABLE 7.3: vNIC features

	FLEXIBLE	E1000	VMXNET 2 (ENHANCED)	VMXNET 3
TSO IPv4	No	Yes	Yes	Yes
TSO IPv6	No	No	No	Yes
Jumbo frames	No	Yes	Yes	Yes
SplitRx	No	No	No	Yes
MSI/MSI-X	No	No	No	Yes
Large ring sizes	No	Yes	No	Yes
RSS	No	No	No	Yes
NAPI	No	No	No	Yes
LRO	No	No	Yes	Yes

vNIC Driver Performance

The VMXNET 3 driver is the best performance choice if the VM is at hardware version 7 or later and the guest OS is able to support it. If you can't use a VMXNET 3 driver, the next best-performing driver is the Enhanced VMXNET, as long the VMware Tools are installed.

From the remaining vNICs, VMXNET performs best. The E1000 then sits between the VMXNET driver and the lowest-performing vNIC, the aging vlance card.

vNIC Interrupt Coalescing

All ESXi vNICs queue network interrupts to reduce the CPU load. These very short burst periods can introduce very small latency to network links, but are limited to never exceed 4 ms. In extremely latency sensitive workloads, for example VOIP servers, you may want to change the settings or disable this feature to minimize the impact. Only vmxnet3 vNICs allow interrupt coalescing to be disabled or statically configured.

MAC Addresses

vSphere automatically sets a VM's MAC address. Ordinarily, the MAC address that's created doesn't need to be altered. However, you may want to change it in the following circumstances:

- There are more than 256 vNICs on a physical host, because conflicts with the automatically generated MAC addresses can occur.
- vNICs on different hosts but the same subnet are allocated identical MAC addresses.
- You need to set a fixed VM MAC address to prevent the address from changing, for example for software licensing reasons.

After a VM's MAC address is created, the MAC address will change only if the VM is turned off and then moved. However, some software installed in VMs ties its licensing to a MAC address, so in this case it's recommended that you set a static address.

You can see where the MAC address is set in the VM configuration settings shown in Figure 7.15. vSphere 5.0 and older doesn't support arbitrary MAC addresses: the allowable range is 00:50:56:00:00:00 to 00:50:56:3F:FF:FF. VMs built with *ESXi 5.1 and later compatibility* (hardware version 9) allow all 48 bits of the MAC address to be controlled. The limitation of using VMware's own OUI allocation isn't enforced anymore. Despite this, unless you are designing very large virtual environments where you are concerned about running out of addresses (with VMware's OUI you get 64,000 addresses per vCenter instance) or conflicts across multiple vCenters, then you are best advised to stay within VMware's own recommended range.

VLAN Tagging

Although vSphere's VLAN tagging options aren't strictly a consideration for a VM's networking design, they're worth mentioning briefly to help you understand how an 802.1q driver can be installed in a VM. Each VM's vNIC connects to one port group on a host. Each port group can use one of three types of VLAN tagging:

EST External switch tagging (EST) is the default port group option, when no VLAN ID is entered (or a VLAN ID of 0 is stipulated). No tagging is performed on the vSwitch, so there is a one-to-one relationship with the VMNICs and the access ports on the physical switch.

VST Virtual switch tagging (VST) is an extremely popular configuration in vSphere deployments to aggregate several VLANs onto a limited number of VMNICs. A VLAN ID number between 1 and 4094 is set on each port group, and any traffic passing out of the port group from a VM is tagged with the VLAN ID.

VGT Virtual guest tagging (VGT) allows you to install 802.1q tagging software in the guest OS. This lets you run several VLANs through to your VM on a single vNIC. This can be particularly useful if you're P2Ving a physical server that used this configuration and you need to preserve the setup. To use VGT, set the port group's VLAN ID to 4095.

Guest Software

vSphere can host a huge variety of guest OSes. Any x86-based OS will install in a VM, but only certain OSes are supported by VMware. Any supported guest OS has a VMware Tools package that can be installed. The list is always being updated, and you can find the latest version at `www.vmware.com/pdf/GuestOS_guide.pdf`.

Generally speaking, all modern versions of Microsoft Windows and Linux distributions from Red Hat, SUSE, Debian, Ubuntu, and FreeBSD or Solaris are supported. Even some versions of older legacy OSes like Microsoft DOS, IBM OS/2 Warp, and Novell Netware are supported. Those not on the list should still work, albeit without support. However, without a compatible version of VMware Tools, you may have to contend with driver issues.

Selecting an OS

vSphere demands 64-bit hardware to run on. But it can virtualize both 32-bit and 64-bit OSes very well. So, which version of the guest OS should you install in a VM? As a general rule, you can treat this decision much the same as if you were installing your OS on the bare metal. 64-bit OSes can address more memory and often perform better even with 32-bit applications.

Because VMs are so portable, it's easy to have OSes around much longer than they used to be. In most enterprises with physical server installs, the hardware is normally replaced at least every five years. If it hasn't been upgraded in that long, it's common to use this as an excuse to rebuild the server and update to the latest OS and application versions. However, infrastructure in vSphere is now abstracted from the hardware, so you may see OSes that are much older. Often, virtualization is used to remove old hardware, and very old OS installs aren't unheard of. It makes sense to install the latest version of any OS when you're deploying, and that should include the 64-bit choice.

Even though 64-bit hardware has been mainstream for many years, there are still some issues with driver support. This was always the one consideration against deploying 64-bit OSes. But with VMware's tight list of hardware and available drivers, this isn't an issue for vSphere VMs unless you need to pass through some legacy hardware. With OS vendors keen for users to migrate to 64-bit as soon as possible, there are no additional licensing costs, so little prevents you from using 64-bit from the outset.

There are some exceptions, however. One example is 16-bit Windows applications that need to be hosted but won't run on a 64-bit OS. Also, if you P2V an existing computer with a 32-bit OS, you're left with no choice.

One other OS option that can be worth considering, if you need a Linux-based guest, is JeOS (pronounced "juice"). JeOS stands for *Just enough OS* and is the term for specially customized

OSes that are fine-tuned for virtualized platforms. Without the need for extra drivers, these OSes can taper their kernels to make them smaller and more efficient. This is possible due to the modular Linux kernel; both Ubuntu and SUSE have their own JeOS-based offerings. These are used as the base of many virtual appliances. VMware has entered into a licensing agreement with SUSE to use its OS as base for some products that can benefit from a JeOS base.

Support for Apple Mac OS X as a Guest

Beginning with vSphere 5.0, VMware introduced support for Mac OS X versions 10.6 and above as a guest. Unfortunately, due to Apple's restrictive EULAs, its use is extremely limited. First, the virtualized guest OS must be the server version. Second, this can only be run on Apple hardware: you can't boot up an OS X image on non-Apple hardware. VMware's HCL limits support to the Xserve 3.1 model, which is no longer sold (although VMware community members have been successful in getting ESXi 5 to successfully run on Mac Minis and Mac Pros).

Guest OS and Application Licensing

OS and application licensing varies between vendors for virtualization platforms. You should look carefully at your options with each vendor. Some vendors base their licensing on the physical hardware the VM is running on at any one time. Confusion can reign, because this may be physical CPU sockets or physical cores, with or without HyperThreading.

Some vendors base their licensing on the VM's hardware, so it may be tied to the number of vCPUs or the VM's RAM. Ordinarily, vCPUs are presented as individual physical sockets, but an advanced VM setting allows them to appear as cores.

Some vendors license on the number of instances on site. Different rules may govern development/staging/test copies of software, and this is of particular interest because these tend to be more prevalent in virtualized environments. Applications can still use hardware dongles, either serial, parallel, or USB based, which have their own support challenges and can impact the VM's mobility within the cluster.

Just understanding Microsoft's licensing rules can be complicated, particularly because they change so regularly. Of particular note is the server licensing, which is based on physical hardware and largely ignores the ability to migrate VMs between hosts. This may change soon, as Microsoft adapts its own hypervisor's capabilities so it can migrate VMs as freely as VMware's hypervisor. Currently, a standard 2008 edition license covers one VM while it's on one host. As soon as the VM migrates to another host, another license is required. An Enterprise edition licenses four VMs. In a large cluster of hosts, you can expect your licensing to become rather costly. For this reason, many opt to use the Datacenter license, which allows unlimited copies per host. You need one 2008 Datacenter license per host, and all the VMs are covered. With the downgrade rights, this license also covers your 2003 instances.

Another Microsoft-specific licensing issue is the activation scheme used in nonvolume license agreement contracts. These VMs can trigger a need to reactivate the licensing if the hardware changes significantly. In these cases, it's always advisable to remove non-essential hardware, install the VMware Tools, and upgrade the VM hardware if required, before activating.

One last special licensing issue worth discussing is that of physical hardware–based licenses. Some vendors, notoriously Oracle, base their licensing on the number of physical CPUs on the host, regardless of the number of vCPUs allocated to the VM. In highly dense hardware, which is commonplace in vSphere hosts, a license for a four-way server may be required even if the VM has access to only one vCPU. Add an eight-way host server to the DRS-enabled cluster, and your licensing costs double, even though the VM remains with one vCPU. These kind of draconian licensing terms create situations where some companies have to physically remove CPUs from servers and isolate VMs on standalone hosts, just to ensure licensing compliance.

vSphere has the ability to create Host-Affinity rules, one of which is known as a *must* rule. This rule is designed specifically for strict licensing terms, and the following chapter explains how you can use it to lock VMs to a particular host. You'll need to check whether this technique is regarded as sufficient by your vendor to satisfy its licensing terms.

Disk Alignment

As disk volumes are laid out on physical storage, it's important that the partitions line up with the RAID chunks. Unaligned partitions mean that write operations are more likely to span several chunks, increasing latency and reducing throughput for those writes as well as subsequent reads. Disk alignment can be an issue for both VMFS datastores and guest partitions, as we mentioned in Chapter 6 when discussing VMFS volumes. Having both unaligned only exacerbates the issue, affecting I/O performance even more. However, as long as VMFS volumes are created with the vSphere client, they will be aligned properly. If a VMFS-5 volume was upgraded from an original VMFS-3 datastore (as opposed to a natively created one) and the VMFS-3 datastore was unaligned, then the resulting VMFS-5 datastore will remain unaligned. Deleting and re-creating this as a native VMFS-5 will ensure that it's aligned correctly.

When you're designing VMs, it's important to understand the impact of unaligned guest partitions and how to create partitions to avoid this potential performance drain. Aligned disks save I/O by reducing the number of times a stripe is crossed and minimize the metadata operations required.

Two settings are fixed when the disks are first initialized by their OS:

Starting Offset The first and most crucial is the starting offset. By default, on older OSes (Windows 2000 and 2003), this is set incorrectly, because these OSes reserve the first 63 sectors for the master boot record (MBR). When the disk is initialized and the first 63 sectors are reserved, they take up 31.5 KB of space, meaning every subsequent cluster is slightly offset from the sectors. From Windows 7 and 2008, this has been fixed, and all disks have an offset of 1,024 KB (although disks initially sized below 4 GB are offset by 64 KB).

Newer 1 MB Sector Drives

Until recently, hard drives were manufactured with 512-byte sectors, so eventually, all read and write operations were broken down into these sectors. SAN vendors vary in the stripe/chunk sizes they use, but commonly they're 32 KB, 64 KB, or 128 KB. With the introduction of GUID Partition Table (GPT), Windows (7 and 2008) and newer Linux partitioning tools have an offset of 1,024 KB. All arrays fitted with new 1 MB drives should work well with this new standard.

Cluster Size The second setting is the cluster size (or file-allocation unit) after the initial offset is applied. Most file systems use 4 KB or larger clusters, so most I/O is a multiple of that. However, applications typically generate certain types of I/O sizes, so you can often customize the partitioning to work as well as possible with the application's needs. You should also check the storage array's advice, because choosing the same cluster size as the chunk/stripe that the array uses maximizes the storage efficiency. But as a rule, ensuring that the offset and the cluster size are cleanly divisible by 4 KB (4,096 bytes) will give you the biggest benefit.

Linux users can use their disk-partitioning tool of choice, fdisk being the most popular, to correctly align the partitions. In Windows, use `diskpart.exe` (or `diskpar.exe` on Windows 2000) to create aligned partitions, and select the appropriate cluster size when formatting the partition.

The easiest way to handle this is to ensure that your template partitions are correctly aligned. That way, you don't need to worry about them in the future. Remember that if you create properly aligned small dummy disks in a Windows 7 or 2008 template, they will have 64 KB offsets even after you expand them beyond 4 GB. This isn't likely to be significant for performance per se, but it can cause inconsistencies across your environment.

If you already have a collection of existing VMs with misaligned disks, various third-party tools are available to perform alignment while preserving the data contained on the virtual disk. However, they all involve some downtime, so the recommendation is to concentrate your efforts on the most I/O-intensive disks. Also, be aware that many P2V solutions, including VMware's own Converter product prior to version 5.0, don't correctly align the disks they create.

An ongoing debate exists about the usefulness of aligning system boot disks. VMware published a white paper several years ago stating that you shouldn't attempt to align boot disks. The company has subsequently withdrawn that advice, but many regard aligning boot disks as unnecessary. Properly aligning a boot disk does take extra effort, because you need to create the partition in a surrogate guest. System boot disks don't normally generate much disk I/O, as long as you split the data off onto its own disk. However, you should certainly try to generate aligned boot disks for your templates.

Defragmentation

File fragmentation in VMs is a hotly debated topic. VMware has produced papers to suggest that fragmentation on VMFS volumes isn't a concern due to the relatively small number of files; the small size of I/O requests compared to the large blocks, which keeps most requests in the same block; and a sub-block allocator technique that reduces disk wastage and encourages file coalescing.

However, many regard defragmentation as an essential performance tune-up for some guest OSes, most notably Windows. Power users have long advocated a regular cycle of defrag jobs to prevent a gradual slowdown. Research continues on the effectiveness of defragging VMs, although this is often sponsored and run by software vendors that sell products associated with its remediation.

Several issues stand against defragmentation of VM disks, particularly those on SAN storage. Storage arrays use several controller methods to optimize I/O to the disks, and the performance impact of fragmented I/O across disks often isn't clear. Most controllers have memory caches to collate I/O before writing it to the disk and read-caching algorithms to grab the files in preparation for their being requested.

If guest OSes are defragged, thin-provisioned storage can lose its effectiveness as it writes blocks to new areas of the virtual disk. This causes the thinly provisioned disks to inflate. In the same way, linked clones in View and vCloud environments, and snapshotted VMs, will grow. If you use any VADP-enabled backup tools that use change block tracking (CBT), backup jobs will balloon after a defrag is run. Replicated storage, such as SRM VMs, will see a sharp increase in WAN data usage after a defrag is run because the moved blocks are interpreted as changes which need replicating. Some deduplication gains can also be lost every time a defragging job is run, and these gains aren't recovered until the deduplication is run again (deduping on primary storage is usually scheduled and not run in-line). Most storage vendors say that defragmenting file systems only increases disk I/O, works against their controllers' smarts, and does nothing to improve performance.

You may decide that you need to defragment some of your VMs. Generally speaking, third-party defraggers work better than the OS built-in ones, and some are starting to become more VMware aware. It's important that if you decide to run these jobs, they're offset and don't run simultaneously (which can put a large amount of I/O pressure on the storage). Defragmenting your templates before using them to deploy VMs may have the most potential benefit without causing an excessive strain. If you're contemplating guest defragmentation, you should run it on the most disk-performance-critical VMs.

Optimizing the Guest for the Hypervisor

Each guest has the opportunity to be optimized for running in a VM. This allows the guest OS and guest applications to run more efficiently and potentially faster, and it can reduce the load on the hosts, the network, and the storage.

Cutting the Fat

Basically, anything that's installed or running in a VM that isn't essential is a waste of host resources. When you multiply all those extraneous cycles by the number of VMs, even small inefficiencies can have an impact on overall performance.

CPU and Memory

You can do several things to reduce the CPU and memory load from VMs. First, optimize your antivirus software. Install the core virus-scanning engine in your VMs, and avoid the antispyware, firewall, and IDS extras that often come prebundled. You can switch to a host-based product that uses VMware's vShield Endpoint to protect the VMs without needing an agent installed in every guest. Also, follow the recommended exclusion lists for the OS and applications in the guest. This should reduce the overhead caused by files that unnecessarily burden a scanning engine. Reducing the CPU and memory allocation to VMs also reduces the memory overhead associated with each VM on the host and in the cluster.

Screensavers are waste of resources in a VM and should always be disabled, including pre-login screensavers. On Linux servers that don't need GUIs running, think about setting the default init level to not start an X Windows session—normally, run level 3.

As we already discussed, consider paravirtualized network and storage adapter drivers, because they can reduce CPU overhead.

Filter through all the running services, and disable anything from starting that isn't required. Look carefully at the default installed software and strip out anything not required. Also, examine the running processes in `top` or Task Manager to find unnecessary items.

Don't enable hot-plugging for the VM's CPUs or memory unless you're likely to use the feature, because hot-plugging reserves the resources in the guest OS for the maximum possible configuration. This will use additional CPU and can degrade the effectiveness of vNUMA calculations.

Disk

Optimizing the disk can mean two things in this context. First, and probably most important, you need to reduce storage I/O wherever possible. Avoid anything that's very disk I/O intensive if it isn't needed. It's also important to avoid too many I/O-intensive jobs happening at once. VDI deployments can suffer from *boot storms*, where all the desktops start up at the same time and the SAN load is overwhelming. These sorts of issues can also happen if scheduled tasks are set to run at the same time—for example, backups, antivirus scans, cron, scheduled tasks scripts or defrag utilities. If these are all set via the same policy or deployed from the same image, then they may all try to run at once. You need to figure out a way to offset the regular tasks so they don't thrash the storage simultaneously.

Second, you can optimize the disk-capacity requirements. Any software you can uninstall will reduce the amount of disk space needed. Store the install sets centrally, to avoid having copies on every VM. Enable circular logging and limit cache sizes, to avoid unnecessary build-up. Clear temporary folders regularly, and use disk-analysis tools to determine where space is being used.

Network

Network load is rarely a bottleneck for VMs; however, for particularly network-demanding loads, it may be possible to save bandwidth. DRS clusters have a function known as *affinity rules*. We'll discuss these in more detail later, but affinity rules tell DRS to try to keep certain VMs together on one host. There are a few reasons why you may want to keep VMs together, but a primary one is that doing so avoids sending the inter-VM network traffic out onto the LAN if the VMs are on the same port group. Two or more VMs involved in a multiserver application will send most of their traffic between themselves. Having all the VMs on one host can make most of that traffic happen locally.

Time Settings

VMs can't match a physical machine's ability to keep time. VMs need to time-share a host's hardware and can be interrupted by being suspended and snapshotted, which wouldn't affect regular computers. Two main options exist to keep a VM's time correctly synchronized. You can use the native guest tools, such as NTP in Linux guests or W32Time in Windows VMs; or you can use the VMware Tools' time synchronization. The VMware Tools have the advantage of knowing that VMs must catch up occasionally and be prepared for the clock to be off significantly. Also, the VMware Tools don't need networking to be configured in the guest because they work directly on the host.

Native NTP and W32Time generally work well in VMs and are usually turned on by default. VMware recommends that you use only one method of time sync; and because the native tools are normally running from the outset, this is how many VMs are configured. Additionally, some OSes and application software need to access the native time service, and sometimes they act as time sources themselves. Just be sure you don't set the VM to sync to its own hardware

clock. For the most accurate timekeeping possible, VMware recommends using the guest OS's native tools.

Different versions of Linux and Windows have had different approaches to time synchronization. Two KB articles cover the best practices for each, and you should consult them to ensure that your design incorporates this advice for the OSes you plan to deploy:

- Timekeeping best practices for Linux guests: `http://kb.vmware.com/kb/1006427`
- Timekeeping best practices for Windows, including NTP: `http://kb.vmware.com/kb/1318`

Clones, Templates, and vApps

Throughout this chapter, we've offered a great deal of specific advice on how to customize and tweak each VM. However, it's likely that in your environment, many of the VMs you plan to build will have very similar requirements. If this is the case, you can use a standard build. Doing so enables you to roll out new VMs much more rapidly.

Along with creating VMs more expediently, standardized builds automate many of the steps involved. You can also allocate specific permissions to users to control new VM deployments.

Standardizing is an important design tool. It simplifies decisions regarding how a large deployment of VMs should be created and configured. Perhaps more important, you're likely to have tens of VMs for every host and maybe hundreds of VMs per vCenter. The payback from a well-designed VM standard build in ongoing management and administration can be very worthwhile. Shaving 5 GB of disk space, reducing the default RAM by 256 MB, or halving the standard vCPUs from two to one can have a massive impact on your overall hardware needs. But being too stingy can mean hundreds of servers or desktops that continually hit performance and capacity problems.

Clones

A straightforward method of deploying a new VM without building it from scratch is to *clone* an existing VM. VM clones are exact copies: the configuration and the disks are duplicated. The only difference is that cloning forces a new VM name for the copy.

Clones are useful for creating quick point-in-time copies as backups. Unlike snapshots, which should be kept only for the short term, can affect performance, and require babysitting, cloning is a wonderful tool to grab a quick backup of the entire machine.

Having an exact copy can be a great way to replicate a production setup, which you can isolate and test. By cloning several VMs in an application setup, you can test changes on exactly the same setup and be confident that upgrades will go smoothly.

Be careful when you make clones, because having exact duplicates on the same network usually creates problems. Being identical, a clone has the same guest hostname and IP address as the original. Bringing a clone up on the network at the same time as the original can therefore create IP and name conflicts. You should ensure that the primary is turned off or disconnected from the network, that the clone is isolated, or that the clone is reconfigured while offline to change its hostname and IP address. Be mindful of tiered applications, because clones can cause inconsistencies on other connected servers if copies are brought online at different times.

You can make clones while a VM is turned on or off. A *hot* clone produces a VM that is crash consistent, so it's always desirable to turn off the VM first. However, if the VM to be cloned is critical, and an outage isn't possible, hot-cloning can still grab an image. Just be sure to test that it hasn't corrupted the OS, applications, or important data. Certain OSes and applications are more robust than others—for example, a database server is more likely to have problems recovering from a hot clone than a fairly static application server.

Another useful cloning facility is vCenter's ability to create a scheduled task to make a clone. If a VM is changing regularly or is particularly critical, vCenter can automate the process of cloning on a regular basis. This shouldn't replace any normal backup procedures you follow but can be useful to augment them as an extra safety net.

Finally, your SAN vendor may have tools for cloning VMs quickly on the storage. This allows for faster cloning, which takes the load off the host and can automate the provisioning of large VM cloning that is often seen in virtual desktop cases. These tools may be useful to consider not just for desktops but also when you need to roll out several VMs at once, such as when you're deploying new branch offices with a standard set of server services.

Templates

A *template* is a master copy VM that is specifically set aside to produce new VMs. A template is a special type of VM that can't be turned on or changed and is only visible in the VMs and Templates view in vCenter. This protection helps to keep a clean and standard image from which other VMs can be created.

VM templates streamline the process of creating new VMs. New VMs are available in minutes instead of after several hours; and often the burden of VM creation can be passed on to other members of the team, allowing greater flexibility and timely deployments.

Any VM can be converted or cloned into a template. However, because templates should be as immaculate as possible, it's advisable to build them from scratch so they're fit for their intended purpose.

Consider building a library of templates for your different OS and application needs. But don't expect to create a template for every scenario. Only include applications in templates if you expect to deploy a reasonable number of them and if their installation or customization is particularly onerous. Remember that it takes time to build up each template and that each one has a maintenance overhead associated with it. Even very large enterprises are likely to have only about a dozen templates, covering all the standard OSes and some of the larger applications.

Templates allow you to set permissions on how users can create new VMs, which again helps to control the types of VMs, the number of VMs, and what hardware can be allocated. For example, you may let only certain users deploy particular templates, limiting their OS or hardware choices. Template- and VM-creation permissions can also help curb the *VM sprawl* that is regularly seen when new vSphere servers appear.

You should consider how the use of templates will fit into your existing OS and application provisioning. If you have an automated process or set methodology for building physical servers and workstations, you can utilize those tools for building the base of your templates. This can lead to an even higher level of standardization and may reduce duplication between teams. However, be mindful that physical build tools often incorporate lots of drivers and specific details to deal with the abundance of hardware they may encounter. One of the advantages of vSphere VMs is its hardware abstraction, which means many of these features to deal with hardware and drivers aren't required.

Templates should be regularly updated. Applying OS and application patches and new antivirus definitions and policies minimizes the post-deployment steps and helps to reduce bandwidth. Rather than patch every new VM, why not patch one template? This means a regular cycle of patching should occur. To update or change a template, convert it into a VM, make the changes, and convert it back.

You should also think about how you'll push out new templates and template updates to multiple sites, if this is a requirement. Although hosts can use a common set of templates, those templates need to be stored on accessible storage. Normally, each site has its own shared storage; so if you're updating your templates regularly, you have two choices. You can either touch every template across every site with the same patches and updates, or you can keep one master set of templates that you update, and replicate these everywhere else. If you already possess some sort of SAN replication technology that only copies delta blocks, you can use it to minimize the bandwidth required to push out all templates again.

Guest Customization

A guest customization wizard automatically starts when you deploy a VM from a template. It asks all the questions required to customize the image and saves you from having to manually configure each piece in the guest. You can also use guest customization after cloning a VM, to make the clone safe to bring online alongside the original.

You can store a number of guest customizations, specific to each template or OS, which contain the majority of the answers needed in the wizard. Each of these guest customization specifications can be managed separately in the vSphere Client. Generally, one specification per OS is sufficient, because it's the license key that will separate them.

The source guest must have VMware Tools already installed and must match the correct OS that is specified in the VM's resources settings. The customization searches for the OS on the first disk, which must be SCSI-based. It won't work if it can't find the OS files it expects. It basically mounts the virtual disk after deployment and makes the changes it needs directly on the guest's file system. Only certain Windows and Linux guest OSes are supported, so check the latest VMware documentation to ensure that your guests can be customized with this tool.

Sysprep

Sysprep is Microsoft's systems preparation tool, which you can use to make Windows images and generalize them. Doing so clears the settings that make each Windows installation unique and creates new hostnames, SIDs, and driver caches.

vCenter can take advantage of these tools during a guest customization if the tools are copied onto the vCenter server. Different versions of sysprep exist for different Windows OSes, so one must be uploaded for each version you expect to deploy. With Windows 2008/Windows 7 and beyond, you no longer need to install external sysprep tools. Normally, when you use sysprep with a disk-imaging system, you need to prepare images, seal them, create answer files, and so on. However, the guest customization process automates these steps and prompts the user during the template wizard.

Preparing a Template

When you're creating a VM to make into a template, start with a fresh image. Create the VM based on the correct OS, and then configure the hardware components and advanced options to

suit the template. Follow the advice from throughout this chapter and think about the purpose of the VMs being deployed from this template. Try to pick what you consider the minimum hardware requirements.

You don't want to initially overprovision, or you'll waste valuable hardware resources for every VM that doesn't use them. But be realistic about the minimum OS and application requirements for the template, particularly memory and disk space. Underprovisioning will lead to extra work when every image has to be changed after you deploy it. Always try to use a single CPU, unless the template is specifically for an application that will always require more than one. Remember that newer OSes require different minimum levels: just because you could get away with 256 MB of memory and a 10 GB hard drive for a Windows XP VM, that doesn't mean a base Windows 2008 template should be the same.

For very large environments, consider a set of hardware tiered templates for the most common OSes. This not only saves you from having to change the hardware setup on each of these VMs after they're built but also helps to standardize the result.

Remember to go through each template and remove or at least disable the hardware that isn't required. Items such as floppy drives and serial and parallel ports are rarely needed and place a tax on the host hardware for every VM deployed. This is also a great opportunity to take advantage of enhanced hardware options such as VMXNET 3 and PVSCSI adapters.

When you're installing the OS and the applications, try to make the image as clean as possible. Make sure it's built from scratch, explicitly for its purpose. Follow the disk-alignment advice from earlier in the chapter to make sure all disks are aligned from the outset. Include all the tools that need to be installed on every VM, such as antivirus software, admin tools, and company templates. Apply OS or application customizations that are individual to your organization, such as specifying wallpaper, creating specific directories, changing drive letters, making registry settings, and modifying profiles.

Avoid including anything that won't be needed. For example, try to avoid the temptation to copy all the application-install files locally, because they will be duplicated every time. Keep one copy centrally. Disabling hibernation on Windows guests removes the very large `hiberfile.sys` file; this feature won't be used. Actively strip out software and stop services that the VMs don't need. Many of the basic premises that you follow when creating a standard workstation image for corporate desktops also apply here.

After the VM is built and the OS and applications are installed, be sure you patch the OS with all the latest updates and install VMware Tools. Shut down the VM, and make any final hardware changes. You may want to think about setting the network adapter to a port group with no external access or to a clean subnet. Then, you can deploy each VM, patch it, and apply its antivirus definition update before attaching it to the corporate network.

Virtual Appliances

Virtual appliance is a generic term for VMs that were built specifically for a particular use case. These are built by third-party software vendors, have stripped down JeOS guest OSes, and are usually patched and maintained as single entity (treating the OS and application[s] together).

Using virtual appliances offers significant advantages. Primarily, they simplify the deployment of applications for end users. A virtual appliance can usually be deployed and configured in less than 30 minutes. Software vendors like the format too: it reduces the initial support calls associated with users installing applications, and the suggested virtual hardware configuration is more likely to be adhered to if it's plugged into the appliance from the outset. Patching

and updates can be controlled via the ISV. The OS has a much smaller footprint, with only the necessary tools included, so there should be fewer patches; and the software vendor can test all patches and updates for compatibility before making them available.

Giving the ISV such control over the design and maintenance of its VMs has lead to criticism in some cases. Often the VM's OSes aren't as minimal as they could be; when the release of OS patches falls far behind that of the OS vendors, then vSphere administrators are left with the dilemma of leaving known security vulnerabilities to stay compatible with the appliance's vendor, or potentially breaking the application (and almost certainly the support).

OVF Standard

VMs can also be distributed in a standard format package known as Open VM Format (OVF). This is a template format created by the Distributed Management Task Force (DMTF) standards group to allow an interchangeable VM, which can be used by software companies to distribute VMs. The vSphere Client can import and export OVF files (or OVA files, which are tarball archives of OVF files encapsulated in one file).

OVF files tend to be used by software vendors to distribute hypervisor-agnostic VMs. These VM appliances normally include tuned OSes and applications that are easy for customers to deploy and easy for software companies to support.

One of the current limitations of the 1.0 standard is the lack of definition for virtual disks. OVF files can include disks in several formats, so non-VMware disk files need to be imported via VMware Converter first.

vApps

vApps are vSphere containers for one or more VMs that allow those VMs to be treated as a single entity. vApps are usually third-party virtual appliances that have the additional vSphere wrapping already included, to take advantage of some of the extra functionality available such as creating service dependencies. You can add vApp features to your own VMs.

vApps can be powered on, powered off, and cloned in one action. You can import entire vApps, which collectively run an application or service, in much the same way that OVF files are distributed for single VM applications. vApps are a potentially useful packaging technique, although they have been slow to amass large adoption.

Virtual Machine Availability

When you're designing VMs to provide greater uptime, remember the difference between VM-level clustering, OS-level clustering, and application-level clustering. They all provide alternative types of failover, but each is aimed at protecting different parts of the stack.

Host-level clustering ensures that the VM is turned on and recovers it if it unexpectedly powers off for any reason (for example, if a host fails). Guest OS-level clustering checks that the OS is up and responding. Common techniques include checking network pings or the VMware Tools' heartbeat. Application clustering monitors application services to ensure that certain software is responding to requests as it should.

vSphere offers a multitude of high-availability options to protect your VMs from unexpected failures. In addition to the built-in solutions, several third-party VM failover and clustering methods exist. Table 7.4 shows how each option protects against the different types of failures.

TABLE 7.4: VM availability options

PROTECTION MEASURES	HOST FAILURE	GUEST OS FAILURE	APPLICATION FAILURE
HA failover	✓		
Host VM startup	✓		
DRS affinity rules	✓		
HA VM monitoring		✓	
HA application monitoring			✓
Fault tolerance	✓		
Microsoft failover clustering	(If CAB or n+1)	✓	✓
Microsoft NLB	(If VMs split with anti-affinity rules)	✓	
Microsoft SQL 2012 AlwaysOn	(If VMs split with anti-affinity rules)	✓	✓

vSphere VM Availability

Most high-availability approaches included in vSphere are functions of DRS and HA clusters, and as such we'll discuss them in much greater length in Chapter 8, "Datacenter Design." Because they will affect your VM guest planning, we'll briefly explain each tool's use and how it can provide a more resilient base for your VMs:

HA Failover HA-enabled clusters protect VMs from host failures. If the cluster determines that a host failure has occurred, the VMs that were running on the failed host are automatically powered up on alternate hosts.

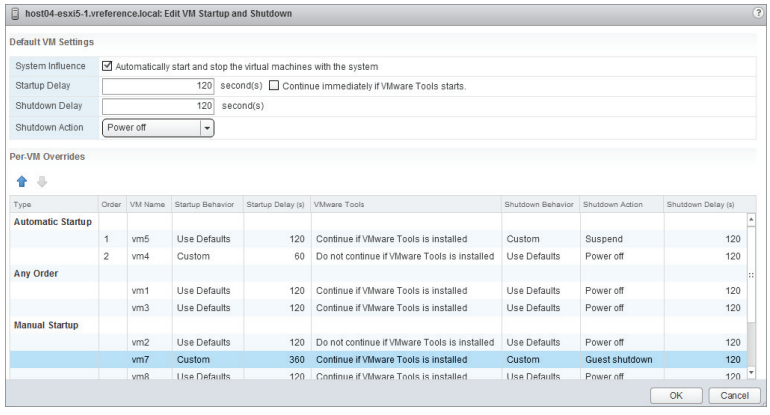
VM Startup On each host, you can set the VMs to start up (and shut down) automatically, controlling their startup order and a delay between each VM's startup. Figure 7.16 shows these settings. This automatic recovery for a host can be very useful in single-server, local storage sites or companies not licensed for HA. When a host is added to an HA-enabled cluster, this option is disabled.

Affinity Rules Affinity rules and anti-affinity rules are functions of DRS and aren't strictly for HA purposes. However, you can use anti-affinity rules to ensure that load-balancing VMs are spread across more than one host. This means that if a host fails, not all VM nodes are lost at the same time. Host affinity rules can be used to keep VMs aligned to certain hosts, allowing you to separate critical-paired VMs across different racks or blade chassis.

Affinity rules can keep VMs together and can be used in availability planning. Keeping VMs together may seem like a contradictory way to provide better redundancy. But if you consider

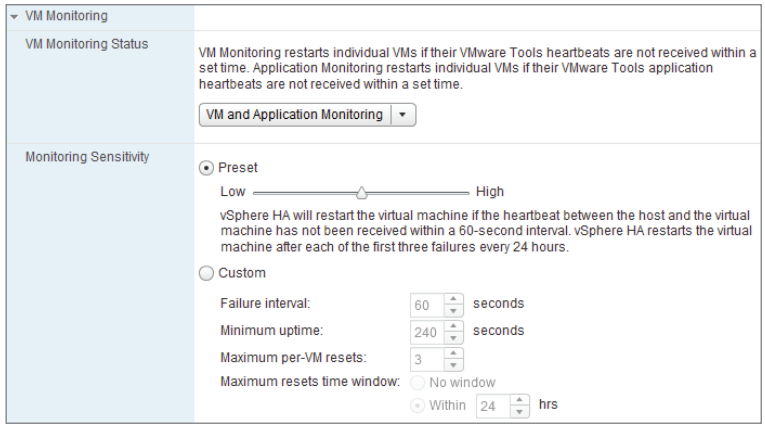
a situation in which multiple VM instances need to be up for an application to work properly, then splitting them apart won't help provide redundancy. Keeping them together on a single host lessens the chance that one of the VMs will be on a host that fails.

FIGURE 7.16
VM Startup /Shutdown



VM Monitoring VM monitoring is a function of an HA cluster. It uses the VMware Tools' heartbeat to check that the OS is still responding, and thus helps to protect against a Blue Screen of Death (BSOD) or kernel panics. The HA daemon passes the heartbeat, which by default is sent out every second to the vCenter server. If a VM's heartbeat is lost, it checks for network and storage I/O to prevent false positives. If nothing is received for a set period, vCenter resets the VM. Different levels of sensitivity are available; you can configure them using the slider shown in Figure 7.17.

FIGURE 7.17
VM monitoring



Application Monitoring vSphere has an application monitoring control that works in the same way, but for known applications. This functionality uses a VMware API, which software vendors must incorporate in their application to allow vCenter to provide monitoring.

Fault Tolerance FT protects against failed hosts by keeping a second image of the VM running on a second host in the same HA cluster. FT impacts both DRS and HA designs; for this reason, a full examination is left to the next chapter.

However, for VM design, it's important to understand that FT places a number of restrictions on the VM being protected:

- It must be on shared storage that's visible to hosts in the HA cluster.
- Only single vCPUs can be protected.
- vSphere Enterprise or Enterprise Plus licensing is required.
- You can't use physical RDMs.
- No snapshots are allowed.
- You can't use Storage vMotion.
- You can't use NPIV.
- VMs with EFI firmware are not allowed.
- FT converts all disks to eager-zeroed thick format.
- You can't use hot-plugging devices.
- No USB devices, 3D video devices, sound devices, serial ports, or parallel ports are allowed.
- No physical or remote CD/floppy devices are allowed.
- Only some guest OSes are supported (see `http://kb.vmware.com/kb/1008027`).
- FT sets the VM's memory reservation equal to its RAM setting to prevent ballooning or swapping.

If you plan to protect a VM with FT, you should look carefully at how it will impact your VM design.

Third-Party VM Clustering

In addition to the host, DRS, and HA tools providing higher availability for VMs, you can use several in-guest clustering techniques. Guest OSes often have built-in clustering features, and many more third-party solutions are available. In this chapter we'll look at the two most common methods used, both for Microsoft OSes: failover clustering and NLB.

Microsoft Clustering

Microsoft Clustering Service (MSCS), or Failover Clustering as it's now known, is a widely used clustering technique for Microsoft servers. It's relatively complicated to configure, particularly in the VMware world. Many alternative options are now available to provide high-availability, but MSCS is so heavily ingrained as a solution that it's still a popular choice.

Microsoft clustering is available on Windows 2003 and 2008 in their Enterprise and Datacenter editions, and Windows 2012 from the Standard edition upwards. It supports 8 nodes on 2003, 16 nodes on 2008 and up to 64 nodes in 2012; but at the time of writing, support for 2012 Failover Clustering had yet to be announced by VMware.The clustering is limited to only 2

nodes in vSphere 5.0. although vSphere 5.1 has raised the supported limit to 5 nodes. Windows 2000 MSCS VMs were supported in vSphere 4.0 but not in version 4.1, so before upgrading hosts to a newer version you must upgrade the guest OSes.

You can configure MSCS in vSphere three ways:

Cluster in a Box CIB is the configuration when both MSCS servers are VMs running on the same vSphere host. Although it's marginally easier to set up, this configuration is somewhat pointless because it doesn't protect against a hardware failure of the single vSphere host. Even as a test setup, it's unlikely to mimic a production situation sufficiently. VMDK files are recommended for the shared disks, although virtual RDMs give you the option to move to a CAB solution in the future if a second host becomes available.

Cluster Across Boxes CAB describes the situation when MSCS is deployed in two VMs, and the two VMs are split across two different hosts. This protects against a single host failure. Physical RDMs are now the recommended disk option with vSphere.

Physical to Virtual Physical to virtual (n+1) clusters allow one MSCS cluster node to run natively on a physical server while the other runs in a VM. This configuration is popular when a physical server is still deemed a necessity, but failover can be handled by a VM. A physical RDM is required in this instance.

MSCS Limitations

MSCS has the following design limitations when run on vSphere:

- Windows 2000 VMs are no longer supported from vSphere 4.1 onward; only 2003 SP2 and 2008 R2 are supported.
- DRS/HA cluster compatibility requires at least vSphere 4.1.
- Only five node clusters are possible (only two nodes in vSphere 5.0).
- You must use at least VM hardware version 7.
- Shared disks need to be the thick provision eager zeroed type. See Figure 7.14 for the check box to enable this setting when creating disks in the client.
- Only Fibre Channel SANs are supported. iSCSI, Fibre Channel over Ethernet (FCoE), and NFS shared storage aren't supported.
- There is no support for vMotion, FT VMs, NPIV, and round-robin multipathing.

Disk Types

Table 7.5 shows the different disk types that are supported for each configuration:

TABLE 7.5: Microsoft clustering disk options (items in bold show VMware's recommended option)

	VMDK	VIRTUAL RDM	PHYSICAL RDM
Cluster in a box (CIB)	**Yes**	Yes	No
Cluster across boxes (CAB)	No	Yes (not 2008)	**Yes**
Physical and virtual (n+1)	No	No	**Yes**

SCSI Controller Settings

SCSI controller settings create the most common design misunderstanding for MSCS VMs. There are two different settings, which sound very similar:

- Disk types (selected when you add a new disk): VMDK, virtual RDM (virtual compatibility mode), or physical RDM (physical compatibility mode)
- SCSI bus-sharing setting: virtual sharing policy or physical sharing policy (or none)

These settings are distinct. Just because you choose a virtual RDM doesn't mean the SCSI controller should necessarily be set to Virtual.

The SCSI bus-sharing setting is often missed, because you don't manually add the second controller (you can't). You need to go back to the settings after you've added the first shared disk. There are settings here:

- None: disks that aren't shared between VMs. This is used for disks that aren't shared in the cluster, such as the VM's boot disks. This is why shared disks must be on a second SCSI controller.
- Virtual: only for CIB shared disks.
- Physical: for CAB and n+1 shared disks.

Design for an HA/DRS Cluster

Since vSphere 4.1, MSCS can be members of HA and DRS clusters. However, to make sure the HA or DRS clustering functions don't interfere with MSCS, you need to apply special settings:

DRS-Only Clusters vMotioning MSCS VMs isn't recommended, so you need to set the VMs with an individual DRS setting of *Partially Automatic.* To ensure that all the cluster's affinity rules are considered *must* rules, you can set the advanced DRS setting ForceAffinityPoweron to 0 (zero).

For CIB VMs, create a VM-to-VM affinity rule to keep them together. For CAB VMs, create a VM-to-VM anti-affinity rule to keep them apart. These should be *must* rules. n+1 VMs don't need any special affinity rules.

HA-Enabled Clusters To run MSCS VMs in an HA cluster, you need to use affinity rules. This means you must also enable DRS and implement the DRS VM-to-VM rules. HA also needs additional Host-to-VM rules, because HA doesn't consider the VM-to-VM rules when recovering VMs.

CIB VMs must be in the same VM DRS group, which must be linked to a host DRS group containing just two hosts, using a *Must run on hosts in group* rule. CAB VMs must be in different VM DRS groups, which must be linked to the different host DRS groups using a *Must run on hosts in group* rule. Again, n+1 VMs don't need any special affinity rules. Figure 7.18 shows how these VMs should be configured in an HA cluster.

Microsoft NLB

Microsoft Network Load Balancing (NLB) is an IP-based clustering technique included in Windows 2000 Advanced Server, 2003, and 2008. All the hosts receive the requests, and a special

network-driver algorithm decides which host should respond while all other hosts drop the request. An NLB cluster can support up to 32 nodes.

The NLB servers exchange a heartbeat to detect failures, and they redistribute requests to servers that continue to operate. NLB has two modes:

Multicast Multicast mode adds a Layer 2 multicast address to the cluster adapter. However, some routers and Layer 2 switches don't support this, and you must add a static ARP entry to map the cluster IP address to the MAC address.

Unicast Unicast has the advantage of working with all Layer 2 devices, but it causes all ports to be flooded with NLB traffic. Unicast NLB VMs need a second vNIC, and you must set the port group on the vSwitch to accept forged transmits.

VMware recommends that you use multicast mode whenever possible, because you don't need to make any changes on the hosts, and no broadcast port flooding will occur.

In a DRS cluster, be sure to set anti-affinity rules to try to protect NLB VMs against host failures.

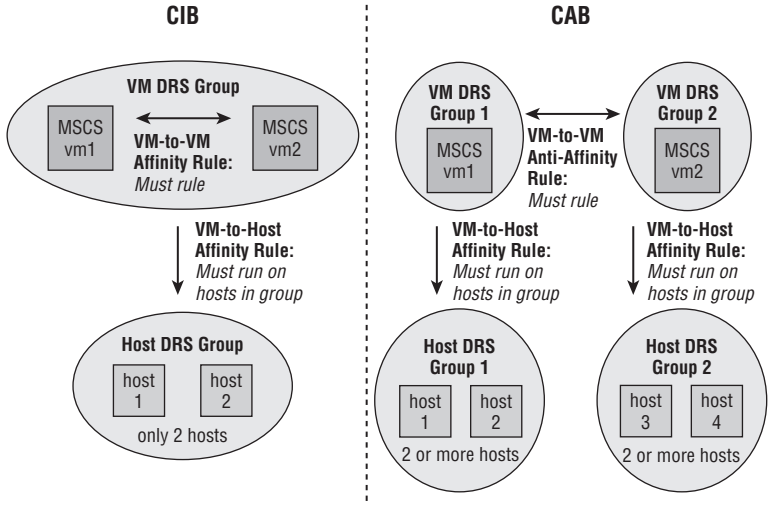


FIGURE 7.18 MSCS VMs using VM/Host DRS groups in an HA cluster

Microsoft Application Clustering

Many of Microsoft's latest core applications have their own built-in application clustering techniques. Two of the most common (and useful) are Exchange 2010 database availability groups (DAGs) and SQL 2012's new AlwaysOn Availability Groups. Both ultimately depend on Windows Failover Clustering, but because they don't rely on a quorum disk like classic failover clustering, they aren't constrained by the same restrictions. The file synchronization happens at the application layer. DAG and AlwaysOn VMs can be treated like any other VM now.

Remember that these VMs are normally very large in size compared to other application servers, very visible to users if they drop offline, and normally considered business-critical these days. You'll probably want to treat them with kid gloves and protect their resources to avoid any form of contention.

Both Exchange DAGs and SQL AlwaysOn VMs are very sensitive to the slightest drops on their network heartbeat. These VMs often have large amounts of memory allocated to them, and they're greedy and keep that memory active. Whereas most VMs drop only one or two pings during a vMotion stun, these large VMs can drop for longer. This is particularly the case if you don't have access to 10GbE networking. To prevent vMotions from causing false positives and initiating an application failover, some advanced settings are available to extend the timeouts.

vCenter Infrastructure Navigator

VMware introduced Infrastructure Navigator as part of the version 5 release of the vCenter Operations Manager suite. It's included as a part of the Enterprise and Enterprise Plus licenses. It fully integrates with the vCenter Web Client in vSphere 5 and adds detail to the VMs' summary page and creates a Navigator-specific tab for each VM. The Navigator tool can be used at many object levels in vCenter, but at its heart it's a tool to examine VM interdependencies. Because it uses the VMs' VMware Tools to gather the required information, no additional agents are needed. Figure 7.19 shows the complex relationship diagrammed between VMs and some of the dependency information that can be garnered from it.

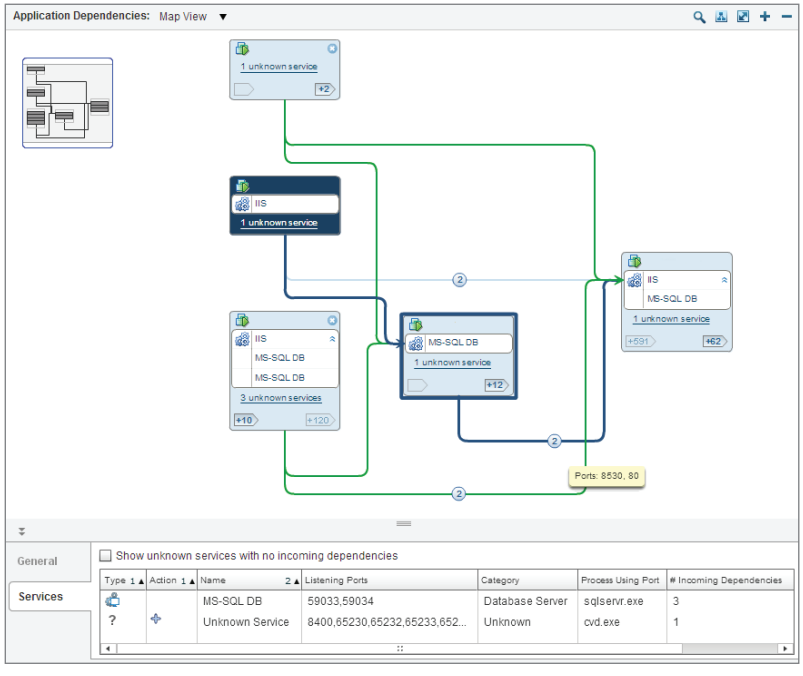


FIGURE 7.19 Infrastructure Navigator displaying the interdependencies between VMs

Infrastructure Navigator automatically discovers in near real-time the relationships between VMs. It identifies services running in the VMs and matches them to a known list of common applications. It can map out all the incoming and outgoing traffic paths between VMs and even shows the ports that the services are using. Infrastructure Navigator also matches all the interdependent VMs to other vCenter objects so it can identify when key infrastructure pieces

are running on common or disparate hosts, subnets, or LUNs. The VM awareness is mapped and also displayed as tables, allowing you to sort and search for key applications, services, and ports.

The ability of Infrastructure Navigator to map out complex application schematics in an ever-changing environment can be a boon for application architects and IS managers. You can use it to help troubleshoot application issues, plan changes or upgrades, and understand the impact of interlinked objects. It's particularly useful in SRM deployments because it can identify how applications are mapped to SRM protection groups, to ensure that all the VM components required for a business service are part of the recovery plan. Infrastructure Navigator can also assist in making sure application start-up dependencies are addressed properly and that cluster affinity and anti-affinity rules are correctly set. Although Infrastructure Navigator isn't part of your VM design per se, in large complicated environments it's becoming an indispensible tool to manage and design your meta-VM landscape.

Summary

VM design is a seminal topic that all too frequently doesn't receive the sort of attention it should during a vSphere design. Its importance can't be stressed too heavily. A strong VM design looks carefully at each VM component, analyzing what can benefit the environment and what is superfluous.

Good VM design makes use of the underlying hardware, identifying the needs of the VMs, and understands how to create a successful model. Undoubtedly the biggest common mistake is treating VMs like physical machines, undervaluing the gains you can make in customizing them and overprovisioning the hardware. Most physical hardware is more than sufficient for most purposes, so there is little benefit in stripping out hardware and software and changing the base system configurations that are offered. However, in vSphere, despite the defaults that give you workable VMs, when you hope to densely pack the host with tens of VMs, that extra 10 or 20 percent of performance can mean significantly more VMs.

VM storage is of particular note, because this is often where performance bottlenecks occur. CPU and memory are also critical because they lay claim to the server hardware and dictate how many VMs you can squeeze onto each, and how much additional capacity you have for growth and redundancy. Overprovisioning some VMs will only have a detrimental effect on the other VMs.

In addition to the VM's hardware, its guest OS, applications, and running processes are ripe for trimming. As we discussed regarding host hardware in Chapter 4, you may be able to scale VMs out instead of upward, avoiding very large VMs.

All these constituent parts can be melded into a small selection of templates. Templates give you the power to effectively enforce your discretionary work in a scalable fashion. It wouldn't be feasible to maintain this level of thoroughness for every VM, but templates provide the mechanism to create this initial consistency.

As you've seen, various options exist to protect your VMs. The next chapter looks at how vCenter design and cluster techniques can make the most of these VM designs, to effectively and efficiently spread resources across hosts, maintain high availability, and organize hosts and VMs.

Chapter 8

Datacenter Design

vCenter functionality gives rise to many design possibilities, particularly the combination of vCenter and the Enterprise and Enterprise Plus licensing features. This chapter explores some of those options, such as the ability to design-in redundancy and share resources efficiently between hosts to offer a fair proportion of hardware while enforcing VM protection. Unused servers can be shut down to reduce power costs, and VMs can automatically balance among hosts in concert with rules to apply control where required.

The chapter looks at the ways in which vSphere objects are flexible enough to create designs optimized for your particular environment. You can control security permissions, collectively manage objects, and monitor and schedule tasks in your datacenters easily and effectively.

This chapter will cover the following topics:

- How objects in vCenter interact and create a hierarchy
- Why clusters are central to your vCenter design
- Resource pool settings
- Using distributed resource scheduling to load-balance, save power, and control VM placement
- How high availability recovers VMs quickly when host failures occur
- Using fault tolerance to provide maximum VM availability

vSphere Inventory Structure

The vSphere Web Client offers the same Home dashboard that is available in the Windows-based vSphere Client. This familiar vCenter hub is a collection of icons organized by function. The first functional area is Inventories, as shown in Figure 8.1.

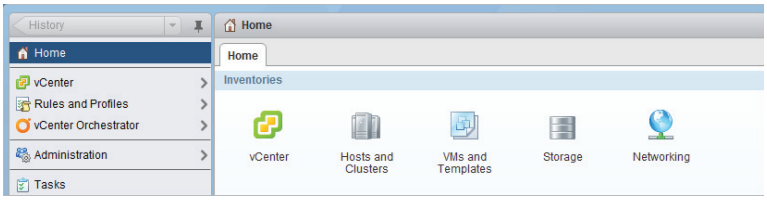


FIGURE 8.1 vSphere Home dashboard

The Windows Client's inventory has four different links, along with a search option. The Web Client replaces the search option with a link to the top of the vCenter hierarchy view. In the Web Client, there is always a search field in the upper-right corner of the browser window no matter where you are. The four common inventory links are as follows:

- Hosts and Clusters
- VMs and Templates
- Storage (labeled Datastores in the Windows Client)
- Networking

These views present items from the inventory, each following the same basic structure but still capable of including its own objects. Although the object types can differ, the hierarchical elements are common to all the views.

The relationship between the elements differs depending on how you use the Client. The Windows Client can connect directly to hosts or vCenter Servers, but the Web Client can still only connect to vCenters, highlighting one of the remaining compelling reasons for an administrator to keep a copy of the Windows Client installed on a workstation. As discussed in Chapter 3, "The Management Layer," you can also connect vCenter Servers via Linked Mode, which aggregates multiple instances.

The inventory structure creates a delineation that serves a number of purposes. It helps you organize all the elements into more manageable chunks, making them easier to find and work with. Monitoring can be arranged around the levels with associated alarms; events trigger different responses, depending on their place in the structure. You can set security permissions on hierarchical objects, meaning you can split up permissions as required for different areas and also nest and group permissions as needed. Perhaps most important, the inventory structure permits certain functionality in groups of objects, so they can work together. This chapter will discuss some of that functionality in much more depth.

The vSphere Web Client in 5.0 had fairly rudimentary options with a small tab for each of the four inventory views that could quickly help navigate from one section to another. The Web Client in 5.1 enhanced the traditional hierarchical view with Inventory Lists, as shown in Figure 8.2. The four views are still available under the Inventory Trees section, but the new Lists section provides jump-points to quickly get to a listing of any of the particular inventory objects.

Figure 8.3 shows the relationship of each of the Inventory Lists jump-points to the overall vCenter hierarchy. These convenient links make managing the environment quicker because objects can be found more directly. To design the most effective structure for vCenter, it's still fundamental to understand the basic relationships between and within the available objects. Figure 8.3 will help you understand the options available in your design.

Each of the following sections describes the main vCenter structural components that are involved in the design of vCenter objects.

Inventory Root

The inventory root object is a special type of folder. It's the starting point of all objects in the Client and the top level for all permissions and alarms. Every object and function cascades from this one element, and their interrelationships can always be traced back to the root.

When the Windows Client is connected directly to a host, the inventory root is the host. When the Client is pointing to a vCenter Server, the root object is effectively the vCenter Server.

If the vCenter Server is part of a Linked Mode group, you'll see several root inventory objects, one for each vCenter. The vCenter root object can contain only folders and datacenter objects.

Folders

Folders are a purely organizational element, used to structure and group other elements. They let you manage objects collectively, such as applying permissions across a group of datacenters. Folders are a vCenter feature, so they're available only when the client is connected to a vCenter Server instance. Folders can also contain child folders, to allow for more complex organization.

Folders can only contain objects of the same type; you can use them at various levels and in different views to consolidate like items. Folders can contain subfolders, datacenters, clusters, hosts, VMs, templates, datastores, or networks. Each inventory view can have its own sets of folders, so the Hosts and Clusters view can have a folder structure that's different than that of the VMs and Templates view. But folders off the root object are common to all views.

They're a very flexible way to organize vCenter items without having to adhere to the normal rules that limit other objects.

Datacenters

Datacenters are the basic building blocks of a vCenter structural design. They make up all the objects needed for virtualization and are visible in all four Inventory views. Folders are useful for organizing elements, but datacenters are always required because they directly house the hosts, clusters, and VMs.

Datacenters are a vCenter-only construct; they aren't available to stand-alone hosts and aren't visible when the Windows Client is directly connected to hosts. They're the boundary to vMotions, which means your datacenter design should consider the network and storage topology, because this is often what separates one datastore from another.

Remember that despite the moniker, a datacenter doesn't necessarily have to align with a physical datacenter or server-room location. However, network and storage connections do tend to be determined by geographical location, so it's common to see this parallel used.

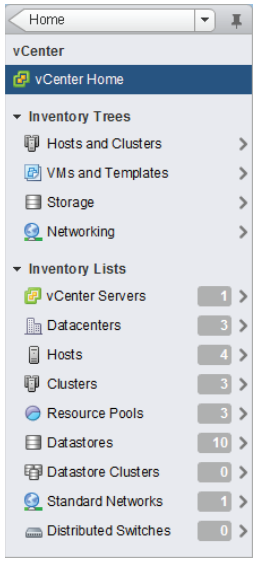


FIGURE 8.2
vCenter Home Inventory Trees and Lists

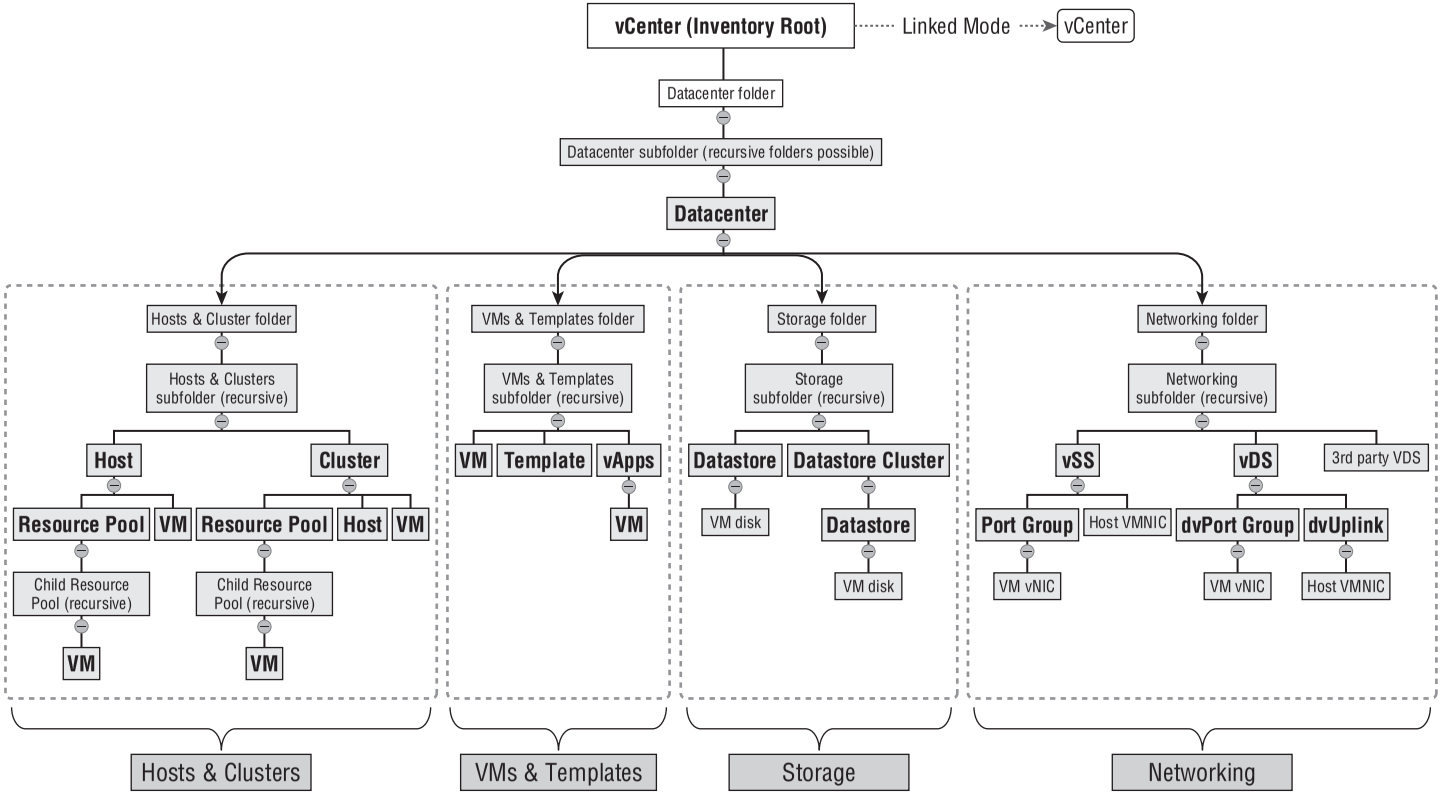


FIGURE 8.3
Hierarchy of vCenter Inventory Lists objects

Clusters

A cluster is a vCenter-only element that collects hosts together from a single datacenter to aggregate compute resources. A datacenter can contain multiple clusters. A cluster groups hosts to provide additional functionality, allowing the hosts and VMs to work together. Clusters can have hosts, resource pools, VMs, and vApps as child objects.

Cluster functionality is further described in later sections of this chapter.

Resource Pools

Resource pools can exist on vCenter instances in clusters or under hosts. They let you divide and apportion the available CPU and memory resources from a cluster or host. You can further subdivide resource pools by using subordinate resource pools. VMs draw their CPU and memory entitlements from the resource pool in which they reside.

To use resource pools on a vCenter cluster, the cluster must have distributed resource scheduling (DRS) enabled. You shouldn't use resource pools as a substitute for folders to organize VMs, but instead use folders in the VMs and Templates view.

Resource pools are examined in more detail later in this chapter.

Hosts

A host represents a physical server running the vSphere ESXi hypervisor, or potentially a pre-5.0 ESX host. Both types can coexist in a cluster, although the host's CPU determines compatibility with other hosts in the same cluster.

The host is the top-level root object when the vSphere Windows Client is logged in to a host directly. Hosts can have resource pools and VMs under them in the structure.

Virtual Machines

VMs can be seen in all four Inventory views, but VM management tends to revolve around the Hosts and Clusters view and the VMs and Templates view. The former concentrates on a VM's physical hardware location: the host, cluster, and resource pool in which it resides. The latter shows the logical groupings you create in the datacenter to organize the VMs.

Templates

A template is a special type of VM that's only displayed in the VMs and Templates view. Because templates are a vCenter-only feature, they aren't visible when the Windows Client is connected directly to a host. Templates were discussed in Chapter 7, "Virtual Machines."

Storage

The Storage view shows all available storage for each datacenter. It collates all local host storage and shared SAN and NAS datastores, allowing common management for all types. You can organize this vCenter-only view using datastore folders to help pool all the sources into logical groupings.

vSphere 5.0 introduced datastore clusters, which were examined in depth in Chapter 6. In a similar fashion to host clusters, datastore clusters aggregate resources for VMs. Folders are still useful, providing organizational structure without the resource implications associated with datastore clusters.

Networks

In the Networking view, all the port groups and dvport groups are collected under each datacenter. As long as different hosts' port groups are named identically, this view treats them together, in the same way that vMotion understands them to be the same. This lets you manage them collectively so you can apply permissions and alarms. You can use folders in the Networking view to group and split port groups, although this view is available only through vCenter connections.

vCenter 5.1 Tagging

Although not a hierarchical element in a vCenter, the tagging feature in vSphere 5.1 can help administrators organize and find objects more effectively. Tagging and its precursor, custom attributes, let you mark objects with custom flags. Tagging objects enables them to be grouped arbitrarily and can provide the basis of searches. If standard inventory structures don't provide the desired detail, then perhaps additional tags can add the required abstract context.

Why and How to Structure

From a design perspective, it's important to reiterate the advantages of using hierarchical structuring, especially because the vSphere 5.1 Web Client provides such convenient access to individual areas that it can seem less important:

- Enables certain functionality via the grouping of similar elements
- Aids management
- Allows granular permissions
- Lets you monitor events and tasks at different levels
- Lets you set alarms and responses based on the structure
- Lets you align scheduled tasks to the same groupings

You can organize your vCenter pieces in several ways. The most successful hierarchical designs usually closely follow the business. This makes it easy for everyone to understand the structure, to apply the appropriate importance and focus to key elements, and to provide suitable resources.

Most designs follow one or more of the following approaches:

Geographical This structure is split according to the physical location of equipment.

Departmental This structure is appropriate if a business's IT equipment and staffing are delivered based on department.

Business Costing Structure This hierarchy works if the OPEX and CAPEX are split into different areas and chargeback figures are subsequently applied.

Business Function This structure is appropriate if a business is naturally split around the products and services it provides.

Projects Certain projects may fund and resource separate parts of the infrastructure.

Priority vCenter elements can enable redundancy and provide resource allocation. If some VMs require different service-level agreements (SLAs) or performance, then this can dictate structure.

Connectivity The link speed and latency of both networks and storage may influence structure.

Equipment Access to different server equipment (such as Intel or AMD), network switches, and shared storage can split infrastructure apart.

Licensing In some cases, application software licensing may require segregated resources.

Usually, businesses use a hybrid solution consisting of several elements from this list. The hierarchy itself is normally scaled on the size of the business's VM deployment. A small company with one or two hosts may have a practically flat structure. However, a large organization can have many tiers. For example, an enterprise may have several linked vCenters to create a hard permissions division, each with a layer of folders to group several datacenters together; more folders under each datacenter to consolidate some hosts and clusters while segregating others; and an entire tree of resource pool levels to tightly control allocation. Figure 8.4 demonstrates some of the structural elements and options you may encounter.

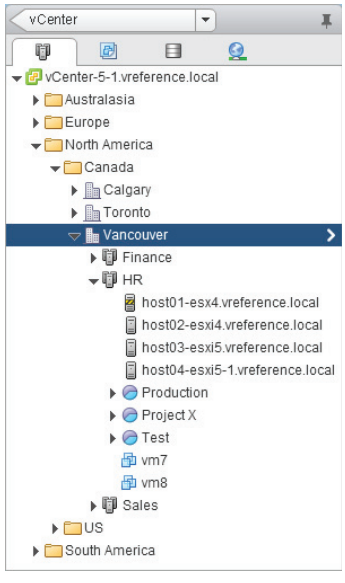


FIGURE 8.4
vCenter structural example

Clusters

vCenter clusters group hosts together for two main reasons. Clusters allow the hosts to work together, enabling the use of both high availability (HA) and DRS resource management. These

two cluster functions are examined more closely in their own sections later in the chapter. But it's worth considering the cluster itself as a vehicle to support hosts and VMs.

Although this isn't a strict requirement of a cluster, the cluster's power is realized when the hosts have access to the same shared storage and networking. This allows HA and DRS to work across the cluster, and prevents VM incompatibilities that would inhibit HA or DRS from functioning effectively.

It's often advisable to zone your shared storage to the clusters, because doing so simplifies I/O and capacity resourcing as well as storage management. vMotion is possible across clusters in the same datacenter; so, to provide more flexibility, you can zone at the datastore level.

To take full advantage of DRS, you should collocate servers with the same CPU manufacturer into the same cluster. As we'll explain in the next section, Enhanced vMotion Compatibility (EVC) can assist with compatibility between different versions from the same chip manufacturer. Note that if you have a mixture of AMD and Intel, and you'll rely on any feature that uses vMotion, then you should aim to split these two manufacturers' chips into separate clusters. However, mixing AMD and Intel hosts in a single cluster is technically possible. If you have only a small, limited number of hosts, you may choose to house all of them in the same cluster to take advantage of HA coverage; just don't expect DRS to work.

You should also keep the host versions consistent in a cluster. It's always advisable to keep all host patching at the same level. Mixing ESXi hosts at different versions, and even alongside older ESX hosts, in a cluster is a viable configuration and fully supported by VMware.

There are several reasons why you may want to create a cluster to house hosts, even if you don't enable DRS or HA:

Future Planning Even though a cluster's benefits aren't realized until there are multiple hosts, when you initially deploy the first host, you may want to include it in a cluster. As your environment grows and hosts are added, the basic cluster will already be available.

Consistency If you have several other datacenter or cluster elements, you may find that even single-node clusters help keep management decisions consistent. For example, you may want to apply all permissions at a cluster level across your organization. Even though you may not be using DRS or HA, the cluster can apply the same settings across all the hosts and to any future hosts you add.

Host Profiles Enterprise Plus licensing includes host profiles, which let you apply a common host configuration to multiple servers. This profile can be deployed to a single host or selected hosts; it can also be used across a cluster of hosts, which means greater host standardization for existing and future cluster hosts. Host profiles not only apply a set configuration during the initial deployment, in the way a scripted install might, but also allow compliance checking throughout the lifetime of the host.

Monitoring When several hosts are members of a cluster, it's easy to compare their performance against each other. Alarms, tasks, and events for all the hosts can be viewed and managed together.

When you're considering clusters without DRS/HA enabled, note that although a stand-alone host can have resource pools, hosts in a cluster with DRS turned off can't. Adding a host to a cluster without DRS strips all the host's resource pool settings. A host without resource pools can still set shares, limits, and reservations, as discussed in Chapter 7, but they will be apportioned in relation to the host and can't be subdivided or spread across the cluster.

You can independently disable HA and DRS at any time at the cluster level. But doing so loses all the associated settings, including any advanced settings that were configured. If you need to temporarily stop a subcomponent of DRS or HA, you should disable the specific undesired function. This way, you can retain the configuration so that when the feature is re-enabled, the same settings are applied again. This is particularly important for DRS, because disabling DRS completely destroys all the resource pools.

There are two other separate cluster settings, which aren't directly related to DRS or HA functionality: EVC and default swapfile locations.

EVC

Enhanced vMotion Compatibility (EVC) is a feature that improves the ability to vMotion VMs between hosts that don't have CPUs from the same family. When you enable EVC, the cluster has to be set for either Intel or AMD chips and must have a minimum baseline level.

Chapter 7 discussed a compatibility feature that hides certain host CPU flags from the VM. Many of the CPU extensions presented by modern CPUs aren't used by VMs and can be safely hidden. EVC is a development of this, and it works by applying CPU masks across all the hosts.

Each CPU type, whether Intel or AMD, has a list of compatibility levels. You should be sure you select the lowest level required to support the oldest CPUs that will be members of the cluster. Add the oldest host first: doing so ensures that the least feature-full host will be checked against the baseline before you start creating or transferring VMs to the cluster.

The general advice is to enable EVC on all clusters. Doing so makes adding hosts easier, saves you from splitting the cluster in future, and shouldn't affect performance. EVC guarantees that each host has the same vMotion compatibility, which reduces the number of calculations the vCenter Server has to undertake. For the most part, EVC can be enabled with the VMs powered on (although to remove the EVC masking from a VM may take a power off/power on), so having the foresight to enable it from the outset isn't as important to prevent mass migrations. The only VMs that need to be shut down are those that use CPU feature sets that are greater than the EVC mode you wish to enable. If you buy all the server hardware for a cluster in one go and completely replace it at the next hardware refresh, then you probably don't need to worry about EVC being enabled. If the company is more likely to add capacity slowly as required, then turn on EVC from the outset.

Swapfile Policy

By default, a VM's swapfiles are stored along with its other files in a location that's determined by the *working directory* setting. The swapfile in this case isn't the guest OS's pagefile or swap partition but the file used by the hypervisor to supplement physical RAM. You can find more details about host memory in Chapter 4, "Server Hardware."

The swapfile location is configurable on a VM-by-VM basis, but this cluster-level setting sets a default for all the VMs. You can choose to move all of these swapfiles to a location set on each host. This can be useful for a couple of reasons:

Moving Swap Off Expensive Storage If the VM is on expensive shared storage, then this approach gives you the option to move it onto less-expensive shared storage (for example, from a RAID-10 LUN to a RAID-6 LUN) or even local host storage.

Preventing Swap from Being on LUNs Using Certain SAN Technologies If the VM's LUNs are using SAN technologies such as replication, snapshots, and deduplication, then it may not be desirable to also store the swap in this area.

Moving swap onto another less-expensive or nonreplicated SAN LUN is practical, but moving it onto the local host disk does have implications. First, vMotions will take considerably longer, because the swapfile will need to be copied across the network every time. In addition, using local-host storage can have unexpected results for other cluster functions. You need to be sure there is sufficient space on all the hosts to hold any possible configuration needs. HA or DRS won't operate effectively if it can't rely on swap space being available on the hosts.

Cluster Sizing

There are several hard limits related to cluster sizing. First, you can have a maximum of 32 hosts per cluster, and 4,000 VMs (3,000 in vSphere 5.0). DRS and HA functions can impact the size of your clusters in their own way. Suffice it to say, just because you can have 32 hosts in a cluster doesn't mean you should plan it that way.

vSphere 5.0 has a limit of 512 VMs per host, regardless of the number of hosts in the cluster. With the limit of 3,000 VMs per cluster, if you fully populate the cluster with 32 hosts, those hosts can have only an average of up to 93 VMs (3,000/32). In reality, not many implementations have more than 93 VMs per host, except perhaps in desktop VDI solutions. But today's largest commodity servers, and an ESXi limit of 2 TB of RAM, mean that it isn't unfeasible to create massive clusters with monster servers that could be restricted by these limits.

Creating very dense clusters has its ups and downs. The benefit of larger HA clusters is that you need less host hardware to provide redundancy, because splitting the cluster in two can double the need for failover hosts. Also, DRS and distributed power management (DPM) have more hosts to load-balance and can spread resources more efficiently. However, creating very large clusters has certain impacts; some companies like to segregate their clusters into smaller silos to create hard resource limitations.

The actual cluster size depends on a number of other factors. It's likely to be designed in conjunction with HA, DRS, and possibly DPM. But the starting point usually depends on factors such as host resilience and VM criticality. For most businesses, n+1 is sufficient; but this is always a numbers game. The concept is similar to Chapters 6's discussion of the availability of storage, which explains how additional redundant pieces reduce the percentage of overall downtime—but with diminishing returns. Also, if you're contemplating an n+1 design, you may want to hold off on patching and hardware maintenance until quieter times; any time you purposefully remove a host from the cluster, you lose that failover capability. If individual VMs are considered very important, you may wish to consider the fault tolerance (FT) functionality discussed later in the chapter; but if you want to protect large proportions of your VMs more carefully, you might consider n+2. Additional hosts for pure redundancy are costly, and most businesses are likely to consider this only for their most crucial clusters.

Generally, you want to opt for larger clusters because they're more efficient, but some guest OSes and applications are licensed by the number of hardware sockets they have access to. As described later in the chapter, you may be able to restrict this with a *must* VM-host rule. Unfortunately, some vendors won't accept this as sufficient segregation and determine it only as the total number of sockets in the cluster. Housing this in a large cluster can cause your software fees to skyrocket.

Previously, the impact of LUN reservations caused concerns as clusters grew, but with Virtual Machine File System (VMFS) 5's optimistic locking or, preferably, vStorage APIs for Array Integration (VAAI) SAN support, this is much less likely to create performance issues. There is still a limit of 256 LUNs per host. It's regarded as good practice where possible to have

all hosts in a cluster connected to all the same datacenters. This may limit the size of the cluster. The larger 64 TB VMFS-5 volumes can relieve this constraint in most instances.

Patching large clusters can take longer, but version 5 of the vCenter Update Manager (VUM) can automatically put multiple hosts into maintenance mode if there are enough spare resources. It can be a problem if you're trying to patch the entire 32-host cluster in one evening's change window—because we all keep the same level of patching on all hosts in the cluster, don't we?

Despite the argument that larger clusters are more efficient, it's interesting to think how much more efficient a 32-host cluster is over a 16-host cluster for DRS. For resource management, you probably want to reserve a certain percentage on each host for spikes and capacity growth. Reducing the ratio of redundant hosts is likely to have little impact above a point if you still want to reserve 10% or 20% spare compute power. When clusters become larger, the notion of n+1 becomes more tenuous as an appropriate measure. The reality is that if you're planning between 16 and 32 hosts in a cluster, you're probably counting on at least n+2.

If you do decide to split the clusters into more manageable chunks, consider grouping like-sized VMs. Perhaps you want a cluster just for your very large VMs so that the HA slot sizes are more appropriate, and to match your very large physical hosts with VMs with more vCPUs and/or large memory requirements. Remember the impacts of different host non-uniform memory architecture (NUMA) configurations from Chapter 4. When a VM is powered on, the guest OS sets its vNUMA based on the residing host. If the cluster has different hosts, then the VM vMotions around it will have a vNUMA setup that may not appropriately match the destination hosts.

Despite EVC, you may want to split unequal hosts because DRS doesn't take into account which hosts have better CPUs or better memory management. Some server generations are quite different from each other, such as the jump to Nehalem from Penryn. In contrast, keeping a Nehalem host and a Westmere host in the same cluster won't create such obvious mismatches.

Often there are internal political resourcing reasons why clusters have to be split up, so consider the failure domains, the levels of redundancy, and projected growth versus hardware lifecycle. From a manageability standpoint, there is definitely a sweet spot. Too many small clusters undoubtedly become more of a burden. But very large clusters have their own management issues, because keeping that many hosts identically configured and having the same networks and storage for all the VMs can become increasingly difficult. Many other aspects are likely to impose cluster-size constraints, but the guiding principles of cluster sizing remain relevant:

Larger but Fewer Clusters Larger clusters are generally more efficient. To provide n+1 or n+2 redundancy takes fewer hosts when you have fewer clusters. Fewer clusters also mean less management overhead; and because there is more scope for consolidation, the design should be less expensive.

Smaller but More Clusters You may want to split your clusters more because they create hard resource divisions. Although resource pools can segregate resources, their shares are proportionate to the rest of the pool, which changes as you add and remove VMs. Splitting hosts into small clusters better guarantees resources.

Resource Pools

Resource pools group VMs to allow dynamic allocation of CPU and memory resources. They can contain VMs but also child resource pools, enabling very fine-grained resource allocation. Resource pools can be found either under stand-alone hosts or as members of a DRS-enabled

cluster. Because resource pools in a cluster require DRS, the cluster's hosts require a minimum of an Enterprise license.

We looked at resource allocation in some depth in Chapter 7, examining the use of shares, reservations, and limits and how they can impact other VMs. But setting these values on every VM is time-consuming, is error-prone, and doesn't scale effectively. Setting these values on a resource pool is much more efficient, and the values dynamically readjust as VMs and host resources are added and removed. VM reservations (and limits), on the other hand, are static and always impact the other VMs in cluster. However, note that HA doesn't consider resource pool reservations, only VM-level reservations for its admission control calculations. VM resource settings can also set shares and input/output operations per second (IOPS) as limits for storage; and network I/O controls can set shares and bandwidth limits for vNetwork distributed switches (vDSs). Resource pools concentrate only on CPU and memory resources.

Resource pools can have sibling pools at the same level and child pools beneath them. Each stand-alone host or DRS cluster is effectively a root resource pool, and all resource pools subsequently derive from that point. Child pools own some of their parents' resources and in turn can relinquish resources to their children.

Each DRS cluster supports resource pools up to eight hierarchal levels deep. To prevent overcomplicating the resource-entitlement calculations and ensure that the hosts' resources are allocated to the VMs in the appropriate manner, the flattest structure should be created. Often, one level of sibling pools is all that is required. Rarely are multilevel child pools desirable. Creating too many sublevels may reduce effectiveness as the environment changes.

Should Your Cluster Consider Overallocation If It's Designed Properly?

"In theory, if you design your clusters properly with the correct amount of hardware provisioned to satisfy the VM's requirements, maybe using these resource controls isn't necessary." Well, that's the argument we hear. A well-designed virtual infrastructure shouldn't hit its limits, shouldn't suffer resource contention so shares are never used, and always allocates all requests for resources, thereby negating the need for VM or resource pool reservations. For many companies, *the risk of underprovisioning far outweighs the cost of overprovisioning*.

But it's usually unrealistic to design an environment that considers the worst-case scenario, for every workload, all the time. Some workloads will be less important to you. As much as a design should incorporate the expected levels of growth and understand that projects will need VMs, using such resource controls offers protection against the unexpected. These techniques allow resource peaks to be leveled and dealt with fairly on a shared workload.

Resource pools are very useful tools, but you shouldn't think of them as a substitute for VM folders. In the Hosts and Clusters view, folders are available at the root of the vCenter instance above datacenters, and below datacenters but above hosts and cluster items. No folder items are allowed inside the clusters themselves. For this reason, resource pools are often misappropriated as a way of grouping VMs into logical silos. However, even with the default values and no adjustment of the unlimited reservations and limits, resource pools still apply normal pool-level shares. Because they're filled with VMs, all the pools continue to have the same value despite some having more VMs than others. During periods of contention, some VMs receive more attention than others. If you create pools purely to group and organize your VMs, then this

unexpected resource allocation will be undesired and unexpected. A better method of grouping VMs is to use the VMs and Templates view, which provides for grouping VMs into folders.

As a general rule, don't place individual VMs at the same level as resource pools. It's almost never appropriate. Resource allocation is always relative to items as the same level, whether they're VMs or resource pools. VMs placed alongside a resource pool will get shares relative to the collective VMs in that resource pool and compete during times of contention. All the VMs one level down could be starved by the VM above them.

Resource Pool Settings

For each resource pool, you set CPU and memory shares, reservations, expandable reservations, and limits, as shown in Figure 8.5.

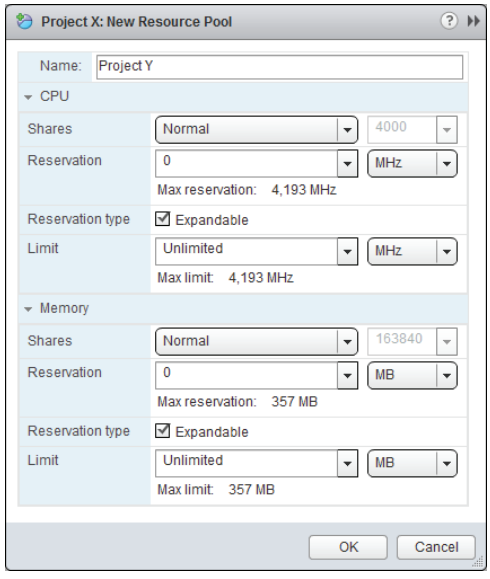


FIGURE 8.5
Resource pool settings

Most of these terms were discussed in Chapter 7, particularly the differences between how CPU and memory are handled. However, it's important to understand the concepts of allocation with regard to resource pools.

SHARES

The CPU or memory shares are relative to any sibling resource pools or VMs. Shares are used only during periods of contention and are always bound first by any reservations or limits. They can only apportion the unreserved memory, and then only up to the limit if one has been set. But if sufficient resources are available to all VMs in the resource pool (there is no contention), then shares are never invoked.

There are no guarantees with shares, and due to their dynamic nature, they can be unpredictable. A VM's resources are relative to the pool it resides in, so as the pool expands to accommodate new VMs, the pool's allocation is spread increasingly thin. Even if you give a particular

resource pool a relatively high level of shares, if it has far more VMs than a sibling pool, you may find that its VMs actually receive fewer resources than those in a less densely populated pool.

Remember, VM shares are relative to the other VMs in the same pool, but resource pool shares are relative to sibling resource pools and sibling VMs. For this reason, it's recommended that you not make resource pools and VMs siblings at the same level in the hierarchy. VMs are unlikely to have share values comparable to the resource pools, so this would result in unbalanced clusters.

Resource pool shares are often the fairest way to allocated resources, but you must check them regularly to be sure you're getting the results you want. They're fickle if neglected for long and can eventually work against you. For example, it isn't uncommon to see a split of high, normal, and low resource pools. It's only human nature that everyone wants their VM in the high resource pool. You quickly end up with an overpopulated high resource pool that performs worse than the other two when resources become oversubscribed.

Reservations

A CPU or memory reservation guarantees resources to its resource pool occupants. Any reservation set is taken from its parent's unreserved amount, even if VMs don't use it. Reservations that are set too high can prevent other resource pools and sibling VMs from being able to operate properly or even from powering on.

Setting a resource pool reservation instantly prevents other siblings or the parent from reserving this amount themselves. Reservations should set the minimum amounts that are acceptable, because leftover resources are still apportioned over and above the reservation. If the resource pool reservations commit all of the cluster's memory, this can prevent VMs from vMotioning between hosts because during the vMotion's pre-copy the equivalent memory resources must exist on the destination host as well as the source host.

Resource pool reservations are a significantly better way to reserve memory than setting it on a per-VM basis. HA ignores the resource pool reservation, so these reservations don't have such a negative affect on HA slot sizes. They allow a guarantee for the VM's memory without the greed that is associated with VM-level memory reservations.

Remember, reservations guarantee a minimum but don't limit the VM to that amount. They can receive more if it's available.

Expandable Reservations

The Expandable Reservation check box in the New Resource Pool dialog indicates whether a resource pool can steal resources from its parent resource pool to satisfy reservations defined at the VM's level. This is used during resource pool admission control, which is explained in the following section.

If powered-on VMs in a resource pool have reservations set that use the resource pool's entire reservation quota, then no more VMs are allowed to power on—that is, unless the pool has an expandable reservation, which allows it to ask upward for more resources. Using expandable reservations offers more flexibility but as a consequence offers less protection.

Limits

Just as with VM limits, resource pool limits artificially restrict the entire pool to certain amounts of CPU or memory. You can use this option to prevent a less important resource pool of VMs

from impacting a more important pool. You should use this setting very sparingly; limits are hard and will take effect even if spare resources are available.

Admission Control

Admission control ensures that reservations are valid and can be met. It works at different points in your virtual infrastructure; hosts, storage DRS, HA, and resource pools are all mechanisms that have their own type of admission control. Resource pool admission control depends on whether the pool's reservations are set as expandable.

Resource pool admission control is checked whenever one of the following takes place:

- A VM in the resource pool is powered on
- A child resource pool is created
- The resource pool is reconfigured

If the reservations aren't expandable (the Expandable Reservation check box isn't selected), then admission control only checks to see whether the resource pool can guarantee the requirements. If it can't, then the VM doesn't power on, the child isn't created, or the pool isn't reconfigured.

If the reservations are expandable (the Expandable Reservation check box is selected, which is the default), then admission control can also consider the resource pool's parent. In turn, if that parent has its reservations set as expandable, then admission control can continue to look upward until the root is reached or it hits a pool without an expandable reservation.

Expandable reservations allow more VMs to be powered on but can lead to overcommitment. A child pool may reserve resources from a parent pool while some of the parent's VMs are powered off. Therefore, subordinates with expandable reservations must be trusted. This is particularly relevant if you use resource pools for permissions.

Distributed Resource Scheduling

DRS in vSphere clusters uses VM placement and the power of vMotion to optimize cluster resources. Its primary function is to load-balance VMs across hosts to provide the best resource usage possible. DRS can use special rules to control VM placement, so that certain VMs can be kept together or apart depending on your requirements. A subfunction of DRS known as distributed power management (DPM) can use vMotion to evacuate hosts and selectively power-down host servers while they aren't needed and power them back on automatically when they're required again.

Load Balancing

DRS monitors the CPU and memory load on the cluster's hosts and VMs, and tries to balance the requirements over the available resources. It can use vMotion to seamlessly move VMs when appropriate, effectively aggregating CPU and memory across the cluster.

DRS does this with two approaches. First, when VMs are powered on, it looks to see which host would be most suitable to run on. Second, while VMs are running, if DRS calculates that the resources have become unbalanced, it decides how to live-migrate VMs to minimize contention and improve performance.

When VMs in the cluster are powered on, DRS performs its own admission control to ensure that sufficient resources are present to support the VM. This is essentially recognition that the DRS cluster is itself a root resource pool and so follows the same resource checks.

DRS Requirements

For DRS to load-balance effectively, you should adhere to a number of design requirements:

Shared Storage In order for VMs to be vMotioned between hosts, they need to be stored on shared storage that all the hosts are configured to use. Despite the new functionality in vSphere 5.1 that allows vMotions between hosts without shared storage, DRS only considers VMs on shared storage.

vMotion-Compatible Hosts The cluster's hosts should be vMotion-compatible with each other. This means their CPU must be of the same processor family. You can enable the EVC feature (discussed earlier) on the cluster to increase the likelihood that hosts will be compatible with each other. vMotion requires that the hosts be able to communicate with each other via TCP port 8000 in both directions.

vMotion-Compatible VMs Chapter 7 explained the factors that prevent VMs from vMotioning across hosts in the cluster. Hardware version levels must be compatible with the hosts, and be sure you don't leave any host-attached hardware connected to the VMs.

No MSCS VMs Microsoft clustering VMs can't vMotion between hosts.

VMkernel network Each host needs to have a minimum 1GbE connection to a shared VMkernel network.

vMotion Improvements in vSphere 5

vSphere 5 has had a number of improvements to vMotion that benefit DRS. Up to four 10 Gbps NICs or sixteen 1 Gbps NICs can be used (or a mixture of both) in combination to accelerate vMotions. This not only speeds up individual vMotions but is particularly helpful in fully automatic DRS clusters when evacuating a host, for example putting a host in maintenance mode and allowing DRS to clear all the VMs off to other hosts in the cluster. To use multiple NICs, all the vMotion interfaces (which are vmknics) on a host should share the same vSwitch, be set active on only one uplink (vmnics) with the other uplinks set to standby, and use the same vMotion subnet.

The improvements in vSphere 5 vMotion techniques also include a more efficient process that should cause fewer issues for guest OSes and applications that previously had problems with the prolonged stun vMotion invoked. The vMotion process now completes more quickly and with fewer interruptions. The performance of vMotions is up to 30% faster to complete, and an improved migration technique helps move those VMs with lots of changing memory whose vMotion would have otherwise failed. These improvements enable a new feature of the Enterprise and Enterprise Plus licensed hosts to do vMotions over stretched links with up to 10 ms latency. Prior to vSphere 5, and with a non-Enterprise licensed host, this is limited to 5 ms.

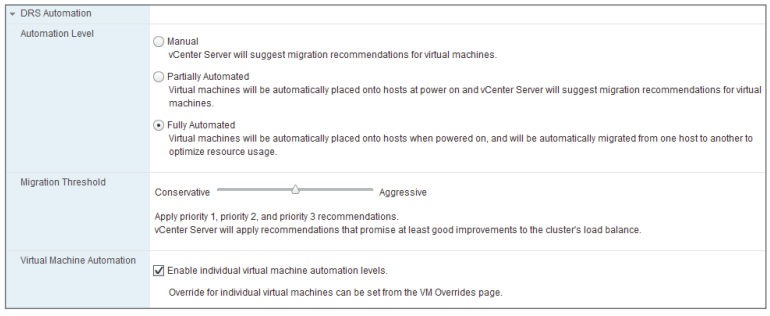
Cross-Host vMotion

The cross-host vMotion capability introduced in vSphere 5.1 is an excellent addition to vMotion and undoubtedly precedes features that will influence DRS. But as of 5.1, cross-host vMotion is not utilized by DRS or DPM, only by manual user-initiated vMotions.

DRS Automation Levels

A DRS cluster has an automation level that controls how autonomous the resource allocation is. Figure 8.6 shows the settings page for DRS levels.

Figure 8.6
DRS Automation levels



Manual At the Manual level, DRS makes recommendations for the most suitable hosts during the VMs' initial placement and also makes ongoing migration suggestions. When the VMs are powered on, DRS decides which is the best host and waits for the user to accept or override the recommendation. It continuously evaluates the cluster resources and proposes vMotions that would benefit the cluster and its VMs. At this level, DRS won't automatically evacuate VMs when the host is put into maintenance mode.

Partially Automated The Partially Automated DRS level takes the initial placement recommendations and implements them without consulting the user. This ensures that from the outset, the VMs are on the best host; and, notwithstanding the VMs' needs changing or hosts being added or removed, this should mean a relatively balanced cluster. A partially automated cluster also makes ongoing migration suggestions but doesn't act on them independently.

Fully Automated Fully automated DRS clusters not only place the VMs automatically on the best host when the VMs are powered on but also react to the resource levels on an ongoing basis, vMotioning VMs to different hosts as appropriate.

As you can see in Figure 8.6, the Fully Automated setting has a slider control that allows finer control over the cluster's propensity to vMotion VMs. The most conservative setting automatically moves VMs only when a top-priority migration is required—for example, if a host has been put into maintenance mode or standby mode. The most aggressive setting takes even the least advantageous recommendations, if it thinks they can benefit the clusters' resource-allocation spread.

VM Options (DRS)

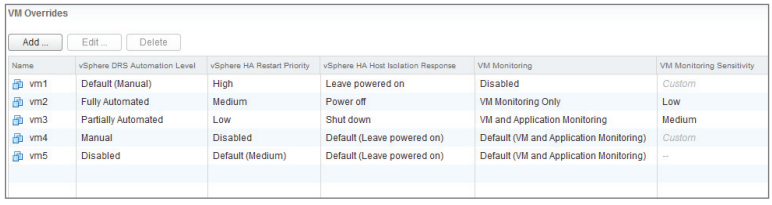
DRS has the ability to override the cluster settings for individual VMs. Figure 8.7 displays all the possible options for a VM.

Each VM by default follows the cluster setting; but by being able to set a different level, you avoid the cluster being too prescriptive. Otherwise, the needs of a single VM can force you to lower the cluster setting. With the per-VM option, if a particular VM needs to avoid being vMotioned, this doesn't have to affect the entire cluster. A common use of this override ability occurs

if your vCenter Server is itself a VM: you can pin that VM to a particular host. This makes it easy to find your vCenter VM in an emergency when it has been powered off.

If your cluster ordinarily is set as fully automated, and you want to shield a VM from this level then you have 3 options: Partially Automated, Manual or Disabled. Setting the VM to disabled or manual means you need to move the VM yourself when you place the host into maintenance mode, whereas partially automated would move it for you. Partially automated and manual VMs, unlike disabled VMs, are included in DRS recommendations and it's expected that those recommendations are followed; otherwise the cluster can become unbalanced. Disabled VMs are included in the DRS calculations, but vCenter understands that they can't move, so although no action is required to keep balancing the cluster there are still less optimal choices for the cluster. For these reasons it is advisable to keep as many VMs at the cluster's default setting as possible.

FIGURE 8.7
VM Overrides options



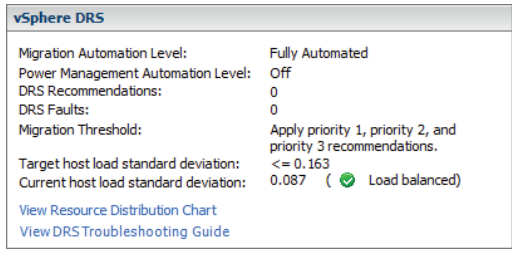
BALANCING DECISIONS

The DRS threshold slider, shown in Figure 8.6, measures how much of an imbalance should be tolerated in the cluster. There are several reasons why the cluster should change after the initial placement of the VMs, such as changes to VM loads, affinity rules, reservations, or host availability.

DRS uses a priority rating to show the varying levels of migration recommendations. The most-recommended migrations are given a priority of 1 or 2.

The DRS summary section in the Windows Client, shown in Figure 8.8, displays the threshold settings.

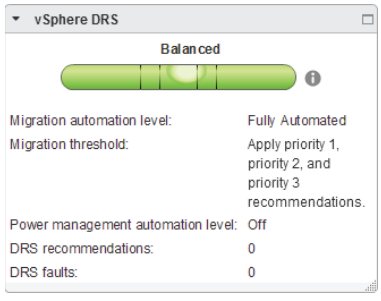
FIGURE 8.8
DRS summary in the Windows Client



The threshold explains which priority level's recommendations are applied automatically. This threshold value, along with the amount of host resources available, determines the Target Host Load Standard Deviation. The Target is effectively a measure of how much the cluster is willing to accept a resource disparity.

Every 5 minutes, the DRS cluster calculates the corresponding Current Host Load Standard Deviation. The Current level is compared to the Target value; if it exceeds the Target, then a recommendation needs to be made. If the cluster is set to Fully Automated, then Current should always be below Target. But if the cluster is only Partially Automated, then a Current value greater than Target shows that there are recommendations that haven't been followed. In vCenter's 5.1 Web Client, as shown in Figure 8.9, the imbalance is illustrated with a spirit level. The threshold settings shown in figure 8.8 are also available via the information link to the right of the spirit level.

FIGURE 8.9
DRS summary in the Web Client



If the cluster has decided that a migration needs to occur, it goes through a selection process to decide which is the next best candidate. It does this by evaluating each VM and seeing which one would make the largest improvement to the cluster. It also considers the vMotion history, and it drops VMs that have had problems migrating in the past. This is why DRS tends to move VMs that have the most vCPUs and biggest memory allocation: they're likely to have the greatest effect on the cluster's performance and reduce the number of vMotions needed to balance the hosts. DRS's main function is to distribute the CPU and memory load across the cluster, not necessarily to distribute the VMs.

DRS Efficiency

Several factors determine how well DRS is able to perform. The hosts should be as similar as possible, to maintain vMotion compatibility and avoid different CPU and memory configurations. This allows the cluster to predict performance outcomes more accurately and consequently make better migration choices.

Try to minimize excessively loading VMs, because those with significantly more vCPUs or memory will ultimately reduce DRS opportunities. Where possible, shut down or suspend VMs that aren't being used, because they consume CPU and memory resources even when not being taxed. Disconnect any unused hardware, such as CD-ROMs, floppy drives, and serial and parallel ports, because they not only use CPU cycles but also reduce the vMotion possibilities. As stated earlier, reservations shouldn't be set too high and limits shouldn't be set too low, because this also affects the DRS calculations.

All DRS hosts should be able to access the same shared storage. If some hosts can only see portions of the storage, this separates the cluster and severely constrains the options for DRS migrations.

Temporarily disabling DRS causes all the resource pool and VM option settings to be lost. To preserve these for the future, you should set the cluster setting to Manual. That way, the

resource pools will remain intact, and you just need to set the level back to what it was to restore the previous settings.

DRS Fully Automated mode ensures that the cluster is as balanced as it can be, without requiring intervention. When your hosts are as uniform as possible and your VMs maintain steady resource needs, you can set the cluster threshold more aggressively.

DRS doesn't solve underlying problems when there aren't enough resources to go around. It will, however, ensure that you're using the available resources in the best way possible.

Affinity Rules

Affinity rules are an additional feature of DRS that let you specify how to place VMs. Two types of affinity rules exist. The original affinity rules, which give you control over keeping VMs together or apart, are now known as VM-VM affinity rules. These are augmented with the VM-Host affinity rules to direct the placement of VMs on the correct hosts.

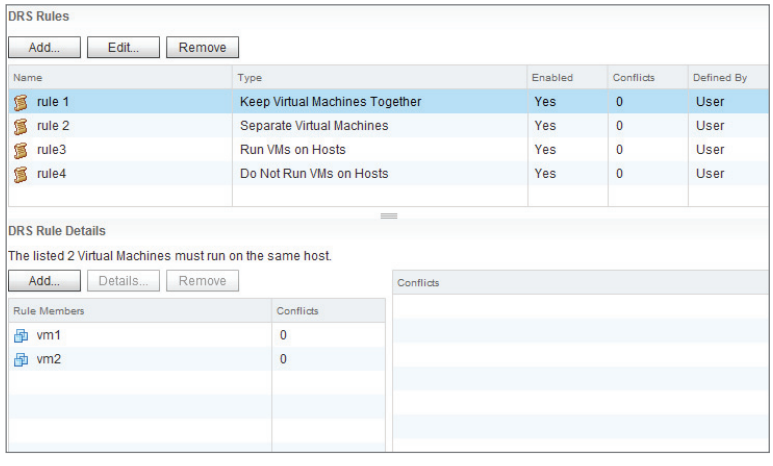
Both types of rules have the basic concept of *affinity* or *anti-affinity*. As you'd expect, affinity rules try to keep objects together, whereas anti-affinity rules aim to keep them apart.

VM-VM Affinity Rules

The VM-VM affinity rules have been around since pre-vSphere days and keep VMs together either on the same host (affinity) or on separate hosts (anti-affinity).

Figure 8.10 shows the DRS affinity rules screen. As you can see, the first rule is keeping VMs together, and the second is keeping them apart.

FIGURE 8.10
DRS affinity rules



Rules can be created and not enabled. However, DRS disregards rules that aren't enabled. The vSphere Client won't let you enable VM-VM rules that conflict with each other. For example, if a rule exists that keeps two VMs together, then although you can create a second rule to keep the same two VMs apart, you can't enable the second rule. The older rule always takes precedence, and the newer rule will be disabled if the older rule is ever enabled. If you have competing affinity and anti-affinity rules, DRS first tries to apply the anti-affinity rule, then the affinity rule.

Keep VMs Together

You may wish to keep VMs together on the same host to minimize the amount of inter-host network traffic. This is useful if two VMs work closely together—for example, an application that has a web server VM and a database VM. These two VMs may benefit from the reduced latency of the network traffic flowing between them, instead of between two hosts via a physical switch.

Also, if you have an application that needs multiple VMs to work, each of them being a potential single point of failure, then you may wish to keep them together. Allowing DRS to separate them only increases the chance that a host failure will affect the application.

Arguably, you can try to keep together VMs that are extremely similar, because you know this will benefit host transparent page sharing the most and reduce memory consumption. But DRS uses very sophisticated algorithms, so such rules are more likely to constrain potential optimizations and may make things worse.

Separate VMs

Rules to separate VMs tend to be used to guarantee that the VMs are always kept on different physical hardware. Applications that can provide redundancy between VMs can be protected against a single host failure. An example is a group of Network Load Balancing (NLB) web servers.

Another reason to keep VMs apart is if you know some VMs are very resource intensive, and for performance reasons you want them on separate hardware. This is particularly relevant if the VMs are inconsistently heavy and you find DRS doesn't do a good job of spreading their load.

With VM-VM anti-affinity rules, although you can state that you want VMs kept on separate hosts, you can't dictate which hosts. For that, you need a VM-Host rule.

VM-Host Affinity Rules

VM-Host affinity rules allow you to keep VMs on or off a group of hosts. To specify the groups of VMs and the groups of hosts for each rule, you use the DRS Groups pane (see Figure 8.11). Here you can create logical groupings of VMs and hosts that you want to use in your VM-Host rules.

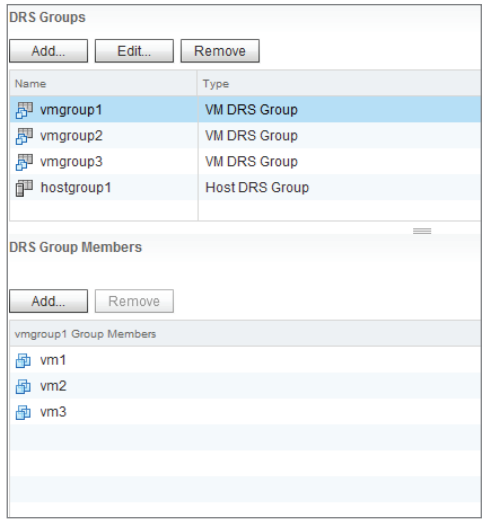


Figure 8.11
DRS Groups

In Figure 8.10, the DRS Rules page, you can see that rules 3 and 4 use the VM groups and the hosts groups. The third rule specifies that a group of VMs should run on a group of hosts, whereas the last rule keeps the group of VMs away from the group of hosts.

In addition to the affinity and anti-affinity rules, VM-Host rules have the concept of *should* rules and *must* rules.

Should *Rule*

A *should* rule is one that DRS treats as a preference. DRS tries to adhere to the rules when it can but may violate them if other constraints are deemed more important. *Should* rules are always a best effort and allow DRS a certain amount of freedom. Similarly, DPM tries to conform to *should* rules but can break them if it needs to.

HA ignores *should* rules and powers on VMs in case of a failure regardless. But if DRS is set to Fully Automated, it then steps in and rebalances the cluster according to the rules.

Must *Rule*

A *must* rule is a mandatory one that can't be broken by DRS, DPM, or HA. This level of strictness constrains cluster functionality and should be used only in exceptional circumstances. DRS won't power-on or load-balance VMs in a way that would violate a *must* rule, DPM won't power-off hosts if doing so would break a *must* rule, and HA will only recover VMs onto hosts that adhere to the *must* rule.

Must rules make certain hosts incompatible with VMs to enforce the rules for DRS, DPM, and HA. If DRS as a whole is disabled, these rules continue to be enforced. DRS must be re-enabled if you need to disable a *must* rule.

Using VM-Host Rules

VM-Host affinity rules are useful in several ways. The classic *must* rule use case is to keep VMs tied to a group of hosts for licensing requirements. Some independent software vendors' (ISVs) licensing depends on the physical hardware—often the CPU socket count. Prior to VM-Host rules, vSphere users created a separate cluster just for this purpose. Despite the fact that the separate cluster complied with the licensing terms, it often wasted spare resources, cost more in both OPEX and CAPEX terms, reduced redundancy possibilities, and gave the regular cluster fewer DRS/DPM/HA options. Such a situation is likely to use *must* rules.

Another good use of VM-Host rules is to keep VMs together or apart on a blade chassis or an entire rack. For the same reasons as VM-VM affinity and anti-affinity, you may consider blade systems one piece of hardware, even though as far as vSphere is concerned, they're all separate hosts. You can use VM-Host rules to keep network traffic together on one chassis or ensure that VMs are on blade hosts that are in different chassis to provide greater physical protection. These sorts of use cases are best set as *should* rules.

Unfortunately, DRS has no inherent way of understanding (or being instructed) which blade servers run in which chassis, or by extension which rack servers or chassis are in which racks. To ensure that VMs are kept on separate chassis, you need to create a host group that has only one member from each chassis. This is more limiting that you'd perhaps like, because all you need is to keep them on *any* blade in each chassis, but by pinning them on a single blade in each chassis at least you prevent them from accidentally pooling together.

You can also use *should* VM-Host rules to provide soft partitioning for areas that previously you may have considered separate clusters in the same datacenters. You may have test and

development environments that you want segregated from other VMs on their own hosts. You don't want the management overhead of complex resource pools and VM resource settings, but you want to keep all the VMs in the same cluster so you can take advantage of a pool of hosts that DRS and HA can optimize. Here you can use *should* rules to keep groups of VMs affined to hosts.

You should use mandatory *must* rules sparingly because they restrict the other cluster operations so much. They can divide the cluster, prevent HA from functioning properly, go against DRS optimizations, and limit DPM selection of hosts to power down. If you're considering a *must* rule for a purpose other than licensing, consider a *should* rule with an associated alarm. You can create vCenter alarms to trigger an email whenever a *should* rule is violated. That way, you're aware of another cluster function breaking a rule without unnecessarily constraining the entire cluster.

Applying several overlapping affinity rules can complicate troubleshooting, because it can be difficult to comprehend the resulting effects. Use affinity rules only when you need to, and avoid creating more than one rule that applies to the same VM or the same host.

Distributed Power Management

DPM is an additional feature of DRS that looks to conserve power by temporarily shutting down hosts during periods of lesser activity. It monitors the total levels of CPU and memory across the cluster and determines whether sufficient resources can be maintained while some hosts are put into standby mode. If so, it uses DRS to migrate VMs off a host to clear it, preparing it for shutdown.

When the cluster deems that additional host resources are needed again, it brings them out of standby in preparation. The cluster either determines this by observing increased CPU or memory levels, or considers historical data to predict busier periods.

By default, if both CPU and memory usage levels on all the hosts in a DPM cluster drop below 45%, then DPM selects a host to power down. It re-evaluates this every 40 minutes. At 5-minute intervals, it checks whether either CPU or memory rises over 81%, and if so, it powers on a host.

To bring the servers back online, it can use one of three different remote power-management protocols. In order, the cluster attempts to use Intelligent Platform Management Interface (IPMI), then HP's Integrated Lights Out (iLO), and finally Wake On LAN (WOL) to send the power-on command.

DPM has a preference for smaller hosts when selecting which server to shut down. It does this because the larger hosts should be more power efficient per VM and probably have better host power-management results. If all the servers are the same size, then it selects the host to power down based on what it considers the least loaded and from which it will be easiest to evacuate the VMs.

DPM Requirements

DPM has a number of requirements in order to function:

- DPM is a subfeature of DRS, which requires an Enterprise or Enterprise Plus license.
- The cluster must have DRS enabled.
- Hosts participating in the DPM set must be running vSphere 3.5 or later.

- Using the IPMI or iLO protocol requires a baseboard management controller (BMC) IP address, a MAC address, and username/password credentials.
- WOL protocol use requires the following:
  - Hosts' vMotion NICs must support WOL.
  - Hosts' vMotion addresses should share a single subnet.
  - The switch ports that the WOL cables connect to must be set to autonegotiate.

You should test each host thoroughly to ensure that it can wake properly when required. This is often forgotten when introducing new additional hosts to an existing DPM-enabled cluster.

DPM Automation Levels

The DPM Power Management page, shown in Figure 8.12, controls the level of automation across all the cluster's hosts:

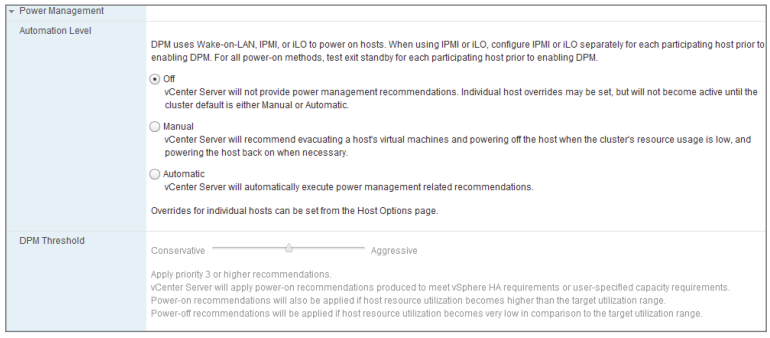


FIGURE 8.12 DPM Power Management

Off This is the default setting on any enabled DRS cluster. DPM isn't used. Changing the setting to Off at any time brings all hosts out of standby if DPM has shut them down.

Manual This mode analyzes the cluster for hosts to shut down and makes recommendations. It's particularly useful for initial testing to see what effect enabling the Automatic level may have.

It's probably less useful in the longer term, because the most likely times for DPM recommendations are during quiet periods (for example, weekends and overnight) when administrative staff are less likely to be available to act on them. But you can use this mode for longer-term warm spares, which ordinarily can be shut down even during busier times based on these recommendations.

Automatic The Automatic option reacts to DPM recommendations and acts to shut down hosts when it sees fit. This setting has a threshold level, which determines how conservative or aggressive the action is. The more aggressive the level, the more recommendations are acted on.

The DPM automation-level setting isn't the same as the DRS automation level. They act independently of each other.

DPM Host Options

When DPM is enabled on a DRS cluster, all the hosts inherit the cluster's settings. However, the Host Options settings allow you to manually override this on a per-host basis. Just as with the cluster itself, you can select Disabled, Manual, or Automatic for each host, or set Host Options to Default to use the cluster's setting.

This is particularly useful when you're introducing new hosts to the cluster and testing a host's ability to wake up when requested. Check a host's Last Time Exited Standby field to see if it was able to resume properly on the last occasion. This individual setting can also be useful if you want to maintain a minimum number of hosts left on despite any DPM recommendations, or if you want to prevent particular hosts from being shut down.

The cluster must be set to either Manual or Automatic for these options to be considered. You can't disable DPM for the entire cluster and enable it only for certain hosts in the Host Options section.

DPM Impacts

DPM takes advantage of spare cluster resources, so the more hosts that are available to it, the better. You should avoid the temptation to exclude too many hosts if they're capable of remotely waking, because the more choices the cluster has, the more potential power savings it can generate. If there are VMs that are set to override the default DRS cluster settings, then you should manually keep them to one host, or as few as possible to allow for the most efficient DPM action. VM templates are not moved off hosts for DPM, so it's a good idea to keep all the templates on one host that is excluded from DPM, unless DPM is only scheduled to run during times when provisioning VMs is unlikely.

The cluster does consider historical patterns to try to predict when there will be an impending resource requirement. However, even when the hosts wake up, performance is always reduced for several minutes. It takes a while for DRS to spread the increasing load across to the newly powered-on hosts, and then some time for the benefits of transparent page sharing (TPS) and memory compression to take effect. If you need the fastest possible time to recover into a fully performant set of hosts, don't set the DRS slider too low.

A scheduled task associated with DPM lets you set a predefined time to disable DPM on a host and therefore wake up the host. This allows you to be sure the resources are ready when you know resource demands will increase, such as when lots of employees start work at the same time. You may know about certain busy periods, such as monthly billing cycles or scheduled work on a weekend, and want the hosts ready beforehand. Just remember, you'll need another scheduled task to re-enable DPM afterward, if you aren't planning to do it manually.

If any host is passing through USB devices, you should consider disabling DPM for that host. Although vSphere can vMotion VMs with USB devices attached, if the host that has the physical connection is powered down, the VMs will obviously lose their connection. DPM can be used with a mixed cluster and supports hosts from version 3.5 onward. However, DPM can operate effectively only if the VMs are able to vMotion between all the hosts; think about VM hardware versions, EVC, attached hardware, and any other factors that will affect the VMs' ability to migrate seamlessly.

When DPM is looking to migrate VMs off hosts, it not only appraises the CPU and memory demands, but also adds the reservation levels set on the VMs and the cluster's resource pools.

It does this to ensure that sufficient compute resource will remain available; it respects the reservation guarantees that are set. This demonstrates how you can use reservations to be sure a suitable number of hosts are left powered on, but also reinforces the advice that reservations have far-reaching impacts, shouldn't be set without reasonable justification, and shouldn't be set excessively.

Finally, remember that DPM also takes into consideration the cluster's HA settings. It doesn't power off all the hosts it otherwise can, if it needs to keep hosts turned on to provide sufficient failover capacity. Again, this reiterates the fact that being overly cautious and setting very conservative values in some areas will have an impact on the efficiency of other features.

When to Use DPM

When designing vSphere clusters, many people question the usefulness of DPM. It can be advantageous in environments where demand varies substantially at different times. Shutting down hosts for extended times can save power and also reduce air-conditioning requirements in a busy server room. But if resource demand is fairly static, which is common in server workloads rather than desktops, the savings are minimal.

Many companies don't pay for their own power or air-conditioning, if they're collocated in third-party datacenters. Often, stringent change-control rules don't allow for automatic host shutdowns. And many power and cooling systems are designed to run at full load, so shutting down a handful of hosts that are potentially spread out through different racks won't provide any savings. Although DPM allows you to set whether specific hosts ignore the cluster's setting, it doesn't currently let you express a preference regarding the order in which hosts should be shut down (to target a certain rack space first). In addition, all the associated storage and networking equipment must remain powered on, even if some hosts are shut down.

DPM does have potential use cases. If a site has a heavy VDI implementation, and the desktop users work predictable hours, then there is obviously scope to power down hosts overnight and on weekends. There may be longer holiday periods or designated shutdown periods when a site is mothballed.

Another possible scenario is disaster recovery (DR) sites that are predominantly required for standby purposes. In the event of a failover, such sites need lots of spare hosts that can be resumed reasonably quickly. DPM allows them to fire up automatically, when more hosts are needed; this provides an excellent solution that requires little additional intervention.

Test and lab environments also tend to vary wildly in their requirements. These can be good candidates for DPM use because they often see much quieter periods, and generally it's acceptable if the time to recover to full capacity is several minutes.

DPM is just one more useful option in a business's toolkit. As hardware manufacturers incorporate more power-saving functionality into their equipment, it will become more useful. ESXi's host power management is available on every host, and is a complementary option to reduce the overall power usage, using ACPI p-states and c-states to calculate when its CPU scaling is appropriate without detrimentally affecting performance. Chapter 4 examines host power management.

Larger cloud data providers with thousands of hosts undoubtedly will be interested in the potentially very large savings. The technology certainly has its place. However, you should remain mindful that in small environments, the savings may be minimal. Don't spend time and resources designing and testing a DPM solution if it's obvious from the outset that you won't

recover enough energy savings to make it worthwhile. Or you may want to apply DPM only in specifically targeted sites, rather than across all your hosts.

High Availability and Clustering

vSphere encompasses several high-availability options. The primary technique is VMware's HA for hosts, but this is supplemented with both VM monitoring and fault tolerance (FT) capabilities.

High Availability

VMware HA is a clustering solution to detect failed physical hosts and recover VMs. It uses a software agent deployed on each host, along with network and datastore heartbeats to identify when a host is offline. If it discovers that a host is down, it quickly restarts that host's VMs on other hosts in the cluster. This is a fast and automated recovery service, rather than what most consider a true high-availability clustering solution to be.

HA uses vCenter to license the feature and deploy the necessary agent software; but after hosts are enabled for the HA cluster, the heartbeat and failure detection are completely independent of the vCenter Server. An offline vCenter won't affect ongoing HA operations, only any reconfiguration that's required, such as adding or removing new hosts.

HA primarily protects against host failures, and as such no special consideration is required for the VM's guest OS or application. There are no requirements for VM-level agents to be installed, and any new VMs in the cluster automatically inherit the HA protection. HA can also protect VM OSes and applications via the VMware Tools software, although it doesn't do this by default. It can detect when OSes aren't responding and allows applications to alert HA when critical services or processes aren't working as expected.

HA also monitors the capacity of the cluster and its ability to fail over VMs from any host. It can enforce policies to ensure that sufficient resources are available.

One common misunderstanding about HA among newcomers to the technology is that HA doesn't use vMotion to recover VMs. The VMs crash hard when the host fails, but they're restarted promptly on alternate hosts. HA uses DRS's load-balancing if it's enabled to make the best restart decisions possible. HA doesn't provide the same level of uptime as some other clustering solutions, but it's very easy to set up and maintain. Expect the VMs to be off for several minutes while they restart after a host failure.

HA Requirements

To enable HA in a vCenter cluster, you should ensure that you meet the following requirements:

- The cluster's hosts must be licensed as Essential Plus or above.
- There must be at least two hosts in the cluster.
- All hosts must be able to communicate with each via their Management Network connections (Service Console if older ESX hosts).
- The HA hosts must be able to see the same shared storage. VMs on local storage aren't protected.

- Hosts must also have access to the same VM networks. For this reason, it's important to use consistent port-group naming across hosts.
- HA's VM and application monitoring functionality needs VMware Tools to be installed in the guest OS.
- TCP/UDP port 8182 must be open between the hosts.

HA in vSphere 5

One of the biggest changes with the initial release of vSphere 5, but one of the least noticeable to users was the replacement of the old HA system with a new purpose-built, superior mechanism. This new HA, known internally as Fault Domain Manager (FDM), does away with the old primary and secondary host model used in pre-vSphere 5 versions. If you're unfamiliar with how this worked previously, see the following breakout section entitled *Pre-vSphere 5 HA* where an overview is provided. This will be useful if you're still responsible for a vSphere 4 environment, want to see how much the design of an HA cluster has simplified, or are simply feeling a tad nostalgic.

The FDM version of HA instead has the concept of a master/slave relationship and no longer has the limit of five primaries that heavily impacted cluster design. This simplifies your design significantly because you no longer need to worry about which room, rack, or chassis the primaries are in. IPv6 is supported; logging is done through a single file for each host (in `/var/log`) with syslog shipping now possible. Reliability has improved, in no small part because HA has dropped its dependency on DNS resolution, a common cause of misconfiguration—everything is IP based. The newer host agents are dramatically faster to deploy and enable regardless of cluster size. This improves cluster reconfigurations, and the status of the master/slave status is clearly revealed in the vCenter GUI.

These HA improvements come as part of vCenter 5. So if there are pre-5 hosts, they receive the benefit of the FDM-based HA if their vCenter is upgraded or they're joined to a vCenter 5 instance. vCenter 5 can manage ESXi and ESX hosts back to and including version 3.5; their HA agents are automatically upgraded if they join an HA-enabled cluster. This is a quick way to take advantage of one of the major vSphere 5 features by upgrading a single infrastructure component.

ESX(i) 3.5 Patch Required for vCenter 5

If you're joining an ESX(i) 3.5 host to a vCenter 5 or upgrading the host's vCenter to version 5, be aware a patch is required to make the host compatible:

- ESX 3.5 hosts require patch ESX350-201012401-SG PATCH.
- ESXi 3.5 hosts require patch ESXe350-201012401-I-BG PATCH.

HA now uses multiple channels for agent-to-agent communication; both network and storage fabrics are utilized. A second heartbeat via the storage subsystem in addition to the network heartbeats improves the reliability of the fault detection, helps to prevent false positives, and clarifies the root cause.

Master Host

In each HA cluster, one host is selected to run in the master role. The host that is connected to the greatest number of datastores and the greatest diversity of datastores (the most arrays) is selected as the master host. Under normal working conditions there is only one master host per cluster. The rest of the hosts are designated as slaves.

The master node is responsible for the following:

- Monitoring the slave hosts and restarting its VMs if a host fails
- Protecting all the powered-on VMs, and restarting them if a VM fails
- Maintaining a list of hosts in the cluster and a list of protected VMs
- Reporting the cluster's state to the vCenter Server

HA communicates using the hosts' management network via point-to-point secure TCP connections (elections are via UDP). It uses the datastores as a backup communications channel to detect if VMs are still alive and to assign VMs to the masters if a network partition occurs, creating multiple masters (more in this later). The datastore heartbeat also helps to determine the type of failure.

vCenter is responsible for informing the master node about which VMs should fall under the protection of HA and any changes to the cluster's configuration, but the recovery of VMs is solely the responsibility of the master. If vCenter is unavailable, the master will continue to protect the VMs; but changes to the cluster's settings, which VMs are protected, and any alarms or reporting will remain unchanged until the vCenter Server is back online.

Slave Hosts and Elections

When a cluster is first enabled for HA, the master host is selected and the remaining hosts in the clusters are designated as slave hosts. A slave in the cluster is responsible for the following:

- Monitoring the state of its own VMs and forwarding information to the master
- Participating in the election process if the master appears to fail

If the slave hosts believe that the master has failed, there is an election between the remaining active hosts to decide which is best placed to take over the master role. The process is identical to when the HA was first initialized—the host with the most datastore connections wins. The entire HA election process takes less than 20 seconds to resolve. The new master takes over all roles and responsibilities immediately.

Failed Hosts

Network heartbeats are sent between the master and each of the slaves every second. If the master doesn't receive a return acknowledgment, it uses the datastore to see if the slave is responding via its datastore file locks. If there is no response to the network or datastore heartbeats, the master tries an ICMP ping to the slave's management address. If the master can't confirm that the host is alive, it deems that the slave has failed and begins restarting its VMs on the most appropriate hosts in the cluster. At this point, the *failed* host may still be running in some capacity (for example it has lost network access, including IP-based storage), but the master still considers it failed and begins seizing the VMs and powering them on elsewhere.

However, if the slave is found to still be responding to its datastores, then the master assumes it has become either network-partitioned or network-isolated and leaves the slave to respond accordingly to its own VMs.

A network *partition* occurs when a host loses its network heartbeat to the master but is still responding to its storage and can see election traffic. A network *isolation* event happens when a host is in similar position but can't see election traffic. When a host can't ping its isolation address, it realizes that it won't be participating in an election, and it must be isolated.

A common scenario to explain a network partition involves hosts split between two locations but with a stretched Layer 2 network. This is often known as a *stretched cluster*, *metro cluster*, or *campus cluster*. If the interroom network connection fails, this can create a partitioned network with grouped hosts split on either side. Another common example of a partition is when the link to a network switch on a chassis fails.

When a partition happens, the master host can still communicate with all the slaves on the same side of the split. The hosts and VMs on that side are unaffected. The master can see that the hosts on the other side of the split are partitioned and not failed (due to the storage fabric heartbeats) and leaves them to their own recovery. In the other room, all the slaves can see the election traffic from each other, but they can't communicate with the master through the network. They hold a reelection between themselves to appoint a new master so they're protected until the partition is resolved. This is why we said "under normal conditions there is only one master per cluster," because when a cluster becomes partitioned a master is elected in each partition.

If the slave doesn't see any election traffic on the network, it realizes that it's isolated from all the other slaves. At this point it carries out the Host Isolation response that was configured in the cluster's settings to leave the VMs running, shut them down, or power them off. The Host Isolation response options are examined further in the "Host Monitoring" section. In a properly designed network environment with appropriately redundant devices and cabling, host isolations should be a very rare occurrence.

When a cluster experiences a network partition, it isn't as worrisome as a host isolation because at least the partitioned hosts have the potential for some redundancy within themselves. Partitioned states should be fixed as soon as possible, because admission control doesn't guarantee sufficient resources for recovery. The VMs aren't fully protected, and it's very likely that some, if not all, of the partitions have lost their connection to vCenter. Until the partitioned state is resolved, the cluster isn't properly protected or managed.

Pre–vCenter 5 HA

Prior to vCenter 5, VMware HA used a different mechanism to monitor and trigger the recovery of failed hosts. This was always known simply as HA, but it used the Automated Availability Manager (AAM) agents. This version of HA had a number of interesting design constraints that thankfully have been removed with the newer FDM agents in vCenter 5. Although no longer applicable to vSphere 5 designs (thank goodness!), a basic understanding of this old HA version is useful when you come across mixed environments or situations where you're working on an upgrade project. If nothing else, it makes you appreciate why you'd deploy vCenter 5, even with older existing hosts.

With AAM, the first five hosts joined to an HA cluster were marked as the *primary* HA hosts. All subsequent hosts were designated as *secondary* HA hosts. If HA was enabled on a preexisting cluster with six or more hosts, then five were randomly chosen to be the primaries.

The primary hosts maintained a replicated cluster state between themselves and coordinated VM failovers when required. HA used this cluster state to tell the approximate state of the resources on each host, by tracking reservations. HA didn't use vCenter or DRS to decide where to restart VMs. Instead, it used these replicated state-resource calculations and additional checks to see whether the VM's network and datastores were available to the host. Secondary hosts sent all their state information to the available primary hosts.

One of the primary hosts was always the active primary, and this primary host decided where to restart VMs, which order to start them in, and how to deal with VMs that fail to start. If the active primary host failed, one of the other primaries stepped up and took the role.

The primary hosts remained primary until they were put into maintenance mode, they became disconnected from the cluster, they were removed from the cluster, or the entire cluster was reconfigured for HA. At that point, a random secondary host was selected and promoted to primary status.

It's important to note that a secondary node wasn't promoted when a primary failed. This meant that if all the primary hosts failed before a secondary could be promoted, HA would stop functioning. No failovers would occur. One of the issues with this limit of five primary hosts was that there was no way in the vCenter Client to tell which hosts were primary and which were secondary. You could get this information from the command line, but there was no supported way to specify which host was which. The only supported action was to reconfigure HA on the cluster, which ran another random reelection and (you hoped) selected the physical servers you wanted. This process was tedious and not something you could plan your design around. The outcome was that for any design, you couldn't group more than four hosts from the same HA cluster in the same blade chassis. If you were particularly risk-averse, you wanted to avoid more than four hosts even being in the same rack.

Host Monitoring

After enabling HA in the cluster settings, you can begin protecting the hosts via their HA agents and heartbeats by switching on host monitoring. The check box to enable this feature is the first option in Figure 8.13.

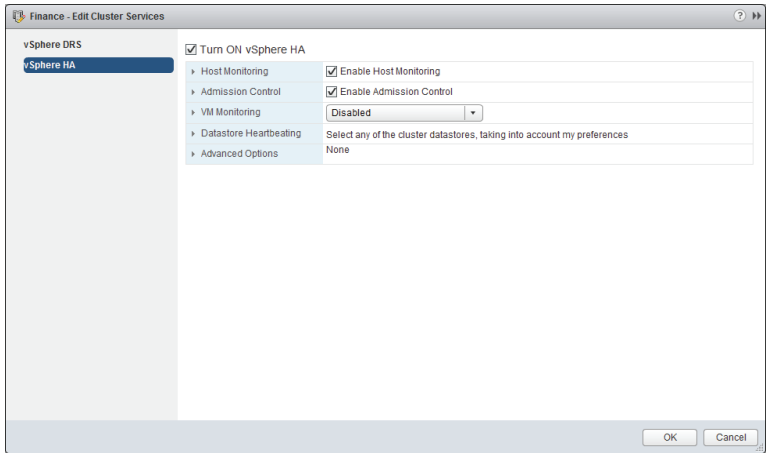


Figure 8.13
HA settings

If you wish to disable HA host monitoring, you should do it with the second setting rather than disable the entire cluster for HA. That way, you keep all your advanced settings, and VM and application monitoring can continue to function.

FT requires that HA host monitoring be enabled in order for it to protect its VMs properly. vSphere keeps the FT secondary copies running if you disable host monitoring, but they may not fail over as expected. This saves the FT clones from being re-created if HA host monitoring is disabled for a short time.

VM Options (HA)

HA's Virtual Machine Options settings let you control the default restart priority and what the cluster's hosts should do if a host becomes isolated from the rest of the network. Figure 8.14 shows these settings.

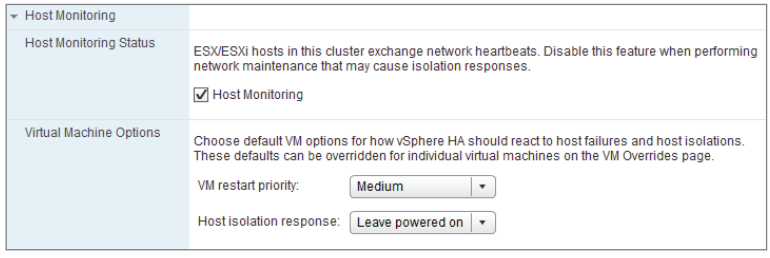


FIGURE 8.14
HA Virtual Machine Options

Setting the VM options gives the default actions for the cluster, but both settings can be overridden on a per-VM basis. Figure 8.14 shows how each VM can have its own restart and isolation response:

Restart Priority The VM Restart Priority setting allows you to specify the order in which VMs are restarted if a host fails. If resources are constrained and you didn't implement proper admission control, it's particularly important to have the more critical VMs start first. Otherwise, when it's time to restart the lower-priority VMs, there may not be enough resources to start them.

You can choose to start certain VMs first if they provide essential services, such as domain controllers and DNS servers. In addition, the order in which some VMs start is important for certain multiserver applications. For example, you may need a database server to start before its application server (as you would for a split virtual vCenter instance).

You can also disable restarting some VMs in the case of a host failure. For example, you may choose not to restart some development and test VMs if you think the extra load on fewer hosts will adversely affect your production VMs. VMs disabled here will react to the VM-monitoring feature on a host that is still up, unless it's disabled in that section as well.

If two hosts fail simultaneously or in quick succession, then the master host begins starting the VMs from whichever it determines was the first host to fail. It restarts all of that host's VMs in their restart priority and doesn't begin to restart the second host's VMs until all the

VMs from the first are finished. This is the case even if there are higher-priority VMs on the second host.

Host Isolation As described previously, a host isolation occurs when an ESXi server loses its connection to the network but continues to see its storage (if it can't keep the datastore heartbeat, then it's recognized by the master and itself to have *failed*, not become isolated). As a final check to differentiate itself from a partitioned host, it pings the cluster's isolation address. By default, the isolation address is the management network's default gateway. If it can't ping the isolation address, then it declares itself isolated and takes the action dictated by the cluster's isolation setting. A master declares itself isolated after 5 seconds if it can't communicate with any other hosts. A slave does this after 30 seconds.

Even though the VMs are still running, this is a less-than-ideal situation. There is clearly an issue with the infrastructure, and the VM's guest OSes are most likely no longer on the network.

The cluster has three possible settings for an isolation response: Leave Powered On, Shut Down, and Power Off. Shut Down attempts to use the VMware Tools to cleanly shut down the VM, thus preventing the guest OS from crashing. If after 5 minutes the VM hasn't shut down, the host powers it off hard. You can change this timing with the advanced setting `das.isolationshutdowntimeout` if there are VMs you know take longer to shut down cleanly.

In vSphere 5, before the isolated host shuts down or powers off each VM, it checks that the master host can lock the datastore. If it can't, then the isolated host doesn't shut down (or power off) the VM. It does this to make sure the master is in a state to be able to recover the VM. If it can't, you're better off keeping the VM running on the isolated host.

In vCenter 5 the default isolation is now Leave Powered On. This setting has changed a number of times during vCenter's lifecycle, so if you've been upgrading this server through the years, then you should check each cluster's setting. Once vCenter has been upgraded, all new clusters are created with the default setting, but existing clusters can maintain the old defaults.

Leave Powered On is the default because it's generally accepted as the safest option in most situations and prevents any false-positive worries. When should you not keep the default setting? Remember that when a host is isolated, it can see the datastore heartbeat volumes but can't connect to other hosts on the network. There are a couple of things to consider here. If the management interface of the host can't connect to the isolation address, then something is wrong with your infrastructure's networking. Depending on where the fault lies (the IP interface, a physical NIC, a cable, a switch), how likely is it that your VM's guest OSes are also offline? If you have a very redundant physical network set up, and the VM networks are physically and virtually separate from the management network, this is highly unlikely. But if you have a converged network of two 10GbE connections through one blade mezzanine card, then this is far more likely. If you're concerned that leaving the isolation response as the default may mean your VMs could stay turned on but not be on the network, then you might choose to shut them down. A second consideration is that although the isolated host can by definition see the heartbeat datastores, that doesn't mean it can

definitely see all the datastores. For example, if you have a mix of array connections, and the heartbeat datastores are on the more robust arrays' connections, then there is the possibility that an isolated host could lose some of the datastores and not others (particularly if you have an IP-based array.

If you opt not to stay with the default of Leave Powered On, then you have the choice of Shut Down or Power Off. VMs can individually override the cluster isolation response setting. Shut Down is obviously the more graceful approach and results in a lower probability of a crash-consistency issue when the VM is powered up on a different host. However, selecting Shut Down over Power Off means the recovery will take longer. If you're confident that a VM won't suffer any ill effect from a sudden power off, and the shortest downtime is paramount, then you can consider dropping down to the Power Off option to provide higher availability—isn't that ironic?

Admission Control

HA admission control is similar to the admission control used by hosts, resource pools, and storage DRS. One notable difference is that HA admission control can be disabled. It's used to control oversubscribing the HA cluster. If enabled, it uses one of three methods to calculate how to reserve sufficient resources to ensure that there are enough hosts for recovery; the following subsections describe these policies. If it determines that the limit has been reached, it can prevent further VMs from being turned on in the cluster or VM reservations from being increased. HA admission control doesn't include hosts that are disconnected, in standby, or in maintenance mode in its calculations.

Without the protection of admission control, there is a greater chance that HA may not be able to power on all the VMs during a failover event. You may want to temporarily disable it if you plan to oversubscribe your cluster during testing or patching. Remember that HA admission control is used to ensure sufficient capacity during failure events. It isn't meant as a general capacity-management tool. Proper host capacity management is about making VMs run well during normal operation.

HA admission control reserves capacity for times when hosts fail. When a failure does occur, it uses that capacity, powering-on VMs that admission control would ordinarily prevent a user from starting. Admission control is driven by vCenter, but recovering from failures is left to the master host to coordinate. Admission control in and of itself doesn't prevent VMs from being powered on during a failure; but without the correct admission control a lack of resources may cause VMs not to be started.

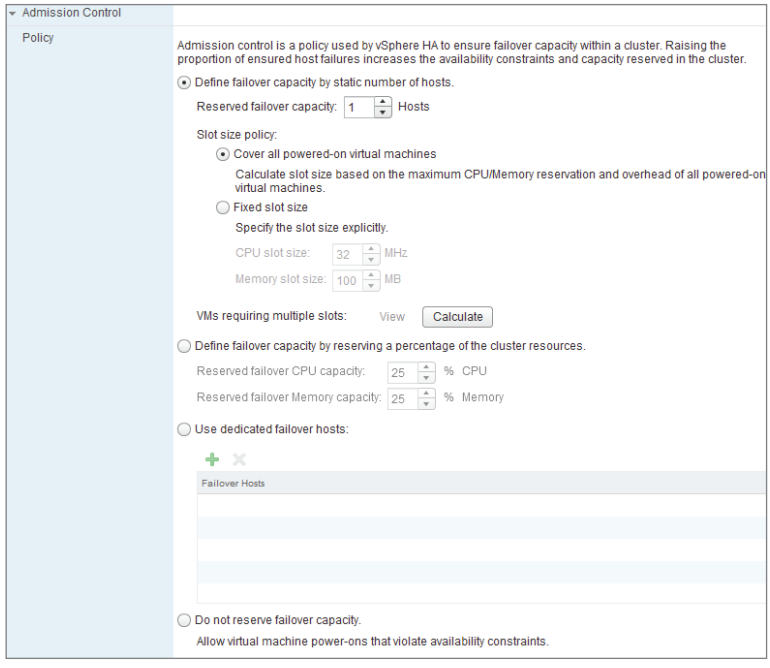
When DRS is enabled on the cluster, it attempts to create contiguous space on hosts. This reduces the potential of the available resources being too divided across hosts and allows larger VMs to be restarted.

Figure 8.15 shows the three policy options available when host monitoring is enabled.

Static Number of Hosts

The first admission control policy—and the default, which for that reason is used in most clusters—is the number of host failures the cluster will tolerate. In other words, it's the amount of capacity kept spare in terms of number of hosts.

FIGURE 8.15
Admission Control policies



HA calculates the amount of spare capacity in terms of number of hosts via an arbitrary value called a *slot size*. This slot size determines how admission control is performed against the cluster:

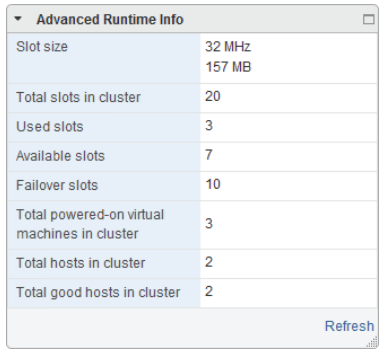
Slot Sizes Slot size is the value of the largest VM requirement in the cluster, or the worst-case scenario. It finds the largest CPU reservation and largest memory reservation of every powered-on VM. If no reservations are set for these values, it assumes a value of 32 MHz for the CPU and 0 MB plus the VM's memory overhead. It then takes the largest resulting CPU figure of any VM and the largest memory figure of any VM, and uses this as the cluster's slot size.

Obviously, very large VMs with large reservations will skew this amount, highlighting the need to keep VM reservations to a minimum. Resource pool reservations don't affect HA slot sizes. In the Windows Client, you can use the advanced settings `das.slotCpuInMHz` and `das.slotMemInMB` to reduce a very high slot setting if you think something lower is appropriate, but doing so can fragment the reserved slots for the larger VMs and cause issue if resources are thinly spread across the cluster when a host fails. Although the spare resources will be available, they may not be enough on a single host to accommodate a large VM. The Web Client in vSphere 5.1 allows you to change these values in the GUI via the Fixed Slot Size radio button option. If you opt to manually reduce the slot sizes to prevent this admission control policy from being too conservative, the GUI shows you which VMs are larger than the fixed slot size chosen. (If you're using the default options for this policy, not manually changing the slot size, resource fragmentation shouldn't be a significant issue because the slot size is set for the maximum reservations.)

Setting specific slot sizes manually can also be useful in situations where VM reservations aren't used. If no reservations are set, then only the most basic slot size is determined for a cluster. This is usually based on the memory overhead for the largest VM in the cluster. However, if all the VMs are very similar in size and no one VM is significantly bigger than the rest, then only the bare minimum needed to power on VMs will be guaranteed. This is the minimum for the hosts to power them on and often not enough to get the guest OS up and running. Setting no VM reservations in this case could lead to a serious overcommitment during a failover event. Manually setting a slot size can prevent this.

Runtime Information As shown in Figure 8.16, the vSphere Web Client has an HA Advanced Runtime Info dialog box. It's available only when you use this method of admission control. It highlights the slot size and host-capacity calculations.

FIGURE 8.16
HA Advanced Runtime Info



HA calculates how much CPU and memory are available for VMs on each host, after the virtualization overhead is taken into account. It does this for all connected hosts that aren't in maintenance mode and works out how many slots each host can hold. Both CPU and memory slot sizes are compared, and the smaller (more conservative) of the two is used. Starting with the smallest host, HA determines how many hosts will accommodate all the slots. Any hosts left over, which will be the larger ones to ensure that worst-case failures are considered, are totaled to give the failover capacity.

This method of admission control is useful primarily because it requires no user intervention after it's initially configured. New hosts are added to the slot size calculations automatically, to ensure that larger hosts are guaranteed protection. As long as you don't use advanced slot-size settings, this method doesn't suffer from the resource fragmentation that is possible with the percentages method discussed next.

But this type of admission control is inflexible and can be overly conservative. It works well if all the VMs have the same CPU and memory requirements; but a small number of large VMs will significantly bias the slot sizes, and you'll need extra hosts to maintain the failover capacity. If you're faced with a cluster with disproportionate VMs, carefully consider how you're using VM reservations, whether the cluster should be split, or if another policy is more appropriate.

Percentage of Cluster Resources

The Percentage of Cluster Resources admission control policy uses cluster totals compared to VM totals to calculate a percentage value of spare capacity.

This policy adds up all the powered-on VMs' CPU and memory requirements, considering their reservations. If the reservations are set to 0 (zero), then it takes 0 MB plus any overhead for memory and adds 32 MHz for CPU. It then adds together all of the hosts' available CPU and memory resources. It creates a percentage difference for both CPU and memory, and if either is less than the admission control policy states, it enforces admission control.

Using percentages for admission control is very useful because it doesn't rely on complex slot size calculations while continuing to respect any set reservations. Unlike the very conservative policy regarding a static number of host failures, the percentage policy is more adaptable. If all the hosts have similar CPU and memory, then calculating the percentages to give a one- or two-host failover capacity is simple. Although appearing reductive, 1/n×100 is commonly enough to give sufficient failover for a cluster (n+1), and its simplicity makes this easy to recalculate whenever the number of hosts in the cluster changes.

Hosts with very different capacities can lead to problems, if the percentage set is lower than the resources of the largest host. Every time a host is added or removed, you may need to recalculate the percentage each host gives to the cluster, to ensure that each is sufficiently protected. Very large VMs may have problems restarting if the spare capacity is spread thinly over the remaining hosts; so, set large VMs to have an earlier restart priority, and always watch to make sure any large VMs can still fit on each host.

This policy is well suited to clusters with a mix of VMs with different CPU and memory sizes. The Web Client in 5.1 allows you to specify different capacities for CPU and RAM if there is significant disparity. DRS does its best to defragment this and make space for VMs when required, but with the type of Admission Control policy there is always this risk.

Dedicated Failover Hosts

This policy designates a particular host or multiple hosts as the ones to fail over to, reserving entire hosts as a hot spares. When this policy is selected, no VMs can power on or migrate to the designated hosts.

This means the hosts are unavailable for use, and they need to be the most powerful hosts in the cluster to ensure that it can recover from any host failures. If you have unbalanced hosts in the cluster, with one host having a very large number of CPUs and another having much more RAM than the rest, then you may find that no particular host is suitable to specify as a single failover host. Being able to select multiple hosts with different hardware profiles can alleviate this but can obviously be costly to implement. This policy is only suitable for the most risk-averse or bureaucratic scenarios where protocols stipulate that reserved hardware has to be on standby to guarantee that no performance degradation can occur during failure.

Do Not Reserve Failover Capacity

This last option is the equivalent of disabling admission control, the nomenclature used in the Windows Client.

HA lets you power on VMs until all of the cluster's resources are allocated. This means that during a failover event, not enough resources might be available to power on all the VMs. We hope you've set the restart priority on the more critical VMs, so at least they can power on first! Needless to say, although operational needs may require you to temporarily switch to this mode, you shouldn't leave it this way for long because the VMs are potentially unprotected. Switching to this often is a result of poor capacity planning. It shouldn't be part of planned design.

VM Monitoring

VM monitoring is an additional feature of HA that watches the VMware Tools' heartbeat, looking for unresponsive guest OSes. If HA determines that the guest has stopped responding—for example, due to a Windows BSOD or a Linux kernel panic—then it reboots the VM.

With vSphere version 4.1, application monitoring was added. For an application to share heartbeat information with HA's monitoring, it relies on either the application's developers adding support via a special API available to select VMware partners, or setting them up yourself using the available SDK. If the application or specific service stops, VMware Tools can be alerted, and vSphere can restart the VM.

Figure 8.17 shows the available VM Monitoring settings. At the top, in the drop-down box, you can select VM, Application, or VM and Application Monitoring. The next section allows you to select the monitoring sensitivity, with an option to customize each setting individually. You can adjust the settings on a VM basis in the VM Overrides section of the cluster, shown previously in Figure 8.7.

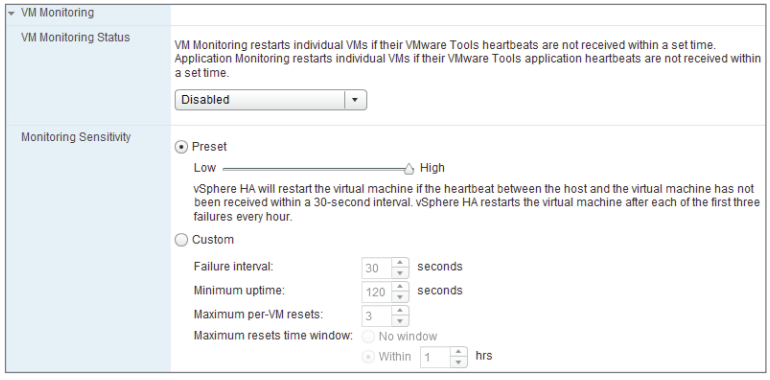


FIGURE 8.17
VM Monitoring

If heartbeats from the guest OS or application are lost from a VM in the Failure Interval period, disk and network I/O are checked to prevent any false results. The I/O activity interval is an advanced cluster setting, which by default is 120 seconds, but you can customize it with the `das.iostatsinterval` setting. HA VM monitoring has a minimum uptime value that waits a period of time after the VM is started or restarted before making monitoring failovers; this allows the VMware Tools to start. If you have a guest OS or application that is known to start very slowly, you should extend this value to be sure everything has started and the initial CPU demands have tapered off. This feature also limits the number of times a VM is reset in a certain period. This prevents a VM from being repeatedly restarted, if the same error continuously reoccurs and causes OS or application failures.

If HA decides a failure has happened after checking heartbeats and I/O, it takes a screenshot of the VM's console and stores it in the VM's working directory. The VM is then immediately restarted. This console screenshot can be very useful in troubleshooting the initial error, because it often contains kernel error messages from the guest OS.

Adjusting the monitor's sensitivity, especially with custom values, affects the monitoring responsiveness and success rate. If you set the sensitivity too high, then you may suffer unnecessary reboots, particularly if the VM is under a heavy workload or the host's resources

are constrained. On the other hand, if you set the sensitivity too low, then it will take longer to respond, and critical services may be down longer than you want.

A successful implementation of VM monitoring depends on testing each OS and application. Every instance may have different sensitivity requirements. For this reason, the VM Overrides section is very useful; it lets you test each VM with its own settings before introducing it to the cluster's settings. It's advisable to have as few VMs with their own individual settings as possible, after you've determined the most appropriate cluster-wide sensitivity settings.

Datastore Heartbeating

As described in the previous sections, HA in vCenter 5 uses the datastores as an additional way to communicate between hosts and prevent false positives. This provides extra redundancy over the network agent-to-agent checking and can more accurately determine faults.

By default, vCenter selects two datastores to use for heartbeating purposes. It picks the data stores that are connected to the most hosts, if possible ones that are split across different storage arrays, with a preference for VMFS volumes over NFS volumes.

Figure 8.18 shows where you can nominate a predilection for specific datastores. You can also use an advanced setting to increase the default of two datastores up to a maximum of five.

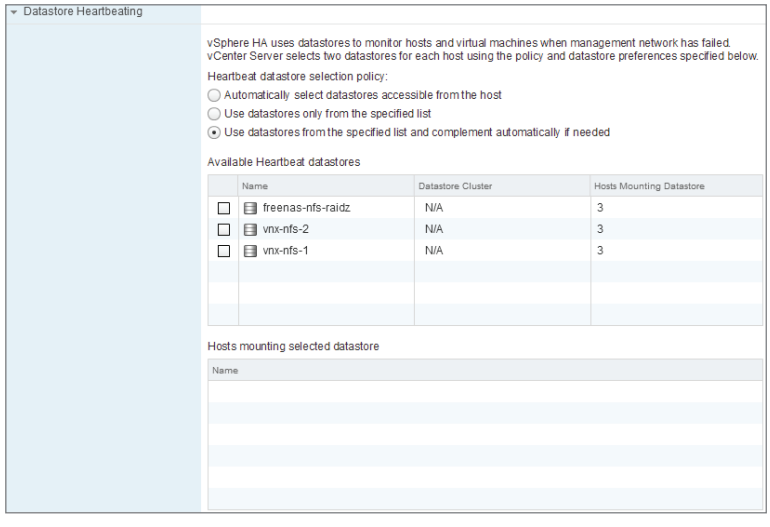


Figure 8.18
Datastore Heartbeating

Making changes to the datastores used for heartbeating isn't usually necessary in a design because vCenter's *smarts* attempt to spread the choice across multiple sources and favor FC fabrics where possible. However, if you have datastores that you know are likely to provide more redundancy or resilience, this is where you can enhance the defaults.

Advanced Options

Some of the advanced options for HA have been discussed throughout this chapter. This is the section in the GUI where advanced options can be manually added for a cluster. To review all

the possible HA advanced options that are currently available, refer to `http://kb.vmware.com/kb/1006421`.

One advanced setting worthy of note because of its absence in vSphere 5 is `das.failuredetectiontime`. This was a popular setting to customize in previous HA configurations because there were different VMware recommendations depending on your extenuating circumstances. This is no longer a customizable setting in vSphere 5. In vSphere 5.1, a setting called `das.config.fdm.isolationpolicydelaysec` was introduced to provide a somewhat similar ability, allowing you to delay the trigger of the isolation response. Ordinarily this isn't a requirement, and is not needed for additional isolation addresses. This setting has a minimum value of 30 seconds, the default isolation response time, and should only ever be increased if the network experiences drops of longer than 30 seconds, but circumstances when you know that you won't want HA to kick-in immediately.

HA Impacts

vSphere has a tight integration with its cluster components and is affected by DRS, DPM, and affinity rules. During failover, if resources are spread thinly over the cluster and HA needs to restart one very large VM, it can ask DRS to make room for it by vMotioning existing VMs around. It can also ask DPM to power-on hosts if it has shut some down, to provide additional resources.

The VM-Host affinity rules can limit the choices HA has to restart VMs after a host failure. If there is a VM-Host mandatory *must* rule, then HA honors it and doesn't restart a VM if doing so would violate the rule. If it's a soft *should* rule, then HA restarts the VM, thus breaking the rule, but creating an event that you can monitor with an appropriate alarm. Multiple VM-Host rules can also fragment the resources in a cluster, enough to prevent HA power-ons. HA asks DRS to vMotion VMs to a point where every VM can be recovered but may not be able to achieve it within the rules. As per the recommendations for VM-Host affinity rules, only use *must* rules if you have to, and be aware that doing so can limit HA so much that VMs may not be recovered.

For clusters that use resource pools, vSphere 5 recovers VMs to the root of the cluster but with adjusted shares relative to their allocation before the failover. This is temporary, because when DRS is also enabled on the cluster (which is required for resource pools), it moves the VMs back into the correct pools automatically.

HA Recommendations

The most crucial factor for HA to work successfully is the communication of the network and storage heartbeats. Therefore two critical design aspects related to this in HA are network redundancy and storage-path redundancy.

The most common risk to an HA cluster is an isolated host, particularly hosts that rely on IP-based storage instead of a FC fabric. You must ensure that there is complete redundancy for the management network (Service Console for ESX hosts) connection at all points through the network to every host. You can do either of the following:

- Configure more than one VMNIC on the management VMkernel's vSwitch.
- Add a second management VMkernel connection to a separate vSwitch on a separate subnet.

The management VMkernel connection must have a default gateway that all the hosts can reach. Hosts use it to decide whether they have become isolated; so, if your management VMkernel subnet doesn't have a gateway or the gateway device doesn't respond to ICMP pings, you should change the `das.usedefaultisolationaddress` parameter to "false" and add an alternative isolation address. To add an alternate isolation address, or if you use a second management VMkernel connection on a separate subnet and want to specify a second isolation address for the alternate subnet, add `das.isolationaddress0`. With the improvements to HA that allow the new default isolation response to be Leave Powered On, there should be less concern regarding false results shutting down VMs. But if you're conducting maintenance tasks on your network devices or the host's networking configuration, you should temporarily disable host monitoring to prevent any false failovers.

Try to minimize the number of network devices between hosts, because each hop causes small delays to heartbeat traffic. Enabling PortFast on the physical switches should reduce spanning-tree isolation problems. If you use active/passive policies for the management VMkernel connections, configure all the active VMkernel links from each host to the same physical switch. Doing so helps to reduce the number of hops.

If only IP-based storage is being used in the design, then you should try to split the networks as much as possible from your management networks. Completely separate fabrics would be ideal, providing physical air-gapped equipment. However, for practical reasons this isn't often possible. Perhaps you have to share the access-layer network switches. Depending on the network topology, the storage may be connected to common aggregate or core switches. If you aren't using converged 10GbE cabling, it may be feasible to allocate a pair of redundant storage-only cables to the northbound switches. If physical separation isn't an option, then at least splitting out the subnets on trunked ports gives a logical split.

A firewall port must be open between all hosts in the same cluster. When HA is enabled on the cluster, the required ports are automatically opened on the host's own firewalls. However, if there are any physical firewalls between the hosts, then TCP/UDP port 8182 needs to be configured.

If the cluster is configured for DRS, you should already have common port group names across all the hosts; otherwise, the vMotions will fail. But if you only enable HA, it's possible to misconfigure your hosts with different port group names between the hosts. Use the Networking view in vCenter of the cluster to confirm consistent naming standards.

Attempt to keep the host versions at the same version and updated to the latest build whenever possible. It's possible to have mixture of vSphere 3.5 hosts and upward, but older hosts don't support all the same features or react quite the same way. To be assured of the most efficient cluster that is as highly available as possible, update the hosts frequently.

Customize the VMs' restart policy to suit your environment. Doing so improves the chances that the more critical servers will be restarted successfully if you don't have admission control enabled or if the failover levels are insufficient. Also, the restart policy will ensure that the servers that need to start first, which subsequent servers rely on, will be ready.

Create an alarm with an associated email action to warn you if an HA failover has occurred. You'll want to check that all the necessary VMs come back online and that services are restored. Then, investigate the state of the failed host to see if it's ready to rejoin the cluster. You can also use alarms to warn you when a cluster becomes overcommitted or invalid.

HA in Stretched Clusters

There is a special cluster design known officially by VMware as a vSphere Metro Storage Cluster (vMSC), although more commonly referred to as simply a stretched cluster (or metro, metropolitan or campus cluster depending on who you talk to). A stretched cluster describes when the hosts and storage are physically *stretched* across two sites, but they all reside within one cluster in vCenter. This allows VMs from both sites to participate in a conjoined DRS logical group to share and balance resources, while offering the potential to protect against several failures including entire room failures with HA automated recovery.

Stretched clusters are an interesting extreme of cluster design concepts, and as such deserve special planning and testing if implemented. VMware has produced an excellent whitepaper on this scenario at `www.vmware.com/resources/techresources/10299` if you are contemplating it. But as a primer, the following section provides an overview of the specific design considerations.

Firstly, for a stretched cluster to meet VMware's certified vMSC guidelines, it requires a minimum of the following:

- Synchronous storage replication of datastores between sites using either FC, FCoE, or iSCSI protocols. Datastores backed by NFS are not supported.
- Storage and network latency less than 5 ms between sites is required. A vSphere Enterprise Plus license includes the Metro vMotion feature which allows the network latency to be up to 10 ms.
- A minimum of 622 Mbps network bandwidth for vMotion traffic between sites. This does not include the storage replication traffic.
- Each datastore must not only be replicated, but the storage array in each room must be able to read and write to the datastore.

There are several recommendations for stretched clusters that improve HA protection. It is important to add these to any such design.

- It is recommended to have an equal number of (and similarly sized) hosts on both sites.
- Use the *Percentage of Cluster Resources* admission control policy and set this to 50% for both CPU and Memory. This ensures that during an HA recovery there will be enough resources to recover from an entire site failure. If you don't have an equal split of host resources on each site, these percentages will need to be even greater.
- Create VM-to-Host *should* affinity rules to group the Hosts into two sides and align the VMs with their primary datastores to prevent excess cross-site replication and network traffic.
- By default each cluster selects two datastores to act as a storage heartbeat for HA. Make sure that a datastore from each site is used, and preferably add another so there is a pair of data stores per site.
- Add a second isolation address (`das.isolationaddress`) so there is a network device in both sites. This help all hosts deal with a split room scenario.

- Since vSphere 5.0 update 1, two new advanced options allow us tune the hosts to respond more appropriately to Permanent Device Loss (PDL) codes from a storage array. When a host sees that a LUN has failed, but can still communicate with the array, it is able to recognize that the failure is not a temporary fault and the LUN is most likely not coming back — this is labeled a PDL. The two advanced options enable HA to restart VMs from a PDL datastore that is still available to other hosts in the cluster. The follow settings should be changed in a stretched cluster:
  - `disk.terminateVMonPDLDefault` set to `true`. Allows HA to kill a VM if its home data store is in PDL state.
  - `disk.maskCleanShutdownEnabed` set to `true`. Makes an HA cluster assume a powered off VM on a PDL datastore as failed and restarts it.

Fault Tolerance

VMware FT is a clustering technology that was introduced with vSphere 4 but whose roots can be traced back to the record/replay feature first introduced in VMware workstation back in 2006. It creates an identical running copy of a VM on another host, which can step in seamlessly should the first VM's host fail.

FT records all the inputs and events that happens on the primary VM and sends them to replay on the secondary VM, in a process known as vLockstep. Both primary and secondary VMs have access to the same VM disks via shared storage, so either can access the disks for I/O.

Both VMs are kept synchronized and can initiate the same inputs and events identically on their respective hosts. Only the primary VM advertises itself on the network, so it can capture all the incoming network I/O, disk I/O, and keyboard and mouse input. These inputs and events are passed to the secondary VM via a dedicated VMkernel connection known as the FT *logging link*. They can then be injected into the running secondary VM at the same execution points. All the outputs from the secondary VM are suppressed by its host before being committed, so only the primary actually transmits network responses to clients and writes to disk. The outside world is oblivious to the secondary VM's existence.

The mechanism to create and maintain the secondary VM is similar to vMotion; but instead of fully transferring control to the second host, it keeps the secondary VM in a permanently asynchronous mirrored state. The initial creation of the FT VM happens over the vMotion network; then, the remaining lockstep occurs over the FT logging network. This clustering is only for host failures and doesn't protect VMs against OS or application failures, because those are replicated to the second VM. But FT does protect VMs against host failures more effectively than HA, because there is no downtime or loss of data, whereas HA needs to restart the VMs, creating a short outage and crash-consistent VMs.

The VMs need to be members of an HA cluster, to detect and restart hosts should a failure occur. Like HA, FT uses vCenter purely for the initial creation of the FT pair but isn't dependent on vCenter and isn't disrupted if it becomes unavailable. The cluster is also responsible for keeping the VMs on different hosts, so when FT is enabled on a VM, the secondary VM is created and never coexists on the same hardware as the primary.

The primary VM is continuously monitored for failures so the secondary can be promoted, be brought onto the network, and spawn a new secondary copy. Failures are detected by checking the VMs for UDP heartbeats between the servers. FT also watches the logging connection to prevent false positives, because regular guest OS timing interrupts should always create a steady stream of logging traffic. FT uses a file-locking technique, similar to vSphere 5's HA, on the shared datastore to tell if there is a split-brain networking issue instead of a host down. This prevents an isolated host from incorrectly trying to promote the secondary VM and advertise itself.

FT by itself only protects a VM from host failure, not OS or application failures. However, you can combine this with HA's VM monitoring and application monitoring. The VM monitoring detects OS problems such as kernel panics/BSODs and restarts the primary VM. This in turn causes the primary to create a new secondary copy. Remember that because the primary and secondary VMs always share the same storage, FT doesn't protect you against a SAN failure.

FT Versions

The FT feature maintains a version number between revisions of vSphere to ensure that hosts are compatible with each other and capable of mirroring VMs. When FT was introduced in vSphere 4.0 with version 1, the build number (effectively, the patching level of the host) was compared: VMs could only run on hosts with exactly the same build number. Since vSphere 4.1 (FT version 2 and above), FT isn't restricted by build numbers, but hosts must still be at the same FT version number.

Version 2 and above FT VMs have a more sophisticated version-control technique that allows the hosts to be slightly different if vCenter can tell FT would be unaffected. This makes patching the hosts considerably simpler. But upgrading hosts between major vSphere revisions still requires a minimum four-host cluster without having to disable FT for an extended period. Ensuring that all hosts in a cluster are at the same vSphere version, and therefore the same FT version, is important in maintaining the most effective environment for FT.

FT can use DRS for both initial placement and load-balancing, but the cluster must be EVC enabled. EVC improves the cluster's ability to place VMs on hosts. The secondary VM assumes the same DRS settings as its primary. If EVC is disabled in the cluster, the FT VMs are DRS disabled as they were with version 1 back in vSphere 4.0. Whenever a FT VM is powered on, it uses anti-affinity rules to ensure that the secondary copy is running on a different host. You can use VM-Host affinity rules to specify the hosts on which you want the FT VMs to run. When you set particular hosts, both the primary and secondary VMs adhere to the rule and only run on those hosts. But with a VM-VM affinity rule, this only applies to the primary, because that is the VM other VMs interact with. If the primary fails, the secondary is promoted, and DRS steps in and moves VMs as the rule requires.

As each version of vSphere, and therefore FT, is released, support for newer CPUs and guest OSes become available. For this reason, it's generally good advice to keep the FT hosts updated as newer releases become available. VMware's online compatibility guide (known as the HCL) at `www.vmware.com/go/hcl` shows the supported hardware and guests: select Fault Tolerant (FT) in the Features section.

vLockstep Interval

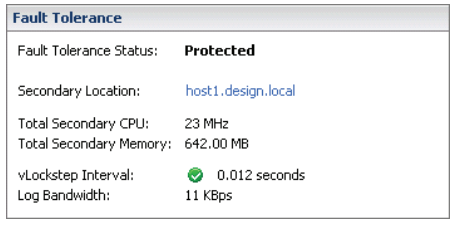
The primary VM always runs slightly ahead of the secondary VM with regard to actual physical time. However, as far as the VMs are concerned, they're both running at the same virtual

time, with inputs and events occurring on the secondary at the same execution points. The lag between the two VMs in real time is affected by the amount of input being received by the primary, the bandwidth and latency of the logging link between the two, and the ability of the host running the secondary VM to keep up with the primary VM.

If a secondary VM starts to significantly lag the primary, then FT slows the primary by descheduling its CPU allocation. When the secondary has caught up, the primary's CPU share is slowly increased.

The lag between the two VMs is known as the *vLockstep interval* and appears in the FT summary shown in Figure 8.19. The vLockstep interval typically needs to be less than 500 ms.

FIGURE 8.19
FT summary



All network transmits and disk writes are held at the primary VM until the secondary acknowledges that it received all the preceding events to cause the output. Therefore, sufficiently large latency on the logging link can delay the primary, although normal LAN-style response times shouldn't cause a problem.

Requirements and Restrictions

The process of recording and replaying VMs is very complex; hence a number of strict requirements and restrictions apply to the environment.

This is a relatively new feature from VMware, and it's still evolving; as such, it has a tendency to change frequently. You should check the latest VMware documentation to make sure these restrictions still apply, because each new version works to remove the existing constraints and improve functionality.

The following lists are aimed at the version of FT that came with vSphere 5.1, but additional prior limitations or differences are highlighted:

Clusters

- At least two hosts that can access the same networks and storage that the FT-protected VMs will use. Hosts must be running a compatible FT version. Host certificate checking must be enabled on the cluster. This has been the default since vCenter 4.1, but upgraded installations may not have it enabled.
- The cluster must have HA enabled with host monitoring.
- Primary and secondary VMs can't span across multiple clusters.
- If cluster hosts are separated from each other by firewalls, FT requires ports 8100 and 8200 to be open for TCP and UDP between all hosts.

Hosts

- Enterprise or Enterprise Plus licensing.
- FT-compatible hardware (see the FT HCL for updated listings).
- FT-compatible CPUs (`http://kb.vmware.com/kb/1008027`).
- Hardware virtualization enabled in the BIOS (Intel VT or AMD-V).
- Access to the vMotion network via a VMkernel connection.
- Access to the FT logging network via a VMkernel connection. The FT logging network should be a separate subnet to the vMotion network. IPv6 isn't supported.

VMs

- Must be at least hardware version 7.
- Must be on shared storage.
- A few guest OSes aren't supported, and some that are supported need to be powered off to enable FT (`http://kb.vmware.com/kb/1008027`).
- Only a single vCPU.
- No more than 16 disks.
- Maximum 64 GB of memory.
- Can't use IPv6.
- Can't use Storage vMotion and therefore can only use the initial placement feature of Storage DRS.
- Can't use or have snapshots, which rules out the VMware Data Protection (VDP) tool and any vStorage APIs for Data Protection (VADP) based backup utilities.
- Can't use or have linked clones.
- No physical raw device mapping disks (RDMs); virtual RDMs are allowed.
- No N-Port ID virtualization (NPIV).
- No NIC or HBA passthrough (DirectPath I/O or SR-IOV).
- No 3D graphics.
- Only virtual BIOS firmware; EFI firmware isn't supported.
- VMs can't have any of the following devices attached:
  - Vlance vNICs
  - VMXNET 3 vNICs with FT version 1
  - PVSCSI devices
  - VMCI connections

- Serial ports
- Parallel ports
- CD drives
- Floppy drives
- USB passthrough devices
- Sound devices

VMware has created a freely downloadable tool called SiteSurvey (`www.vmware.com/support/sitesurvey`) that can connect to your vCenter instance and analyze clusters, hosts, and VMs to check whether they're suitable for FT. It highlights any deficiencies and makes recommendations.

Enabling FT

A two-step process is involved in using FT on a VM. First FT is *turned on*, which prepares the VM for FT. When FT is turned on for a VM, it disables the following features:

- Nested page tables (Extended Page Tables/Rapid Virtualization Indexing [EPT/RVI]). If this has been enabled, then the VM must be turned off to disable it. FT only does this per VM, so the host remains enabled, and other VMs on the host can still take advantage of this hardware optimization.
- Device hot-plugging.
- DRS is disabled on the primary and secondary unless EVC is enabled on the cluster.

Turning on FT also converts all of the VM's disks to thick-provision eager-zeroed format, removes any memory limits, and sets the VM's memory reservation to the full allocation of RAM.

The second step is known as *enabling FT*, which creates the secondary VM. If the VM is already powered on when you turn on FT, it's automatically enabled at the same time.

When to Use FT

FT provides excellent protection against host failures, despite not protecting VMs from OS or application faults. But some of the restrictions listed here make FT less useful for some designs.

In particular, two requirements severely limit how it's used. First, the licensing needed to enable FT on VMs is Enterprise or above, meaning that many companies can't take advantage of this feature. Second, the fact that only single-vCPU VMs can be protected really restricts its use. The sort of Enterprise customers who pay for additional licensing features are also the entities most likely to run their more critical VMs with multiple vCPUs.

With the long list of restrictions and the additional demands to run the secondary VMs, most businesses only protect their most crucial VMs with FT. Not only does the cluster need to provide the extra resources for the secondary, but enabling FT also sets the full memory reservation for the primary, and in turn the secondary, to prevent any chance of ballooning or swapping. Enabling FT on the VM prevents you from changing any CPU or memory shares, limits, or reservations.

Enabling FT can also reduce the performance of VMs by up to 10% just with the additional overhead. If the secondary VMs CPUs can't keep up with the primary, this can reduce its performance even more.

DRS also restricts the number of primary and secondary VMs per host to a default of four. You can change this value with the advanced setting `das.maxftvmsperhost`; if you set it to 0 (zero), the limit is ignored.

However, the higher protection afforded by FT makes it extremely useful. Most popular guest OSes are supported; and FT is unaware of the application stack running, so the best use cases tend to rely on the services that are most important to the business. Because this feature is in addition to HA, it makes sense to use it only when a quick HA failover is unacceptable. You may want to consider using FT in your design in these cases:

- For the most business-critical applications
- During periods when a particular application is more crucial to the business
- When there are no other appropriate clustering techniques for the application

Scheduled tasks are available to turn FT on and off, so VMs can be protected on demand. You can automate when FT is used, so the additional resources are consumed only during those periods when you need the protection.

FT Impacts

FT works very closely with HA and also interacts with DRS functions. You should note some of these impacts while considering your design.

As we stated earlier, VM-VM affinity rules apply only to the primary VM, because that is where all the I/O traffic is directed. FT keeps both VMs apart through its own hidden anti-affinity rule, so you can't force them together on the same host.

If the primary has an additional VM-VM affinity rule, and there is an affinity or anti-affinity between the primary and another VM, then when a failover happens a violation can occur. DRS can't move the newly promoted secondary, and its original position may be in conflict with the rule. With EVC enabled on the cluster, DRS can move the promoted secondary VM to avoid any rule violation.

The VM-Host affinity rules apply to both primary and secondary, so both are effectively kept in the same VM group and have an affinity or anti-affinity with a group of hosts. This means that even if the FT fails over to the secondary VM, it will also be in the correct group of hosts for licensing or hardware association as required.

FT reserves the full allocation of memory on the primary VM and removes any previously set limits. To assist the secondary VM with keeping pace with the primary's host, you have the option of setting a CPU reservation as well. Both these reservations can have a significant impact on HA. The likely candidates for FT protection will probably have a reasonable amount of RAM allocated to them, and setting several FT VMs in the same cluster will change the HA slot sizes and so HA's efficiency. With larger VM reservations set in the cluster, DRS finds it harder to turn on more and more VMs if strict admission control is set. HA includes both the primary and secondary FT VMs when it's calculating admission control.

Without EVC enabled, you can't use DRS for active load-balancing the FT VMs, which means DPM may not be able to evacuate a selected host to shut it down. DPM won't recommend powering off any hosts running an FT VM if it can't vMotion it.

HA is able to protect FT VMs when a host fails. If the host running the primary VM fails, then the secondary takes over and spawns a new secondary. If the host running the secondary VM fails, then the primary creates a new secondary. If multiple hosts fail, which affects both primary and secondary, then HA restarts the primary on an alternate host, which in turn automatically creates a new secondary. During an HA failover, HA can intelligently place the new primary, and FT should create the new secondary on the most suitable host.

HA's VM monitoring feature can detect if the primary VM's guest OS or a specific service fails. When it does detect this, it restarts the primary VM. Once FT sees that the sync between the primary and secondary has failed, it instantly re-creates a replacement secondary VM. In this way, FT can work alongside VM monitoring to provide a solution that can protect against host, VM, and application failures.

FT Recommendations

Along with the long list of requirements and restrictions that FT places on hosts and VMs, a number of design recommendations improve the efficiency of FT:

Clusters

- All cluster hosts should have access to same networks and shared storage. Otherwise, the hosts selected for the secondary VMs will be limited.
- It's important that all the hosts have similar performance characteristics so primary and secondary VMs can run at similar speeds. VMware recommends that host CPUs shouldn't vary more than 400 MHz.
- All cluster hosts should have the same power-management, CPU instruction sets, and HyperThreading BIOS settings.
- Enable EVC on the cluster so the FT VMs can participate in DRS.
- You can create a resource pool for all your FT VMs with excess memory, to ensure that VM overhead is accounted for. Because FT is turned on, the VM's memory reservation is set to the full amount of allocated RAM. But this doesn't account for any memory overhead.

Hosts

- You should have fully redundant network and storage connections. FT will fail over a VM if the host loses all paths to its Fiber Channel (FC) connected datastore, but this shouldn't replace redundant FC links.
- The use of active/passive links with other VMkernel traffic provides a simple way of providing logging-traffic redundancy.
- You should split the vMotion and FT logging subnets and use separate active links for each.
- You should use at least 1 GbE links for the logging network. Using jumbo frames can improve the efficiency of the logging network even further. The FT logging connection is likely to be the limiting factor on the number of FT VMs you can have on each host. If you're consolidating host traffic via 10 GbE links, you need to implement some sort of QoS for the FT logging link to prevent any bursty traffic from saturating the link. For example, vMotions can use up to 8 Gbps of bandwidth, and each host can vMotion eight VMs at once. Without control, this can flood the connections shared with FT logging.

- Adding more uplinks on the logging connection's vSwitch doesn't necessarily increase distribution of the traffic across multiple links. You can use IP-hash load-balancing with EtherChannel to provide some load-balancing across multiple links. Source port ID or source MAC address load-balancing policies won't balance logging traffic.
- Host time should be synchronized to an available Network Time Protocol (NTP) server.
- VMware advises that you run at most four primary VMs and four secondary VMs on each host. Because the logging traffic is largely asymmetric from primary to secondary, spreading them between hosts improves the use of the logging connection. Also, because all the network I/O and disk I/O is being directed to and from the primary, the primary's host has more load that can be balanced by splitting primary and secondary VMs.
- The network used for FT logging should be sufficiently secure because this traffic is unencrypted and contains network and storage I/O that could potentially carry sensitive data.

VMs

- If the secondary VM is struggling for CPU resources on its host, causing the primary to slow down, you can add a CPU reservation to the primary that will be duplicated to the secondary. This should help with CPU contention on the secondary host.
- When you initially turn on FT for a VM, the disks are converted to eager-zeroed thick, which can take time to process. You can convert the disks ahead of time, during quieter periods, to ensure that FT can be turned on more quickly.
- Turning on FT disables some performance features, such as nested page tables. Don't turn on FT unless you're going to use it.
- If you're enabling FT on VMs with more than 64 GB of memory, you can encounter issues because there may be insufficient bandwidth to transfer all the changes in the default vMotion timeout of 8 seconds. You can increase the timeout in the VM's configuration file, but doing so may lead to the VM being unresponsive longer when you enable FT or during failovers or vMotions.
- Enabling FT causes spikes in CPU and saturates the vMotion network while the secondary VM is created. Try to avoid enabling and disabling FT frequently.
- The secondary VM uses additional resources on its host: CPU and memory levels equivalent to that of the primary. So, don't enable VMs unnecessarily.
- High guest OS timer interrupts cause increased logging traffic. Some Linux OSes allow you to reduce this setting, and you may wish to consider doing so if the logging bandwidth is constrained.
- Each FT VM's guest OS should have its time synchronized to an external time source.

Summary

This chapter has looked carefully at vCenter's datacenter structures and the functionality available to protect and optimize your clusters.

A clever datacenter design allows you to manage the hosts and VMs more efficiently and more intuitively. Fortunately, many of the structural pieces can be changed and honed easily, often with little or no downtime, so retrospective design changes may be relatively painless. It's worth considering your entire vSphere environment, because the power of vCenter's permissions, events/tasks, alarms, scheduled tasks, and views can make a big difference; this central infrastructure tool is simple to manage and effective, which enables staff instead of burdening them.

There are several cluster tools to make the most of your physical resources. First, you can use resources pools to divide and allocate CPU and memory as required, ensuring that the VMs that need resources receive them fairly and as you want them apportioned. DRS is critical to resource management; it places VMs appropriately on hosts and keeps the cluster suitably balanced with vMotion. You can achieve more granular control over VM placement with affinity rules and VM-Host rules. Finally, DPM makes power savings possible by shutting down hosts that aren't required.

In addition to all the resource optimizations, you can design clusters to provide differing levels of protection for your VMs. HA can restart VMs on alternate hosts, VM monitoring watches for unresponsive guest OSes and applications, and FT maintains online stand-in copy VMs. The rewritten HA mechanism in vSphere 5 not only improves the stability and scalability of this feature but also removes many of the awkward design constraints necessary to accommodate its predecessor.

Datacenter design is very important to the overall efficiency of your vSphere environment. It can enhance the working ability of your hosts and their VMs, making the most of your CAPEX layout and reducing the accompanying OPEX overhead. It's one part of the design worth reevaluating regularly, because nothing remains static and there are usually areas that can benefit from further optimizations. When you're looking at the datacenter design, it's important to examine it as a whole, because so many of the elements interact, work together, and even constrain each other. Think of all the elements in this chapter, and apply each part to best effect.

Chapter 9

Designing with Security in Mind

In this chapter, we'll change our point of view and look at a design from the perspective of a malicious user. We won't say *hacker*, because typically your virtual infrastructure isn't exposed to the outside world. Because this book isn't about how to protect your perimeter network from the outside world, but about vSphere design, we'll assume the security risks that need to be mitigated come primarily from the inside. We'll discuss these topics:

- The importance of security in every aspect of your environment
- Potential security risks and their mitigating factors

Why Is Security Important?

We're sure you don't need someone to explain the answer to this question. Your personal information is important to you. Your company's information is no less important to your company. In some lines of business, the thought of having information leak out into the public is devastating. Consider the following theoretical example. Your company is developing a product and has a number of direct competitors on the market. Due to a security slip, the schematics for a new prototype of your product have found their way out of the company. Having the schematics in the open can and will cause huge damage to the company's reputation and revenue. There's a reason that security is one of the five principles of design that we've discussed throughout this book.

At VMworld 2010, the slide in Figure 9.1 was presented, showing that the number of virtual OS instances is larger than the number deployed on physical hardware—and that number is only expected to rise.

Your corporate databases will run as VMs, your application servers will be virtual, and your messaging servers will also run as VMs. Perhaps all of these are already running as VMs in your environment. There is also a good chance that in the not-too-distant future, your desktop and perhaps even your mobile phone will be VMs (or, at the very least, will host one or more VMs). You need to ensure that the data stored on all these entities is as safe and secure as it can possibly be.

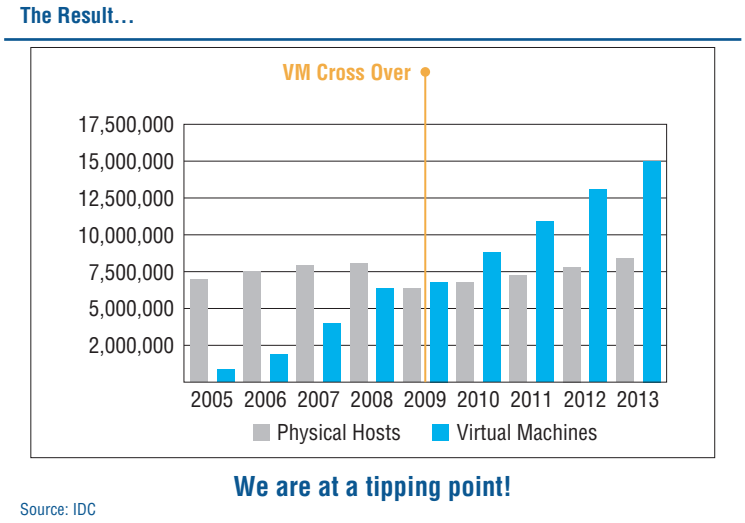


FIGURE 9.1
The number of VMs deployed will only increase in the future.

Because this chapter focuses specifically on security, we'll discuss these top-level security-related topics:

- Separation of duties
- vCenter Server permissions
- Security in vCenter Linked Mode
- Command-line access to ESXi hosts
- Managing network access
- The demilitarized zone (DMZ)
- Protecting the VMs
- Change management
- Protecting your data
- Auditing and compliance

In each section, we'll provide an example risk scenario followed by recommendations on mitigating that potential risk. Let's start with a discussion on the separation of duties.

Separation of Duties

A centralized model of administration can be a good thing but can also be a major security risk.

Risk Scenario

John, your VI Admin, has *all* the keys to your kingdom. In Active Directory, John has Domain Admin credentials. He also has access to all the network components in your enterprise. He has

access to the physical datacenter where all the ESXi hosts are located. He has access to the back-end storage where all your VMs are located and to all of the organization's Common Internet File System (CIFS) and Network File System (NFS) shares that store the company data.

And now John finds out that he will be replaced/retired. You can imagine what a huge security risk this may turn out to be. John could tamper with network settings on the physical switches. He could remove all access control lists (ACLs) on the network components and cause damage by compromising the corporate firewall, exposing information to the outside, or perhaps even opening a hole to allow access into the network at a later date.

John could also tamper with domain permissions. He could change passwords on privileged accounts and delete or tamper with critical resources in the domain.

John could access confidential information stored on the storage array, copy it to an external device, and sell it to the competitors.

John could then access the vCenter Server, power off several VMs (including the corporate mail server and domain controllers), and delete them. Before virtualization, John would have had to go into the datacenter and start a fire to destroy multiple servers and OSes. In the virtual infrastructure, it's as simple as marking all the VMs and then pressing Delete: there go 200 servers. Just like that. Clearly, this isn't a pretty situation.

Risk Mitigation

The reason not to give the keys to the kingdom to one person or group is nightmare scenarios like the one we just described. As we've discussed elsewhere, part of a comprehensive vSphere design is a consideration of the operational issues and concerns—including separation of duties and responsibilities.

Ironically, the larger the environment, the easier it becomes to separate duties across different functions. You reach a stage where one person can't manage storage, the domain, the network, and the virtual infrastructure on their own. The sheer volume of time needed for all the different roles makes it impossible.

Setting up dedicated teams for each function has certain benefits, but also some drawbacks:

Benefits Each member's expertise is concentrated on one field and not spread out over a number of duties. This allows them to focus on their specific duties and become experts in their field.

Each team manages its own realm and can access other realms but doesn't have super-user rights in areas not under its control. Active Directory admins don't have full storage rights, network admins don't have Domain Admin privileges, and so on.

Drawbacks There will be only one (maybe more) expert in any certain field. If that expert is sick, is away on vacation, or leaves the company, the rest of your team has to do a crash course in solving complex issues in a time of need. You'll find it difficult to have complete coverage on your team.

Having only one person dealing with the infrastructure doesn't allow for out-of-the-box thinking. The same person could be the one who designed and implemented the entire infrastructure and has been supporting it since its inception. Therefore, this person could be deep in a certain trail of thought, making it extremely difficult to think in new ways.

Structuring your IT group according to different technology areas furthers the IT silos that are common in many organizations. This can inhibit teamwork and cooperation between the different groups and impair the IT department's ability to respond quickly and flexibly to changing business needs.

Although ensuring a proper balance in the separation of duties is primarily an operational issue, there are technical aspects to this discussion as well. For example, vCenter Server offers role-based access controls that help with the proper separation of duties. Some of the potential security concerns—and mitigations—for the use of vCenter Server's role-based access controls are described in the next section.

vCenter Server Permissions

vCenter Server's role-based access controls give you extensive command over the actions you can perform on almost every part of your infrastructure. This presents some challenges if these access controls aren't designed and implemented carefully.

Risk Scenario

Bill is a power user. He has a decent amount of technical knowledge, knows what VMware is, and knows what benefits can be reaped with virtualization. Unfortunately, with knowledge come pitfalls. Bill was allocated a VM with one vCPU and 2 GB RAM. The VM was allocated a 50 GB disk on Tier-2 storage. You delegated the Administrator role to Bill on the VM because he asked to be able to restart the machine if needed.

Everything runs fine until one day you notice that the performance of one of your ESXi hosts has degraded drastically. After investigating, you find that one VM has been allocated four vCPUs, 16 GB RAM, and three additional disks on Tier-1 storage. The storage allocated for this VM has tripled in size because of snapshots taken on the VM. As a result, you're low on space on your Tier-1 storage. This VM has been assigned higher shares than all others (even though it's a test machine). Because of this degradation in performance, your other production VMs suffered a hit in performance and weren't available for a period of time.

Risk Mitigation

This example could have been a lot worse. The obvious reason this happened was bad planning and a poor use of vCenter Server's access controls.

Let's make an analogy to a physical datacenter packed with hundreds of servers. Here are some problems you might encounter. Walking around the room, you can do the following in front of a server:

- Open a CD drive and put in a malicious CD.
- Plug in a USB device.
- Pull out a hard disk (or two or three).
- Connect to the VGA port, and see what's happening on the screen.
- Switch the hard disk order.
- Reset the server.
- Power off the server.

Going around to the back of the server, you can do the following:

- Connect or disconnect a power cable.
- Connect or disconnect a network port.
- Attach a serial device.
- Attach a USB device.
- Connect to the VGA port, and see what's happening on the screen.

As you can see from this list, a lot of bad things could happen. Just as you wouldn't allow users to walk into your datacenter, open your keyboard/video/mouse (KVM) switch, and log in to your servers, you shouldn't allow users access to your vSphere environment unless they absolutely need that privilege. When they do have access, they should only have the rights to do what they need to do, and no more.

Thankfully, vSphere has a large number of privileges that can be assigned at almost any level of the infrastructure: storage, VM, and network cluster. Control can be extremely granular for any part of the infrastructure.

Let's get back to our analogy of the physical server room. In theory, you could create a server rack that was completely secure so the only thing a specific user with access could do is reset a physical server. Granted, creating such a server rack would be complicated, but with vSphere, you can create a role that allows the user to do only this task. Figure 9.2 shows such a role.

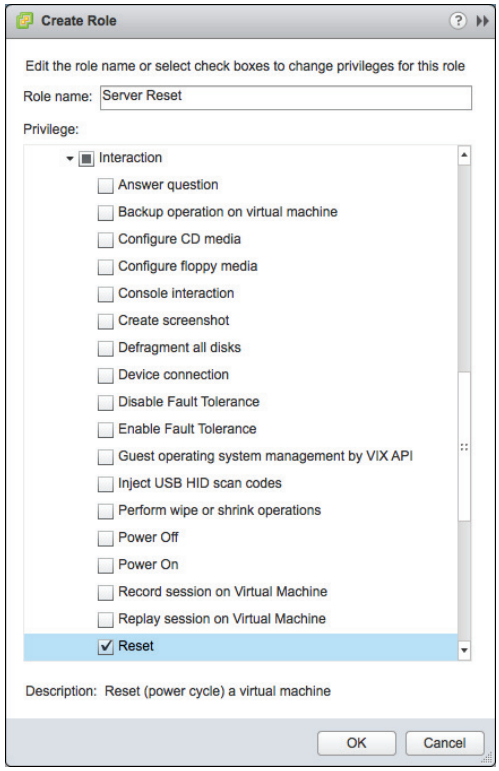


FIGURE 9.2
vCenter Server allows you to create a role that can only reset a VM.

With vSphere, you can assign practically any role you'd like. For example, you can allow a user to only create a screenshot of the VM, install VMware tools on a VM, or deploy a VM from a template—but not create their own new VM. You should identify the minimum tasks your user needs to perform to fulfill the job and allocate only the necessary privileges. This is commonly known as the *principle of least privilege*. For example, a help-desk user doesn't need administrative permissions to add hosts to the cluster but may need permission to deploy or restart VMs. This is also clearly related to the idea of separation of privileges—it allows you to clearly define the appropriate roles and assign the correct privileges to each role.

To return to the risk example, if Bill had been allocated the privileges shown in Figure 9.3, your production VMs wouldn't have suffered an outage.

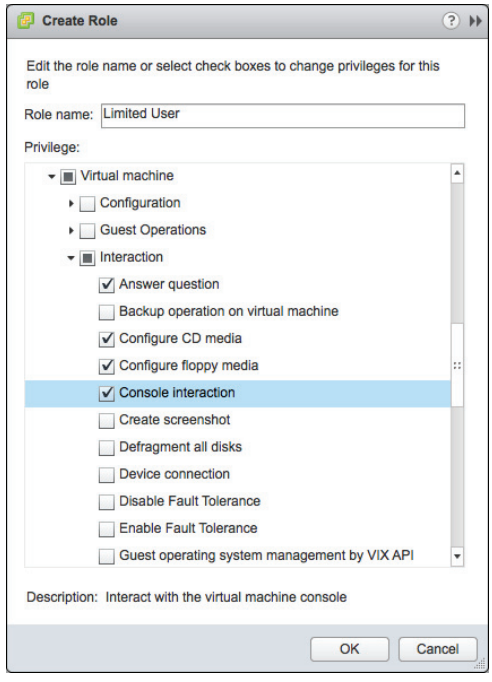


FIGURE 9.3
Very granular permissions can be combined to create a limited user role.

If Bill only had the options shown, he wouldn't have been able to

- Create a snapshot.
- Add more hard disks.
- Change his CPU/RAM allocation.
- Increase his VM shares.

With the proper design strategy, you can accommodate your users' needs and delegate permissions and roles to your staff that let them do their tasks, but that also make sure their environment is stable and limited to only what they should use. Be sure your vSphere design takes this key operational aspect into consideration and supplies the necessary technical controls on the backend.

The use of Linked Mode with multiple vCenter Server instances is another area of potential concern, which we explore in the next section.

Security in vCenter Linked Mode

Linked Mode allows you to connect different vCenter Server instances in your organization. We discussed the design considerations in Chapter 3, "The Management Layer," but let's take a closer look at security.

Risk Scenario

XYZ.com is a multinational company. It has a forest divided into child domains per country: US, UK, FR, HK, and AU. At each site, the IT staff's level of expertise differs—some are more experienced than others.

Gertrude, the VI Admin, joined the vCenter Servers together in Linked Mode. Suddenly, things are happening in her vCenter environment that shouldn't be. Machine settings are being changed, VMs are being powered off, and datastores are being added and removed.

After investigating, Gertrude finds that a user in one of the child domains is doing this. This user is part of the Administrators group at one of the smaller sites and shouldn't have these permissions.

Risk Mitigation

Proper planning is the main way to mitigate risk here. When you connect multiple vCenter Servers, they become one entity. If a user/group has Administrator privileges on the top level and these permissions are propagated down the tree, then that user/group will have full Administrator permissions on every single item in the infrastructure!

In a multidomain structure with vCenter Servers in different child and parent domains, there are two basic ways to divide the permissions: per-site permissions and global permissions.

Per-Site Permissions

With per-site permissions, each site maintains its administrative roles but isn't given full administrative rights to the other sites. Figure 9.4 shows such a structure.

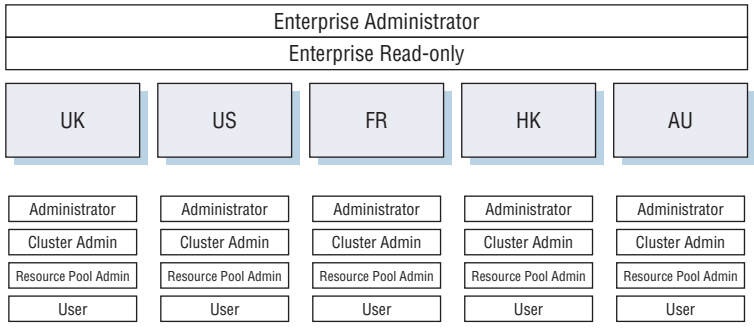


Figure 9.4
The per-site permission structure is one way to handle permissions in Linked Mode environments.

You can see that each site has its own permissions. The top level has two groups: Enterprise Administrator and Enterprise Read-only. The need for the Enterprise Administrators group is

obvious: you'll have a group that has permissions on the entire structure. But what is the need for the second group?

To allow the insight from one site into the other sites requires a certain level of permissions. For most organizations, this isn't a security risk because the permission is read-only; but in some cases it's unacceptable to allow the view into certain parts of the infrastructure (financial servers and domain controllers). In these cases, you can block propagation at that level. Figure 9.5 depicts such a configuration, where a specific user is blocked from accessing a cluster called Cluster 1.

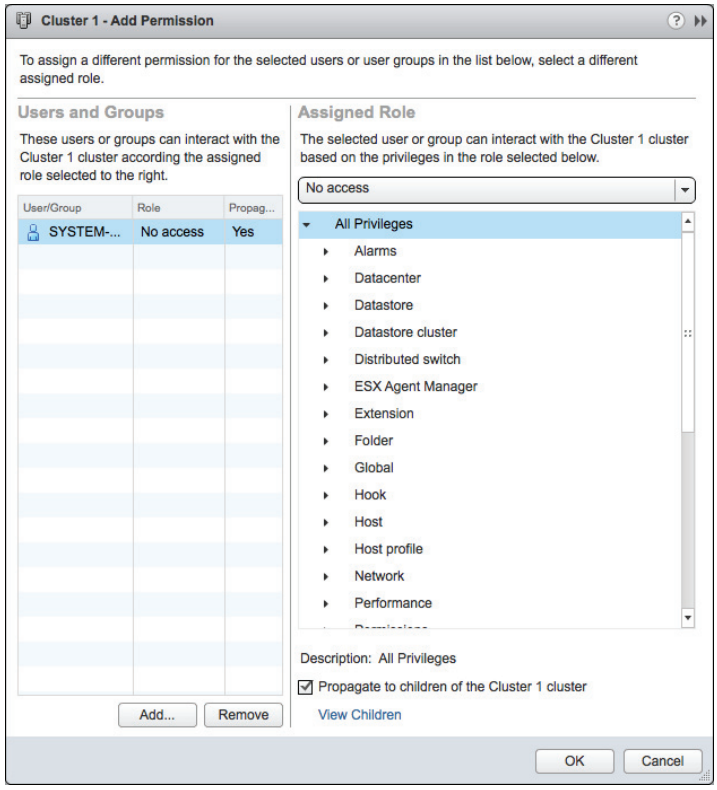


FIGURE 9.5
To protect specific areas within linked vCenter Server instances, use the No Access permission.

GLOBAL PERMISSIONS

In the global permissions model, roles are created for different tasks, but these roles are global. They must be applied at every vCenter tree, but this only needs to be done once. Figure 9.6 shows such a structure.

Which is the preferred structure? That depends on your specific environment and needs, and is determined by the design factors (functional requirements, constraints, risks, and assumptions). When we said the functional requirements were key to the entire design process, we were serious.

In the risk scenario, Gertrude should have completed her homework before allowing people she didn't trust to access the vCenter environment. If there were areas that people shouldn't access, she should have denied access using the No Access role.

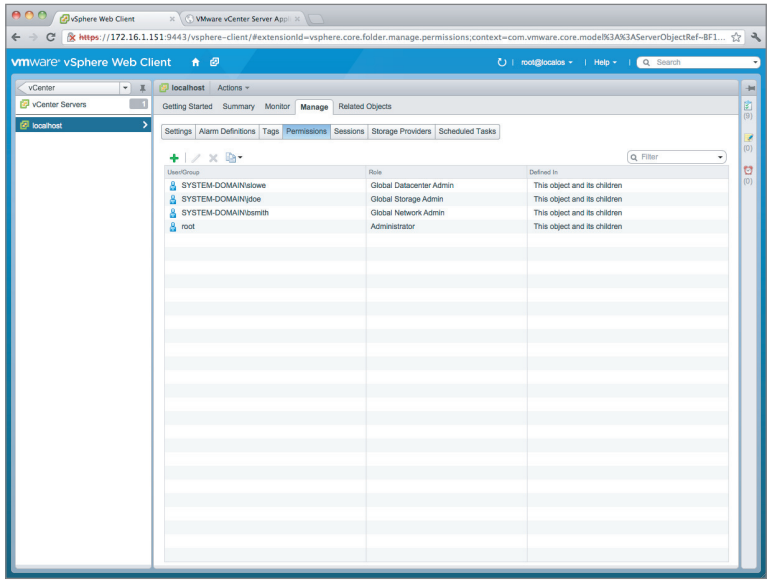


FIGURE 9.6
The global permission structure uses broad roles that are applicable across multiple Linked Mode instances.

Access control in all its forms—including the role-based access controls that vCenter Server uses—is a key component in ensuring a secure environment. Let's look at another form of access control: controlling access to command-line interfaces for managing ESXi hosts.

Command-Line Access to ESXi Hosts

If only everything could be done from the vSphere Client (some administrators cringe at the thought) … but it can't. The same is true for vSphere 5.1's Web Client. Although both the original Windows-based client and the new Web Client are powerful administrative tools, sometimes there are still certain functions that need to be done from a command line. This can have certain security risks, as you'll see.

Risk Scenario

Mary, your VI Admin, was with the company for many years. But her relationship with the company went downhill, and her contract ended—not on a good note, unfortunately.

One morning, you come into the office, and several critical servers are blue-screening. After investigation, you find that several ESXi hosts and VM settings have been altered, and these changes caused the outages.

Who changed the settings? You examine the logs from vCenter Server, but there's no record of changes being made. The trail seems to have ended.

Risk Mitigation

How could the changes have been made without vCenter Server having a record of them? Simple: Someone bypassed vCenter Server and went directly to the hosts, most likely via a command-line interface.

In older versions of vSphere that shipped with both ESX and ESXi, vSphere architects and administrators had to lock down and secure ESX's Red Hat Enterprise Linux (RHEL)-based Service Console. Prior to vSphere 4.1, ESXi had an unsupported command-line environment, but it didn't really present a security risk; in vSphere 4.1 and later, the ESXi Shell could be activated (but it issued an alert about this configuration change). In all cases, root SSH access to ESX hosts (and ESXi hosts with the shell enabled) was disabled by default (and rightfully so). Denying root access is also part of VMware's security best practices.

But as we all know, in certain cases it's necessary, or just plain easier, to perform actions on the host from the command line. How do we mitigate this potential security risk?

Disabling ESXi Shell and SSH Access

This might be a bit of overkill, but one way to mitigate the risk of users bypassing vCenter Server and performing tasks directly on the ESXi hosts is to simply turn off the ESXi Shell (referred to as Tech Support Mode [TSM] in earlier versions of vSphere) and SSH access to ESXi hosts. Both of these commands are available from the Direct Console User Interface (DCUI), as you can see in Figure 9.7.

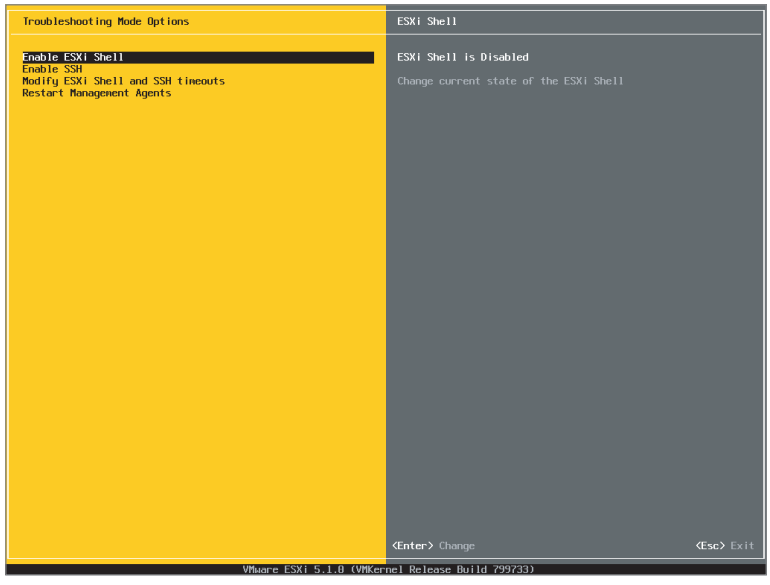


Figure 9.7
By default, the ESXi Shell and SSH access are disabled.

Although these settings aren't a panacea, they're a good first step toward mitigating the risk of direct command-line access. However, keeping the ESXi Shell and SSH disabled still doesn't

address access to the ESXi hosts via other CLI methods, such as the vSphere Management Assistant (vMA). Let's look at how to secure the vMA.

vMA Remote Administration

VMware usually releases a version of the vMA with the relevant releases of vSphere. As we discussed in Chapter 3, the vMA is a Linux-based virtual appliance. Whereas ESXi uses a limited BusyBox shell environment (and thus has limited controls), the vMA is a full Linux environment, and you can apply the full set of security controls to what can—or more appropriately, can't—be done by users using the vMA.

You can use a number of methods to tighten the security of the vMA:

- You can use `sudo` to control which users are allowed to run which commands. For example, using `sudo` you could let some users run the `vicfg-vswitch` command but not the `vicfg-vmknic` command. This technique almost demands an entire section just for itself; `sudo` is an incredibly flexible and powerful tool. For example, you can configure `sudo` to log all commands (that would have been handy in this scenario). It's beyond the scope of this book to provide a comprehensive guide to `sudo`, so we recommend you utilize any of the numerous "how to" guides available on the Internet.
- You can integrate the vMA into your Active Directory domain, which means one less set of credentials to manage. (Note that in vSphere 5.1 the vMA doesn't take advantage of vCenter's new Single Sign-On [SSO] functionality; however, SSO integrates into Active Directory so the end result is much the same.) You can use Active Directory credentials with `sudo` as well.
- You can tightly control who is allowed to access the vMA via SSH, using either the built-in controls in the SSH daemon or the network-based control (ACLs and/or firewalls).

Keeping the ESXi Shell disabled, leaving SSH disabled, and locking down permissions in the vMA are all good methods. However, it's still possible to bypass vCenter Server and go directly to the hosts. To close all possible avenues, you'll need Lockdown Mode.

Enable Lockdown Mode

Lockdown Mode provides a mechanism whereby ESXi hosts can only be managed via vCenter Server. In vSphere 4.1 and later, when Lockdown Mode is enabled access to ESXi hosts via the vSphere API, vSphere command-line interface (vCLI, including vMA), and PowerCLI are restricted. This provides an outstanding way to ensure that only vCenter Server can manage the ESXi hosts.

Lockdown Mode can be enabled via the vSphere Client, via the vSphere Web Client, or via the DCUI. Figure 9.8 shows a screenshot of the option to enable Lockdown Mode in the vSphere Web Client.

Note that it's generally recommended to enable or disable Lockdown Mode via the vSphere Client or the vSphere Web Client; disabling Lockdown Mode via the DCUI can undo certain permission settings. We recommend using the DCUI to disable Lockdown Mode only as a last resort if vCenter Server is unavailable.

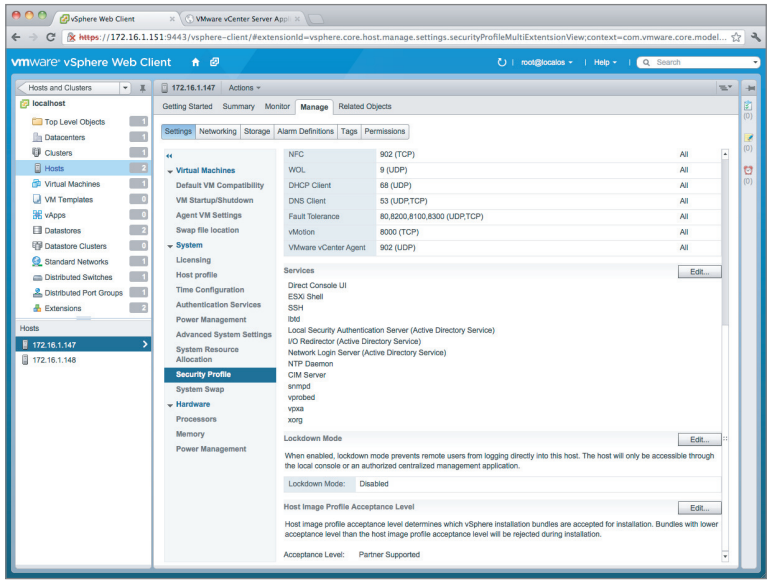


FIGURE 9.8
Lockdown Mode can be enabled using the vSphere Web Client.

In this scenario, keeping the ESXi Shell and SSH disabled on the ESXi hosts would have forced users through other channels, like the vMA. The use of `sudo` on the vMA would have provided the logging you needed to track down what commands were run and by whom. Further, if you really wanted to prevent any form of access to the ESXi hosts except through vCenter Server, Lockdown Mode would satisfy that need.

In the next section, we discuss how to use the network to manage access to your vSphere environment.

Managing Network Access

We touched on this topic in Chapter 5, "Designing Your Network," when we discussed planning your network architecture and design. Let's go into a bit more detail here.

Risk Scenario

Rachel, a hacker from the outside world, has somehow hacked into your network. Now that Rachel has control of a machine, she starts to fish around for information of value. Using a network sniffer, she discovers that there is a vCenter Server on the same subnet as the machine she controls. Rachel manages to access the vCenter Server, creates a local administrative user on the server, and then logs in to the vCenter with full Admin privileges.

She sees where all the ESXi hosts are and where all the storage is, and she performs a vMotion of a confidential Finance server from one host to another. She captures the traffic of the VM while in transit and is able to extract sensitive financial information. The potential damage is catastrophic.

Risk Mitigation

We won't get into how Rachel managed to gain control of a computer in your corporate network. Instead we'll focus on the fact that all the machines were on the same network.

Separate Management Network

Your servers shouldn't be on the same subnets as your users' computers. These are two completely different security zones. They probably have different security measures installed on them to protect them from the outside world.

You can provide extra security by separating your server farm onto separate subnets from your user computers. These could even be on separate physical switches if you want to provide an additional layer of segregation, although most organizations find that the use of VLANs is a sufficient barrier in this case. The question you may ask is, "How will this help me?"

This will help in a number of ways:

- If you suffer an intrusion such as the one described, malicious attackers like Rachel can't use a network sniffer to see most of the traffic from the separate management network. Because your servers reside on a different subnet, they must pass through a router—and that router will shield a great deal of traffic from other subnets. (Speaking in more technical terms, the dedicated management network will be a separate broadcast domain.)

- You can deploy a security device (an intrusion-protection system, for example) that will protect your server farm from a malicious attack from somewhere else on the network. Such appliances (they're usually physical appliances) examine the traffic going into the device and out to the server. With the assistance of advanced heuristics and technologies, the appliance detects suspicious activity and, if necessary, blocks the traffic from reaching the destination address.

- You can deploy a management proxy appliance, such as that provided by HyTrust, to proxy all management traffic to and from the virtual infrastructure. These devices can provide additional levels of authentication, logging, and more granular controls over the actions that can be performed.

- You can utilize a firewall (or simple router-based ACLs) to control the traffic moving into or out of the separate management network. This gives you the ability to tightly control which systems communicate and the types of communication they're permitted.

Fair enough, so using a separate management network is a good idea. But what traffic needs to be exposed to users, and what traffic should be separate? Typically, the only traffic that should be exposed to your users is your VM traffic. Your end users shouldn't have any interaction with management ports, the VMkernel interfaces you use for IP storage, or out-of-band management ports such as Integrated Lights Out (iLO), Integrated Management Module (IMM), Dell Remote Assistance Cards (DRACs), or Remote Supervisor Adapter (RSA). From a security point of view, your users shouldn't even be able to reach these IP addresses.

But what *does* need to interact with these interfaces? All the virtual infrastructure components need to talk to each other:

- ESXi management interfaces need to be available for vSphere High Availability (HA) heartbeats between the hosts in the cluster.
- The vCenter Server needs to communicate with your ESXi hosts.
- ESXi hosts need to communicate with IP storage.
- Users with vSphere permissions need to communicate with the vCenter Server.
- Monitoring systems need to poll the infrastructure for statistics and also receive alerts when necessary.
- Designated management stations or systems running PowerCLI scripts and/or using the remote CLI (such as the vMA) to perform remote management tasks on your hosts will need access to vCenter Server, if not the ESXi hosts as well.

Depending on your corporate policy and security requirements, you may decide to put your entire virtual infrastructure behind a firewall. In this case, only traffic defined in the appropriate rules will pass through the firewall; otherwise, it will be dropped (and should be logged as well). This way, you can define a very specific number of machines that are allowed to interact with your infrastructure.

Designated Management Stations

A related approach is to use designated management stations that are permitted access to managing the virtual infrastructure. First, you'll need to determine what kind of remote management you need:

- vSphere Client
- vCLI
- PowerShell

From this list, you can see that you'll need at least one management station, and perhaps two. The vSphere Client and PowerShell need a Windows machine. The vCLI can run on either Windows or Linux, and the vMA can provide the vCLI via a prepackaged Linux virtual appliance. This might make it reasonably easy, especially in smaller environments, to lock down which systems are allowed to manage the virtual infrastructure by simply controlling where the appropriate management software is installed. Further, with these management stations, you can provide a central point of access to your infrastructure, and you don't have to define a large number of firewall rules for multiple users who need to perform their daily duties. (You will, of course, have to provide the correct security for these management stations.)

However, with the introduction of the vSphere Web Client in vSphere 5.1, this model begins to break down. Now, any supported browser can potentially become a management station, and the idea of designated management stations might not be a good fit. At this point, you might have to look at other options, such as controls that reside at the network layer.

Network Port-Based Access

The Cisco term for this is *port security*. You define at the physical switch level which MAC addresses are allowed to interact with a particular port in the switch. Utilizing this approach,

you can define a very specific list of hosts that interact with your infrastructure and which ports they're allowed to access. You can define settings that allow certain users to access the management IP but not the VMkernel address. Other users can access your storage-management IPs but not the ESXi hosts. You can achieve significant granularity with this method, but there is a trade-off—solutions like this can sometimes create additional management overhead. The potential additional management overhead should be evaluated against the operational impact it might have.

Separate vMotion Network

Traffic between hosts during a vMotion operation isn't encrypted *at all*. This means the only interfaces that should be able to access this information are the vMotion interfaces themselves, and nothing else.

One way you can achieve this is to put this traffic on a separate, nonroutable network/VLAN. By doing so, you ensure that nothing outside of this subnet can access the traffic, and you prevent the potential risk that unauthorized users could somehow gain access to confidential information during a vMotion operation or interfere with vMotion operations.

Note that this does create certain serious challenges related to long-distance vMotion. However, long-distance vMotion isn't a reality for most organizations, because it requires considerable additional resources (storage and/or network connectivity between sites). Latency between different sites and, of course, stretching the network or the VLANs between the sites can be quite a challenge. Comprehensive network design is beyond the scope of this book.

Going back to the earlier example, if you had segregated the infrastructure network from the corporate traffic, your dear hacker friend Rachel wouldn't have discovered that there was a vCenter server. She wouldn't have been able to access the vCenter Server because her IP wouldn't have been authorized for access, and she wouldn't have been able to sniff the vMotion traffic. Even though she could still have compromised some users' computers, this wouldn't have led to the compromise of the entire virtual infrastructure.

In addition to planning how you can segregate network traffic types to enhance security, you also need to think carefully about how you consolidate network traffic and what the impact on security is as a result. The use of a demilitarized zone (DMZ) is one such example and is the topic of our next section.

The DMZ

Your infrastructure will sooner or later contain machines that have external-facing interfaces. It's best to plan that design properly to avoid mistakes like the one we'll describe next.

Risk Scenario

Harry, the sysadmin in your organization, wanted to use your virtualization infrastructure for a few extra VMs that were needed (yesterday, as always) in the DMZ. So, he connected an additional management port to the DMZ and exposed it to the outside world. Little did he know that doing so was a big mistake.

The ESXi host was compromised. Not only that, but because the host was using IP storage (NFS in this case), the intruder managed to extract information from the central storage and the network as well.

Risk Mitigation

In the previous section, we mentioned that the end user shouldn't have any interaction with the management interfaces of your ESXi hosts. The only thing that should be exposed are the VM networks. The primary mitigation of risk when using vSphere with (or in) a DMZ is proper design—ensuring that the proper interfaces are connected to the proper segments and that proper operational procedures are in place to prevent accidental exposure of data and/or systems through human error or misconfiguration.

With that in mind, let's take a look at some DMZ architectures. A traditional DMZ environment that uses physical hosts and physical firewalls would look similar to the diagram in Figure 9.9.

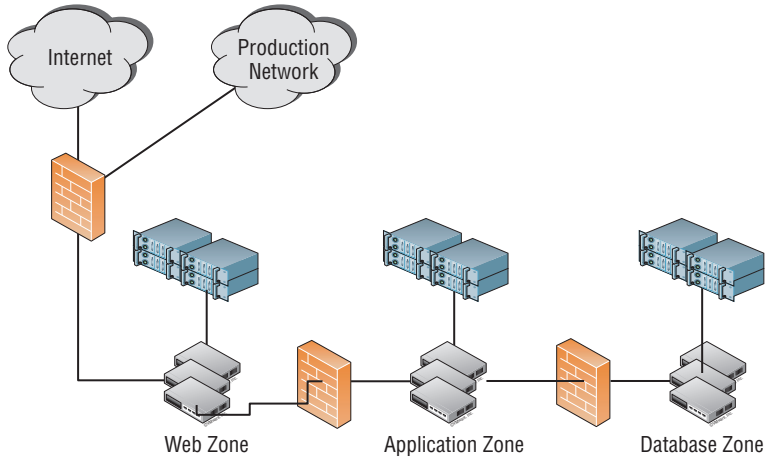


FIGURE 9.9
This traditional DMZ environment uses separate physical hosts and physical firewalls.

A virtual DMZ will look similar to Figure 9.10.

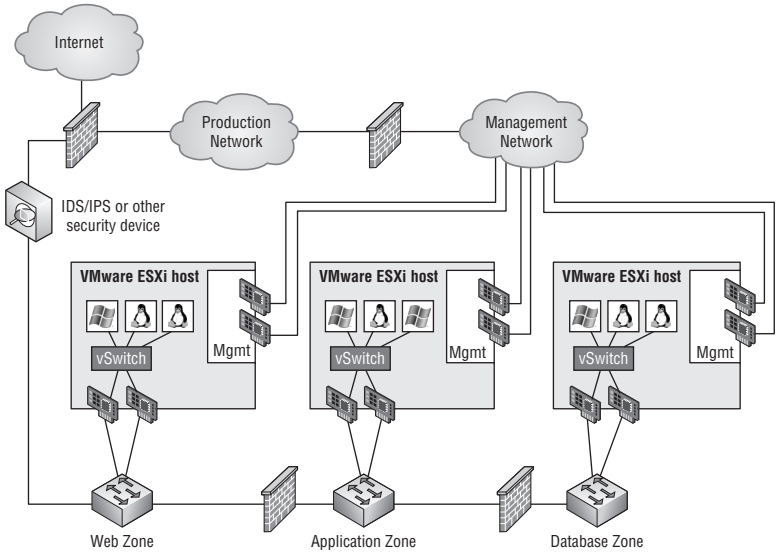


FIGURE 9.10
This virtualized DMZ uses VMs and physical firewalls.

Notice that the management ports aren't exposed to the DMZ, but the VM network cards are exposed—just like their physical counterparts from Figure 9.9.

DMZ configurations can be divided into three categories:

- Partially collapsed with separate physical zones
- Partially collapsed with separate virtual zones
- Fully collapsed

Partially Collapsed DMZ with Separate Physical Zones

Figure 9.10, shown earlier, graphically describes this configuration. In this configuration, you need a separate ESXi host (or cluster of hosts) for each and every one of your zones. You have complete separation of the different application types and security risks. Of course, this isn't an optimal configuration, because you can end up putting in a huge number of hosts for each separate zone and then having resources stranded in one zone or another—which is the opposite of the whole idea of virtualization.

Partially Collapsed DMZ with Separate Virtual Zones

Figure 9.11 depicts a partially collapsed DMZ with separate virtual zones.

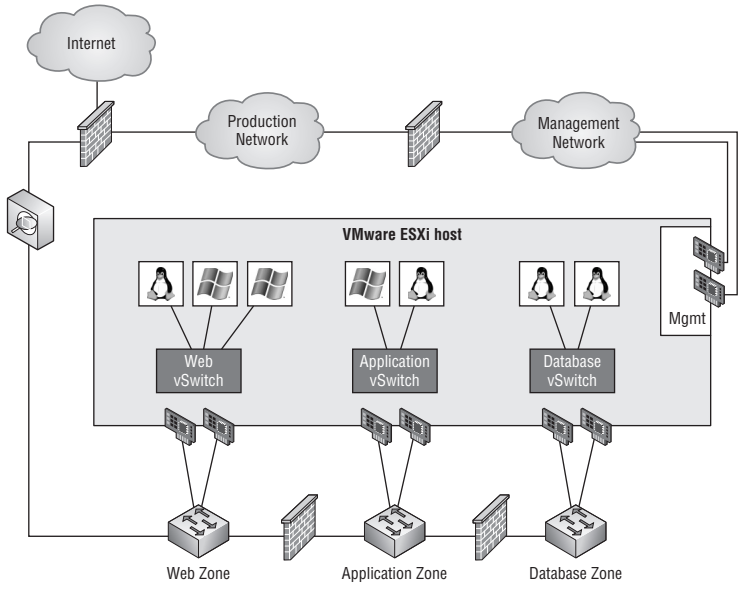


Figure 9.11
In this configuration, a single host (or cluster) spans multiple security zones.

In this configuration, you use different zones in a single ESXi host (or a cluster of hosts). The separation is done only on the network connections of each zone, with a firewall separating the different network zones on the network level. You have to plan accordingly to allow for a sufficient number of network cards for your hosts to provide connectivity to every zone.

This configuration makes much better use of your virtualization resources, but it's more complex and error prone. A key risk is the accidental connection of a VM to the wrong security zone. Using Figure 9.11 as an example, the risk is accidentally connecting an application server to the Web vSwitch instead of the Application vSwitch, thereby exposing the workload to a different security zone than what was intended. To mitigate this potential risk, you should apply the same change-control and change-management procedures to the virtual DMZ as those you have in place for your physical DMZ.

Fully Collapsed DMZ

This is by far the most complex configuration of the three, as you can see in Figure 9.12.

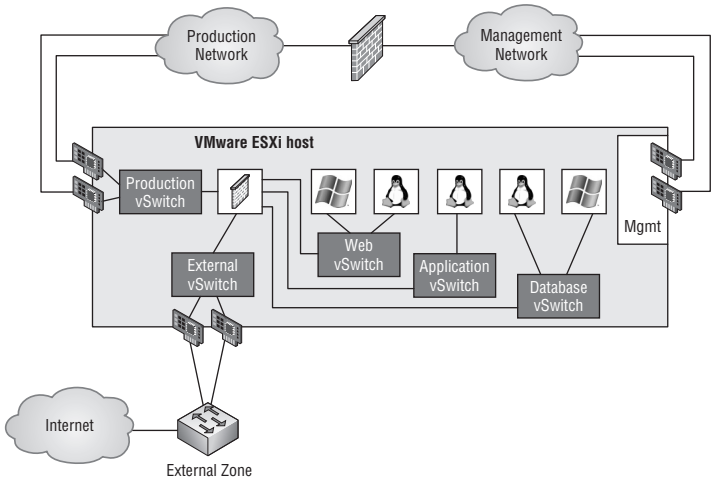


Figure 9.12
A fully collapsed DMZ employs both VMs and virtual security appliances to provide separate security zones.

In this configuration, there are no physical firewalls between the different security zones: the firewall is a virtual appliance (running as a VM on the ESXi cluster) that handles network segregation. This virtual appliance could be VMware's vShield Edge, Cisco's ASA 1000v, or some other third-party virtual firewall or virtual security appliance. As noted in the previous configuration, you should make sure your processes are in effect and audited regularly for compliance.

In all three DMZ configurations we've reviewed—partially collapsed with separate physical zones, partially collapsed with separate virtual zones, and fully collapsed—you'll note that management traffic is kept isolated from the VM traffic, which exposes VMs to a DMZ while preventing the ESXi hosts from being exposed to potentially malicious traffic.

However, protecting management traffic is only part of the picture. You also need to protect the storage that is backing the vSphere environment.

Separation of Storage

How possible is it for someone to break out of a compromised ESXi host onto a shared network resource? That depends who you ask. Has it been done? Will it be done? These are questions that have yet to be answered. Is it worth protecting against this possibility? Many organizations would say yes.

With that in mind, you can use the following methods to help protect your storage resources in DMZ environments:

- Use Fibre Channel (FC) host bus adapters (HBAs) to eliminate the possibility of an IP-based attack (because there is no IP in a typical FC environment). It *might* be possible for an attack to be crafted from within the SCSI stack to escape onto the network, but this would be a difficult task to accomplish.
- Theoretically, the same is true for Fibre Channel over Ethernet (FCoE), although the commonality of Ethernet between the FCoE and IP environments might lessen the strong separation that FC offers.
- Use separate physical switches for your storage network, providing an air gap between the storage network and the rest of the network. That air gap is bridged only by the ESXi hosts' VMkernel interfaces and the storage array's interfaces.
- Never expose the storage array to VM-facing traffic without a firewall between them. If the VMs *must* have access to storage array resources via IP-based storage, tread very carefully.

The theory behind all these approaches is that there is as little network connectivity as possible between the corporate network (the storage array) and the DMZ, thus reducing the network exposure.

Going back to the example, the situation with Harry and the exposed management port on the external-facing network should never have happened. Keeping these ports on a dedicated segment that isn't exposed to the outside world minimizes the chance of compromising the server. In addition, using the FC infrastructure that is currently in place minimizes the possible attack surface into your network from the DMZ.

You should take into consideration two more important things regarding the DMZ. First, it's absolutely possible to set up a secure DMZ solution based completely on your vSphere infrastructure. Doing so depends on proper planning and adhering to the same principles you would apply to your physical DMZ; the components are just in a slightly different location in your design. Second, a breach is more likely to occur due to misconfiguration of your DMZ than because the technology can't provide the proper level of security.

Firewalls in the Virtual Infrastructure

VMs are replacing physical machines everywhere, and certain use cases call for providing security even in your virtual environment. You saw this in the previous section when we discussed fully collapsed DMZ architectures (see the section "Fully Collapsed DMZ").

What are the security implications of using virtual firewalls? That's our focus in this section. Note that we won't be talking about risk and risk mitigation because you can provide a secure firewall in your environment by continuing to use your current corporate firewall. Doing so, however, presents certain challenges, as you'll see.

The Problem

Herbert has a development environment in which he has to secure a certain section behind a firewall. All the machines are sitting on the same vSwitch on the same host. Herbert can't pass the traffic through his corporate firewall because all traffic that travels on the same vSwitch never leaves the host.

He deploys a Linux machine with `iptables` as a firewall, which provides a good solution, but the management overhead makes this too much of a headache. In addition, in order for this to work, he has to keep all the machines on the same host, which brings the risk of losing the entire environment if the host goes down.

The Solution

There are two ways to get around this problem. One is to use the physical firewall, and the second is to use virtual firewalls that have built-in management tools to allow for central management and control.

Physical Firewall

The main point to consider is that in order to pass traffic through the physical firewall, the traffic must go out of the VM, out through the VM port group and vSwitch, out through the physical NIC, and from there onto the firewall. If the traffic is destined for a VM on the same host, then the traffic does a hairpin turn at the firewall (assuming the firewall even supports such a configuration; many firewalls don't) and traverses the exact same path back to its destination. Traffic on that network segment is doubled (out and then back again), when ideally it should have been self-contained in the host. Although the vast majority of workloads won't be network constrained, it's important to note that network throughput VM-to-VM across a vSwitch is much higher than it would be if the traffic had to go out onto the physical network and back again.

Virtual Firewall

A virtual firewall is a VM with one or more vNICs. It sits as a buffer between the physical layer and the VMs. After the VMs are connected to a virtual firewall, they're no longer connected to the external network; all traffic flows through the virtual firewall.

You might think that this sort of configuration is exactly what Herbert built earlier using Linux and `iptables`. You're correct—sort of. To avoid the issues that normally come with this type of configuration, some virtual firewalls operate as a bump in the wire, meaning they function as transparent Layer 2 bridges (as opposed to routed Layer 3 firewalls). This allows them to bypass issues with IP addresses, but it does introduce some complexities (like the VMs needing to be connected to an isolated vSwitch with no uplinks—a configuration that doesn't normally support vMotion).

VMware provides a suite of applications called VMware vShield that can protect your virtual environment at different levels. Table 9.1 describes some of the differences between the products in the vShield product family. Note that as of vSphere 5.1, vShield Endpoint is now included with all editions of vSphere and no longer needs to be purchased separately (although the purchase of a supported antivirus engine to use Endpoint is still necessary).

Other third-party vApps provide similar functionality. This is an emerging market, so you should perform a full evaluation of which product will fit best into your organization's current infrastructure and best meet your specific security requirements.

TABLE 9.1: VMware vShield product comparisons

FEATURE	VSHIELD EDGE 1.0	VSHIELD ZONES 4.1	VSHIELD APP 1.0	VSHIELD ENDPOINT 1.0
Deployment method	Per port group	Per host	Per host	Per host
Enforcement	Between virtual datacenter and untrusted networks	Between VMs	Between VMs	Within the guest VM
Antivirus, anti-malware	No	No	No	Yes
Site-to-site VPN	Yes	No	No	No
NAT, DHCP services	Yes	No	No	No
Load balancing	Yes	No	No	No
Port group isolation	Yes	No	No	No
Stateful firewall	Yes	Yes	Yes	No
Change-aware	Yes	Yes	Yes	No
Hypervisor-based firewall	No	Yes	Yes	No
Application firewall	Yes	Yes	Yes	No
Flow monitoring	No	No	Yes	No
Groupings for policy enforcement	Only 5-tuple* based policies	Only 5-tuple based policies	1) 5-tuple 2) Security groups: resource pools, folders, containers, and other vSphere groupings	Any available vCenter groupings for VMs

Source: VMware

* A 5-tuple is defined as the combination of source IP address, destination IP address, source port, destination port, and protocol.

Change Management

As we mentioned in the section "The DMZ," a security breach is more likely to occur due to misconfiguration (or human error) than because the software can't provide the appropriate level of security. This is why we want to discuss change management in the context of security. Although this isn't really a security feature, it's still something that should be implemented in every organization.

Risk Scenario

Barry, your junior virtualization administrator, downloaded a new vApp from the Internet: an evaluation version of a firewall appliance. He deployed the appliance, started the software setup, answered a few simple questions, and filled in a few fields (including his administrative credentials to the virtual infrastructure). All of a sudden, a new vSwitch was created on all ESXi hosts, and all traffic was routed through the appliance.

You start to receive calls saying that some applications aren't working correctly. Web traffic isn't getting to where it should go. You begin to troubleshoot the problem from the OS side. After a long analysis, you discover the change that Barry made.

The result is far too much downtime for the applications and far too much time spent looking in the wrong direction.

Risk Mitigation

You don't want changes made to your infrastructure without documenting and testing them before the fact. Bad things are sure to happen otherwise. To mitigate the risk of untested and undocumented changes, organizations implement *change control* or *change management*.

The problem with discussing change management with companies is that the idea of change management is different for every organization. Different companies have different procedures, different regulations, different compliance requirements, and different levels of aversion to risk (and different risks). Is there, then, a way to discuss change management in some sort of standardized way? That's the attempted goal behind the Information Technology Infrastructure Library (ITIL).

ITIL attempts to provide a baseline set of processes, tasks, procedures, and checklists that a company can use to help align IT services with the needs of the organization. ITIL is very broad, but for the purposes of this discussion we'll focus on the Service Transition portion where change management is discussed. The purpose of change management in ITIL is to ensure that changes are properly approved (through some sort of approval/review chain), tested before implementation so as to minimize risk to the organization, and aligned to the needs of the business (such as providing improved performance, more reliable service, additional revenue, or lower costs).

There are those who hate ITIL with a passion and those who swear by it. Our purpose here isn't to say that ITIL is perfect and every organization should adopt it, but rather to emphasize that proper change management is an important part of every IT organization's operational procedures. Let's take a look at a couple of items you might want to include in your design to help emphasize the importance of change management as a way to mitigate risk.

Test Environment

To mitigate problems, prepare a test environment that is as close as possible to your production environment. We say "as close as possible" because you can't always have another SAN, an

entire network stack, or a blade chassis. You need a vCenter Server, central storage, and some ESXi hosts. The beauty of this is that today, all of these components can be VMs. Numerous sessions at the VMworld conference and a large number of blogs explain how to set up such a home lab for testing purposes. To lower the licensing cost for this lab, you can purchase low-end bundles (Essentials and Essentials Plus) for a minimal cost.

Play around in your test environment before implementing anything on your production systems. Document the changes so they can be reproduced when needed when you implement for real.

Before you make the changes, answer these questions:

- What are the implications of this change?
- Who or what will be affected?
- How much of a risk is this change?
- Can the change be reversed?
- How long will it take to roll back?

Change Process

All the questions in the previous list should be answered by all the relevant parties involved. This way, you can identify all the angles—and not only from the point of view in your position.

We aren't always aware of the full picture, because in most cases it isn't possible to be. In most organizations, the storage person, the network person, the help-desk team, and the hardware person aren't the same person. They aren't even on the same team. You can benefit from the different perspectives of other teams to get a fuller picture of how changes will impact your users. Having a process in place for involving all stakeholders in change decisions will also ensure that *you're* involved in changes initiated by other groups.

In the previous risk example, Barry should never have installed this appliance on the production system. He should have deployed it in the test environment, and he would have seen that changes were made in the virtual network switches that could cause issues. Barry also should have brought this change through a proper change-management process.

Not every organization implements ITIL, not every organization wants to, and not every organization needs to. You should find the process that works for you: one that will make your job easier and keep your environment stable and functioning in the best fashion.

Protecting the VMs

You can secure the infrastructure to your heart's content, but without taking certain measures to secure the VMs in this infrastructure, you'll be exposed to risk.

Risk Scenario

Larry just installed a new VM—Windows 8, from an ISO he got from the help desk. What Larry did *not* do was install an antivirus client in the VM; he also didn't update the OS with the latest service pack and security patches. This left a vulnerable OS sitting in your infrastructure.

Then, someone exploited the OS to perform malicious activity. They installed a rootkit on the machine, which stole information and finally caused a Denial of Service (DoS) attack against your corporate web server.

Risk Mitigation

The OS installed into a VM is no different than the OS installed onto a physical server. Thus, you need to ensure that a VM is always treated like any other OS on your network. It should go without saying that you shouldn't allow vulnerable machines in your datacenter.

In order to control what OSes are deployed in your environment, you should have a sound base of standardized templates from which you deploy your OSes. Doing so provides standardization in your company and also ensures that you have a secure baseline for all VMs deployed in the infrastructure, whatever flavor of OS they may be.

Regarding the patching of VMs, you should treat them like any other machine on the network. Prior to vSphere 5.1, VMware Update Manager (VUM) can provide patching for certain guest OSes, but this functionality was discontinued in vSphere 5.1. VMware has decided that it should patch only the vSphere environment and leave the guest OSes to the third-party vendors. This is actually a good thing. Most of the large OS vendors already provided patch-management functionality (think of Microsoft and Software Update Services/Windows Server Update Services [SUS/WSUS]) that offered benefits over VUM with regard to features, reliability, and OS integration. It didn't make sense for VMware to try to continue to push this functionality, in our opinion.

You don't only need to patch the VMs—you also need to patch the templates from which those VMs are deployed! Ensure that you have operational procedures in place to regularly update the templates with the latest security patches and fixes. You'll also want to test the changes to those templates before rolling them into production!

Don't forget about providing endpoint protection for your VMs. You can deploy antivirus/anti-malware directly into the guest OS in every VM, but a more efficient solution might be to look at hypervisor-based protection. We mentioned vShield Endpoint in the earlier section "Firewalls in the Virtual Infrastructure," but let's discuss it in a bit more detail here.

What is vShield Endpoint? The technology adds a layer to the hypervisor that allows you to offload antivirus and anti-malware functions to a hardened, tamper-proof security VM, thereby eliminating the need for an agent in every VM. vShield Endpoint plugs directly into vSphere and consists of a hardened security VM (delivered by VMware partners), a driver for VMs to offload file events, and the VMware Endpoint security (EPSEC) loadable kernel module (LKM) to link the first two components at the hypervisor layer.

There are multiple benefits to handling this method of protection:

- You deploy an antivirus engine and signature files to a single security VM instead of every individual VM on your host.
- You free up resources on each of your VMs. Instead of having the additional CPU and RAM resources used on the VM for protecting the OS, the resource usage is offloaded to the host. This means you can raise the consolidation ratios on your hosts.
- You eliminate the occurrence of antivirus storms and bottlenecks that occur when all the VMs on the host (and the network) start their malware and antivirus scan or updates at the same time.
- You obscure the antivirus and anti-malware client software from the guest OS, thus making it harder for malware to enter the system and attack the antivirus software.

As of the writing of this book, only a few vendors offered vShield Endpoint-compatible solutions. Kaspersky, McAfee, and Trend Micro all offer solutions that integrate with vShield

Endpoint. If you use a vendor other than one of these three, you'll have to protect your VMs the old way. And you'll have to plan accordingly for certain possibilities.

Antivirus Storms

Antivirus storms happen all the time in the physical infrastructure, but they're much more of an issue when the infrastructure is virtual. When each OS is running on its own computer, each computer endures higher CPU/RAM usage and increased disk I/O during the scan/update. But when all the OSes on a host scan/update at the same time, it can cripple a host or a storage array.

You'll have to plan how to stagger these scans over the day or week in order to spread out the load on the hosts. You can do so by creating several groups of VMs and spreading them out over a schedule or assigning them to different management servers that schedule scans/updates at different times. We won't go into the details of how to configure these settings, because they vary with each vendor's products.

Ensuring That Machines Are Up to Date

Each organization has its own corporate policies in place to make sure machines have the correct software and patches installed. There are several methods of achieving this.

One option is checking the OSes with a script at user login for antivirus software and up-to-date patches. If the computer isn't compliant, then the script logs off the user and alerts the help desk.

A more robust solution is Microsoft's Network Access Protection (NAP). Here, only computers that comply with a certain set of rules are allowed to access the appropriate resources. Think of it as a DMZ in your corporate network. You connect to the network, and your computer is scanned for compliance. If it isn't compliant, you're kept in the DMZ until your computer is updated; then you're allowed full access to the network. If for some reason the computer can't be made compliant, you aren't allowed out of the DMZ. This of course is only one of the solutions available today; there are other, similar ones.

Looking back at the risk example from the beginning of the section, if you had only allowed the deployment of VMs from predefined templates, Larry wouldn't have been able to create a system that was vulnerable in the first place:

- The VM would have been installed with the latest patch level.
- Antivirus software would have been deployed (automatically) to the VM, or the VM would have been protected by software at the hypervisor level.
- No rootkit could have been installed, and there would have been no loss of information.
- Your corporate web server would never have been attacked, and there would have been no outage.

And you would sleep better at night.

Protecting the Data

What would you do if someone stole a server or a desktop in your organization? How much of a security breach would it be? How would you restore the data that was lost?

Risk Scenario

Gary is a member of the IT department. He was presented with an offer he couldn't refuse, albeit an illegal one: he was approached by a competitor of yours to "retrieve" certain information from your company.

Gary knew the information was stored on a particular VM in your infrastructure. So, Gary cloned this VM—not through vCenter, but through the storage (from a snapshot), and he tried to cover his tracks. He then copied the VM off the storage to an external device and sneaked it out of the company.

Three months down the road, you find out that your competitor has somehow managed to release a product amazingly similar to the one you're planning to release this month. Investigations, audit trails, and many, many hours and dollars later, you find the culprit and begin damage-control and cleanup.

Risk Mitigation

This example may be apocryphal, and you can say that you trust your employees and team members, but things like this happen in the real world. There are people who will pay a lot of money for corporate espionage, and they have a lot to gain from it.

In this case, put on your paranoid security person hat, and think about the entry points where you could lose data.

A Complete VM

Walking out past the security guard with a 2U server and a shelf of storage under your coat or in your pocket is pretty much impossible. You'd be noticed immediately. But with virtualization, you can take a full server with up to 2 TB of data on one 3.5" hard disk, put it in your bag or in your coat pocket, and walk out the door, and no one will be any the wiser.

One of the great benefits of virtualization is the encapsulation of the server into a group of files that can be moved from one storage device to another. VMs can be taken from one platform and moved to another. This can also be a major security risk.

How do you go about protecting the data? First and foremost, trust. You have to trust the staff who have access to your sensitive data. Trust can be established via corporate procedures and guidelines. Who is allowed to access what? Where? And how?

At the beginning of the chapter, we talked about separation of duties. Giving one person too much power or too much access can leave you vulnerable. So, separation of duties here has clear benefits.

In the risk example, the clone was performed on the storage, not from the vCenter Server. vCenter provides a certain level of security and an audit trail of who did what and when, which you can use to your advantage.

You should limit access to the VM on the backend storage. The most vulnerable piece here is NFS, because the only mechanism that protects the data is the export file that defines who has what kind of access to the data. With iSCSI, the information is still accessible over the wire, but it's slightly more secure. In order to expose a logical unit number (LUN) to a host, you define the explicit LUN mask allowing the iSCSI initiator access to the LUN. And last but not least, don't forget FC. Everything we mentioned for iSCSI is the same, but you'll need a dedicated HBA to connect to a port in the fabric switch as well. This requires physical access to the datacenter, which is more difficult than connecting to a network port on a LAN and accessing the IP storage.

How can the data be exported out of your datacenter? Possibly over the network. If that is the case, you should have measures in place alerting you to abnormal activity on the physical switches. You can use the corporate firewall to block external uploads of data.

If this data has to be copied to physical media, you can limit which devices are able to connect to USB ports on which computers and with which credentials. A multitude of security companies thrive on security paranoia (which is a good thing).

Backup Sets

What about the backup set of your data? Most organizations have more than one backup set. What good does it do, if your backups are located in the same location as your production environment—and the building explodes? To be safe, you need a business continuity planning/disaster recovery (BCP/DR) site.

And here comes the security concern: how is data transported to this site? Trucking the data may be an option, but you'll have to ensure the safety and integrity of the transport each time the backups are moved offsite, whether you use sealed envelopes, armed guards, a member of your staff, or the network. When sending data over the network, you should ensure that your traffic is secured and encrypted. You should choose which solution is suited for you and take the measures to protect the transport.

Virtual Machine Data

In this case, there are no significant differences between a VM that is accessible from the network and its counterpart, the physical server. You should limit network access to your servers to those who need and have the correct authority to access these files. The ways of exporting the files from the VM are the same as with a full VM, as we discussed earlier.

Back to your "faithful" employee, Gary. Gary shouldn't have been granted the same level of access to all the components, the virtual infrastructure, and the storage backend. Your security measures should have detected the large amount of traffic that was being migrated out of the storage array to a non-authorized host. In addition, Gary shouldn't have been working in your organization in the first place. Some organizations periodically conduct compulsory polygraph tests for IT personnel who have access to confidential information.

And last but not least, you should protect your information. If data is sensitive and potentially damaging if it falls to the wrong hands, then you should take every measure possible to secure it. Be alert to every abnormal (or even normal) attempt to access the data in any way or form.

Cloud Computing

Any virtualization book published in this day and age that doesn't mention cloud computing is pretty bizarre, so let's not deviate from this norm.

Risk Scenario

Carrie set up the internal cloud in your organization. She also contracted the services of a provider to supply virtual computing infrastructure somewhere else, to supplement the internal cloud's capacity when needed and in a cost-effective manner.

Due to a security issue at the cloud provider, there was a security breach, and your VMs were compromised. In addition, because the VMs were connected to your organization, an attack was launched from these hosts into your corporate network, which caused additional damage.

Risk Mitigation

VMware announced its private/public cloud solution at VMworld 2010. Since then, several large providers have begun to supply cloud computing services to the public using VMware vCloud. In addition, Amazon's Elastic Compute Cloud (EC2) has seen tremendous uptake, and many organizations—both large and small—are supplementing their compute capacity with instances running in Amazon's datacenters.

Again, put on your paranoid thinking cap. What information do you have in your organization? HR records, intellectual property, financial information—the list can get very long. For each type of information, you'll have restrictions.

But how safe is it to have your data up in the cloud? That depends on your answers to a few questions.

Control

Do you have control over what is happening on the infrastructure that isn't in your location? What level of control? Who else can access the VMs? These are questions you should ask yourself; then, think very carefully about the answers. Would you like someone to access the data you have stored at a provider? What measures can be taken to be sure this won't happen? How can you ensure that no one can access the VMs from the ESXi hosts on which the VMs are stored?

These are all possible scenarios and security vulnerabilities that exist in your organization as well—but in this case, the machines are located outside your network, outside your company, outside your city, and possibly in a different country.

Using the cloud may also present several different legal issues. Here's an example. You have a customer-facing application on a server somewhere in a cloud provider's datacenter. The provider also gave services to a client that turned out to be performing illegal activities. The authorities dispatched a court order to allow law-enforcement agencies to seize the data of the offender—but instead the law-enforcement agencies took a full rack of servers from the datacenter. What if you had a VM on one of the servers in that rack? This is a true story that happened in Dallas in 2009. It may be pessimistic, but it's better to be safe than sorry.

Here's something else to consider: will you ever have the same level of control over a server in the cloud that you have in your local datacenter, and what do you need to do to reach that same level? If you'll never have the same level of control, what amount of exposure are you willing to risk by using an external provider?

Data Transfer

Suppose you've succeeded in acquiring the correct amount of control over your server in the cloud. Now you have to think about how to transfer the data back and forth between your corporate network and the cloud.

You'll have to ensure the integrity and security of the transfer, which means a secure tunnel between your network and the cloud. This isn't a task to be taken lightly; it will require proper and thorough planning. Do you want the set of rules for your data flow from your organization

out to the cloud to be the same as that for the flow of information from the cloud into your corporate network?

Back to the risk example. Carrie should have found a sound and secure provider, one with a solid reputation and good security. In addition, she should have set up the correct firewall rules to allow only certain kinds of traffic back into your corporate network, and made sure only the relevant information that absolutely needed to be located in the cloud was there.

The future will bring many different solutions to provide these kinds of services and to secure them as well. Because this technology is only starting to become a reality, the dangers we're aware of are only the ones we know about today. Who knows what the future will hold?

Auditing and Compliance

Many companies have to comply with certain standards and regulatory requirements, such as HIPAA, the Sarbanes-Oxley Act, the Data Privacy and Protection Act, ISO 17799, the PCI Data Security Standard, and so on. As in the section "Virtual Firewall," this isn't so much a question of risk as it is a problem and solution.

The Problem

You've put your virtual infrastructure in place, and you continue to use it and deploy it further. You add hosts, import some VMs, add new storage, deploy a new cluster—things change, and things grow. And with a virtual infrastructure, the setup is more dynamic than before. How do you make sure everything is the way it should be? Are all your settings correct? How do you track all these changes?

The Solution

There are several ways you can ease your way into creating a compliant and standardized environment, including using host profiles, collecting centralized logs, and performing security audits.

Host Profiles

VMware has incorporated a feature called Host Profiles that does exactly what its name says. You can define a profile for a host and then apply the profile to a host, to a cluster, and even to all the hosts in your environment.

A *profile* is a collection of settings that you configure on a host. For example:

- vSwitch creation
- VMkernel creation
- Virtual machine port group creation
- NTP settings
- Local users and groups

This is just a short list of what you can do with host profiles. To enhance this, Host Profiles not only configures the hosts with the attached profile but also alerts you when a change is made to a host that causes it to no longer be compliant.

For example, someone may add a new datastore to the ESXi host or change the name of a port group. Why are these changes important? Because in both cases, the changes weren't made across all the hosts in the cluster—vMotion may fail or may succeed, but the destination host won't have the port group, and the VM will be disconnected from the network after the migration.

Using Host Profiles can ease your deployment of hosts and keep them in compliance with a standard configuration.

NOTE Host Profiles is an Enterprise Plus feature only, which means you'll need to have all hosts in your cluster at this license level in order to use this feature.

Centralized Log Collection

Each ESXi host is capable of sending its logs to a syslog server. The benefit of doing so is to have a central location with the logs of all the hosts in your environment. It's easier to archive these logs from one location, easier to analyze them, and easier to perform root-cause analysis, instead of having to go to every host in the environment. The use of a centralized logging facility is even more critical when you move to vSphere 5.0 or later, where only ESXi is present. ESXi can't preserve logs over system reboots/failures. This can be particularly problematic if an outage was unexpected and you have no logs previous to the outage. A syslog server will be a critical element in your design, now that ESXi will be the default hypervisor.

Security Audits

You should conduct regular audits of your environment. They should include all components your environment uses: storage, network, vCenter, ESXi, and VMs. You shouldn't rely on the external audit you undergo once a year, because there is a significant chance that if something has changed (and this is a security risk), you don't want to wait until it's discovered in the external audit (if you're lucky) or is exploited (in which case you aren't so lucky any more).

As we said earlier, environments aren't static: they evolve, and at a faster pace than you may think. The more hands that delve into the environment, the more changes are made, and the bigger the chance of something falling out of compliance with your company policy.

Create a checklist of things to check. For example:

- Are all the hosts using the correct credentials?
- Are the network settings on each host the same?
- Are the LUN masks/NFS export permissions correct?
- Which users have permissions to the vCenter Server, and what permissions do they have?
- Are any dormant machines no longer in use?

Your list should be composed of the important issues in your environment. It shouldn't be used in place of other monitoring tools or compliance tools that you already have in place but as an additional measure to secure your environment.

Summary

Throughout this chapter, you've seen different aspects of security and why it's important to keep your environment up to date and secure. You must protect your environment at every level: guest, host, storage, network, vCenter, and even outside your organization up in the cloud. Identifying your weakest link or softest spot and addressing the issue is your way to provide the proper security for your virtual infrastructure.

In the next chapter, we'll go into monitoring and capacity management.

Chapter 10

Monitoring and Capacity Planning

Monitoring the VMware vSphere environment and anticipating future growth via capacity planning are integral parts of a good vSphere design. Although related, monitoring and capacity planning aren't the same. Monitoring is more about the present or the recent past, whereas capacity planning is about the future.

This chapter will cover the following topics:

- The importance of monitoring and capacity planning
- Selecting monitoring and capacity-planning tools
- Incorporating monitoring into your design
- Different aspects of monitoring and capacity planning
- Building capacity planning into your design

Nothing Is Static

The best VMware vSphere designs are capable of changing, growing, and evolving. Just as the needs of the business that is implementing VMware vSphere in its environment will change, grow, and evolve over time, the vSphere design must be able to do the same thing. As the organization increases in size, the vSphere environment must also be able to increase in size. As the organization brings on new types of workloads, the environment must be able to handle, or adapt to handling, these new types of workloads. The design can't remain static. This ability to be flexible and scalable on demand is what experts refer to as *elasticity*. This is one popular entry point for a cloud solution discussion because Infrastructure as a Service (IaaS) or Platform as a Service (PaaS) resources are immediately available to accommodate both trending growth and immediate spikes in infrastructure utilization while sparing some of the costs of ownership that go along with infrastructure growth without compromising on critical points such as security.

The challenge for a vSphere architect, then (and we use the term *architect* here to mean anyone who designs vSphere environments), is to build environments that can grow and adapt. What does this mean, exactly? Specifically, a design that is capable of growing and adapting has the following qualities:

- The design should be *flexible*—it should accommodate both the addition as well as the removal of resources. There must be a framework present in the design whereby administrators can add servers, storage, or networking to or remove them from the design without having to redo the entire design.

- The design should be *instrumented*—it should incorporate monitoring as an integral component, providing the organization implementing the design with a means to know about the current utilization and behavior of the components within the design. Without instrumentation, it's impossible to know if resources must be added to or removed from the design.

Throughout this book so far, we've attempted to show you how to accomplish the first item in this list: how to build frameworks into your design so you can add or remove resources in a way that doesn't compromise the overall design.

This chapter is primarily focused on the second item: instrumentation. Specifically, this chapter discusses how and why architects should incorporate monitoring and capacity planning into VMware vSphere designs. Monitoring and capacity planning are the two sides of the instrumentation coin: one side tells you what's happening in your environment right now, and the other side tells you what will happen in your environment in the future.

Let's start by looking at how and why you should incorporate monitoring into your designs.

Building Monitoring into the Design

Designing a VMware vSphere environment that lacks sufficient monitoring is like designing a car without an instrument panel—there is no information available to operate it! Hence, our use of the term *instrumentation*. It enables operators of the vSphere environment you've designed to run it, to know what it's doing, and to be aware of the behaviors and limitations of the components that make up the design. Just as the various instruments in a car notify the operator of what's happening in the various systems that form an automobile, the monitoring that you incorporate into your vSphere design will notify the operator of how the various systems that form your design—compute, memory, storage, networking, security—are behaving and operating. This information makes it possible for the operator to use the flexibility you designed into the environment to add resources, remove resources, or reassign resources as needed.

Like so many other aspects of design, incorporating monitoring into a VMware vSphere design means answering a number of questions:

- What monitoring tools will the design use?
- What items are monitored?
- What values, or thresholds, will represent abnormal operation? What thresholds will represent normal operation? In other words, how do the operators determine if something is wrong with the design?
- What will the environment do, if anything, when an abnormal threshold is detected?
- Will the monitoring tools alert the operators? If so, how? Using what mechanisms?

The following sections address each of these questions.

Determining the Tools to Use

Like so many other areas of vSphere design, the first two questions are somewhat interrelated: the choice of tools can affect what items can be monitored, and the items that must or should be monitored can sometimes determine what tools need to be used.

For the purposes of this chapter, we'll group monitoring tools into three categories:

- Built-in tools that are supplied with VMware vSphere as part of the typical installation
- VMware vSphere tools integrated outside of a typical installation
- Third-party tools written by independent software vendors (ISVs)

Among these three categories, your job as a VMware architect or lead designer is to select the right tools or combination of tools that will meet the organization's functional requirements with regard to monitoring. You may end up using only built-in tools, if those meet the requirements. Alternately, you may end up with a mix of tools from all three categories in order to satisfy the requirements for the design.

Let's take a closer look at some of the tools that fall into these categories.

Using Built-in Tools

VMware vSphere offers a number of tools that are built into, or provided as part of, the standard product suite. Depending on the size of the intended VMware vSphere environment and the monitoring needs of the organization, it's entirely possible that the built-in tools will be sufficient.

What are some of these built-in monitoring tools? A few of them are described next:

vCenter Server's Alarms vCenter Server offers a fairly extensive set of alarms to monitor and alert on things like loss of network uplink redundancy, cluster high availability (HA), host memory usage, and VM CPU utilization. Figure 10.1 shows a screenshot taken from the vSphere Web Client when connecting to vCenter Server; this figure shows some of the alarms that are predefined in vCenter Server.

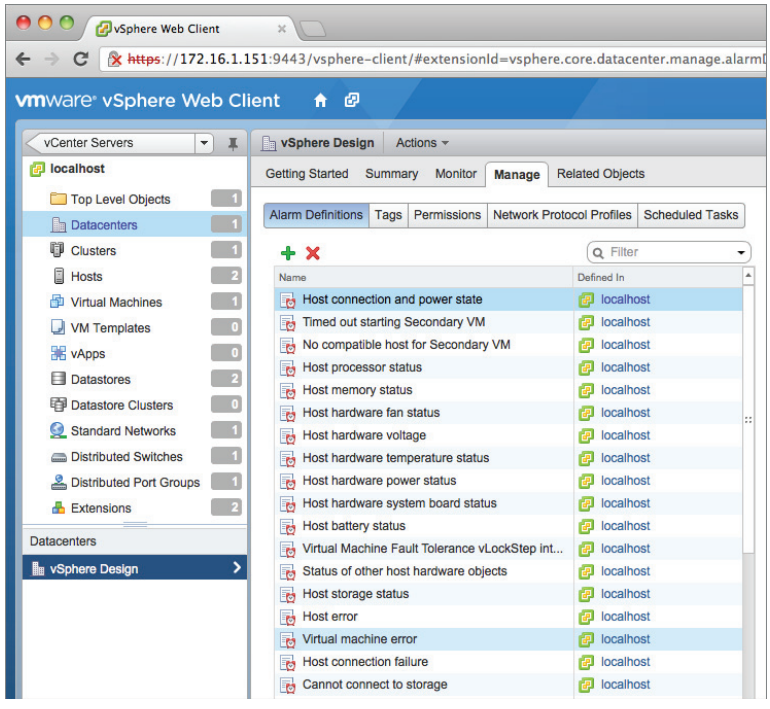


FIGURE 10.1 vCenter Server's alarms functionality can alert on many different conditions in the virtualized environment.

vCenter Server's Performance Charts In addition to alarms, vCenter Server offers an extensive set of performance charts, which can help administrators get a better view of the behavior of the environment. Administrators also have the option of building custom performance charts, to see the specific data in which they're interested. This may be specific performance counters or all counters for a specific time window. Figure 10.2 shows some of the standard performance charts in vCenter Server.

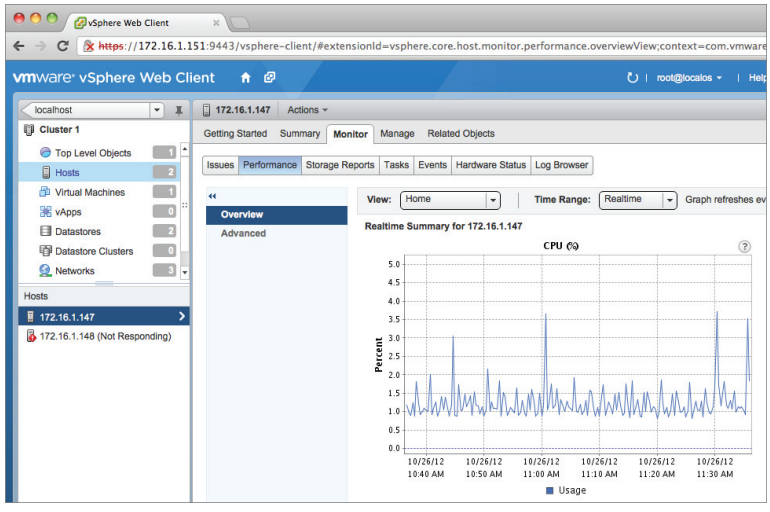


FIGURE 10.2 vCenter Server offers both standard and custom performance charts to provide the specific information administrators need to see.

As an aside, you'll also note that Figure 10.2 shows a host that isn't responding. (It's actually powered off at the moment.) This triggered one of the built-in alarms shown in Figure 10.1 (the "Host connection and power state" alarm), although it's hard to tell that in the screenshots.

Performance Monitor Counters (Windows Guests Only) When the VMware Tools are installed into VMs running a Windows guest OS, VMware-specific Performance Monitor counters are installed. Two VMware-specific objects, VM Memory and VM Processor, contain a number of different performance counters that are specific to virtualized instances of the Windows OS. Using these tools, administrators can capture more detailed information about the behavior of Windows instances running in a VMware vSphere environment. Figure 10.3 is a screenshot taken from a virtualized instance of Windows Server 2008 R2, showing the VMware-specific performance objects and counters.

VMware provides other tools that can provide insight into the operation of a VMware vSphere environment, but we wouldn't necessarily classify them as monitoring tools. They include tools like `esxtop` and `resxtop`, both of which provide useful and detailed information about the operation and behavior of specific VMware ESXi hosts. But neither of these utilities was designed as a long-term monitoring solution. vCenter's alarms and performance charts can not only present near real-time information, but also operate in a longer-term arrangement to provide current as well as past information. There is also a measure

of consistency in reporting and monitoring across environments in an organization when using vSphere's native tools; this consistency can be helpful in baselining and trending.

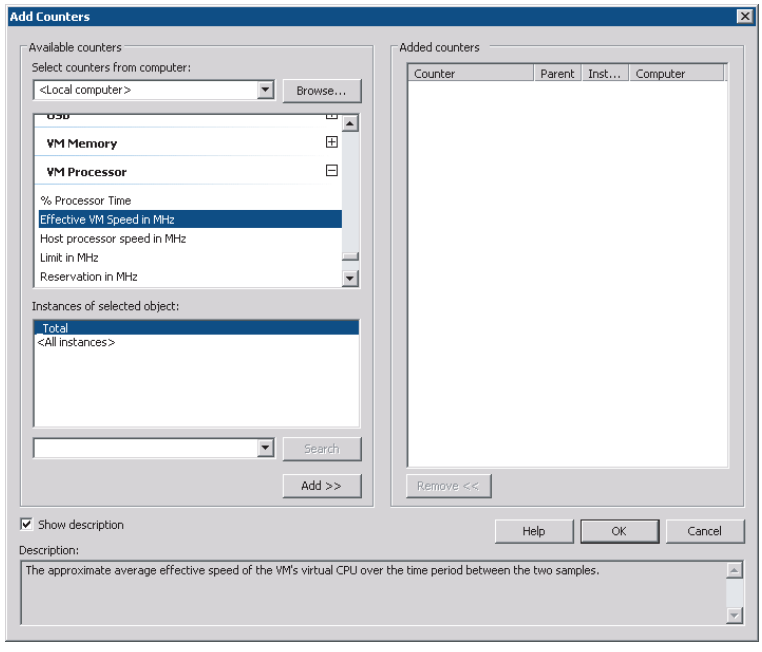


FIGURE 10.3
VMware Tools adds VMware-specific performance objects and counters to virtualized Windows Server instances.

The decision whether to use the built-in tools boils down to whether they meet the functional requirements of your organization. Further, because these tools are supplied in the standard environment, the question is most often not whether the tools will be used, but whether the tools will be used alone.

The answer to this question lies in the organization's functional requirements. The vCenter Server alarms, for example, only work within the framework of vCenter Server and are therefore bound to the same integration mechanisms as the rest of vCenter Server. Does your organization need more than the ability to send an email message, throw an SNMP trap, or run a script or an application if an alarm occurs? If so, then using vCenter's alarms alone may not meet your functional requirements. Does your organization need more detailed performance information than the vCenter Server or Windows Performance Monitor can provide? If so, then you may need to employ other tools in addition to or even instead of the built-in tools.

In cases like this—where the built-in tools don't meet the functional requirements for monitoring—VMware offers some additional tools that may help.

Using Additional VMware Tools

As the vSphere product line has matured, VMware has also added more products, especially management products. Some of these management products may find a way into your VMware

vSphere design where warranted by your organization's monitoring needs. Several of these management products provide monitoring functionality in some form or fashion; one example is VMware vCenter Operations Manager, more commonly known just as vCenter Operations (see Figure 10.4).

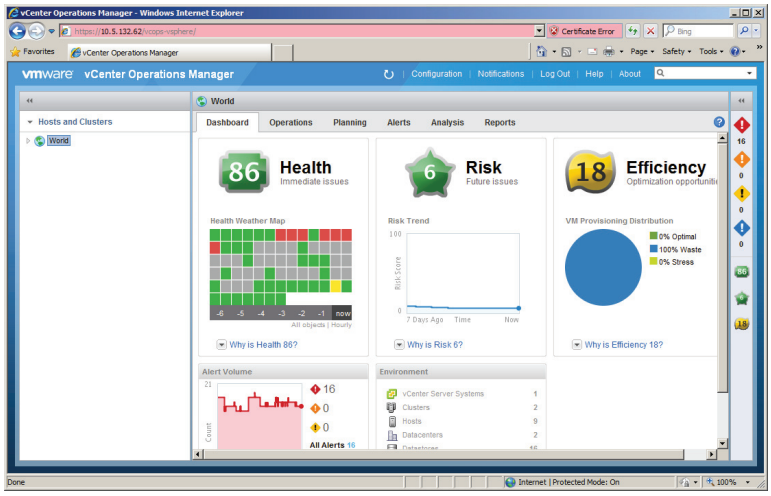


FIGURE 10.4
vCenter Operations provides both high-level monitoring functionality and the ability to drill down into specific areas for more information.

vCenter Operations (often referred to as vC Ops, or vCOps) came out of VMware's acquisition of a company called Integrien, and it's considered by many to be the centerpiece of VMware's management and monitoring strategy. vCenter Operations gathers key metrics from objects at all levels of the vSphere environment. For example, vCenter Operations may collect metrics from VMs, vCenter Server, ESXi hosts, clusters of ESXi hosts, and datastores. Using all these metrics, vCenter Operations then performs analysis to provide the following:

- Scores for health, efficiency, and capacity risk
- The range of normal behavior for all metrics, so it can highlight abnormalities in the metrics over time
- Graphical representations of the current and historical states of the vSphere environment (or portions of the environment)

The ability for vCenter Operations to establish the normal range of metrics is an important part of the product's functionality. Using this understanding of what is normal for a given metric, vCenter Operations can establish dynamic thresholds that change over time. Consider an environment in which disk I/O is high every week because of data being dumped from a particular application. vCenter Operations can learn that this is normal behavior and then alert you via dynamic thresholds only when disk I/O is abnormal during that time period. In contrast, vCenter Server itself only supports hard thresholds, which simply alert any time a metric exceeds a specified value (even if the utilization of that metric is expected and normal).

In addition to vCenter Operations, VMware offers a couple of other applications that you may also need to incorporate into your design, based on the monitoring requirements that have been specified:

- VMware vFabric Hyperic for application management of web and custom applications
- VMware vCenter Infrastructure Navigator for application-level discovery and dependency mapping

NOTE VMware's public website at `www.vmware.com/products` has more detailed information about the features and functions found in the products mentioned in this section.

In continuing to evaluate which tools to use in monitoring your VMware vSphere environment, if neither the built-in tools nor additional tools from VMware provided the necessary functionality to meet the organization's requirements, you must incorporate third-party tools into the design.

Using Third-Party Tools

In addition to the tools supplied with the VMware vSphere suite and the other VMware products, a number of third-party tools have been created by a variety of ISVs. These products vary widely in functionality, complexity, and cost, and architects designing VMware vSphere environments should carefully evaluate these products to find the one that most closely matches the needs of the environment.

A few third-party monitoring tools include the following:

- Zenoss (`www.zenoss.com`) is an open source monitoring solution that claims to offer agentless visibility into both physical and virtual environments.
- Dell, through its acquisition of Quest (`www.quest.com`), offers a number of management and monitoring solutions. In the virtualization space, Quest vFoglight claims to offer virtual infrastructure monitoring and VMware ESX(i) management in a single product. Prior to the Dell acquisition, Quest had purchased a company called VKernel that offered some monitoring solutions; that product is now called vOPS Server.
- Veeam (`www.veeam.com`) offers several virtualization-focused products, including Veeam ONE. Veeam ONE provides monitoring and reporting functionality.
- eG Innovations (`www.eginnovations.com`) provides an agent-based monitoring solution that supports multiple virtualization platforms as well as a number of specific applications.
- PHD Virtual (`www.phdvirtual.com`) has a solution, PHD Virtual Monitor for VMware vSphere, that is agentless and uses the vSphere API to gather metrics.
- Xangati's StormTracker (`www.xangati.com`) offers dynamic thresholds and the ability to review the state of the environment at any point in time.

NOTE There are many, many more third-party tools than the few that we've listed; inclusion in our list doesn't constitute an endorsement. A quick Google search will reveal a wide variety of third-party monitoring solutions that you can evaluate for inclusion in your VMware vSphere design. Further, as you can tell from the text, this space changes rapidly, so the information provided here can quickly become dated due to acquisitions and mergers.

Incorporate tools such as these into your designs on an as-needed basis to meet the functional requirements of your organization. The features and functions of these products vary widely from product to product, as do the system requirements to run the products. Be sure to perform the necessary testing to ensure that the products work in your environment and that you have a solid understanding of the resource requirements before folding a third-party solution into your design.

Your testing should include more than functional testing, though. When you're testing third-party tools for inclusion in your environment, we recommend that you include scaling and integration testing. You'll want to be sure the tool you're considering scales properly with your environment. As the environment grows, will the tool continue to perform consistently and acceptably? Does the tool hit a performance ceiling before the environment scales to its anticipated limits? Integration is another important point. Will you be able to extract data from the tool for use elsewhere? That may be necessary in some environments, and the use of proprietary database engines can limit integration into existing environments.

Licensing is another area to evaluate carefully. Some tools are licensed on a per-socket basis, and other tools are licensed on a per-VM basis. Will the costs of the tool fall within acceptable limits as your environment grows? Or will spiraling licensing costs make you unable to keep the tool as your environment scales?

When you've determined what tools to use, you're ready to proceed with selecting what items those tools will monitor.

Selecting the Items to Monitor

As we mentioned earlier, sometimes the choice of what items to monitor will affect the tools that should be used to monitor those items. Because of this fact, you'll need to have a reasonably solid idea of what items you want to monitor in your design as you evaluate tools and products to incorporate into the design.

Just as the list of third-party tools is far too long to try to include each and every product, providing a list of things you should monitor in your VMware vSphere environment probably isn't possible. There are simply too many things, and the list of items you need to monitor is heavily influenced by your organization, functional requirements, business needs, and other factors.

Common things that you may want to monitor include the following:

- Storage performance, typically monitored by watching the latency in milliseconds (ms) of transactions or storage operations
- Storage capacity, as in the number of terabytes (TB) available on the storage platform(s)
- Storage overallocation, if you're using thin provisioning at either the vSphere layer or in the storage array itself
- Storage utilization, defined as a comparison of the total storage available and the amount of storage actually being used

- CPU utilization on both VMware ESXi hosts as well as within virtualized guest OS instances
- RAM utilization on both the VMware ESXi hosts as well as in the virtualized guest OS instances
- Memory overcommitment, to ensure that excess memory pressure doesn't significantly impact performance in a negative fashion
- Network utilization, which lets you see how much network bandwidth is actually being used
- VM availability, which monitors for the availability of individual VMs
- Host availability, checking to ensure that VMware ESXi hosts are up and running and accessible across the network
- Storage availability, monitoring whether the storage is accessible and responsive to I/O requests from hosts and guests
- Application availability, to ensure that applications in guest OS instances are running and responding to requests in an appropriate manner

This is just a sample list; your final design should specify a much more comprehensive list of the items you'll monitor. One way to approach this may be creating a list of questions about the environment and then ensuring that the correct items are being monitored to answer those questions. For example, consider the question "How many terabytes of storage are available for use?" By monitoring storage capacity, either at the array level or the hypervisor level or both, you can be sure that the information to answer that question is available to the operators of the virtualization environment.

Earlier in this section and in this chapter, we've mentioned the close relationship between the items to be monitored and selecting the tools you use to monitor. These two items often influence one another. Consider this example: your design requires application-level awareness, such as the ability to verify the availability of specific services in a guest OS instance. What tools are most appropriate?

In this case, the built-in tools are insufficient. They lack application-level awareness and the ability to monitor individual processes within VMs. For this reason, you probably need to select an agent-based third-party tool. With an agent installed in the guest OS instance, the monitoring solution gains more application-level awareness and knowledge and can monitor for specific service availability.

Selecting the items to monitor must also extend beyond the virtualization environment. As a core technology in your datacenter, virtualization will touch many other areas, as you've seen throughout this book so far. Your VMware vSphere design should also specify the items that need to be monitored in other areas of the infrastructure. Some examples may include the following:

- Storage-capacity and storage-performance monitoring, taken from the array using tools provided by the storage vendor (these metrics may provide additional insight at a deeper level than would be available from the virtualization layer)
- Network utilization or network errors, taken from the networking equipment that sits upstream of the virtualization environment

- Hardware-related errors reported via hardware agents or via Common Information Model (CIM) to a centralized server management console (these may also be reported up through vCenter Server, but with less detail)
- Application-level errors or reports that are specific to the particular applications running in the guest OS instances on the VMware vSphere environment (for example, performance reports taken from a database)

Failure to properly define how all these metrics will be monitored means you've potentially omitted a source of data that could provide ongoing feedback about the health and performance of your VMware vSphere environment. This is why it's important to be as thorough and detailed as possible when defining the items and metrics that are included in the design for monitoring.

After you've selected the items to monitor, you'll need to determine the thresholds for those items.

Selecting Thresholds

A monitoring threshold determines a behavior or operation that is considered normal or abnormal. Because every design and every implementation is slightly different, it's not practical for us to provide a comprehensive list of thresholds and items. These need to be determined on a per-project basis.

Let's look at an example. In the early phases of virtualization adoption, customers were virtualizing low-level workloads that had very low CPU utilization. These included things like Active Directory domain controllers, web servers, DHCP or DNS servers, and similar workloads. It was easy to stack lots of these workloads together and achieve high consolidation ratios while still seeing host CPU utilization less than 50–60%. As a result, many organizations tuned their thresholds so that high host CPU utilization, in excess of 80%, would result in an alarm. Why? Simple: because these CPU values were uncommon given the workloads running on the virtualized infrastructure.

As virtualization has matured, however, customers are now virtualizing more substantial workloads, and these more substantial workloads—which include applications like Microsoft SQL Server, Microsoft Exchange, SAP, and others—naturally generate higher CPU loads. Consolidation ratios are lower with these types of workloads, and overall host CPU utilization is higher. As a result, organizations have to retune their thresholds to accommodate the fact that host CPU utilization is now generally higher.

We present this example to reinforce the statement made at the start of this section: thresholds are intended to identify abnormal behavior and generally need to be defined on a per-customer, per-project, or per-implementation basis. Although VMware can and does present general guidelines for thresholds, both in the form of predefined alarms as well as in performance white papers and similar documents, in the end the thresholds you use should be defined based on the specific workloads you are or will be running. For this reason, we don't provide any recommendations for thresholds in this book.

NOTE If you're using a product like vCenter Operations that offers dynamic thresholds, then it might not be necessary to establish hard (fixed) thresholds for resources and metrics in the environment. Even with the use of a product that offers dynamic thresholds, though, you might still want to define hard thresholds for some behaviors or conditions.

What if you don't know what workloads you'll be running on the virtualized infrastructure? There will always be an amount of uncertainty with regard to the workloads that will run on your environment. After all, who can tell the future? Who can know what sorts of workload the organization will need to deploy six months from now or a year from now? The answer lies in capacity planning, which we'll discuss later in this chapter in the section "Incorporating Capacity Planning in the Design."

With the items to monitor, the tools with which to do the monitoring, and thresholds defined, it's now time for you to decide what action to take when a threshold is reached.

Taking Action on Thresholds

For most, if not all, monitoring solutions, an action is taken when a threshold is reached. When host CPU utilization exceeds the threshold you've defined as acceptable and normal, "something" happens. That "something" may be sending an SNMP trap to your SNMP management system, or it may be running a script. The "something" may be different based on which threshold was reached. Whatever the "something" is, you'll need to define this action in your design.

As you'll see with many different areas of virtualization design, this area both is heavily influenced by and heavily influences the selection of the monitoring tools. If you aren't considering the influence that this has on your selection of monitoring tools, you're missing a key component. For example, if you decide to use vCenter Server's alarms as the sole monitoring tool for your virtualized environment based only on the types of items it monitors and the granularity it provides, but you haven't considered the types of actions vCenter Server can take, you've missed half the picture. It's important to consider both aspects when selecting a monitoring solution.

If the monitoring solution offers the ability to execute a script or an application, then you have tremendous flexibility in generating actions. Essentially, the actions you can take are limited only by the limitations of your chosen scripting language. Given that VMware has enthusiastically embraced PowerShell and offers extensive PowerShell integration, having the ability to execute a PowerShell script in response to a threshold having been reached or exceeded is a useful feature.

Here's an example of how you can use scripting (with potentially any scripting language, but especially with PowerShell) with alarms. Let's say you create an alarm in vCenter Server that monitors datastore free space. When that threshold of available free space in the datastore triggers the alarm, you can have the action set to execute a PowerShell script that initiates a storage vMotion operation to migrate a VM to another datastore. (Or you could use Storage DRS and accomplish the same thing, but without a script. Refer to Chapter 6, "Storage," for more information on Storage DRS.)

We hope this gives you some ideas about how you can use scripting and automation to broaden the reach of the actions taken by your alarms when thresholds are reached. However you use actions, though, you should be sure to thoroughly document the actions (and any accompanying scripts, if they exist) in your design documentation.

NOTE If you'd like to learn more about PowerShell and how you may be able to use it to help create custom actions for your monitoring thresholds, one good resource is *VMware vSphere PowerCLI Reference: Automating vSphere Administration* (Sybex, 2011), by Alan Renouf and Luc Dekens.

One final area remains to be addressed: alerting the operators when abnormal values are detected. In some instances, you may include alerting the operators in the actions you've defined when a threshold is reached. If alerting is handled separately, you'll need to define it. We'll cover this in the next section.

Alerting the Operators

In the event that alerting the operators of the virtualization environment is handled separately by the monitoring solution, you also need to decide how and when operators are alerted by the monitoring solution. In many cases, alerting is integrated with the actions that are taken when a threshold is met or exceeded. This is the case, for example, with vCenter Server alarms, where one of the possible actions is sending an email message via SMTP.

For those solutions that don't offer integrated alerting, or for those situations where alerting may be handled via an existing alerting mechanism such as an enterprise systems management tool, you'll need to define, in the design, the alerting structure that will apply to the monitoring system. Will administrators be alerted every time a threshold is met or exceeded, or only after a threshold is exceeded a certain number of times within a specified time window? Will the same type of alert be used in all cases, or will a different type of alert be used for more urgent situations? For example, excessive CPU utilization or high memory utilization within a virtualized instance of a guest OS may not need the same type of alert as a problem with storage availability or host availability. Your design needs to include a description of the types of alerts that will be provided and which thresholds will generate alerts. Some thresholds may generate alerts, but some thresholds may not. This all has to be included in your design.

Building an appropriately configured monitoring solution and strategy into your VMware vSphere designs is a key task. Without the instrumentation that a monitoring solution provides, it would be difficult, if not impossible, to truly understand the behavior of various workloads in the environment.

Capacity planning involves shifting the focus on monitoring from problem resolution to problem prevention. A solid monitoring solution tells you when a bottleneck is present; capacity planning attempts to prevent things from become a bottleneck. Monitoring is reactive; capacity planning is proactive. We'll dive into capacity planning in the next section.

Incorporating Capacity Planning in the Design

We've already said this, but it's a useful distinction between monitoring and capacity planning: monitoring tells you what *has happened* or *is happening*, and capacity planning tells you what *will happen*. Further, adequate capacity planning goes a long way toward helping an IT organization be proactive instead of reactive. A lack of capacity planning, on the other hand, often results in a reactive scramble for resources and unplanned long nights and weekends. Which would you prefer?

Capacity planning comes in two varieties, both of which are important to the architect of a VMware vSphere design:

- Capacity planning before virtualization occurs, such as that done by VMware Capacity Planner
- Capacity planning after virtualization, such as that done by VMware vCenter Operations, VMware vCenter Chargeback Manager, or similar products

In this section, we'll look at both types of capacity planning and why they're important to a solid VMware vSphere design.

Planning before Virtualization

Previrtualization capacity planning invariably involves the assessment of nonvirtualized systems by gathering information such as physical hardware; utilization of resources like CPU, memory, disk, and network; inventorying installed software on the systems; interaction with other systems on the network to understand dependencies; and analyzing all the data to provide estimates or suggestions of what a virtualized environment would look like. This process can be done manually using built-in tools supplied by OS vendors with their OSes, but we recommend the use of an automated tool to help with this task. If you prefer to manually assess your environment before virtualization, we discuss that process in more detail near the end of this section.

Using Tools for Previrtualization Capacity Planning

Should you choose to use a tool, a few tools provide the necessary functionality for previrtualization capacity planning:

- VMware Capacity Planner (`www.vmware.com/products/capacity-planner`)
- NetIQ PlateSpin Recon (`www.netiq.com/products/recon`)
- CiRBA (`www.cirba.com`)

Although the tools differ in feature sets, functionality, implementation, and operation, they share the same end result: providing sufficient information for a virtualization architect to properly design an environment that can support the workloads that are proposed to be migrated into this environment.

These tools enable a proper virtualization design by providing critical information necessary for a good design. This information includes the following:

- Resource usage in each of the major resource categories (CPU, memory, storage, and networking)
- Resource usage patterns, such as when workloads are their busiest or when certain types of resources are most heavily used
- Hardware inventory information that you can use to determine whether existing hardware can be repurposed into a virtualized environment
- Software inventory information

Although these technical details are important, sometimes nontechnical details are also important. Facts like business unit ownership, compliance, and regulatory requirements can also have a significant impact on the design of a VMware vSphere environment.

For example, consider the idea of business unit ownership. In some organizations, IT assets are owned by individual business units rather than by a central IT group. In these instances, consolidation can typically only occur within a business unit's IT assets rather than across the IT assets of multiple business units. If the previrtualization capacity-planning tool can't account

for this additional dimension, then the consolidation recommendations will be skewed because workloads will be stacked without consideration of IT asset ownership, thus potentially placing one group's workload on another group's physical assets. The results won't properly reflect the reality of how consolidation or virtualization will actually occur.

Some of the more advanced tools account for this functionality by incorporating additional business rules. You can use these business rules to model how the virtualized environment would be affected by including other criteria into the planning process. For example, if PCI-compliant and non-PCI-compliant systems must be kept separate, how does that affect the design? If DMZ and internal systems must not share common storage, how does that affect the design?

In many cases, VMware vSphere architects aren't given the opportunity to choose the previrtualization planning tool; they're called in after the initial analysis is performed and the results prepared. Although this situation isn't ideal, it's fairly common. In the cases where you're allowed to select the previrtualization capacity-planning tool, carefully compare the features of the tools to ensure that you select the tool that incorporates and analyzes the information necessary for your organization. If your organization is one in which individual business units own IT assets, then you may need a tool that incorporates business ownership as a factor for helping to determine consolidation ratios and how to stack applications onto a virtualized environment.

The importance of previrtualization capacity planning and the information it provides should be pretty obvious. Without a clear understanding and knowledge of the types of workloads that will be placed into the virtualized environment, it's almost impossible to craft an effective design. How will you size the storage if you don't know how much storage capacity (in gigabytes or terabytes) is required, or if you don't know how many input/output operations per second (IOPS) are required? How many hosts will you need? Without some sort of assessment of current CPU utilization, you'll be guessing.

Manually Performing Previrtualization Capacity Planning

So far, our discussion of previrtualization capacity planning has focused on the use of capacity-planning tools. As we mentioned earlier, it's possible to do previrtualization capacity planning without the use of additional tools. In these situations, you'll need to manually perform the tasks that are automated by tools. These tasks are summarized at a high level next:

1. Gather performance-utilization information for the physical servers that are included in your list of potential virtualization candidates. Applications and utilities provided by the OS vendors are key here, so use things like Performance Monitor on Windows Server–based systems or tools such as `vmstat` and `top` on Linux-based systems.

   Gather information about processor utilization, memory utilization, memory-paging activity, disk utilization for both local disks and storage area network (SAN) attached disks, and network utilization. You need to decide whether you'll gather average utilization only, peak utilization only, or some combination of the two.

   This utilization data also helps you establish a performance baseline, a reference point against which you can compare later. Performance baselines are helpful in a couple of ways: they help protect against skewed perceptions after virtualization ("It's slower now that it's virtualized") and can provide assistance when you're troubleshooting by making it easier to identify abnormalities in the environment.

Gather Both Peak and Average Utilization Data

Gathering only peak utilization data or only average utilization data will leave you exposed in your analysis. If you gather only peak utilization data, then your consolidation estimates will be much lower than they probably need to be. However, you'll be guaranteed that resources will exist to meet the peak requirements.

If, on the other hand, you gather only average utilization data, then you're likely to come up with consolidation estimates that are much closer to what is actually required. However, in this case you run the risk of stacking workloads whose peaks will cause a resource shortage to occur when they all ask for resources at the same time. This may cause the virtualized infrastructure to be unable to satisfy the resource requirements during these periods of peak demand.

What is the best approach? You should gather both peak utilization data and average utilization data, and do your best to incorporate both aspects into your consolidation assessment.

2. Gather inventory information for your virtualization candidates. Inventory information is necessary because utilization and performance data are typically reported as percentages. Although 90% CPU utilization may be a concern for a workload running on the latest and greatest 3.0 GHz processors, 90% CPU utilization on a 400 MHz CPU is an entirely different story. The same goes for memory usage, disk usage, and network usage.

3. Standardize the data you've gathered across different OSes. Virtualization makes it possible to run Windows Server–based systems and Linux- or Unix-based systems on the same physical hardware. However, the utilization/performance data provided by these OSes is in different formats and measurements. If you don't standardize the data across different OSes, then you'll only be able to estimate consolidation within groups of systems running the same OS instead of across all systems and all OS instances. This will generally result in lower consolidation ratios. Depending on the organization, this may be acceptable; but in many cases, organizations want to achieve the highest possible consolidation ratio.

   The process for standardizing the data depends on the tools being used and the OSes involved, but typically it involves converting the values into standard values that apply across OSes. For example, you can convert CPU utilization to gigahertz or megahertz and convert memory utilization into gigabytes or megabytes. Converting the percentage-based utilization data to solid numbers is also a byproduct of this step and a necessary part, as you'll see in the next step.

4. After you've standardized the data, you're finally ready to begin the process of stacking workloads. What do we mean by *stacking workloads*? For a group of workloads, you take the standardized resource-usage numbers created from the previous step and add them together. The result represents a rough analysis of what those workloads would look like if they were all running on the same hardware. You then compare the combined values against a reference point. The reference point is typically a set of limits on an anticipated hardware platform. This boils down to a trial-and-error process in which you compare various combinations of workloads against the reference point until you find the optimal combination for your environment.

For example, let's say you're planning to use servers that have two quad-core 2.53 GHz CPUs. That is a total of just over 20 GHz of CPU capacity. If you say you only want your hosts to be 80% utilized for CPU capacity, then you can stack workloads until you reach 16 GHz of utilization. You must repeat this process for each of the four major resource areas: CPU, memory, disk, and network. As you stack workloads, you also need to account for other sorts of limits, such as limiting the number of VMs per logical unit number (LUN) due to IOPS or limiting the number of VMs per host in order to minimize the size of the fault domain. Finally, if you can, you should also factor in nontechnical rules such as business-unit ownership or regulatory compliance when performing your analysis, because these rules will also affect the consolidation ratio as well as the placement of workloads on physical hosts.

As you can see, this can quickly become a complex multidimensional analysis. You'll probably need to perform this process multiple times against multiple reference points (for example, different hardware platforms or hardware configurations) in order to find the optimal combination of workloads, hardware platform, and hardware configuration. Keep in mind that this process isn't about the raw consolidation ratio—it's about finding the right balance of workloads on physical servers for maximum efficiency.

A Manual Analysis Will Provide a Rough Guess at Best

Performing a manual capacity-planning exercise before virtualization will provide a rough estimate, at best, of what the consolidation results will look like. Why? A manual analysis probably won't take into account VMware vSphere's memory-management functions, such as transparent page sharing, memory ballooning, and memory compression. A manual analysis also isn't likely to be able to take into account the effect of combining I/O workloads and the effect this will have on I/O performance, for example. Still, it will suffice to provide a rough estimate of what will be required.

5. Your final product provides at least three pieces of information. First, it gives you a rough idea of how many servers are required to virtualize the selected candidates. Second, it provides a list of which workloads are stacked with which other workloads after consolidation. Third, it identifies workloads relying on hardware that can't be virtualized (fax servers relying on a fax board, for example) and workloads whose resource utilization exceeds limits you've defined.

As you can see by this high-level description, manually performing previrtualization capacity planning is time consuming and difficult, especially in larger environments and environments that have nontechnical business rules affecting the results. For this reason, we recommend that you use one of the tools mentioned earlier for the majority of instances.

Before moving to the next section on capacity planning during virtualization, we feel it's important to pause for a moment and review the idea of a consolidation ratio. This is something we've mentioned a couple of times so far but haven't formally defined. What is a consolidation ratio? Is it important? What role does it play? These are all valid questions that individuals relatively new to virtualization might have, and the idea of a consolidation ratio is an important concept within the context of VMware vSphere design.

The *consolidation ratio* is the ratio of VMs to hosts, typically reduced to the number of VMs on a single host. For example, if your environment has (or will have) a consolidation ratio of 300:10—that is, 300 VMs running on 10 ESXi hosts—then you have a consolidation ratio of 30:1. The actual number of VMs that are (or will be) running might vary, especially considering the effect of features like vSphere HA and vSphere DRS, but on average you would expect to see about 30 VMs running on every ESXi host.

Consolidation ratios are important because, invariably, just about every capacity-planning exercise comes down to them. In step 4 of the process for manually performing a previrtualization assessment, we talked about stacking workloads. Stacking workloads is another way of talking about creating a consolidation ratio, especially as environments grow in scale and complexity. When environments are small, remembering the relationship between certain groups of workloads and the underlying ESXi hosts is (relatively) easy. As environments grow, and more workloads are created, it becomes harder to remember that relationship, and you need to take a higher-level view of the environment. A consolidation ratio is one way of taking that higher-level view. We'll revisit consolidation ratios again in the next section, which talks about capacity planning during (as opposed to before) virtualization.

NOTE Don't get caught up in using the consolidation ratio as a measure of the success of your project. It should be reasonably apparent by now that different workloads have different resource requirements and will therefore generate different consolidation ratios. Although you might expect to see very high consolidation ratios early in a virtualization project as you virtualize the low-hanging fruit in your datacenter, it's also natural to see consolidation ratios drop as you begin to move more resource-intensive workloads into the vSphere environment. This is natural and expected. The "best" consolidation ratio is the one that most efficiently satisfies the functional requirements of the design while minimizing potential risks.

Planning during Virtualization

Capacity planning after virtualization (or during virtualization, if you prefer) typically involves the use of historical performance and monitoring data to perform trending analyses. The results of these trending analyses are used to create projections of virtualization usage that help administrators understand how quickly resources are being consumed and when additional resources will need to be added to the virtualization infrastructure.

Using Tools for Capacity Planning during Virtualization

As with previrtualization capacity planning, a number of products on the market provide this functionality. In fact, most—if not all—of the products mentioned in this chapter offer both performance-monitoring functionality as well as capacity planning/forecasting features.

Although capacity planning during virtualization is, in large part, about forecasting trends in resource usage, a number of other features have come to be included in this category as well. These features include the following:

- Identifying inactive or idle VMs that may be consuming capacity. This is targeted at encouraging more effective VM lifecycle management. By decommissioning inactive VMs that may no longer be in use, organizations gain more efficient use of virtualization resources.

- Identifying VMs that aren't right-sized—that have been configured with more resources than they typically use based on historical performance data. Again, by right-sizing VMs, organizations can use virtualization resources more efficiently.
- Identifying orphaned VMs and VM resources, such as virtual disks that are still on the storage platform but are no longer referenced by an active VM.

Through the addition of these features and their association with tools that also perform capacity planning, many of the products and solutions are now referred to as *capacity-management* or *capacity-optimization* solutions. The fundamental purpose is the same, though: providing insight into resource usage in the virtualized environment and helping virtualization administrators know when more resources need to be added.

When it comes to selecting a tool to help with capacity planning (or capacity management, if you prefer), you'll want to answer some questions. The answers will help you determine which tool is right for your environment:

What Is the Resource Impact of This Tool? One of the basic principles of quantum physics is that you can't observe something without changing it. This principle holds true in virtualized environments as well—you can't observe the virtual environment without changing it. Usually, this change comes in the form of resource usage by the tools that are intended to watch resource usage. What you need to know, then, is how many resources this tool consumes. How much additional memory will it require? Will it require additional memory on every VMware ESXi host in your environment, or just on one? Does it require an agent installed in your guest OS instances? If so, what is the resource impact of that guest OS agent? How often does the agent need to be upgraded? What sort of maintenance or management does that agent require? How CPU intensive is the tool, or how CPU intensive are certain features of the tool?

Does It Meet Your Functional Requirements? At the risk of sounding like a broken record, we can't stress enough the importance of solid functional requirements. What specific features do you or your business require from this solution? Do you need the ability to do trending analysis of resource usage? Most solutions offer this functionality. Do you need the ability to identify orphaned VM resources, such as snapshots or virtual disks, which are no longer in use? Not all tools offer this functionality. Without a clear understanding of the basic functional requirements, you'll be unable to select the tool that best meets your needs.

What Impact Does This Tool Have on Your Design? Even if the capacity-planning tool is very lightweight and doesn't consume a great deal of resources, it will still have an impact on your design. Will it require specific configurations, such as probe VMs, to be installed on every VMware ESXi host? Does it require a certain type of networking configuration to be used, or does it only support a particular storage protocol? What is the financial impact on the design? In other words, how much does the tool cost? What is the operational impact on the design? Put another way, who will operate this software, and how does that fit into the design's existing or proposed operational model?

Based on the answers to these questions, you can begin to go about incorporating a capacity-planning tool into your overall design. After the selection of the capacity-planning tool is complete, you can amend your design, where necessary, to account for increased resource usage by the tool or to adjust for any changes to the operational procedures required in the design. You should also amend your design to account for the new functionality this tool adds; for

example, you may want to add operational procedures that discuss creating regular reports about inactive VMs, orphaned VM resources, or resource-usage trends. You may also consider building a design that is a bit leaner with regard to extra resources, knowing that the capacity-planning tool can provide recommendations about when additional resources will need to be added. This may help reduce initial acquisition costs and make the design more palatable for your organization.

Manually Performing Capacity Planning During Virtualization

As with previrtualization capacity planning, it's possible to perform capacity planning after virtualization without the use of additional tools. There will almost certainly be a feature gap between the use of third-party tools and a manual capacity-planning process, but this is nevertheless a viable approach for any VMware vSphere design.

What's involved in manually performing capacity planning? As with previrtualization planning, there are several steps:

1. Determine the specific aspects of the design for which you'll perform capacity planning. At the very least, we recommend that you include processor utilization, memory utilization, storage utilization from both a capacity and a performance perspective, and network utilization. As you add more utilization information, the analysis will become more complicated and more in-depth, so balance depth of information against your own skill in analyzing and correlating the data.

2. Begin gathering utilization data for the selected resource types. For example, begin periodically logging utilization data. You may be able to gather this utilization data directly from your VMware ESXi hosts, or you may need to extract it from vCenter Server.

   For example, if you want to gather data about guest OS storage latency as reported by Storage I/O Control in vSphere 4.1 and later, you need to get that information from vCenter Server. If you want information about CPU usage, you can get it directly from a host or from vCenter Server. It's possible that you can use vCenter Server's existing performance data, as long as vCenter Server's polling frequency and data-rollup schedule are acceptable for your purposes. (Many third-party tools rely on vCenter Server's database.) The same tip regarding peak and average utilization mentioned earlier in our discussion about manually performing previrtualization capacity planning also applies here.

3. When you have the utilization data, you need to analyze it in some fashion to get an idea of any trends that are hidden in the data. For example, you can use an Excel spreadsheet and chart to show you trends in average CPU usage over time.

4. Extrapolate those trends to see where things will stand one month, two months, or three months into the future. It's up to you how far out you want to look. Extrapolating data will tell you that you'll run out of memory in two months based on the current growth data, for example.

A simpler, but potentially less accurate, means of managing capacity centers on planning around VM growth. For this method, we'll use the idea of the consolidation ratio, which we defined and explained in the previous section. Because the consolidation ratio is itself an

approximate figure, this method may not be as accurate as more detailed assessments involving the use of performance metrics and data analysis, but it's still a valid approach.

For example, let's say you know that the number of VMs in your environment will increase by 25% per year. As a result, you can use the following formula to calculate how many additional hosts will be required in the next year:

$$(\text{Growth rate} \times \text{VM count} \times \text{Length of time}) \div \text{Consolidation ratio}$$

An environment with 200 VMs and a current (or expected) consolidation ratio of 15:1 results in the following calculation of additional hosts needed in one year:

$$(25\% \times 200 \times 1) \div 15 = 4 \text{ additional hosts (rounded up)}$$

Or consider an environment with 300 VMs, an expected (or measured) consolidation ratio of 12:1, and an expected growth of 20% over the next year:

$$(20\% \times 300 \times 1) \div 12 = 5 \text{ additional hosts}$$

Although this formula gives you an idea of how many hosts will be needed over the course of the next year, it doesn't tell you when those hosts will be needed. For that, you need to go back to monitoring resource usage and extrapolating data. Between these two methods, though, you should be able to get a fairly good handle on managing the growth of your VMware vSphere environment.

Other aspects of capacity management, such as identifying inactive VMs or orphaned VM resources, can be addressed through the definition of operational procedures that specify routine and regular audits of the environment and the VM configurations. This can be time consuming, but the cost savings resulting from more efficient use of virtualization resources may offset the expense of the additional operational overhead. Further, using automation tools such as vCenter Orchestrator and/or PowerCLI can reduce operational overhead and streamline tasks.

Capacity planning, like monitoring, can be an extremely useful and important part of your vSphere design. In some ways, it's every bit as important as the storage, networking, and cluster designs. In the next chapter, we'll put your design skills to the test in a review of a sample design intended to help you pull it all together.

Summary

In this chapter, we've discussed the importance of incorporating monitoring and capacity planning into your VMware vSphere design. A design can't be static; it must be flexible enough to grow or shrink as the company adopting the design also grows or shrinks. Monitoring provides the instrumentation necessary to determine the need to grow or shrink. When combined with capacity planning, operators not only know the immediate needs but can also attempt to forecast future needs. In many ways, monitoring and capacity planning are two sides of the same coin: one is reactive and targeted for the here and now (monitoring), and the other is proactive and targeted for the future (capacity planning).

When you're selecting a monitoring solution, ask yourself questions that help you determine the company's specific organizational and functional requirements. These requirements will often play a significant role in selecting the appropriate monitoring solution.

Because both performance monitoring and capacity planning/forecasting rely on data gathered from the environment, we see an increasing number of products that offer both features. This is true for VMware's own products—like vCenter Operations—as well as third-party products. As you've see, building capacity planning into the design allows future operators or IT directors to get a better idea of the growth trends and forecasted rate of acquisition for VMware ESXi hosts, network capacity, memory, and storage. Using this trending and forecasting data, administrators can add capacity to their environments before running out of resources.

Chapter 11

Bringing a vSphere Design Together

In this chapter, we'll pull together all the various topics that we've covered so far throughout this book and put them to use in a high-level walkthrough of a VMware vSphere design. Along the way, we hope you'll get a better understanding of VMware vSphere design and the intricacies that are involved in creating a design.

This chapter will cover the following topics:

- Examining the decisions made in a design
- Considering the reasons behind design decisions
- Exploring the impact on the design of changes to a decision
- Mitigating the impact of design changes

Sample Design

For the next few pages, we'll walk you, at a high level, through a simple VMware vSphere design for a fictional company called XYZ Widgets. We'll first provide a business overview, followed by an overview of the major areas of the design, organized in the same fashion as the chapters in the book. Because VMware vSphere design documentation can be rather lengthy, we'll include only relevant details and explanations. A real-world design will almost certainly need to be more complete, more detailed, and more in-depth than what is presented in this chapter. Our purpose here is not to provide a full and comprehensive vSphere design but rather to provide a framework in which to think about how the various vSphere design points fit together and interact with each other and to help promote a holistic view of the design.

We'll start with a quick business overview and a review of the virtualization goals for XYZ Widgets.

Business Overview for XYZ Widgets

XYZ Widgets is a small manufacturing company. XYZ currently has about 60 physical servers, many of which are older and soon to be out of warranty and no longer under support. XYZ is also in the process of implementing a new ERP system. To help reduce the cost of refreshing

the hardware, gain increased flexibility with IT resources, and reduce the hardware acquisitions costs for the new ERP implementation, XYZ has decided to deploy VMware vSphere in its environment. As is the case with many smaller organizations, XYZ has a very limited IT staff, and the staff is responsible for all aspects of IT—there are no dedicated networking staff and no dedicated storage administrators.

XYZ has the following goals in mind:

- XYZ would like to convert 60 existing workloads into VMs via a physical-to-virtual (P2V) process. These workloads should be able to run unmodified in the new vSphere environment, so they need connectivity to the same VLANs and subnets as the current physical servers.
- Partly due to the ERP implementation and partly due to business growth, XYZ needs the environment to be able to hold up to 200 VMs in the first year. This works out to be over 300% growth in the anticipated number of VMs over the next year.
- The ERP environment is really important to XYZ's operations, so the environment should provide high availability for the ERP applications.
- XYZ wants to streamline its day-to-day IT operations, so the design should incorporate that theme. XYZ management feels the IT staff should be able to "do more with less."

These other requirements and constraints also affected XYZ's vSphere design:

- XYZ has an existing Fibre Channel (FC) storage area network (SAN) and an existing storage array that it wants to reuse. An analysis of the array shows that adding drives and drive shelves to the array will allow it to handle the storage requirements (both capacity and performance) that are anticipated. Because this design decision is already made, it can be considered a design constraint.
- There are a variety of workloads on XYZ's existing physical servers, including Microsoft Exchange 2007, DHCP, Active Directory domain controllers, web servers, file servers, print servers, some database servers, and a collection of application servers. Most of these workloads are running on Microsoft Windows Server 2003, but some are Windows Server 2008 and some are running on Red Hat Enterprise Linux.
- A separate network infrastructure refresh project determined that XYZ should adopt 10 Gigabit Ethernet and Fibre Channel over Ethernet (FCoE) for network and storage connectivity. Accordingly, Cisco Nexus 5548 switches will be the new standard access-layer switch moving forward (replacing older 1 Gbps access-layer switches), so this is what XYZ must use in its design. This is another design constraint.
- XYZ would like to use Active Directory as its single authentication point, as it currently does today.
- XYZ doesn't have an existing monitoring or management framework in place today.
- XYZ has sufficient power and cooling in its datacenter to accommodate new hardware (especially as older hardware is removed due to the virtualization initiative), but it could have problems supporting high-density power or cooling requirements. The new hardware must take this into consideration.

Your Requirements and Constraints Will Likely Be Much More Detailed

The requirements and constraints listed for XYZ Widgets are intentionally limited to major design vectors to keep this example simple while still allowing us to examine the impact of various design decisions. In real life, of course, your requirements and constraints will almost certainly be much more detailed and in-depth. In fact, you should ensure that your design constraints and requirements don't leave any loose ends that may later cause a surprise. Although some loose ends could be categorized as assumptions, you'll want to be careful about the use of assumptions. Assumptions should not encompass major design points or design points that could be considered pivotal to project success or failure. Instead, carefully consider all assumptions you make and, where possible, gather the information necessary to turn them into requirements or constraints. Don't be afraid of being too detailed here!

Now that you have a rough idea of the goals behind XYZ's virtualization initiative, let's review its design, organized topically according to the chapters in this book.

Hypervisor Design

XYZ's vSphere design calls for the use of VMware vSphere 5.1, which—like vSphere 5.0—only offers the ESXi hypervisor, not the older ESX hypervisor with the RHEL-based Service Console. Because this is its first vSphere deployment, XYZ has opted to keep the design as simple as possible and to go with a local install of ESXi, instead of using boot from SAN or AutoDeploy.

vSphere Management Layer

XYZ purchased licensing for VMware vSphere Enterprise Plus and will deploy VMware vCenter Server 5.1 to manage its virtualization environment. To help reduce the overall footprint of physical servers, XYZ has opted to run vCenter Server as a VM. To accommodate the projected size and growth of the environment, XYZ won't use the virtual appliance version of vCenter Server, but will use the Windows Server–based version instead. The vCenter Server VM will run vCenter Server 5.1, and XYZ will use separate VMs to run vCenter Single Sign-On and vCenter Inventory Service. The databases for the various vCenter services will be provided by a clustered instance of Microsoft SQL Server 2008 running on Windows Server 2008 R2 64-bit. Another VM will run vCenter Update Manager to manage updates for the VMware ESXi hosts. No other VMware management products are planned for deployment in XYZ's environment at this time.

Server Hardware

XYZ Widgets has historically deployed HP ProLiant rack-mount servers in its datacenter. In order to avoid retraining the staff on a new hardware platform or new operational procedures, XYZ opted to continue to use HP ProLiant rack-mount servers for its new VMware vSphere environment. It selected the HP DL380 G8, picking a configuration using a pair of Intel Xeon E5-2660 CPUs and 128 GB RAM. The servers will have a pair of 146 GB hot-plug hard drives configured as a RAID 1 mirror for protection against drive failure.

Network connectivity is provided by a total of four on-board Gigabit Ethernet (GbE) network ports and a pair of 10 GbE ports on a converged network adapter (CNA) that provides FCoE support. (More information about the specific networking and shared storage configurations is

provided in an upcoming section.) Previrtualization capacity planning indicates that XYZ will need 10 servers in order to virtualize the 200 workloads it would like to virtualize (a 20:1 consolidation ratio). This consolidation ratio provides an (estimated) VM-to-core ratio of about 2:1. This VM-to-core ratio depends on the number of VMs that XYZ runs with only a single vCPU versus multiple vCPU. Older workloads will likely have only a single vCPU, whereas some of the VMs that will handle XYZ's new ERP implementation are likely to have more vCPUs.

Networking Configuration

As we mentioned, each of the proposed VMware vSphere hosts has a total of four 1 GbE and two 10 GbE network ports. XYZ Widgets proposes to use a hybrid network configuration that uses both vSphere Standard Switches as well as a vSphere Distributed Switch.

Each ESXi host will have a single vSwitch (vSphere Standard Switch) that contains all four on-board 1 GbE ports. This vSwitch will handle the management and vMotion traffic, and vMotion will be configured to use multiple NICs to improve live migration performance times. XYZ elected not to have vMotion run across the 10 GbE ports because these ports are also carrying storage traffic via FCoE.

The vSphere Distributed Switch (VDS, or dvSwitch) will be uplinked to the two 10 GbE ports and will contain distributed port groups for the following traffic types:

- Fault tolerance (FT)
- VM traffic spanning three different VLANs

A group of Cisco Nexus 5548 switches provides upstream network connectivity, and every server will be connected to two switches for redundancy. Although the Nexus 5548 switches support multichassis link aggregation, the VDS won't be configured with the "Route based on IP hash" load-balancing policy; instead, it will use the default "Route based on originating virtual port ID." XYZ may evaluate the use of load-based teaming (LBT) on the dvSwitch at a later date. Each Nexus 5548 switch has redundant connections to XYZ's network core as well as FC connections to the SAN fabric.

Shared Storage Configuration

XYZ Widgets already owned a FC-based SAN that was installed for a previous project. The determination was made, based on previrtualization capacity planning, that the SAN needed to be able to support an additional 15,000 I/O operations per second (IOPS) in order to virtualize XYZ's workloads. To support this workload, XYZ has added a four 200 GB enterprise flash drives (EFDs), forty-five 600 GB 15K SAS drives, and eleven 1 TB SATA drives. These additional drives support an additional 38.8 TB of raw storage capacity and approximately 19,000 IOPS (without considering RAID overhead).

NOTE This environment was designed for server virtualization, so this drives the storage configuration. If you were designing for an environment to support virtual desktops (VDI), which has very different storage I/O requirements and I/O profiles, then your design would need to be adjusted accordingly. For example, VDI workloads are read heavy during boot, but write heavy during steady state—so the storage configuration needs to take that I/O profile into account.

The EFDs and the SAS drives will be placed into a single storage pool from which multiple LUNs will be placed. In addition to supporting vSphere APIs for Array Integration (VAAI) and

vSphere APIs for Storage Awareness (VASA), the array has the ability to automatically tier data based on usage. XYZ will configure the array so that the most frequently used data is placed on the EFDs, and the data that is least frequently used will be placed on the SATA drives. The EFDs will be configured as RAID 1 (mirror) groups, the SAS drives as RAID 5 groups, and the SATA drives as a RAID 6 group. The storage pool will be carved into 1 TB LUNs and presented to the VMware ESXi hosts.

As described earlier, the VMware ESXi hosts are attached via FCoE CNAs to redundant Nexus 5548 FCoE switches. The Nexus 5548 switches have redundant uplinks to the FC directors in the SAN core, and the storage controllers of XYZ's storage array—an active/passive array according to VMware's definitions—have multiple ports that are also attached to the redundant SAN fabrics. The storage array is Asymmetric Logical Unit Access (ALUA) compliant.

VM Design

XYZ has a number of physical workloads that will be migrated into its VMware vSphere environment via a P2V migration. These workloads consist of various applications running on Windows Server 2003 and Windows Server 2008. During the P2V process, XYZ will right-size the resulting VM to ensure that it isn't oversized. The right-sizing will be based on information gathered during the previrtualization capacity-planning process.

For all new VMs moving forward, the guest OS will be Windows Server 2008 R2. XYZ will use a standard of 8 GB RAM per VM and a single vCPU. The single vCPU can be increased later if performance needs warrant doing so. A thick-provisioned 40 GB Virtual Machine Disk Format (VMDK) will be used for the system disk, using the LSI Logic SAS adapter (the default adapter for Windows Server 2008). XYZ chose the LSI Logic SAS adapter for the system disk because it's the default adapter for this guest OS and because support for the adapter is provided out of the box with Windows Server 2008. XYZ felt that using the paravirtual SCSI adapter for the system disk added unnecessary complexity. Additional VMDKs will be added on a per-VM basis as needed and will use the paravirtualized SCSI adapter. Because these data drives are added after the installation of Windows into the VM, XYZ felt that the use of the paravirtualized SCSI driver was acceptable for these virtual disks.

Given the relative newness of Windows Server 2012, XYZ decided to hold off on migrating workloads to this new server OS as part of this project.

VMware Datacenter Design

XYZ will configure vCenter Server to support only a single datacenter and a single cluster containing all 10 of its VMware ESXi hosts. The cluster will be enabled for vSphere High Availability (HA) and vSphere Distributed Resource Scheduling (DRS). Because the cluster is homogenous with regard to CPU type and family, XYZ has elected not to enable vSphere Enhanced vMotion Compatibility (EVC) at this time. vSphere HA will be configured to perform host monitoring but not VM monitoring, and vSphere DRS will be configured as Fully Automated and set to act on recommendations of three stars or greater.

Security Architecture

XYZ will ensure that the firewall on the VMware ESXi hosts is configured and enabled, and only essential services will be allowed through the firewall. Because XYZ doesn't initially envision using any management tools other than vCenter Server and the vSphere Web Client, the ESXi hosts will be configured with Lockdown Mode enabled. Should the use of the vSphere

command-line interface (vCLI) or other management tools prove necessary later, XYZ will revisit this decision.

To further secure the vSphere environment, XYZ will place all management traffic on a separate VLAN and will tightly control access to that VLAN. vMotion traffic and FT logging traffic will be placed on separate, nonroutable VLANs to prevent any sort of data leakage.

vCenter Server will be a member of XYZ's Active Directory domain and will use default permissions. XYZ's VMware administrative staff is fairly small and doesn't see a need for a wide number of highly differentiated roles within vCenter Server. vCenter Single Sign-On will use XYZ's existing Active Directory deployment as an identity source.

Monitoring and Capacity Planning

XYZ performed previrtualization capacity planning. The results indicated that 10 physical hosts with the proposed specifications would provide enough resources to virtualize the existing workloads and provide sufficient room for initial anticipated growth. XYZ's VMware vSphere administrators plan to use vCenter Server's performance graphs to do both real-time monitoring and basic historical analysis and trending.

vCenter Server's default alerts will be used initially and then customized as needed after the environment has been populated and a better idea exists of what normal utilization will look like. vCenter Server will send email via XYZ's existing email system in the event a staff member needs to be alerted regarding a threshold or other alarm. After the completion of the first phase of the project—which involves the conversion of the physical workloads to VMs—then XYZ will evaluate whether additional monitoring and management tools are necessary. Should additional monitoring and capacity-planning tools prove necessary, XYZ is leaning toward the use of vCenter Operations to provide additional insight into the performance and utilization of the vSphere environment.

Examining the Design

Now that you've seen an overview of XYZ's VMware vSphere design, we'd like to explore the design in a bit more detail through a series of questions. The purpose of these questions is to get you thinking about how the various aspects of a design integrate with each other and are interdependent on each other. You may find it helpful to grab a blank sheet of paper and start writing down your thoughts as you work through these questions.

These questions have no right answers, and the responses that we provide here are simply to guide your thoughts—they don't necessarily reflect any concrete or specific recommendations. There are multiple ways to fulfill the functional requirements of any given design, so keep that in mind! Once again, we'll organize the questions topically according to the chapters in this book; this will also make it easier for you to refer back to the appropriate chapter where applicable.

Hypervisor Design

As you saw in Chapter 2, "The ESXi Hypervisor," the decisions about how to install and deploy VMware ESXi are key decision points in vSphere designs and will affect other design decisions:

XYZ has selected local (stateful) installations of ESXi rather than boot from SAN or AutoDeploy. What are some drawbacks of this decision? For an organization that is new to VMware vSphere, using local (stateful) installations of ESXi is simpler and easier to

understand and might be the best approach—operationally speaking—for that particular organization. Remember that it's important to consider not only the technical impacts of your design choices, but also the organizational and operational impacts. However, while this design choice does have some advantages, it also has disadvantages. Upgrading or patching the ESXi hosts might be more complex, and expanding the capacity of the vSphere environment requires new local installs on new hardware. This could make it difficult for XYZ's IT organization to respond quickly enough to changing business demands as XYZ Widgets' business grows.

What impact would it have on XYZ's design to switch to AutoDeploy for the ESXi hosts? Switching from local (stateful) installations to using AutoDeploy would have several impacts on the design. XYZ would need to add DHCP and TFTP services to the server subnet (where they might not have been present before), and this addition might impact other servers and equipment on the same subnet. Using AutoDeploy would introduce a dependency on these other network services and require that XYZ also use Host Profiles (if it wasn't using them already). The Host Profiles requirement, in turn, would introduce a dependency on vCenter Server as well.

vSphere Management Layer

We discussed design decisions concerning the vSphere management layer in Chapter 3, "The Management Layer." In this section, we'll examine some of the design decisions XYZ made regarding its vSphere management layer:

XYZ is planning to run vCenter Server as a VM. What are the benefits of this arrangement? What are the disadvantages? As we discussed in Chapter 3, running vCenter Server as a VM can offer some benefits. For example, XYZ can protect vCenter Server from hardware failure using vSphere HA, which may help reduce overall downtime. Depending on XYZ's backup solution and strategy (not described here), it's possible that backups of vCenter Server may be easier to make and easier to restore. XYZ's hybrid design, illustrated in Figure 11.1, also sidesteps one perceived concern with running vCenter Server as a VM: the interaction between vCenter Server and the VDS it manages. Although changes in vSphere 5.1 greatly mitigate this concern (through VDS configuration rollback, for example), the placement of management traffic on a standard vSwitch eliminates this potential concern.

Are there other disadvantages that you see to running vCenter Server as a VM? What about other advantages of this configuration?

What would be the impact on XYZ's design if it wanted to run vCenter Server as a physical computer instead of as a VM? Ignoring the fact that VMware recommends running vCenter Server as a VM—and keeping in mind that best practices aren't to be followed blindly without an understanding of the reasoning behind them—it's possible for XYZ's design to be modified to run vCenter Server as a VM. However, a number of impacts would result. How will XYZ provide HA for vCenter Server? Will all of vCenter Server's components run on a single system, or will multiple physical systems be required? What about HA for the other components? What about environmental (power, rack space, cooling) considerations?

What is the impact of running all of vCenter Server's components, SQL Server, and vCenter Update Manager in the same guest OS instance? XYZ has opted to distribute these workloads across multiple VMs. If it consolidated them into a single VM, the resource needs of that

VM would clearly be much greater than they would have been without combining these applications. For running multiple components on the same system, VMware recommends a minimum of 10 GB RAM, and that doesn't account for the SQL database. Taking the SQL database into account, you're looking at a VM with at least 16 GB RAM and at least two vCPUs.

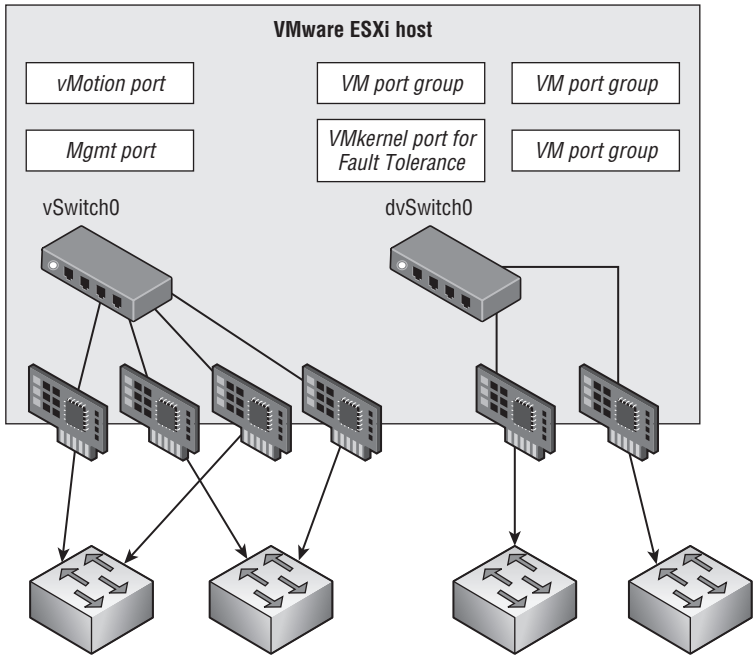


FIGURE 11.1 A hybrid network configuration for XYZ's vSphere environment

Overall, the configuration complexity is slightly reduced because there is no need for a dedicated service account for authentication to SQL Server and because there are fewer VMs to manage (only one VM running all three applications instead of three VMs, each running one application). On the downside, a fault in this VM will affect multiple services, and running all these services in a single VM might limit the scalability of XYZ's design.

Server Hardware

Server hardware and the design decisions around server hardware were the focus of our discussion in Chapter 4, "Server Hardware." In this section we ask you a few questions about XYZ's hardware decisions and the impact on the company's design:

What changes might need to be made to XYZ's design if it opted to use blade servers instead of rack-mount servers? The answers to this question depend partially on the specific blade-server solution selected. Because XYZ was described as using primarily HP servers, if the blade-server solution selected was HP's c7000 blade chassis, a number of potential changes would arise:

- The design description indicates that XYZ will use 10 physical servers. They will fit into a single physical chassis but may be better spread across two physical chassis to protect against the failure of a chassis. This increases the cost of the solution.

- Depending on the specific type of blade selected, the number and/or type of NICs might change. If the number of NICs was reduced too far, this would have an impact on the networking configuration. Changes to the network configuration (for example, having to cut out NFS traffic due to limited NICs) could then affect the storage configuration. And the storage configuration might need to change as well, depending on the availability of CNAs for the server blades and FCoE-capable switches for the back of the blade chassis.

What if XYZ decided to use 1U rack-mount servers instead of 2U rack-mount servers like the HP DL380 specified in the design description? Without knowing the specific details of the 1U server selected, it would be difficult to determine the exact impact on the design. If you assume that XYZ has switched to an HP DL360 or equivalent 1U rack server, you should ensure that the company can maintain enough network and storage connectivity due to a reduced number of PCI Express expansion slots. There might also be concerns over the RAM density, which would impact the projected consolidation ratio and increase the number of servers required. This, in turn, could push the cost of the project higher. You should also ensure that the selected server model is fully supported by VMware and is on the hardware compatibility list (HCL).

Would a move to a quad-socket server platform increase the consolidation ratio for XYZ? We haven't given you the details to determine the answer to this question. You'd need an idea of the aggregate CPU and memory utilization of the expected workloads. Based on that information, you could determine whether CPU utilization might be a bottleneck.

In all likelihood, CPU utilization wouldn't be a bottleneck; memory usually runs out before CPU capacity, but it depends on the workload characteristics. Without additional details, it's almost impossible to say for certain if an increase in CPU capacity would help improve the consolidation ratio. However, based on our experience, XYZ is probably better served by increasing the amount of memory in its servers instead of increasing CPU capacity.

Keep in mind that there are some potential benefits to the "scale-up" model, which uses larger servers like quad-socket servers instead of smaller dual-socket servers. This approach can yield higher consolidation ratios, but you'll need to consider the impacts on the rest of the design. One such potential effect to the design is the escalated risk of and impact from a server failure in a scale-up model with high consolidation ratios. How many workloads will be affected? What will an outage to that many workloads do to the business? What is the financial impact of this sort of outage? What is the risk of such an outage? These are important questions to ask and answer in this sort of situation.

Networking Configuration

The networking configuration of any vSphere design is a critical piece, and we discussed networking design in detail in Chapter 5, "Designing Your Network." XYZ's networking design is examined in greater detail in this section.

What would be the impact of switching XYZ's network design to use only 1 Gigabit Ethernet, instead of 1 and 10 Gigabit Ethernet? Naturally, XYZ needs to ensure that the servers could provide enough network throughput without the 10 GbE links. Further, because the 10 GbE links are running FCoE, XYZ needs to provide some sort of connectivity back to XYZ's existing SAN; that probably means the addition of FC HBAs into the servers. This raises additional questions—are FC HBAs available for this server model? Are enough FC ports

available on the SAN? What other operational impacts might result from this change? Most likely, XYZ needs to add not only FC HBAs but additional 1 GbE ports as well, which could really present an issue depending on the number of available PCIe slots in the server. Finally, XYZ needs to revisit the network equipment selection, because Nexus 5548 switches would no longer be needed to handle the 10 GbE/FCoE connectivity.

What benefit would there be, if any, to using link aggregation with the 1 GbE links in XYZ's design? The traffic that is going across the 1 GbE links is largely point-to-point; the management traffic is from the ESXi host to vCenter Server, and the vMotion traffic is host-to-host. Thus there would be very little benefit from the use of link aggregation, which mostly benefits one-to-many/many-to-one traffic patterns. Further, vSphere's support for multi-NIC vMotion already provides an effective mechanism for scaling vMotion traffic between hosts. The link-aggregation configuration is also more complex than a configuration that doesn't use link aggregation.

As a side note regarding link aggregation, the number of links in a link aggregate is important to keep in mind. Most networking vendors recommend the use of one, two, four, or eight uplinks due to the algorithms used to place traffic on the individual members of the link-aggregation group. Using other numbers of uplinks will most likely result in an unequal distribution of traffic across those uplinks.

TIP For more information about how the load-balancing algorithms on Cisco's switches work, refer to `www.cisco.com/en/US/tech/tk389/tk213/technologies_tech_note09186a0080094714.shtml`.

What changes would need to be made, if any, to XYZ's design if it decided to use only standard vSwitches instead of the current hybrid approach? What are the advantages and disadvantages to this approach? Switching to vSphere standard switches (vSwitches) offers a slight decrease in complexity but results in significant impacts on operational concerns. Figure 11.2 shows how a vSwitch could be swapped for a dvSwitch in XYZ's design.

As the result of using only vSwitches, the administrative overhead is potentially increased because changes to the network configuration of the ESXi hosts must be performed on each individual host, instead of being centrally managed like the VDS. The fact that vSwitches are managed per host introduces the possibility of a configuration mismatch between hosts, and configuration mismatches could result in VMs being inaccessible from the network after a vMotion (either a manual vMotion or an automated move initiated by vSphere DRS).

On the flip side, given that XYZ is using Enterprise Plus licensing, it could opt to use host profiles to help automate the management of the vSwitches and help reduce the likelihood of configuration mismatches between servers.

There is also a loss of functionality, because a distributed switch supports features that a standard vSwitch doesn't support, such as Switched Port Analyzer (SPAN), inbound and outbound traffic shaping, and private VLANs. It's also important to understand that some additional VMware products, such as vCloud Director, require a VDS for full functionality. Although XYZ doesn't need vCloud Director today, switching to vSwitches might limit future growth opportunities, and this consideration must be included in the design analysis.

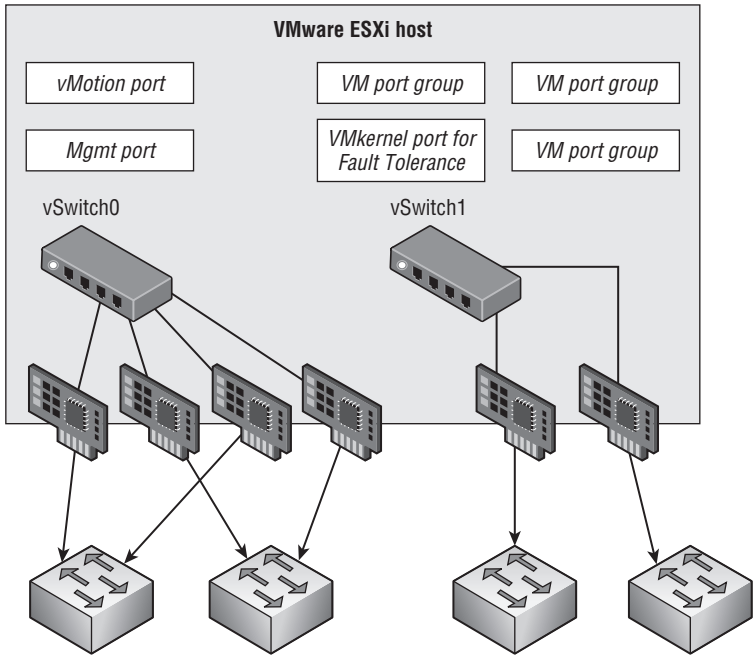


FIGURE 11.2
A potential configuration for XYZ using vSphere standard switches instead of a VDS

Shared Storage Configuration

The requirement of shared storage to use so many of vSphere's most useful features, like vMotion, makes shared storage design correspondingly more important in your design. Refer back to Chapter 6, "Storage," if you need more information as we take a closer look at XYZ's shared storage design:

How would XYZ's design need to change if it decided to use NFS exclusively for all its storage? Are there any considerations to this design decision? Some changes are immediately apparent. First, XYZ would no longer need FCoE CNAs in its servers and could get by with straight Ethernet adapters, although it would most likely want to keep the 10 GbE support. Second, the way in which the storage is presented to the hosts would likely change; XYZ might opt to go with larger NFS exports. The size of the NFS exports would need to be gated by a few different factors:

- The amount of I/O being generated by the VMs placed on that NFS datastore, because network throughput would likely be the bottleneck in this instance.
- The amount of time it took to back up or restore an entire datastore. XYZ would need to ensure that these times fell within its agreed recovery time objective (RTO) for the business.

XYZ's design already calls for 10 GbE, so network throughput won't generally be an issue. The use of link aggregation won't provide any benefit in an NFS environment, so the fact

that XYZ's design doesn't utilize link aggregation is a nonissue. On the other hand, XYZ might want to consider the use of Network I/O Control (NIOC) to more carefully regulate the IP-based traffic and ensure that the traffic types are given the appropriate priorities on the shared network links.

Historically speaking, VMware has had a tendency to support new features and functionality on block storage platforms first, followed by NFS support later. Although this trend isn't guaranteed to continue in the future, it would be an additional fact XYZ would need to take into account when considering a migration to NFS.

Finally, a move to only NFS would prevent the use of raw device mappings (RDMs) for any applications in the environment, because RDMs aren't possible on NFS.

How would XYZ's design need to change if it decided it wanted to use datastore clusters and Storage DRS in its design? The use of datastore clusters and Storage DRS could offer XYZ some operational benefits with regard to storage management and VM placement, so there are reasons XYZ might consider this as part of its design. If XYZ also wanted to use the array's autotiering functionality, though, it would probably need to configure Storage DRS to make its decision only on capacity, not on I/O latency. The use of datastore clusters might also lead XYZ to use a larger number of smaller datastores, because the vSphere administrators wouldn't need to worry about managing more datastores (these would be managed via Storage DRS for capacity and/or latency).

How would XYZ's design need to change if it decided to use iSCSI (via the VMware ESX software iSCSI initiator) instead of FC? As with replacing FCoE with NFS, the hardware configuration would need to change. XYZ would want to replace the FCoE CNA, although it would likely want to retain its 10 GbE connectivity. The company might also want to change the network configuration to account for the additional storage traffic, but the use of dual 10 GbE NIC ports doesn't provide much flexibility in that regard. Instead, XYZ might need to use NIOC and/or traffic shaping to help ensure that iSCSI traffic wasn't negatively affected by other traffic patterns.

The storage configuration might also need to change, depending on the I/O patterns and amount of I/O generated by the VMs.

Finally, using the software iSCSI initiator would affect CPU utilization by requiring additional CPU cycles to process storage traffic. This could have a negative impact on the consolidation ratio and require XYZ to purchase more servers than originally planned.

The default multipathing policy for an active/passive array is usually most recently used (MRU). Does XYZ's array support any other policies? What would be the impact of changing to a different multipathing policy if one was available? We noted that XYZ's storage array is an active/passive array, so the multipathing policy would typically be MRU. However, we also indicated that XYZ's array supports ALUA, which means the Round Robin multipathing policy is, in all likelihood, also supported. Assuming that typical storage best practices were followed (redundant connections from each storage processor to each SAN fabric), this means the VMware ESXi hosts will see four optimal paths for each LUN (and four non-optimal paths) and can put traffic on all four of those active paths instead of only one. This would certainly result in a better distribution of traffic across storage paths and could potentially result in better performance. It's important, though, to ensure that you follow the configuration recommendations available from the storage vendor where applicable.

VM Design

As we described in Chapter 7, "Virtual Machines," VM design also needs to be considered with your vSphere design. Here are some questions and thoughts on XYZ's VM design:

Does the use of Windows Server 2003 present any considerations in a VMware vSphere environment? In general, the only real consideration with regard to Windows Server 2003 comes in the form of file-system alignment within the virtual disks. Windows Server 2003 is a fully supported guest OS, and VMware vSphere offers VMware Tools for Windows Server 2003. However, by default, NTFS partitions created in Windows Server 2003 aren't aligned on a 4 KB boundary, and this misalignment can potentially have a significant impact on storage performance as the environment scales. Based on the scenario given, the number of Windows Server 2003 workloads is and will be relatively small; therefore, the impact on the storage environment is likely to be quite limited in most cases. Nevertheless, XYZ should take the necessary steps to ensure that file-system partitions are properly aligned, both for systems that are converted via P2V and for systems that are built fresh in the virtual environment. For systems built fresh for the virtual environment, XYZ can streamline the process by using VM templates and correcting the file-system alignment in the VM templates.

Many variations of Linux are also affected, so XYZ should ensure that it corrects the file-system alignment on any Linux-based VMs as well.

Note that both Windows Server 2008 and Windows Server 2012 properly align partitions by default.

What impact would using thin-provisioned VMDKs have on the design? The performance difference between thick-provisioned VMDKs and thin-provisioned VMDKs is minimal and not an area of concern. Potential concerns over SCSI reservations due to frequent metadata changes aren't an issue in an environment of this size and would be eliminated entirely if XYZ used the VAAI support in its array (which is enabled by default when ALUA is configured). Operationally, XYZ would need to update its monitoring configuration to monitor for datastore oversubscription to ensure that it didn't find itself in a situation where a datastore ran out of available space.

VMware Datacenter Design

The logical design of the VMware vSphere datacenter and clusters was discussed at length in Chapter 8, "Datacenter Design." Here, we'll apply the considerations mentioned in that chapter to XYZ's design:

What impact would it have on the design to use 2 clusters of 5 nodes each instead of a single cluster of 10 nodes? Cluster sizing affects a number of other areas. First, a reduced cluster size might give XYZ more flexibility in the definition of cluster-wide configuration settings. For example, does XYZ need an area where DRS is set to Partially Automated instead of Fully Automated? Do regulatory factors prevent XYZ from taking advantage of automated migrations that might drive this requirement? It's possible to set DRS values on a per-VM basis, but this practice grows unwieldy as the environment scales in size. To reduce operational overhead, XYZ might need to create a separate cluster with this configuration.

Reducing cluster size means you reduce the ability of DRS to balance workloads across the entire environment, and you limit the ability of vSphere HA to sustain host failures. A cluster of 10 nodes might be able to support the failure of 2 nodes, but can a cluster of 5 nodes

support the loss of 2 nodes? Or is the overhead to support that ability too great with a smaller cluster?

Does the use of vCenter Server as a VM impact XYZ's ability to use VMware Enhanced vMotion Compatibility? EVC will be very helpful to XYZ over time. As XYZ adds servers to its environment, EVC can help smooth over differences in CPU families to ensure that vMotion can continue to migrate workloads between old and new servers.

However, the use of vCenter Server as a VM introduces some potential operational complexity around the use of EVC. VMware has a Knowledge Base article that outlines the process required to enable EVC when vCenter Server is running as a VM; see `kb.vmware.com/kb/1013111`. To avoid this procedure, XYZ might want to consider enabling EVC in the first phase of its virtualization project.

Security Architecture

We focused on the security of vSphere designs in Chapter 9, "Designing with Security in Mind." As we review XYZ's design in the light of security, feel free to refer back to our security discussions from Chapter 9 for more information:

Does the default configuration of vCenter Server as a domain member present any security issues? If so, how could those issues be addressed? Recall that, by default, the Administrators local group on the computer where vCenter Server is installed is given the Administrator role in vCenter Server. When vCenter Server is in a domain, the Domain Admins group is a member of the local Administrators group. This confers the Administrator vCenter role on the Domain Admins group, which may not be the intended effect. To protect against this overly broad assignment of rights, you should create a separate local group on the vCenter Server computer and assign that group the Administrator role within vCenter Server. Then, remove the local Administrators group from the Administrator role, which will limit access to vCenter Server to only members of the newly created group.

Monitoring and Capacity Planning

Chapter 10, "Monitoring and Capacity Planning," centered on the use and incorporation of monitoring and capacity planning in your vSphere design. Here, we examine XYZ's design in this specific area:

If XYZ needs application-level awareness for some of the application servers in its environment, does the design meet that requirement? As currently described, no. The built-in tools provided by vCenter Server, which are what XYZ currently plans to use, don't provide application awareness. They can't tell if Microsoft Exchange, for example, is responding. The built-in tools can only tell if the guest OS instance is responding, and then only if VM Failure Monitoring is enabled at the cluster level.

If XYZ needed application-level awareness, it would need to deploy an additional solution to provide that functionality. That additional solution would increase the cost of the project, would potentially consume resources on the virtualization layer and affect the overall consolidation ratio, and could require additional training for the XYZ staff.

Summary

In this chapter, we've used a sample design for a fictional company to illustrate the information presented throughout the previous chapters. You've seen how functional requirements drive design decisions and how different decisions affect various parts of the design. We've also shown examples of both intended and unintended impacts of design decisions, and we've discussed how you might mitigate some of these unintended impacts. We hope the information we've shared in this chapter has helped provide a better understanding of what's involved in crafting a VMware vSphere design.

Chapter 12

vCloud Design

In this chapter, we'll examine how server profiling, networking design, storage design, high availability, DRS, and many other technologies apply to a vCloud Director design. This chapter assumes that you already understand many of the definitions and the terminology used with vCloud Director.

This chapter will cover the following topics:

- Differences between cloud and server virtualization
- Role of vCloud Director in cloud architecture
- vCloud Director use cases
- Components of the vCloud management stack
- vCloud cell and NFS design considerations
- Management vs. consumable resources
- Database concepts
- vCenter design
- vCloud management physical design
- Physical side of provider virtual datacenters
- Logical side of provider virtual datacenters
- Virtual network switch
- Network pool decisions
- External networks
- Designing organizations, catalogs, and policies
- Correlating organizational networks to design
- End users and vApp networking
- Designing organization virtual datacenters
- Multiple sites
- Backup and disaster recovery

Differences between Cloud and Server Virtualization

We've often sat in presentations and heard the question, "Who is running a private cloud today?" and watched 95% of the hands in the room reach for the sky. Of course, everyone has their own opinion on this topic, because the term *cloud* is always open for debate. This conversation has taken place with many influential people in the industry, and the same question continually arises: What's the difference between cloud and server virtualization?

Server virtualization (or a virtualized datacenter) is what many of us have been doing for years: acquire a couple of servers, switches, and a storage array; install vSphere; and make the components talk to one another. From here we can begin to do some P2Vs or create new VMs through wizards, templates, and scripting. The main drivers behind virtualization are consolidation, simplifying disaster-recovery (DR) efforts and administration, and achieving a lower total cost of ownership. At the heart of this is running a multitude of virtualized operating systems on a hypervisor that virtualizes the underlying hardware. Pretty fascinating stuff, but is that cloud? Of course not; it's just a cool technology.

Transitioning to a cloud operating model is completely different. You've probably heard this analogy a thousand times: cloud is like electricity. When you flip on a light switch, the light comes on. You don't care if that electricity was generated by coal, solar, or water, but that the light is on. It takes more than just virtualization to move to a cloud operational model. Virtualization is a key enabler of cloud because without it, we couldn't dynamically create resources at such a rapid pace.

What does it take to move beyond server virtualization and into cloud? Let's break this down into a few components:

Product There must be a product that consumers demand. In this case, IT departments or service providers have compute and storage capacity as well as network connectivity to sell as a product.

Multitenancy Every customer is different. Service providers have direct customers from different organizations, and internal IT organizations meet the demands of different departments, such as marketing, engineering, and HR. Your goal as a cloud architect is to ensure that each tenant has no visibility into another tenant's data.

Self-service Provisioning End users can use resources independently of the IT department. This point is critical in any cloud deployment. The ability of the end user to use products or resources without the assistance of the IT department streamlines processes and gives end users responsibility and control.

Catalog Users should have a variety of options from which to choose. Customers can access a self-service catalog provided by IT and provision vApps consisting of VMs of different types, sizes, prebuilt OSes, and even applications. Giving customers a choice makes adoption successful.

Automation and Orchestration A streamlined workflow creates efficiency. Many IT organizations still need to follow business procedures through approvals and signoffs. Service providers can produce distinguished technologies. An example through the use of orchestration could be as simple as a tenant ordering a SQL server. A workflow can be kicked off to find out who ordered that VM and to add user credentials for role-based access or even change the SA password. Consistency is critical in a cloud platform, and too much human intervention can lead to costly mistakes and errors. Automation is a critical part of the cloud experience.

Chargeback/Showback The cloud isn't free, so a process for charging or showing costs should be instituted. Billing, charging, and metering of the storage, compute, and networking of the entire infrastructure is how service providers make a profit. In the enterprise space, this is how IT can turn the table. IT is historically viewed as a cost center. In a cloud, business units are the parties responsible for the costs of IT infrastructure. Even if IT isn't actually charging the business units, IT can create showback reports to demonstrate that IT isn't the cost center. VMs being used by the different business units can be metered and reported.

Capacity Modeling and Planning Cloud providers need to plan for growth, and the amount of data generated will only increase. Capacity modeling is important so businesses can predict costs and budget for the future. Using proper tools allows a business to take an educated approach instead of blindly throwing money at resources.

Role of vCloud Director in Cloud Architecture

How does vCloud Director address the components of cloud architecture we've discussed? Does vCloud Director cover all the facets of cloud? The definition of cloud computing and VMware vCloud taken from vCAT 3.0 states, "Cloud computing leverages the efficient pooling of an on-demand, self-managed, virtual infrastructure that is consumed as a service. VMware vCloud is the VMware solution for cloud computing that enables delivery of Infrastructure as a Service (IaaS)."

VMware's definition covers many of these components, but it doesn't cover them all. Here is how vCloud Director addresses some of these components:

Product vCloud Director is an abstraction of vSphere resources that are transformed into consumable resources of the cloud.

Multitenancy vCloud Director has a concept of organizations or tenants that can create security zones through vCloud Networking and Security Edge devices and types of organizational networking.

Self-Service Provisioning vCloud Director contains an intuitive portal where tenants can provision resources into the cloud.

Catalog vCloud Director can contain multiple catalogs in a global fashion or dedicated to specific organizations.

Automation and Orchestration vCloud Director is responsible for automated provisioning of VMs, virtual networking, and consumable resources in the form of resource pools. However, vCloud Director doesn't have the native capability to do any advanced orchestration such as workflow approvals, triggering emails, and populating CMDBs.

Chargeback/Showback vCloud Director shows the amount of resources being consumed by organizations' virtual datacenters, but it's not a chargeback product.

vCenter Chargeback uses vCloud Director polling to collect the data necessary to assign dollar values to virtual and physical resources and to provide automated reporting.

Capacity Modeling and Planning vCloud Director shows the amount of resources being consumed by organizations' virtual datacenters, but it isn't a capacity-planning product.

vCloud Director is complemented by vCenter Operations with a vCloud Director plug-in to collect data for operational readiness, offering a proactive approach to troubleshooting and to model the capacity of consumed resources and future planning.

It's fair to say that vCloud Director doesn't cover all facets of cloud architecture. vCloud Director is one component of an entire cloud infrastructure, but depending on your use case it may be all that is needed. Many vendors, including VMware, have additional products to fill in the gaps where vCloud Director is lacking in terms of portal use, automation, and chargeback. Like any good architect, it's your job to determine requirements and define the products that will fit your design. We'll examine these pieces further in the next section.

vCloud Director Use Cases

Before beginning to create a cloud architecture, you need to understand if the vCloud Director product is necessary for a particular scenario. There are often misunderstandings about vCloud Director's functionalities, and as an architect you need to know when and where it fits. Project Redwood was the internal codename of vCloud Director, and it was touted as the new generation of VMware's IaaS cloud offering. Vendors are working on vCloud integration by creating plug-ins with their products, and partners, contractors, and vendors are pushing for rapid adoption. VMware has a vision of vCloud as the next step in datacenter transformation. What does this mean for you as an architect? Virtualization is a key component, but it's only a stepping stone. If you're thinking of adopting vCloud, you have to ask yourself, "What am I really trying to accomplish?" The answer to this question is unique to each scenario.

Are you architecting for yourself or a service provider, an enterprise customer, or a Small to Medium Business? Are you looking for a portal with a self-service catalog? Are you trying to create multitenant networks? The answer to this question is unique for everyone.

Let's look at what vCloud Director offers in terms of a product. From VMware's definition of cloud and vCloud Director, we can examine what vCloud offers and start identifying requirements.

Do you require the pooling of vSphere resources or multiple vSphere environments? This question is tailored for large vSphere farms with different types of infrastructures. Whether you have brand-new hosts with high-end storage arrays, pods of converged infrastructure, or a mixture of other low-end arrays and old servers, vCloud Director can inherit all of these resources. They're further subdivided into pools of consumable cloud resources. If you have a vSphere environment with minimal hosts, you'll most likely end up with a single infrastructure offering. If your environment is small, don't let that steer you away from vCloud Director, but understand that the logical configuration will be a bit different.

Do you require logical multitenancy? This is a typical case for service providers and many Fortune 500 companies. Does your enterprise require that HR, engineering, development, and other business units have a separation of IT resources for security, and chargeback purposes? Or does IT control the entire infrastructure, regardless of who owns it? This is a change in corporate thinking that needs to occur eventually if you want to move to a cloud-operating model. Just because that's the way it has always been done, doesn't mean that's the way it will always have to be done. There are use cases to satisfy the service provider as well as traditional IT for multitenancy that we'll examine.

Do you need a portal where users can access or request IT services? Enabling end users is always a key requirement, and it helps move innovation forward. You want to make processes simple for end users because complexity leads to failure.

If you've looked at the vCloud Director user interface for an end user, it may not be that simple. Many times, you need to demonstrate how an end user deploys a vApp after vCloud

is installed. Many users may find it very complex and that it won't meet their expectations and standards. Depending on the technical capabilities of the end user, you may need another off-the-shelf product to build a simple portal or to custom build a new portal from scratch to hook back into vCloud Director through APIs. After determining the requirements, you may discover that a portal with a few simple orchestrated workflows into vSphere is all that is needed to satisfy a customer's need and that vCloud Director isn't a necessary component.

A key point to mention is that vCloud Director can only provision virtual resources into vSphere. What if, in addition to VM provisioning, you also want to provide bare-metal provisioning, or to poke holes in a firewall somewhere, or to allow a user to request a new IP phone for their desk in a single catalog? vCloud Director won't be able to accomplish these tasks. This is another case where a custom-built portal or off-the-shelf product with integrations into orchestration tools will accomplish this goal.

What items do your users need to request from a self-service catalog? This feeds into the previous question about the portal: what do you want to offer? Without vCloud Director, you can offer pretty much anything because it can be virtual or physical, but more work is involved in creating custom portals, catalogs, and workflows. vCloud Director offers virtual resources but in ways that are unique.

First, vCloud can contain multiple global catalogs instead of a single ordering mechanism. Perhaps one global catalog has standard operating-system images of Win2K8R2, WinXP, WinXP_x64, Win7, Win7_x64, Ubuntu, and SuSE. Another global catalog offers ISOs of applications, such as SQL, Office, and Exchange. Yet another global catalog contains sets of VMs and applications packaged as a vApp, such as vApp1 = DB, app, and web server; and vApp2 = vCenter on 2K8, SQL on 2K8, and two ESXi hosts for a nested deployment of vSphere.

The other unique feature gives control to organizations so they can manage their own private catalogs. If a user in the development organization has a new beta code and they want to give other developers access to try it, they can upload that vApp into the development catalog to allow other developers to deploy it and test it out. This unique feature enables end-user capabilities without the constant need for IT intervention.

Do you need isolated and secure networks? You probably think you do, but again it depends on your requirements. Many architects misunderstand the implications that appliances from the vCloud Networking and Security Manager (vCNS Manager) suite throw into the mix. This is usually a standard requirement in service-provider environments where it's a guarantee that two tenants won't be able to see each other's traffic. In an enterprise environment, that may not be the case. Do the business units care if the HR server and the engineering server can ping each other? Some of this is accomplished today through Active Directory and Group Policies or at an L3 device with access control lists (ACLs). You also need to think about communication between VMs that exist on external networks, which we'll examine later in "External Networks."

Do you need automation and orchestration capabilities? Of course you do. We all do! vCloud Director can bidirectionally communicate with vSphere and provision resource pools, folders, port groups, and VMs. What about workflows with email approvals to deploy a certain VM? That isn't a part of vCloud Director, so you may need another custom portal and orchestrated design.

Let's examine some use cases and see if vCloud Director will fit.

Use Case #1

ACME Inc. has asked you to evaluate its environment to determine how it can become more streamlined. Today, a user requests a VM by sending IT an email with an attached Word document that specifies which OS and sets of applications are needed. The user has permission to request this VM from their manager. The VM will have an in-house application and will be used for test and development purposes. The VM can't interfere with the production network where the production application lives.

Is vCloud Director a good fit here? You have identified that the process to request VMs isn't efficient. The vCloud Director portal can easily accommodate itself to enable end users' requests. No approvals of workflows are necessary after IT receives the document; therefore, no additional orchestration is needed. The VM in question needs to be on a segregated network. You can assume that the network security team must use VLANs and ACLs to maintain segregation and not interfere with the production network. vCloud Director can create segregated Layer 2 networks to maintain isolation while using pools, so as not to burn up VLANs. This virtual machine is being requested for test and development teams, which is a good fit for vCloud Director.

Use Case #2

ACME Inc. wants to automate more of its processes. Users currently request everything through IT via an email ticketing system. The requests can range from fixing Outlook, to provisioning new applications on VMs, to facility maintenance. The infrastructure on the backend is completely segregated, and every department is billed for every request that comes into the ticketing system. For legal reasons, the security team is very stringent about making sure there isn't any information sharing between departments.

Is vCloud Director a good fit in this case? ACME has a system in place that creates tickets for requests beyond VM provisioning. There is also a chargeback system in place, but it's generic and doesn't take into account actual CPU, memory, storage, and network utilization. ACME has a critical need for segregated networks. vCloud Director could be a very good fit here. The vCloud Director portal wouldn't be used, but the API combined with an orchestration engine can substitute. When a new VM request is submitted through the email ticketing system, an orchestration engine can take over to complete approval emails and begin the provisioning of the vApps to vCloud Director. Because vCloud Director functionality includes segregated multitenant networks, it's much easier to satisfy requests in a shorter period of time. vCloud Director can use network pools to quickly create segregated Layer 2 networks without any interaction from the network team or security team. In addition, the chargeback process can be more granular based on certain VM types and utilization instead of a fixed cost per VM.

Use Case #3

ACME Inc. wants to enable its development teams to provision their own VMs but not have access to vCenter. The requirements state that there should be a portal with a catalog containing the VMs available to a team. After the team chooses the VM they wish to provision from the portal, the VM should be customized, added to the domain, and given an IP address on a specific VLAN in the corporate network so it can be easily accessed to test against production systems.

Is vCloud Director a good fit? In this case, vCloud Director wouldn't be a necessary component. Instead, as an architect you should focus on a series of orchestrated events through a

custom-built portal. This custom-built portal can have a series of drop-down and text boxes for the user to specify the OS, application, computer name, and Active Directory forest for customizations. There isn't any stringent access control or segregation policy for the development teams, and the VM needs to have corporate network access.

As we dive further into vCloud networking, you'll see that external networks will satisfy this request without the need for vCloud's segregated Layer 2 networks. If the customer still wanted vCloud Director, it could be a component in this stack, but it wouldn't be necessary. The only thing vCloud Director can offer is a portal, a simple catalog, and workflows of creating VMs already prepackaged.

Use Case #4

ACME Inc. is a service provider that has built a successful vSphere hosting environment. Many of the tasks done today are scripted and automated to streamline the efforts of getting new customers online. ACME is continuing to expand within its datacenter and plans to add an additional datacenter 100 yards away to meet its growing needs. It needs a solution that can scale to meet future business-development needs.

Is vCloud Director a good fit in this situation? ACME has a good system, but it needs to be able to scale, and vCloud Director has that ability. You can assume that ACME has varying degrees of hardware available to its customers and charges based on the service-level agreement (SLA). Because vCloud Director can consume multiple vCenter Servers, the portal and orchestration engines will enable end users to choose the SLA that meets their needs more quickly instead of relying on homegrown logic. As ACME continues to grow, its range of VLANs and IP address space will diminish. If ACME's current solution is to dedicate 1 VLAN per customer, then its plan to grow beyond 4,000 customers in a single location is limited. In most cases with service providers, customers have 1 to 5 VLANs dedicated. vCloud Director can play an important role by creating segregated Layer 2 networks and making VLANs and IP address space go further with network pools.

vCloud Director was built with the service provider in mind—so much so that it requires a change in thinking. For an enterprise customer, the adoption of vCloud Director means IT becomes the service provider for their organization. It's hard for IT to own everything in vCloud Director, but it depends on the architecture. In some instances, the end user becomes responsible for many aspects of the VM, such as patching and policies. vCloud Director can contain mission-critical production VMs, but making sure they adhere to correct policies for continual maintenance is a different process.

Components of the vCloud Management Stack

You need to understand the components of vCloud Director so you can design for logical and physical management. To get vCloud Director up and running, the following minimum components are required:

- 1 vCloud Director cell (a *cell* is an instance of the software in a single server) installed on Red Hat Enterprise Linux (as of vCloud 5.1, the vCD Virtual Appliance isn't intended for production use)
- 1 vCenter Server (Windows or the Virtual Appliance can be used with 5.1)
- 1 DRS-enabled vSphere cluster

- 1 vCNS Manager server (formally known as vShield Manager)
- 1 SQL Server (contains the database for vCloud Director and vCenter)

vCloud Director may not satisfy all the requirements for a cloud environment, so other supplemental products are available to round out the portfolio. Adding any of the following products can potentially make up a cloud offering based on requirements:

- vCenter Server management components
- vCenter Server for vCloud resources
- Database servers, SQL/Oracle (1 required, 2 optional)
- Multiple vCloud Director cells (the number of nodes depends on the size of the vCloud environment and the level of redundancy)
- VMware vCenter Chargeback server (additional nodes can be added for data collectors)
- vCenter Orchestrator server (optional if other workflows need to be initiated)
- RabbitMQ server (Advanced Message Queuing Protocol [AMQP] based messaging; optional)
- vCenter Operations servers (1 database and 1 UI; optional components for monitoring and capacity planning)
- vCloud Automation Center server (originally DynamicOps)
- vCloud Connecter server
- vCloud Connector node
- vCloud Request Manager
- vFabric Application Director
- Load balancer (for incoming connections to vCloud nodes)

In this chapter, we'll focus solely on vCloud Director's required components and not on the entire ecosystem.

Figure 12.1 is a representation of the communication between components in a vCloud Director configuration. vSphere 5.1 added single sign-on (SSO) capabilities and can be used against vCenter and vCloud Director. SSO can only be used for cloud administrators and not for organizations within vCloud. In this diagram, vCenter has a DRS cluster of three hosts; vCenter has configured storage profiles for the datastores, and the DRS cluster can access all datastores; vCNS Manager has deployed edge devices to the cluster; and vCloud Director has a line of communication to the vCenter Server, vCNS Manager, and vSphere Hosts.

The vCloud Director software works as a scale-out application. You can install vCloud Director on multiple servers, and they will all handle incoming connections from a load balancer to satisfy end-user requests. As a cloud continues to grow, and so do end-user requests, additional cells can be added to satisfy those requests. Adding cells increases the resiliency of the application as well as redundancy. Every cell is mapped to the same database to keep changes consistent across the cloud. Best practice requires two cells at minimum for every production instance of vCloud Director. Having two or more cells provides cell redundancy,

allows for planned upgrades and maintenance of the cells, and requires a shared NFS datastore for storing SSL certificates and the `response.properties` file for adding new cells.

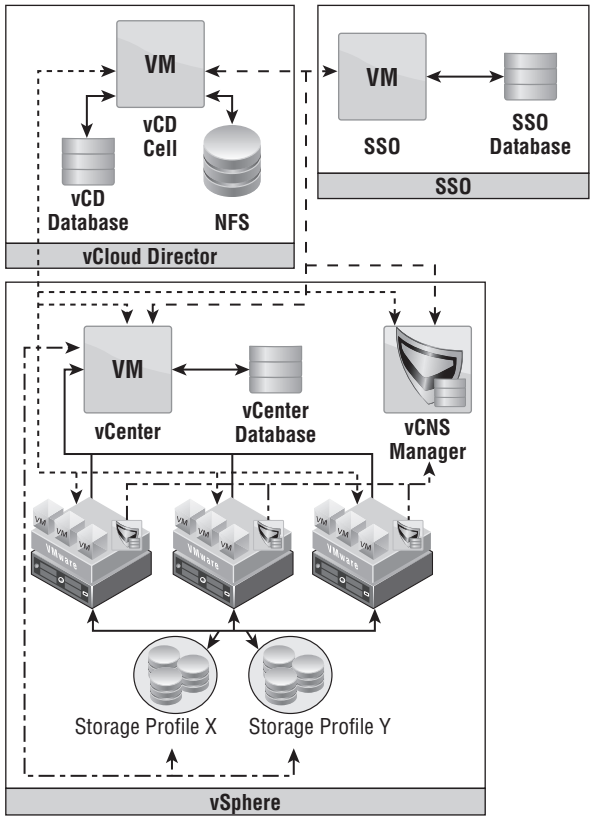


FIGURE 12.1
There are many dependencies in the vCloud Director stack.

Every vCenter Server instance is paired with a vCNS Manager server. These two pieces work in a 1:1 fashion and are presented to vCloud Director as a pair when you add a new vCenter to vCloud Director.

Figure 12.2 shows the architecture of a multinode environment where two vSphere farms are added to vCloud Director.

vCloud Cell and NFS Design Considerations

Cell servers make fine candidates for VMs, but it's worth pointing out that cell servers can be physical as well—a design decision that you may need to make if a management or infrastructure cluster doesn't exist. Choosing a physical server may also be attractive for repurposed P2V hardware.

Depending on the vCloud requirements and vSphere architecture, vCloud Director cells may be fitted with various forms of hardware. Every vCloud Director cell should be provisioned with a minimum of two virtual NICs. One NIC is required for vCloud communication tasks,

and the other NIC is bound to vCloud console connections. The vCloud console connection is pretty straightforward. It brokers the communication of the vCloud cell and the VM console to the end user. vCloud communication tasks are a bit more complex and include common server communication like DNS, vCenter Servers for API calls, the NFS server share for transfers, and the vCloud Director user interface.

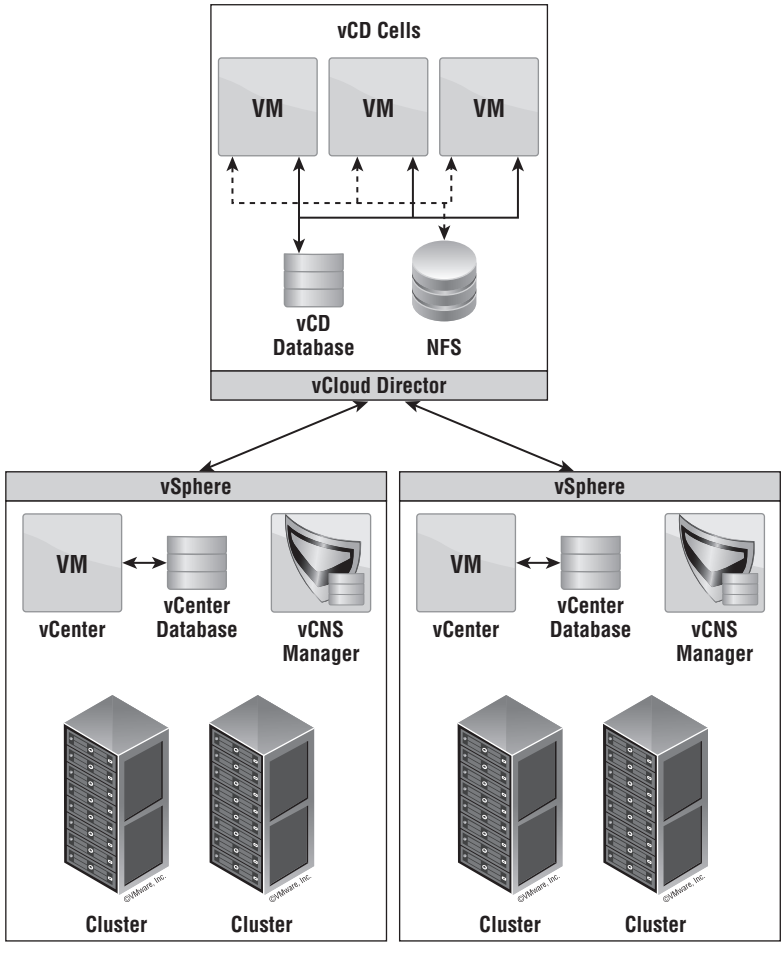


FIGURE 12.2
vCloud Director relies on a 1:1 mapping of vCenter Server to a vCNS Manager server.

If you plan to make the vCloud Director portal publicly accessible from the Internet, you may want to add an additional vNIC. Two vNICs are responsible for the vCloud portal and remote console connections from the Internet, and other vNICs are responsible for communication to internal systems and NFS shares. Other customizations on the interfaces are required, such as static routing to satisfy communication.

Every production instance of vCloud Director should include a NFS share that is greater than or equal to 200 GB. This NFS share is mapped to the vCloud cell's transfer directory and satisfies the transfer of vApps between cloud cells, transfers between vCenter Servers, and uploading and downloading vApps into catalogs. NFS servers may or may not be accessed

via Layer 3. Therefore, an additional vNIC mapped to the NFS VLAN may be necessary to satisfy Layer 2 communication.

The NFS share design is dependent on the architecture of your cloud. Deploying vApps between vCloud cells and vCenter Servers into different Provider virtual datacenters (vDCs) relies on the NFS share. There can be three or more copy processes that need to take place for a vApp to finally find its home. The input/output operation (IOP) capabilities of the NFS share and the IP connection between vCloud components (1 GB versus 10 GB) play a role in how fast vApps are copied between locations. If your cloud doesn't consist of multiple vCenter Servers, then the NFS server isn't used because native vSphere cloning is used to speed up the process.

The NFS share can be hosted in a multitude of places. The preferred form is to have the NFS share created on the storage array that is also hosting the vCloud Director cells. This method puts the copy traffic very close to the source. Many storage arrays are not equipped with file capabilities and must rely on block storage. In this scenario, we can create a VM on a VMFS datastore to serve as the NFS share using a product like OpenFiler or any other standard operating system. Standard vSphere HA protection for this VM is suitable because traffic is only occurring during copy processes. The final option is to create an NFS share on a vCloud cell. This is not a suggested approach because the share may be inaccessible during vCloud upgrades on the hosted cell.

Management vs. Consumable Resources

When beginning to architect a vCloud Director design, you should always first identify two logical constructs. First is the *infrastructure management cluster* and second are the *vCloud consumable resources*:

- The infrastructure management cluster is a vSphere cluster that contains VMs responsible for the management construct of vCloud Director. This includes the core set of vCloud components such as vCloud Director, vCenter Server, vCenter Chargeback, vCenter Orchestrator, and vCNS Manager.
- vCloud consumable resources are groups of vSphere clusters managed by vCenter Server(s) that are designated as vCloud consumable resources where provisioned vApps will live. This is typically where SLAs are tied to the infrastructure, such as Gold, Silver, or Bronze.

Identifying these two key constructs allows you to scale vCloud Director in parallel ways. As your cloud continues to expand, so must your management footprint. Separating these two constructs allows the following:

Delineation of Responsibility When vCloud Director is adopted, a vSphere administrator is responsible for the vSphere infrastructure (infrastructure management cluster), whereas a cloud administrator is responsible for the vCloud pieces (vCloud consumable resources). This keeps the change-management process in place for the ESXi teams to treat the management cluster differently than the consumable resources. We'll discuss multiple vCenter Servers in the next section and why the separation is important.

Eliminating False Positives in vCloud If you were to place VMs from the infrastructure management cluster into vCloud consumable resources, vCloud Director wouldn't show an accurate representation of the resources available because of resource-pool calculations and the actual workloads of the infrastructure management VMs. This same rule applies for any workload running outside of vCloud Director on vCloud consumable resources.

Higher Availability Creating separate clusters ensures that infrastructure management VMs aren't interrupted from the variable-workload characteristics of the cloud (including possible denial-of-service attacks); thus, resource contention isn't a factor. A key point in any design is to never have something managing itself.

Scalability As vCloud resources are added to satisfy additional workloads, the management cluster may also need to be upgraded to satisfy additional requests. This is a building-block approach where one manages the other and the growth of one directly impacts the other's design.

Disaster Recovery This architecture simplifies the constructs of facilitating disaster-recovery efforts by enabling Site Recovery Manager (SRM) to work on a supported workload. As of SRM 5.1, it can only recover the infrastructure management cluster. Disaster recovery options will be discussed later.

Figure 12.3 shows how a management cluster is responsible for the management VMs, whereas vCloud consumable resources are used to create Provider vDCs.

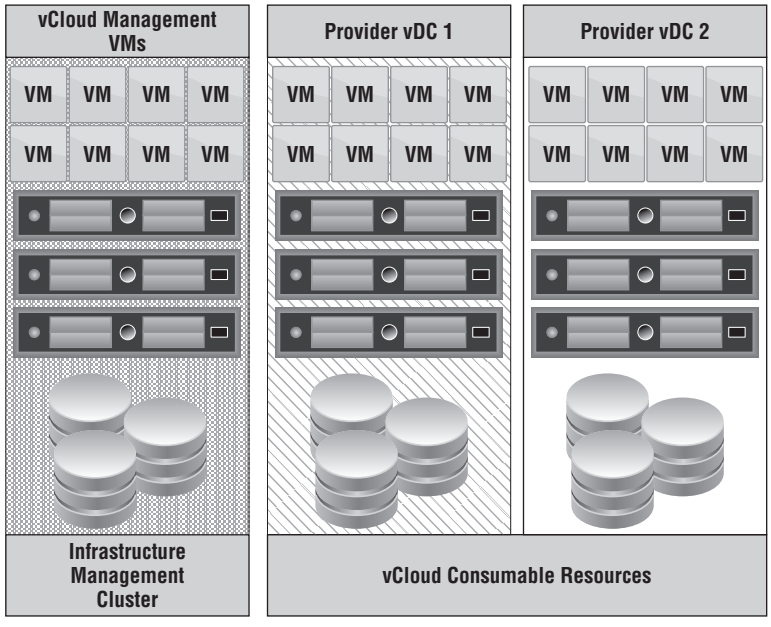


FIGURE 12.3 As your cloud continues to grow, so must your management footprint.

Database Concepts

A vCloud Director cell is considered a scale-out application. Even though there are multiple vCloud Director cells, only a single database is created and shared among all cells. The design concept for the vCloud Director database is dictated by physical location and security access. The vCloud Director database won't incur high input/output (I/O) even during peak usage; many standard database servers can handle the additional database load from vCloud Director. vCloud Director only needs to write to its database for changes to the UI, to map resources to vCenter, and to handle other small pieces. It's mainly responsible for sending API calls to

the vCenter Server and the vCNS Manager to deploy objects that affect their database I/O and not vCloud Director's. In addition, during the configuration of the database, parameters are set to make sure the database doesn't grow out of hand.

There are a few locations where you can place the vCloud Director database:

- You can use a highly resilient SQL cluster with sufficient bandwidth (>= 1 GB) to the vCloud cells.
- Depending on the size of the cloud infrastructure, the vCloud database can live on the same database server that is hosting other databases such as vCenter, vCenter Orchestrator (vCO), and SRM. This scenario keeps new database servers from being provisioned; additional databases can be added to the regularly scheduled backups or replication without much administrative overhead. This VM can live in the same management infrastructure as the cloud cells or have >= 1 GbE communication.
- A dedicated database server can be provisioned for vCloud Director. The vCloud Director database must use local authentication (LDAP isn't supported); therefore, it may be in the best interest of the security team to not compromise a primary database server with local logins. This creates a separation of management and allows the cloud administrator to be responsible for database activity.
- The SQL Server resources in all these scenarios should be identified and considered according to the input/output profile or workload that will be running on them.

vCenter Design

Two vCenter Servers are mentioned in the cloud portfolio model. This design concept correlates to the infrastructure management cluster and vCloud consumable resources discussion.

A vCloud Director recommended practice is to have at least two vCenter Servers. The first vCenter Server is responsible for hosting vSphere and/or vCloud infrastructure components related to the infrastructure management cluster. This vCenter is called the *vCloud Infrastructure vCenter*. The second vCenter Server (and subsequent vCenter Servers) is called the *vCloud Resource vCenter*(s) and is responsible for hosting vCloud consumable resources. Why are two vCenter Servers necessary?

Separation of Management Domains As mentioned earlier, creating a clear delineation of responsibility is critical. The vCloud Infrastructure Management VMs live on a vSphere cluster and are treated as production VMs with default vSphere administrative privileges. The vCloud Resource vCenter is responsible for managing consumable resources in vCloud Director. The vCloud Resource vCenter is consumed by vCloud Director; therefore, the management of this vCenter is treated differently.

vCenter Becomes Abstracted In typical virtualized datacenters, ESXi abstracts the hardware layer, and vCenter becomes the central management point. vCloud Director abstracts the resources that belong to vCenter and presents them to vCloud as Provider vDCs. Therefore, the vCenter responsible for vCloud consumable resources shouldn't be treated as a normal vCenter instance, and administration should be performed at the vCloud UI level. Access to the vCloud Resource vCenter is only necessary during initial configuration, software updates to vSphere, and troubleshooting.

Saving vSphere Administrators from Making Mistakes The vCloud Resource vCenter is responsible for vCloud consumable resources and should be considered owned by vCloud Director. As operations happen in vCloud Director, many objects are created, such as folders,

resource pools, port groups, and appliances. Everything created by vCloud Director has a set of unique identifiers. For instance, if a vSphere administrator has access to a distributed virtual switch (DVS) and notices what looks like a random port group ending with a long set of characters, they will be tempted to delete it. If objects are deleted directly from the vCloud Resource vCenter without vCloud interaction, vCloud Director will attempt to re-create them, but if it can't, then the database may get out of sync.

Relieving Stress on vCenter When tenants of the cloud issue a multitude of requests, a single vCenter Server may be rendered unusable by the flow of API calls. By separating the workload between two vCenter Servers, you won't impact the vCloud Infrastructure vCenter Server responsible for management functions.

Figure 12.4 depicts a scenario where a large organization already has a management cluster with servers such as Active Directory, DNS, SQL, and an existing vCenter Server to manage current operations. In this case, the existing vCenter Server becomes the vCloud Infrastructure vCenter. If the physical resources are available, you want to create a new cluster called a *vCloud management cluster*. This management cluster houses the vCloud Resource vCenter, SQL, vCNS Manager, vCD cells, and potentially more VMs. We're choosing to add a second SQL server because the vCloud Resource vCenter, vCloud Resource vCenter Update Manager, and vCloud Director applications all need access to a database where transactions don't have to traverse the network a far distance to limit latency and unexpected downtime. As shown in Figure 12.4, the vCloud Infrastructure vCenter owns the management cluster and the vCloud management cluster. The vCloud Resource vCenter owns the vCloud Provider vDC Resource clusters.

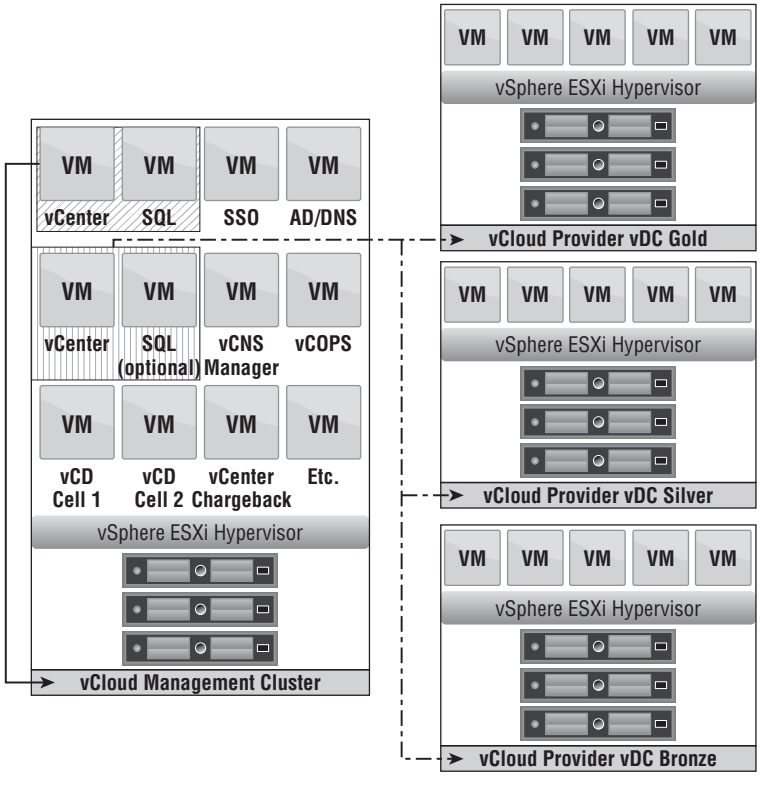


FIGURE 12.4
An existing management cluster can be used for vCenter design.

Figure 12.5 shows a singular global management infrastructure. Instead of having a dedicated vCloud management cluster (sometimes as a cost-saving measure), it's integrated into a global management cluster. This is a pooling of all the VMs required for the management operations of the infrastructure as well as the cloud. The second SQL server paired with the vCloud Resource vCenter is optional because the existing SQL server may support the new necessary databases, depending on requirements.

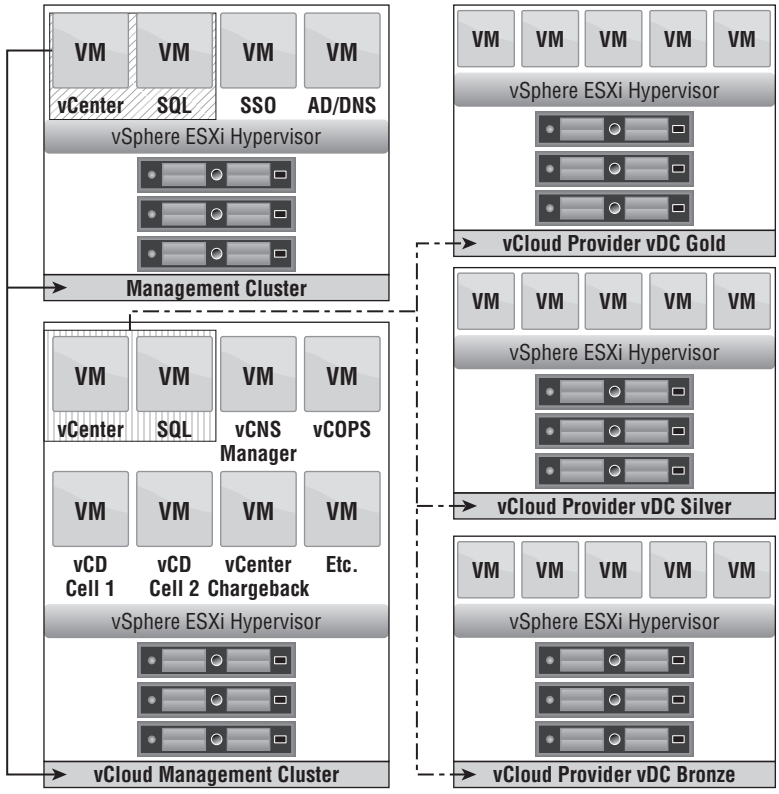


FIGURE 12.5
vCenter design can rely on a single management infrastructure.

Converged infrastructure is beginning to achieve market traction, and adoption is accelerating. In many converged infrastructure solutions, a vCenter Server is configured as part of the delivery process and can't be integrated into an existing vCenter instance. This can be seen as a constraint with organization vDC (Org vDC) design, but the design is very simple and makes the procurement and integration of converged infrastructure much easier. In this case, every vCenter delivered in a converged infrastructure pod can be defined as a vCloud Resource vCenter, as shown in Figure 12.6. The preferred method is to integrate with an existing vCenter Server to use elastic vDCs whenever possible.

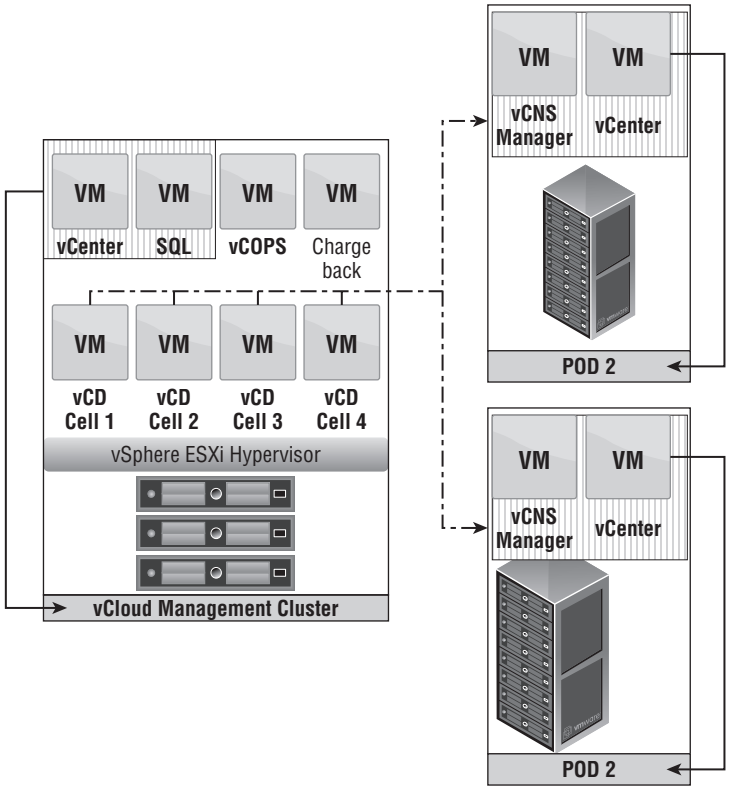


FIGURE 12.6
Adopting converged infrastructure with an integrated vCenter into vCloud Director

vCloud Management: Physical Design

The physical design is unique in every situation. As an architect, you're responsible for determining the requirements to derive assumptions and constraints. The size of your management infrastructure depends on a single question: "How big is the cloud I have to manage?" This question must be answered in concert with the vCloud maximums.

vCloud Director recommended practice suggests a management infrastructure for vCloud Director infrastructure resources. This management infrastructure is beyond a single cluster of resources. It suggests dedicated servers, networking, and storage. This is a typical design in large vSphere implementations as well. The goal of this design is to make sure an environment isn't managing itself. If there is an issue on the production systems, how can the management tools troubleshoot it if they're experiencing the same issues? This can be related to physical outages and misconfigurations in the logical components. Having a dedicated management infrastructure ensures that a problem with the production infrastructure can be accessed and troubleshot through the management infrastructure. And vice versa: if the management infrastructure experiences an outage, it doesn't affect the production infrastructure. You take the same approach with vCloud Director, but more caveats are involved. The management infrastructure is critical to the survival of vCloud Director.

In a normal vSphere environment, the loss of vCenter doesn't impact running workloads, and HA continues to function. With vCloud Director, the loss of a vCloud Resource vCenter can introduce unanticipated consequences. The communication between vCloud Director and a vCloud Resource vCenter is responsible for instantiating the provisioning of new vApps and new networks, and brokering access to the remote console of VMs. The vCloud Resource vCenter(s) become critical components of the functioning cloud. In the case of the vCloud management infrastructure, you should adhere to recommended practices for vSphere design.

The market is seeing a growing adoption of vCloud Director. Some cases are not for production workloads but instead for specific use cases, such as development and test environments or proof of concepts. Therefore, cost may be a limiting factor. vCloud Director infrastructure management VMs can be aggregated into an existing vSphere management infrastructure farm where Active Directory, DNS, and SQL already exist. The assumption is that this existing cluster has ample capacity to satisfy vCloud Director cells; vCNS managers; additional vCenter Servers; and any ecosystem VMs, such as vCenter Operations Manager (vCOPs) or vCenter Chargeback, as shown in Figure 12.5.

A second option is to create a management cluster, alongside Provider vDC resources. The vCloud infrastructure management VMs use the same networking and storage infrastructure as the Provider vDC clusters, but you have separation at the cluster level. This option works well for most cases, assuming the storage and networking infrastructures are resilient. If problems arise in the storage or networking infrastructure, it will directly impact both the vCloud management cluster and Provider vDCs.

The physical design for the management infrastructure looks like a typical vSphere infrastructure based on capacity and the size of the cloud it manages. A VMware best practice is a minimum of three servers to satisfy HA, DRS, and N+1 capacities. If this design is for a proof of concept (POC), cost plays a larger role. POCs, test and development, and other use cases might dictate that more consolidation is necessary. With these scenarios, you may opt for a two-server cluster until it's time to move into production. As the use case progresses toward production, you can add additional servers for resiliency and scale. The number of servers depends on the number of VMs you plan to host, such as multiple vCenter Servers (perhaps adding vCenter Heartbeat as well for a total of three or four vCenter Servers), vCenter Chargeback that can expand to multiple collectors, vCNS Managers, multiple vCloud Director cells, a SQL server or two (perhaps more if you want to implement clustering services), redundant AD/DNS, a load balancer for the cells, and vCenter Orchestrator. The number and types of servers depend on requirements.

Storage design of the management cluster also depends on the number and types of VMs living in this cluster. This cluster is the center of your entire cloud, so you need a production-ready storage solution. The amount of I/O driven by these VMs varies. The core vCloud Director products (vCloud Director, vCenter, SQL, and vCNS Manager) don't generate a heavy I/O load. As an architect, defining the ecosystem products that generate high I/O load is a constraint. For example, vCOPs Enterprise can require up to 6,000 input/output operations per second (IOPS) to monitor more than 6,000 VMs.

Network design of the management cluster depends on the number of connections the cluster is servicing. The management cluster is responsible for API calls to vCenter and ESXi hosts to issue the creation, deletion, and modification of objects in vSphere. The networking connections for this can be satisfied with both 1 GbE and 10 GbE connections. As more consumers of your cloud access the vCloud Director portal, many console sessions can occur simultaneously. Network bandwidth monitoring will be an administrative effort as it relates to the NICs

dedicated to standard VM networking. As consumers of the cloud increase, additional 1 GbE NICs may be needed, or a transition to 10 GbE may be necessary.

The Physical Side of Provider Virtual Datacenters

In vCloud Director, physical resources must be provided for organizations to consume, as shown in Figure 12.7. These resources are considered Provider vDCs. Everyone has a different Provider vDC strategy. Provider vDCs in vCloud Director can consist of any type of vSphere infrastructure. The key point to understanding what a Provider vDC can accomplish is to tie it to an SLA. The SLA defined by a Provider vDC depends on many different options. Most people are familiar with Gold, Silver, and Bronze approaches, and we'll use them going forward for our examples.

To simplify, a Provider vDC can be a cluster or clusters of servers with associated datastores mapped to storage profiles. A standard best practice is to associate a cluster of servers and datastores as a tier of service. Stay away from using resource pools as the root of a Provider vDC. It's also important to note that a good vSphere design is crucial to a good vCloud design.

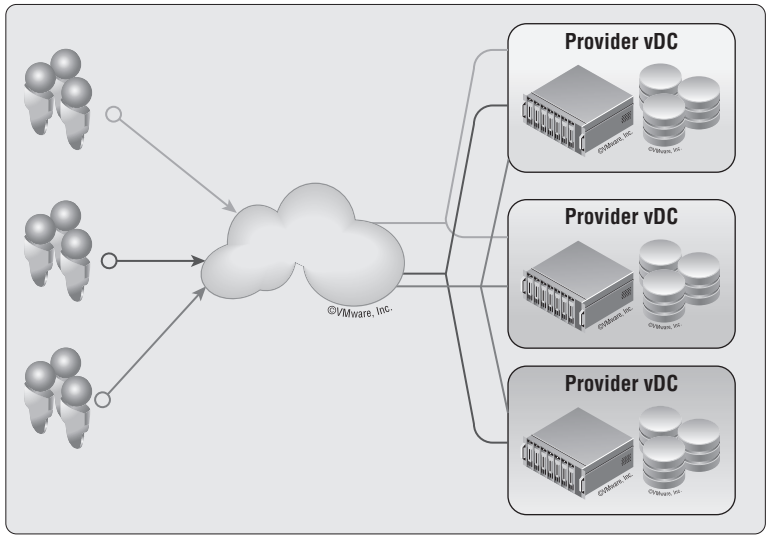


FIGURE 12.7 Provider vDCs are the resources for deployment of vApps in vCloud Director.

The simplest approach to tie an SLA to a Provider vDC is based on the types of disks shown in Figure 12.8. This should be relatively easy because everyone understands the differences in performance between EFD/SSD, Fibre Channel/SAS, and SATA drives. Assigning an appropriate SLA is simple because you know the Gold service level is aligned with EFD/SSD, Silver with FC/SAS, and Bronze with SATA, based on performance characteristics. The disadvantage of this method is the inability to appropriately estimate the number of each type you'll need and the wasted costs. If you fail to determine your tenants' needs, then you'll over or under purchase for a particular tier. Perhaps you wasted a capital expenditure on Gold EFD/SSD drives, and you don't have single tenant that wants to pay for that sort of premium. The wasted costs are risky.

A second approach also relates to disks, but it builds on multiple tiers by using multiple types of RAID groups as shown in Figure 12.9. This is a tough scenario to standardize because there are lots of RAID offerings, and you could once again waste money on unused disks.

Different applications may warrant RAID 5 versus RAID 6 versus RAID 1+0 for performance characteristics. Now you have to decide where to spend your money on types of disks. An example would be setting a Gold tier as SAS/FC Raid 1+0, Silver as SAS/FC in RAID 5, Bronze Plus as SATA in RAID 5, and Bronze as SATA in RAID 6.

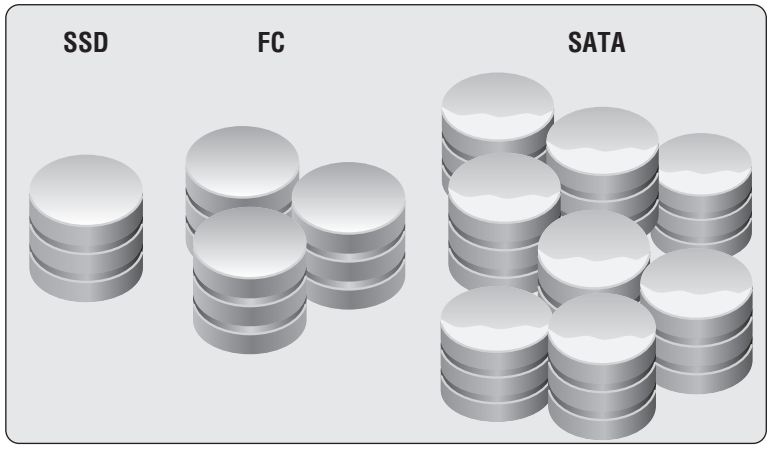


FIGURE 12.8
Storage capability is an easy differentiator for service levels.

Not only do types of media have varying performance characteristics, but they also offer differing levels of redundancy. We opted not to include an EFD/SSD tier to keep everything simple. We could just as easily add EFD/SSDs and more RAID offerings on all tiers of media to make a multitude of offerings. The goal is to keep costs in mind and find that sweet spot for return on investment.

Going with RAID types as a differentiating factor might not be the most efficient because the applications hosted in vCloud Director probably aren't critical enough to warrant this lengthy thought process. Sticking with one standard RAID type and moving forward may be a better plan to make sure you aren't over- or under-allocating resources.

Another piece of information to keep in mind is that VMs inside of vCloud Director can only belong to a single datastore. In vSphere, we could take a high I/O VM and place the operating system VMDK on a RAID-5 datastore, and the VMDK needing more IOPS can be placed on a RAID 1+0 datastore. As of vCloud Director 5.1, this is not possible and all VMDKs belonging to a VM must reside on a single datastore.

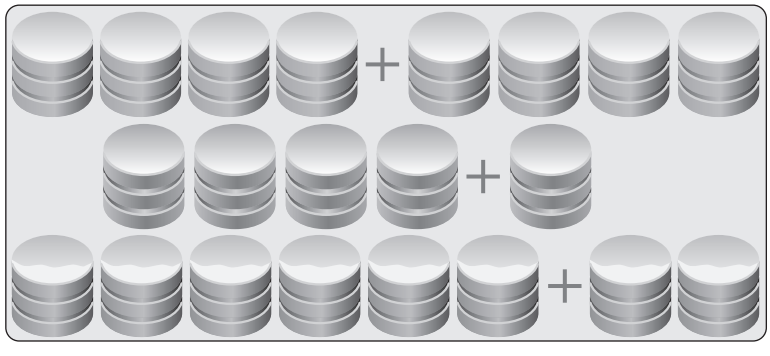


FIGURE 12.9
RAID type can be an acceptable form of differentiating service levels but isn't recommended.

The third approach to tie an SLA to a Provider vDC still uses types of media, but it focuses on storage technology. Many storage vendors have a feature that allows the dynamic movement of blocks to different media depending on accessibility, as shown in Figure 12.10. Some storage vendors that offer this type of technology include EMC, Dell Compellent, HDS, HP 3PAR, IBM, NetApp, and many others. In this example, we'll use EMC's fully automated storage tiering (FAST) technology.

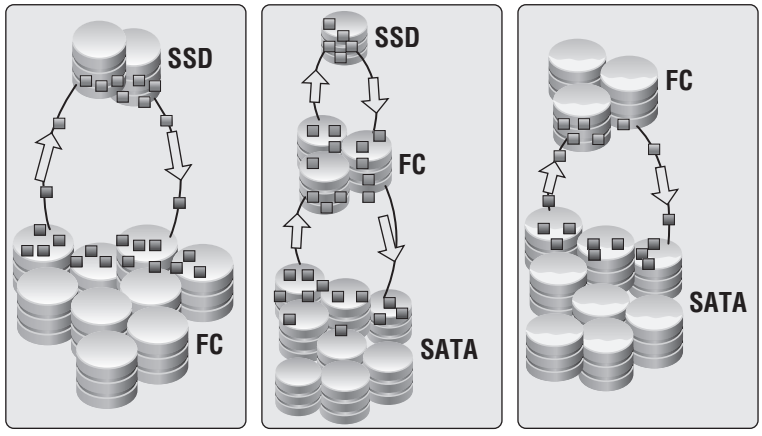


FIGURE 12.10 Dynamic movement of blocks can dictate levels of performance.

FAST can determine your Provider vDC strategy based on the types of disks because you can offer this technology based on single datastores. FAST allows multiple types of disks to be aggregated into a single LUN/datastore while an algorithm determines where data will be stored. You can put SSD, FC, and SATA into a single pool, and datastores can then be carved up. The algorithm determines when hot blocks need to be moved to a higher tier of disk, and other unused blocks can be moved to a lower tier. If those lower-tier blocks start seeing action, then they can potentially move up a tier or two based on how often the algorithm runs.

FAST lets cloud administrators offer multiple kinds of disk-based SLA offerings. For example:

- Gold = 30% EFD and 70% FC, giving Gold tenants more room to burst into EFD while not paying a premium for EFD drives in the short term.
- Silver = 5% EFD, 70% FC, and 25% SATA, which gives tenants an offering that allows a little burst room but warrants good performance when needed.
- Bronze Plus = 25% FC and 75% SATA, allowing tenants to burst into FC-type performance while still keeping costs minimal.
- Bronze = 100% SATA without FAST technology, for a predictable performance tier.

This strategy gives the cloud provider greater options for the level of service they can offer tenants while also saving money on expensive EFD drives. The only downside to a FAST offering is that you can't guarantee tenants a predictable I/O pattern or particular level of performance. vCloud Director sees datastores equally in a Provider vDC, and if multiple tenants use the same FAST datastore, they will compete for those higher-grade tiers based on their workload.

Of course, we can stretch this same type of thinking to servers, as shown in Figure 12.11. Perhaps you still had a single SAN, but you were refreshing or expanding your compute cluster. You can use old servers to create vSphere clusters that run older dual- or quad-core processors and that are assigned a Silver or Bronze SLA, and give a Gold SLA to newer hex-core servers that have greater clock speeds and RAM densities. Both clusters still rely on the same backend storage array, but the differentiating factor is the processing power given to the VMs. Typical vSphere design comes into play here as well. Don't cross-pollinate datastores between clusters. vSphere hosts have a maximum datastore connection threshold, and cross-pollinating can lead to reaching that maximum.

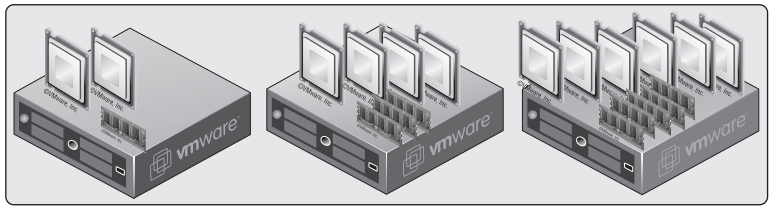


FIGURE 12.11
Cluster together hosts with similar CPU speeds, and create offerings based on processing power.

Stay away from tying an SLA to FC/block versus NFS/file. Both solutions are great, and they both achieve what you need. Instead, think about how you would tie SLAs to connections on 1 GbE versus 10 GbE NFS and 4 GbE versus 8 GbE FC. If there is a mixed environment, you could have 1 GbE IP = Bronze, 4 GbE FC = Silver, and 8 GbE FC = Gold or 10 GbE NFS = Gold. The battle of block- versus file-based storage will never end, so stay neutral about how you tie an SLA to a type of network medium.

In addition to speed, take reliability into account. What type of switch or fabric switch is in the middle? Are the fabric switches redundant? If the loss of a switch occurs, what is the impact to the throughput and availability?

Now that we have looked at a few types of Provider vDC approaches, let's start thinking a bit bigger. Many companies are adopting converged infrastructure or pod types of computing. Basing your Provider vDC on disk is good for use in a single pod because it can easily be managed. The great thing about vCloud Director is that it gives the cloud provider the freedom and control to adopt multiple infrastructures that can determine Provider vDC offerings.

Many companies have older VMware farms, or somewhat new VMware farms, but are looking either for a refresh or to expand. You can now use vCloud Director Provider vDCs in a pod approach instead of thinking in terms of granular disk. For instance, suppose you have a collection of Dell R610 servers connected to two Cisco 3650s via iSCSI 1 GbE to a Hitachi array. You also have a few clusters of HP DL380 G7 servers connected to a single Cisco 4507R where storage is supplied from a NetApp FAS6080 via 10 GbE NFS. You've also purchased a new Virtual Computing Environment (VCE) Vblock 300HX of converged infrastructure. For simplicity's sake, let's say each pod has a single cluster of eight hosts and datastores of only FC/SAS storage. From this, you can derive a few differentiating factors. First, the servers keep getting newer, and you can tie appropriate SLAs to them. In addition, the connection medium is capable of higher throughput and is also more redundant. Pod 1 has 1 GbE connections on 2x Cisco 3560s of which only one is used for the uplink to overcome Spanning-Tree Protocol (STP). Pod 2 has much better throughput using a 10 GbE connection but falls short of true redundancy because the 4507-R is a single-chassis solution, even though it has two supervisor engines. Pod 3 uses

8 GbE FC and 10 GbE NFS storage for maximum throughput and is fully redundant by using Virtual Port Channels (vPCs) between Nexus 5548UP Switches and a redundant FC network. In all of these vSphere infrastructures, the backend storage stays the same. Sure, the storage processors are fresher on the newer arrays, but it's still the same 300 GbE FC disks spinning in RAID 5 delivering the same number of IOPS. This is an example of thinking in a pod-based approach because all of this equipment can still be used by a cloud provider; the provider can reuse older hardware as a service tier to continue to realize profits.

As we start thinking further down the line about newer capabilities of vCloud Director and integrations into more products, we can imagine more capable Provider vDC scenarios. Today, we can replicate datastores across the WAN to create a Provider vDC offering with built-in DR, as shown in Figure 12.12. For this scenario, you can create a few datastores that are characterized as replicated and have a higher cost. This is where a new service offering can be created, such as Gold Plus or Silver Plus.

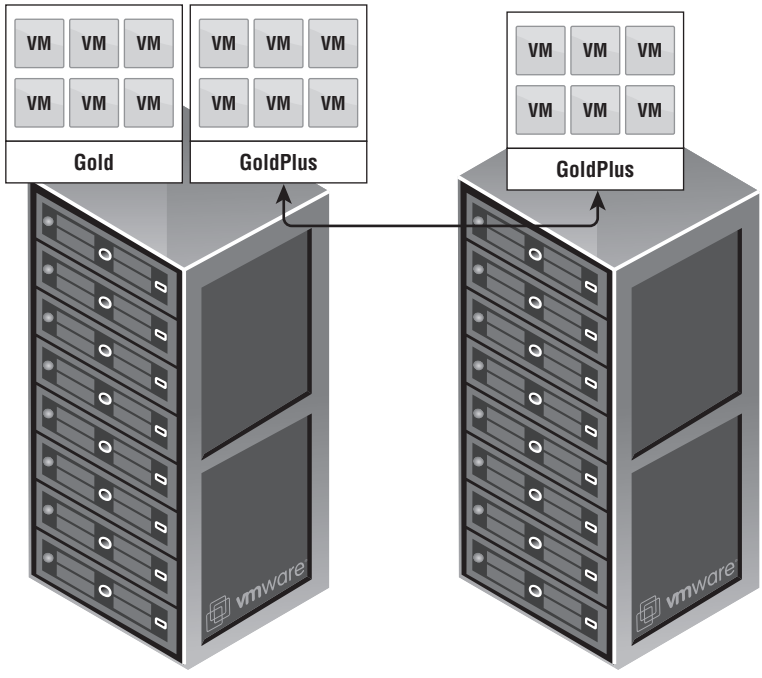


FIGURE 12.12
Use replication as a value added to types of offerings.

Many companies are also looking at implementing stretched clusters for mission-critical applications that require little to no downtime. As vCloud becomes an increasingly trusted platform, more mission-critical workloads will be placed there. To satisfy the needs of these mission-critical workloads, the Provider vDC may need to be modified with technologies that can introduce these possibilities. Today, technologies like EMC VPLEX along with Cisco Nexus 7000s for Overlay Transport Virtualization (OTV) and Locator/ID Separation Protocol (LISP) can create stretched-cluster scenarios to give that level of availability; see Figure 12.13. In both of these scenarios, you must take into account the architecture of the management infrastructure to accompany a successful DR failover.

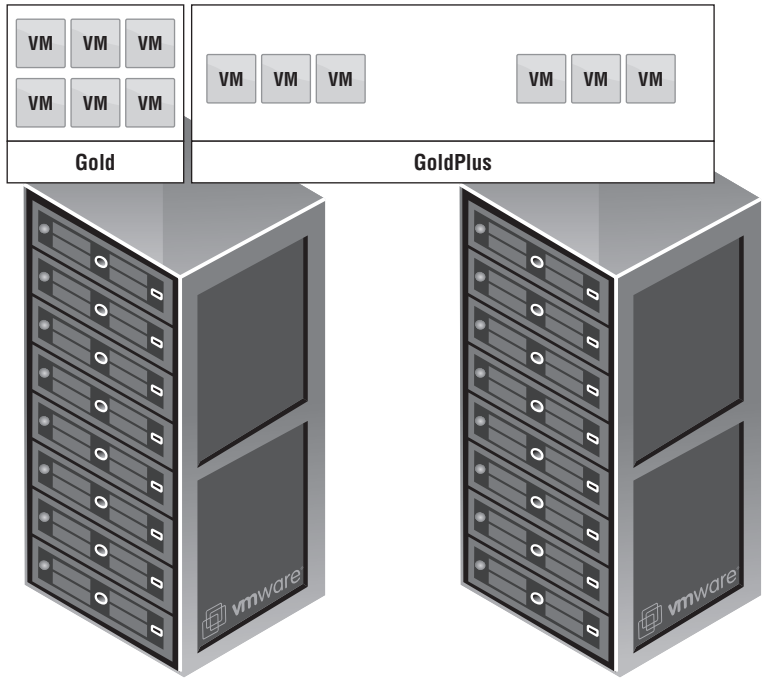


FIGURE 12.13
Stretched clusters can give an offering a lower recovery point objective (RPO) and help avoid disasters.

The Logical Side of Provider Virtual Datacenters

In previous versions of vCloud Director (1.0, 1.5, and 1.5.1), a Provider vDC was mapped to a physical cluster, and the datastores were attached to that cluster. If you wanted to create a Gold offering at the cluster level, all datastores attached to those hosts had to have similar characteristics. vCloud 5.1 brings more feature parity from vSphere: datastore clusters and storage profiles that enable much greater flexibility.

When you're designing Provider vDCs in vCloud 5.1, the server becomes less of an issue and there is a greater focus on storage. With vCloud 1.5.1 and earlier, the datastores defined in a Provider vDC were the ones you needed to use, as shown in Figure 12.14. vCloud Director didn't know you were combining SATA and FC drives in a single Provider vDC offering, so when a VM was provisioned, it could go to either SATA or FC.

When you're buying storage for a cloud infrastructure, you probably won't buy an entire storage array with just one kind of disk. Storage profiles bring a new type of architecture to vCloud Director: they let Provider vDCs be more flexible in their offerings. Traditionally, a Provider vDC could be considered Gold or Silver depending on the types of storage backing it. Storage profiles can create Provider vDCs with a mix of storage types. They allow better utilization of the server environment because clusters aren't dedicated to specific types of disk. In addition, vSphere 5.1 can now expose datastore clusters to vCloud 5.1, which lets you balance workloads across resources and enables flexibility of the pools being offered.

Figure 12.15 shows that provisioning a vApp to a particular Provider vDC dictates the type of storage depending on the storage profile assigned. In this case, the datastores in the FC datastore

cluster have been assigned the storage profile Gold, whereas the datastores in the SATA datastore cluster have been assigned the storage profile Silver.

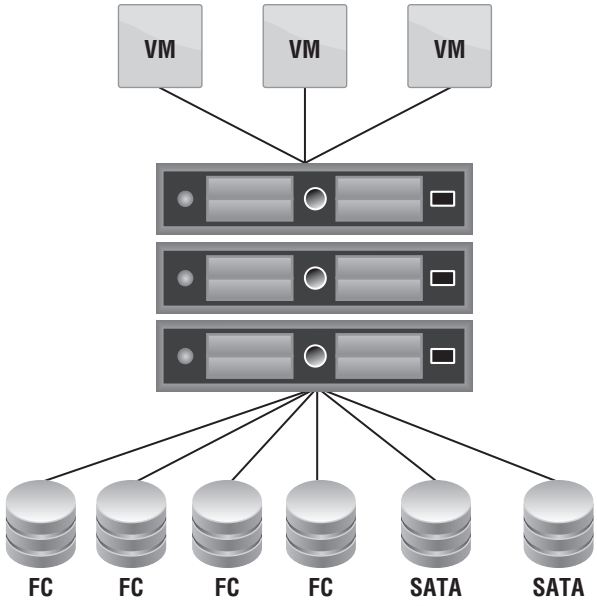


FIGURE 12.14
vCloud 1.5.1 and earlier couldn't use storage profiles; therefore, deploying vApps was unintelligent and used any datastore available to a cluster of hosts.

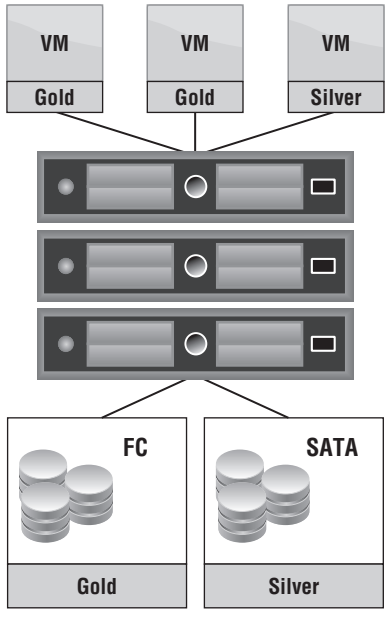
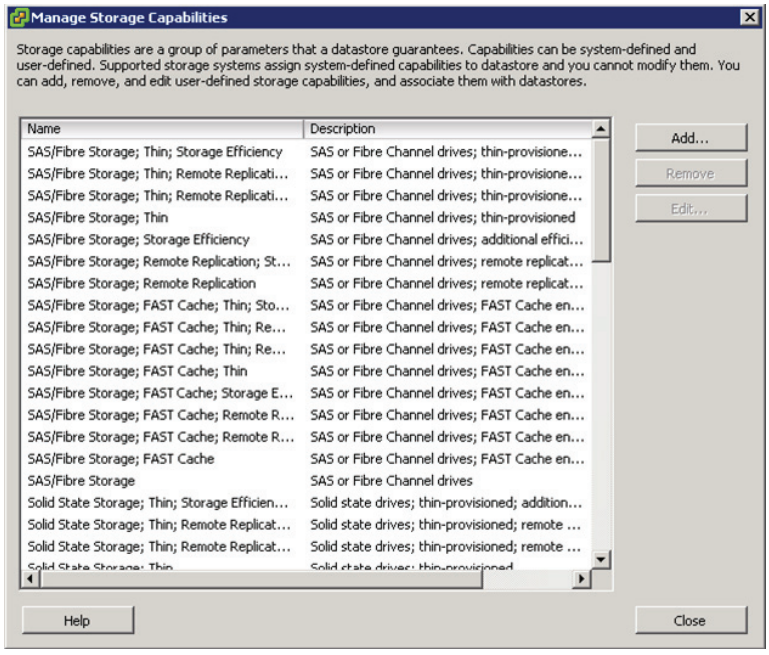


FIGURE 12.15
Choosing a storage profile type allows better utilization of the server environment.

Before you configure storage profiles, individual datastores need to be defined with capabilities. There are two ways to create datastore capabilities: use vSphere APIs for Storage Awareness (VASA), or create them manually as user-defined. When you're deciding on a storage array for the architecture, the ability to use VASA can be a time saver. Many storage manufacturers are adopting VASA to make using storage profiles much easier. Figure 12.16 shows the storage capabilities exposed to vSphere after adding an EMC VNX array as a storage provider

FIGURE 12.16
VASA integration can natively define datastore characteristics.



Notice in Figure 12.16 that many of the capabilities are almost duplicated. For instance:

- SAS/Fibre Storage
- SAS/Fibre Storage; Thin
- SAS/Fibre Storage; Thin; Remote Replication

The reasoning behind this methodology is that a datastore can only be defined with a single datastore characteristic string. If datastores need to be more granular and well defined, you can manually create your own specific user-defined profile. If you had a sample datastore that was based on SAS drives and was being thin-provisioned, you could give it a name such as

SAS/Fibre Storage; 15k RPM; RAID-5; Thin Provisioned

User-defined storage capabilities can be more granular. Giving your datastores a specific characteristic makes it much easier to define a storage profile.

As your capabilities are defined, you may or may not want to begin using datastore clusters. Datastore clusters let you group datastores with similar characteristics to achieve better storage

utilization. Datastore clustering allows vSphere to manage the initial placement of VMs into datastores based on the space available. In addition, vSphere can also be responsible for the real-time migration (via Storage vMotion or Storage DRS) of VM files based on available capacity or I/O latency thresholds. Datastore clusters are supported in vCloud 5.1 with vSphere 5.1 or later. They're recommended in most scenarios, contingent on an array vendor's best practice, because they relieve the administrator of having to monitor storage use.

There is a particular scenario in which you should disable Storage DRS for datastore clusters: when a datastore is backed by an auto-tiering technology (such as EMC FAST). The array is responsible for moving blocks of data, and a Storage vMotion event would place the VM's storage on another datastore. The placement of blocks could be unpredictable, and performance could suffer. These types of datastores should be placed in a datastore cluster for initial VM placement during provisioning, but Storage DRS capabilities should be disabled to allow the storage array to perform its duties.

Storage vMotion in vCloud Director is supported and enables the live migration of VM disk files to another datastore. This is only possible when

- The target datastore is part of the same Org vDC as the original vApp.
- All the virtual disks for a single VM are migrated to the same datastore (the VM can't have virtual disks in separate datastores).
- The vCloud API is invoked to initiate the Storage vMotion for fast-provisioned VMs to preserve the tree (performing a Storage vMotion using the vSphere Client can cause the inflation of the delta disks).

Using datastore clusters isn't required in vCloud Director but is recommended. Always consult your storage-array manufacturer on its recommended practice. The correlation of storage profiles for use in Provider vDCs is directly related to defining storage characteristics; a storage profile can be mapped to multiple datastore clusters.

You're now ready to implement storage profiles in the design. The key is mapping all three of the following storage components together: assign a datastore capability to each individual datastore, create a datastore cluster for datastores that share similar or identical characteristics, and then create a storage profile to define a class of service. This tree is depicted in Figure 12.17.

When you're defining a name or a class of service in the storage profile, it needs to be both relevant and relative to what is being offered. The name needs to be easily discernable by end users who choose to provision to this particular storage class. Some suggested names are as follows:

- Extreme Performance = SSD/EFD
- Standard = SAS/FC
- Standard with Replication = replicated SAS/FC datastores
- Capacity = SATA

Of course, the standard Gold, Silver, and Bronze may be suitable. The key thing to understand is that technology will mature over time. A Gold level of service today could be tomorrow's Bronze level. In addition, how do you differentiate your cloud against someone else's if you're competing in that space? Your Silver offering could be someone else's Gold. The

definition of your storage profile should be apparent to the end user to simplify their placement decision, as shown in Figure 12.18.

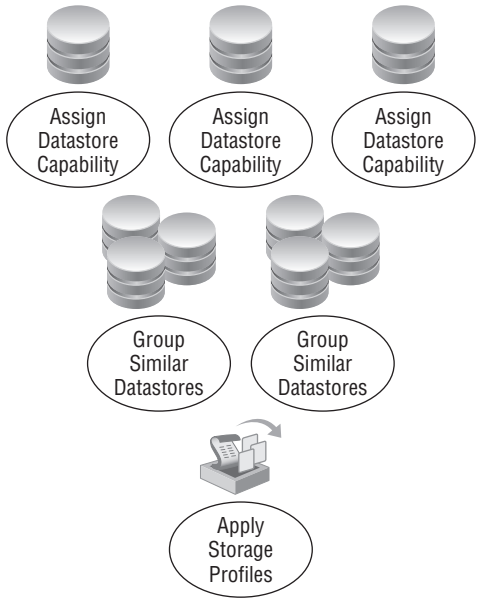


FIGURE 12.17
A storage profile depends on datastore capabilities. You can use datastore clusters to group similar datastores.

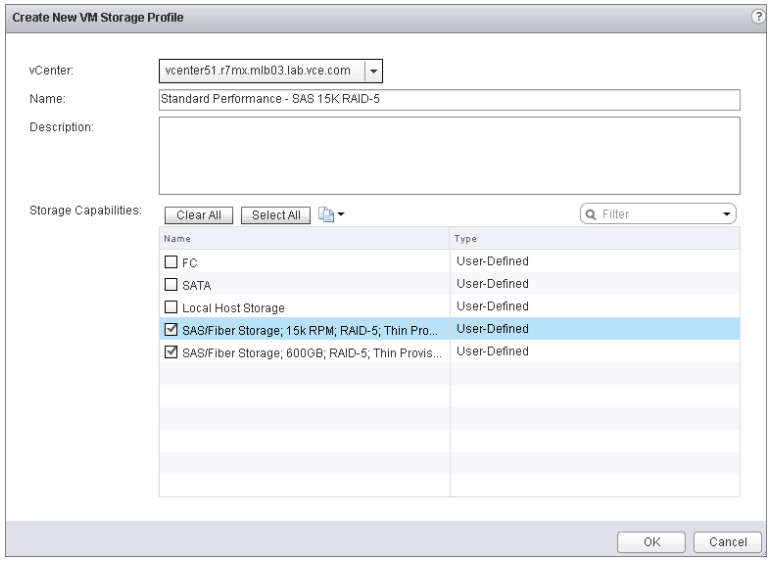


FIGURE 12.18
Create easily understood names for storage profiles.

Figure 12.19 is an example of what it looks like to provision a vApp as a tenant in vCloud Director. Each individual VM in a vApp can be associated with a particular storage profile. Making these names easy for the end user to understand improves the experience.

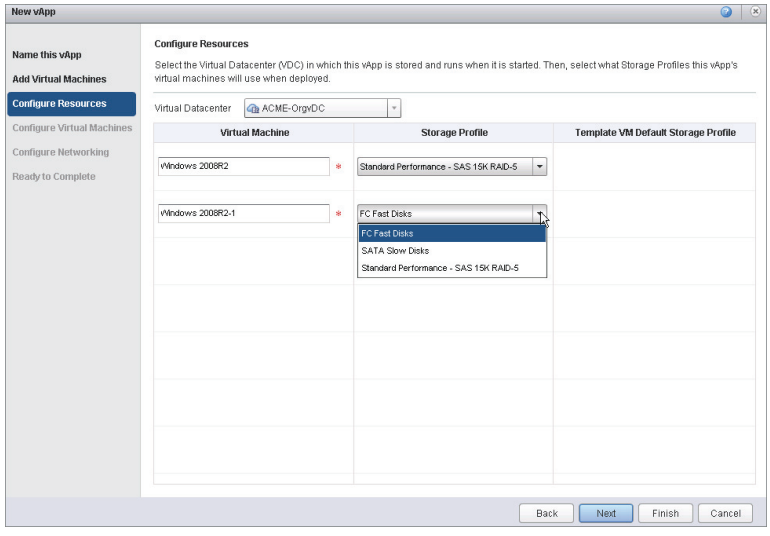


FIGURE 12.19
The easily understood names make it easier for an end user to choose where to deploy vApps in vCloud Director

For scalability knowledge, a storage profile is associated to a single vCenter. Creating the same storage profile on multiple vCenter instances will be viewed as independent Provider vDC resources to vCloud Director.

On the server side, vCloud 5.1 has extended the capabilities of block-based datastores along with fast provisioning to support 32-node clusters if vSphere 5.1 Virtual Machine File System 5 (VMFS-5) volumes are on the backend. Traditionally, the maximum number of supported nodes in a cluster when using fast provisioning was eight hosts because of a limitation with VMFS volumes. NFS datastores, on the other hand, could support more than 8 hosts but less than or equal to 32 host clusters. vCloud 5.1, along with vSphere 5.1 VMFS-5, brought feature parity at the host and cluster level to support vSphere's maximum cluster size of 32 with Fault Domain Manager (FDM—vSphere's new and improved HA agent). It's still a best practice to combine into clusters servers with similar characteristics, such as amount of memory, processor families, and number of cores. Different Provider vDCs have differing hardware capabilities, so tying an SLA and chargeback profile to each will be unique.

vCloud 1.5 featured the ability to have elastic Provider vDCs. This lets an existing Provider vDC add computing, memory, and storage resources. As discussed earlier, a cluster can scale to 32 hosts. If the resources in that cluster are being consumed at a rate that is maxing out all consumable resources, then additional clusters can be added to the existing Provider vDC. This capability has the constraint that all clusters in a single Provider vDC must exist in the same vCenter Server, in the same logical datacenter, and must use the same distributed virtual switch (DVS). The easiest implementation of an elastic Provider vDC uses a single DVS to make sure all clusters have similar networking properties. The vSphere Distributed Switch (vDS) is mapped at the logical datacenter level.

Elastic Provider vDCs play a role in design because they can cause vSphere, vCenter, and vDS pieces to potentially reach their maximums. Depending on the number of hosts you're adding to an elastic Provider vDC, you may create conflict with vSphere maximum thresholds.

The Virtual Network Switch

One of the constraints that you may run into is the level of licensing. Enterprise and Enterprise Plus licensing are the only two forms available to run vCloud Director. One of the most critical pieces required to let vCloud Director function with less intervention is to use the vDS that is only available with Enterprise Plus licensing. The Cisco Nexus 1000v also has functionality parity with the vDS in vCloud 5.1 in terms of network pools. As of this writing, the Cisco Nexus 1000v doesn't match the scalability of the vDS.

For this chapter, we'll focus on the vDS. If you must use Enterprise licensing, then the vSphere Standard Switch (vSS) is all that is available for you to use. It's compatible with vCloud Director but poses many functionality constraints. These constraints will be discussed in the following section.

Network Pool Decisions

The role of a network pool is to enable the creation of Layer 2 segmented networks, including organization routed networks, vApp networks, and isolated networks (discussed later). Every Layer 2 segmented network can reuse the same IP address space, which allows networks to scale. Let's examine what's available and how to architect for each type:

Port Group–Backed Network Pools This is the only network pool type that is compatible with Enterprise vSphere licensing. This means this type of network pool can be used on both the vSS as well as the vDS. The downside of this network pool is that it must be manually provisioned. If you're going to need 100 networks, then it's your responsibility to manually create 100 different port groups for use by vCloud Director. No automation is supported by vCloud Director.

VLAN-Backed Network Pools This type of network pool is automatically provisioned on a vDS and uses a range of specified VLANs. If you want to enable the creation of 100 different Layer 2 networks, you can give vCloud Director a range of VLANs such as 100–199. Whenever a new network is being provisioned, a VLAN is taken from the pool; and when that network is destroyed, the VLAN is added back to the pool.

The isolation of the networks relies on the configuration of the upstream switches. This requires that VLANs given to vCloud Director for pool creation must be configured on the upstream switches. The constraint is that the number of VLANs is finite, and this method eliminates usable VLANs in the network. The vDS is a requirement for this type of pool.

vCloud Director Network Isolation–Backed (VCD-NI) Network Pools This type of networking uses MAC-in-MAC encapsulation to create a transport network. This network can create up to 1,000 Layer 2 networks in a single VLAN and is automatically provisioned using the vDS. The impacts of this type of networking are that jumbo frames are required to mitigate packet fragmentation (1,600 maximum transmission units [MTUs]), CPU overhead is consumed on the ESXi hosts to do the encapsulation and decapsulation of additional packet headers, and the vDS is a requirement.

There can be a maximum of only 10 VCD-NI network pools per vCloud Director instance. This option is more secure than VLAN-backed network pools because vCloud Director is in control of the networking and not relying on outside configuration.

Virtual Extensible LAN (VXLAN) Network Pools This pool is the successor to VCD-NI and is moving toward an industry standard of using Layer 2 over Layer 3 MAC-in-UDP encapsulation. This type of network pool uses a multicast address mapped to a VXLAN segment ID for isolation and uses multicast for learning nodes on the network. The ESXi hosts become configured with a VXLAN Tunnel End Point (VTEP) to enable traffic between VMs to communicate over Layer 3 while only seeing Layer 2. This network pool can create up to 16 million networks in a single Layer 3 multicast network and doesn't rely on VLANs for the separation. The constraints related to this network pool are that jumbo frames are required because 1,600 MTU packets are transmitted over a multicast-enabled network and a multitude of other configurations. In the current release of vSphere 5.1 and vCloud 5.1, VXLAN can only be paired with a single vCenter and vCNS Manager instance. Multiple vCenter Servers can't share VXLAN networks at this time.

If you plan to implement VXLAN, many prerequisites must be defined early in the process, such as the following:

- Acquiring a Layer 3 VLAN with a default gateway and addressable IP space
- Enabling Internet Group Management Protocol (IGMP) snooping on switches taking part in multicast traffic
- Assigning an IGMP querier address on the routers taking part in multicast traffic
- Use of IGMP by hosts and adjacent routers to establish multicast group membership
- Enabling Protocol Independent Multicast (PIM) on the router if VTEPs are on different Layer 2 segments
- Aligning the vDS uplinks properly with failover, Link Aggregation Control Protocol (LACP), or EtherChannel (LACP and EtherChannel require port groups to be configured with the Route Based on IP Hash option)

You need to know your NIC teaming policy ahead of time as well. vSphere 5.1 brought the functionality of true LACP and EtherChannel for configuration of vDS uplinks. This functionality was made available because VXLAN can only use LACP or EtherChannel when teaming multiple NICs together for communication. If you want to use LACP or EtherChannel, it must be supported and configured on your upstream switch. The other option, which may be the path of least resistance, is failover. The failover option will choose only one NIC for communication; the other NIC is put in standby. This option still works very well when network I/O control (NIOC) is enabled and all other port groups are set to Route Based on Physical NIC Load. vSphere is smart enough to move traffic on any uplink that has free resources in this scenario.

External Networks

Talking about the vDS is a good segue into the next portion about networking. As you design your vCloud environment, one key is defining the external networks. An *external network* is a shared-services network that can be given to any organization or tenant in the cloud.

This external network comes in many different forms. The key question to ask is, "What do the tenants in the cloud need access to?" Every situation is different, but here are some of the most common external networks:

- Public Internet access
- Tunneled Internet access
- Dedicated access to an existing network per business function
- Backup network for agent-based solutions
- Initial implementation of vCloud Director networking

One key thing to understand is that anything sitting on an external network is supposed to be viewed as sitting beyond a firewall. Anything connected to a shared external network can communicate with one another, unless of course the VM itself has a firewall with rules configured. Therefore, this configuration is different for every environment.

How does the external-network configuration relate to vSphere networking? The external network is the simplest form of vCloud networking. It's no different than a manually provisioned port group. Yes, it's that simple. Today, every vSphere administrator knows how to configure a port group on a virtual switch, and that directly maps as an external network in vCloud Director.

The external network you're creating as a port group should follow many of the best practices set today:

- Use static binding with elastic port allocation.
- Reject forged transmits, MAC address changes, and promiscuous mode.
- Assign a VLAN ID to limit broadcast traffic.
- Set a proper load-balancing technique.
- Use uplinks properly in conjunction with the load-balancing technique.

In previous versions of vCloud Director and vSphere, the external port group was manually changed to Ephemeral Port Binding. This is because the cloud administrator was never conscious of how many devices would be consumed on the external network, and this allowed the dynamic creation and deletion of ports. The new static binding with elastic port allocation brings the security features of static port binding while also dynamically increasing the number of ports available on the port group. Thus you'll make fewer administrative mistakes during installation and configuration of vCloud Director because the port-binding types can't be modified until all ports are free.

When you're configuring external networks in vCloud Director, there are some prerequisites for the port group that has been provisioned:

- Gateway address
- Network mask
- Primary DNS
- Secondary DNS (optional)
- DNS suffix/FQDN (optional)

- Static IP pool
- Unique name for identification

Many of these attributes are common knowledge, so we'll touch on the two design pieces. The static IP pool is a range of IP addresses that vCloud Director can use to allocate external access to tenants in the cloud. These IP addresses are necessary for external communication in organization routed networks, which we'll discuss later. In addition, any virtual NIC or device placed on this network will consume an IP address from the static pool that is defined. It's important to note that the same external network can be created with the same VLAN multiple times, and the range of IP addresses can be segmented out based on each tenant. This is more complicated, but such a use case may exist. Understanding how to design for vCloud networking is important.

Let's examine some possible scenarios of using external networking in vCloud Director. For this exercise, we'll keep things simple and not dive into organizational networks but will show where the logical mapping takes place for accessing resources outside the cloud.

In Figure 12.20, there are four external networks, and each one has access to different types of networks. You may also notice that a VM is connected to each of these networks. For this exercise, the VM may represent a live VM or a vCloud Networking and Security gateway appliance. For simplicity's sake, we've chosen to use VMs; we'll examine vCloud Networking and Security gateway appliances more in depth in the next section.

The first use case in Figure 12.20 is the *public Internet*. This use case will resonate with service providers more than enterprise customers. In this scenario, a VM placed on the external network is given an IP address from the static pool of 74.11.123.2–74.11.123.40 along with the corresponding gateway and network mask. Perhaps you want to give a VM that will sit on the public Internet a static IP because the nature of a static IP pool is that if the VM is powered off, the static IP from the pool is released back to the pool. On reboot, the VM may be given a different IP address from the static pool. Because this Class A subnet address has 74.11.123.41–74.11.123.62 still available, you can allocate that to a VM as well, outside of the specified static IP pool. Giving a VM an IP from the static pool could create IP address conflicts. If you want to retain the IP assigned from the static pool, then select the Retain IP/MAC Resources check box (see Figure 12.21).

To create a scenario in which a single organization or tenant is given a certain number of public IPs—for instance, five—you can create multiple external networks with the same VLAN: 90. The static IP pool given to this tenant can be five addresses from the large pool of 74.11.123.2–74.11.123.62. This makes managing network subnet masks much easier, instead of assigning 255.255.255.248 network masks to everyone as well as future upgrades when a tenant needs more IP addresses. To create this scenario, select the Allow Overlapping External Networks check box, as shown in Figure 12.22.

The next scenario in Figure 12.20 is *tunneled Internet*. Both service providers and enterprise users can relate to this type of external network. This scenario is very similar to the previous scenario where there was direct access to the public internet. In this case, you create another layer of security by using an internal network address class and a traditional corporate firewall before accessing the Internet. This method allows the network security teams to remain in control of what enters and leaves the network by having granular control over an advanced firewalling appliance. This type of scenario could also be used for many enterprise networks where VMs in the cloud (behind organization routed networks) can access other system servers such as Active Directory or SMTP.

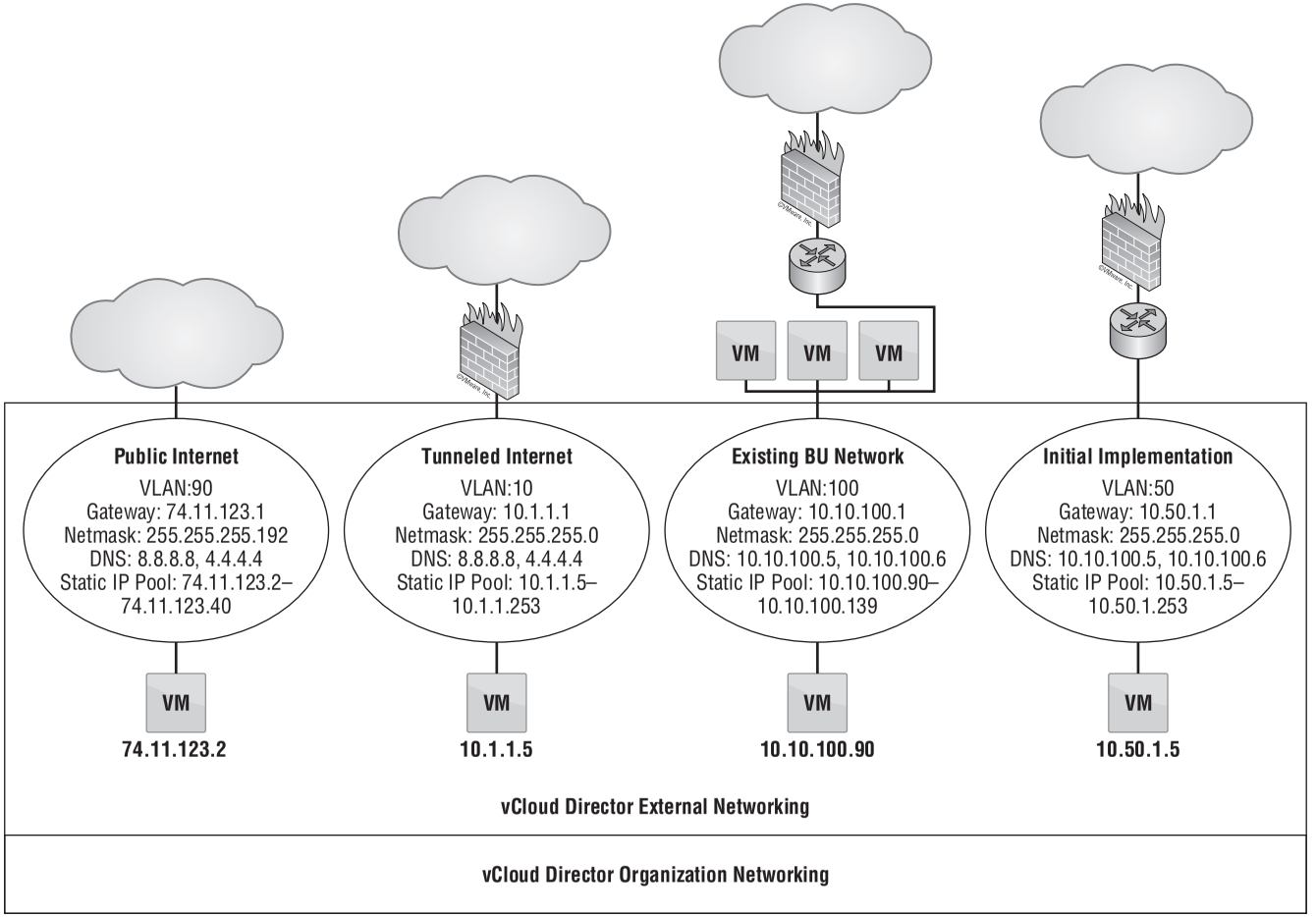


FIGURE 12.20
An external network is a shared-services network. All vApps that bind to this network can communicate with one another.

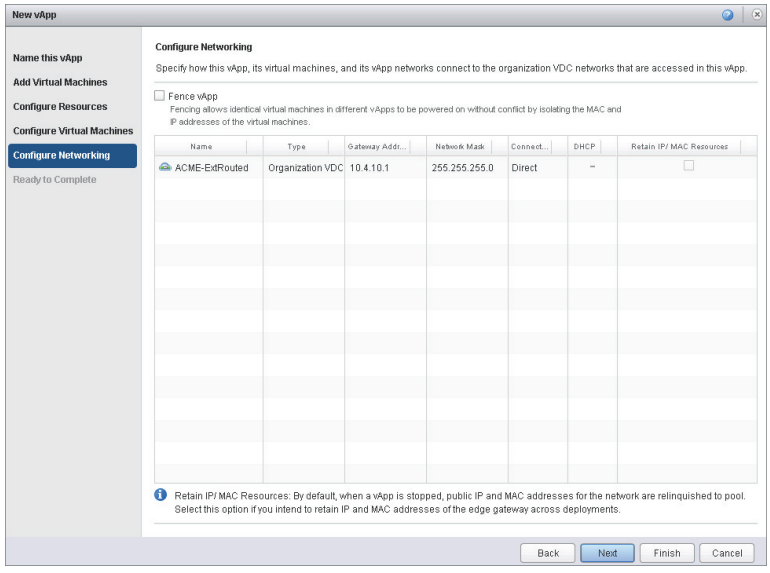


FIGURE 12.21
Selecting the Retain IP/MAC Resources check box enables the vApp to consistently maintain the same IP characteristics even after you power off the vApp.

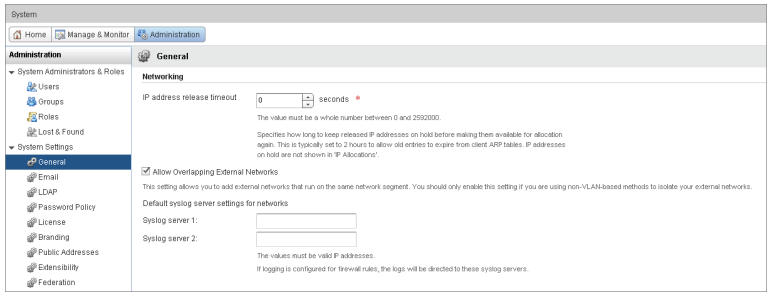


FIGURE 12.22
Enabling overlapping networks lets you create multiple external networks using the same VLAN.

The *existing business unit (BU) network* is a use case where VMs in the cloud need access to existing resources outside of the cloud but must be segregated. For instance, if VLAN 100 is given to the engineering group, then this ensures that VMs accessing this external network will only be able to talk to resources on this network. As you can see in Figure 12.20, the static IP pool given to this external network is 10.10.100.90–10.10.100-139, giving it a total of 50 IP addresses. This IP pool must be excluded from the Dynamic Host Configuration Protocol (DHCP) range that may or may not exist for this VLAN from an outside DHCP server so there aren't IP conflicts. This type of scenario relates to any network resource outside vCloud Director, such as a backup network for access to a backup proxy for agent-based backups.

The last scenario in Figure 12.20 is called *initial implementation*. As companies continue to adopt vCloud Director, there is a major transition to its networking aspect. This scenario is also known as an *organization direct connected external network*. As we've discussed, the external network is nothing more than a direct mapping of a vSphere port group. The easiest way to transition into vCloud Director is not to create vast amounts of change within your company.

Keeping existing processes in place without making users of the cloud learn new concepts right away allows for a smoother adoption of the cloud throughout the company. As users deploy VMs on this network, each VM receives an IP address on an existing VLAN or a new VLAN and integrates seamlessly with existing processes. We'll explore this development more in the next section on organization networks.

Designing Organizations, Catalogs, and Policies

vCloud Director allows multiple tenants to consume shared resources without traversing or seeing another tenant's data. Within the constructs of vCloud Director, a tenant is called an *organization*.

The cloud administrator is responsible for creating all organizations in vCloud Director. The first organization that should be created for the cloud is for the service provider of the cloud. This usually maps to IT or the name of the actual service-provider company. The cloud provider must have an organization created for it because you need an authority for the public catalog offering, which we'll touch on later.

When you're creating an organization in vCloud Director, give it a simple, short name that can be remembered easily, as shown in Figure 12.23. For instance, if Action Creators of Mechanical Engineers comes to you for business, the simple name is "acme." Many enterprises can be given their department name because that should be easy to recognize. This short name is used for creating the URL that tenants use to access their organization.

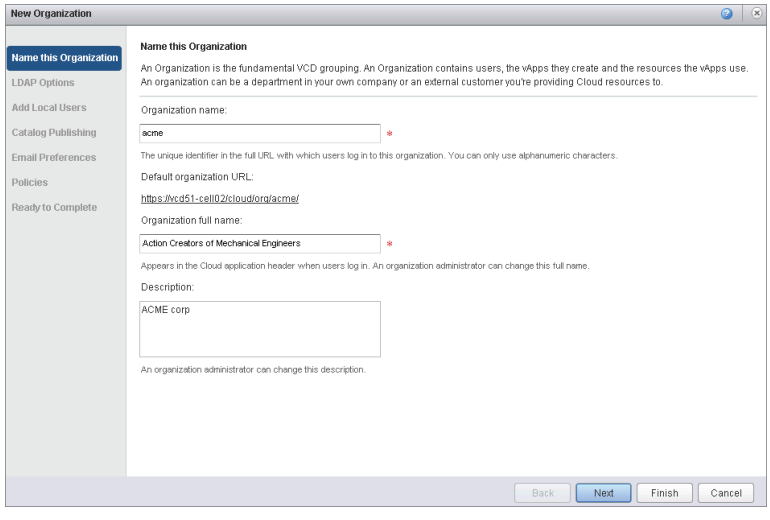


FIGURE 12.23
Create a short name for an organization so it can be easily remembered.

When you create an organization in vCloud Director, you have the option to dictate how catalogs will be handled. This is where the cloud provider will need to have the rights to publish a public catalog. Figure 12.24 shows the options for publishing catalogs that all organizations can use. Choosing the option Allow Publishing Catalogs to All Arganizations allows the organization to turn on the sharing capability among all organizations and tenants of the cloud. The cloud provider can create a public catalog with sample templates such as Windows servers,

multiple types of Linux distribution, and any type of ISO media. On the other hand, most organizations should select the Cannot Publish Catalogs option for security reasons. This option is crucial to the security of data between differing organizations, because giving an organization administrator the ability to accidentally create a public catalog could compromise information.

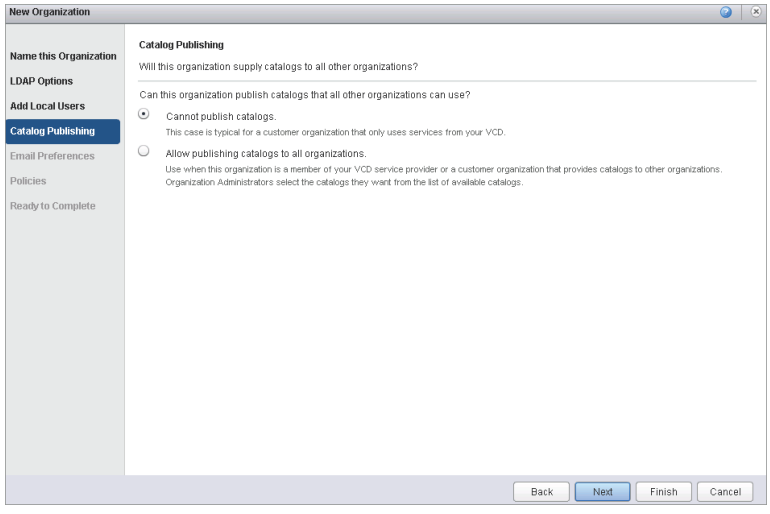


FIGURE 12.24
Allowing an organization to publish catalogs to all organizations gives the organization the ability to create global catalogs accessible to all tenants in the cloud.

Catalogs play a significant role in enabling end users of the cloud. Every organization can create private catalogs and offerings. As suggested earlier, the cloud provider can provide standard OS templates and ISO images. In addition, every organization administrator can create a catalog accessible to only their organization. This catalog can contain OS images with corporate applications or additional ISO types that are different from what is in the public catalog. After deploying an image from the public or organization catalog, an end user can customize that image with whatever they need. Whether it's a hardened OS, installation of a database, or installation of customized applications, the end user (if given proper permissions) can move it into another catalog. Organization catalogs can be created to give Read, Read/Write, or Full Write access to every user in an organization, certain users (local or LDAP), or certain groups (local or LDAP).

Let's look at a use case for this type of operation. A group of developers is experimenting with a new upgrade for their application. The original version of this application lives in a vApp in the organization-wide catalog. The organization administrator creates a catalog called ProjectX-Catalog and gives the developers Read/Write access to it. The first developer provisions the original version from the organization-wide catalog into their cloud and performs the first stage of the upgrade process. Once the upgrade is completed on the original version, the developer uploads this upgraded version to ProjectX-Catalog. The next piece of the upgrade is to see which developer has created the best plug-in. Each developer provisions the upgraded vApp from ProjectX-Catalog and implements their plug-in. After implementation, they upload their vApp into ProjectX-Catalog, and everyone can provision any vApp to see what the others have done. After ProjectX is completed, the organization administrator copies the upgraded vApp to the organization-wide catalog. This process allows end users to keep track of their own code

and bug changes without having to share VMs. It also gives IT a simple way to allow end users to enable provisioning of their own resources.

During the creation of an organization, another key concept to keep in mind is the policy of leases. Leases play a crucial role in the amount of resources that can continue to be consumed in the cloud. vApp leases and vApp template leases have similar concepts. The difference is that a vApp lease pertains to vApps that have been provisioned from catalogs, whereas vApp templates are the vApps that are in the catalog. The values represented for each type of lease will differ among organizations.

Before designing leases, you have to understand what each type of lease means. The length of a lease can range from a minimum of 1 hour to indefinite:

- The *vApp maximum runtime lease* dictates how long a vApp will run before it's automatically stopped. By default, this setting is set at seven days. This stopgap is put in place to make sure vApps aren't consuming CPU and RAM during extended periods. The timer for this lease begins as soon as the vApp is powered on. After seven days, when the lease expires, the vApp will be stopped even if a user is controlling the vApp actively and has a remote console session open to it. The only way to mitigate this lease is either to come to an agreement with the cloud provider to extend the lease or for the end user to reset the vApp lease by powering the vApp off and then back on or by resetting the lease in the vApp properties.

- The *vApp maximum storage lease* timer begins immediately after the vApp maximum runtime lease expires. This lease ensures that space isn't being wasted on storage. The default for this setting is 30 days. If the vApp sits dormant for 30 days without being powered on, it's handled by the Storage Cleanup option. Storage Cleanup has two possible values: Move to Expired Items or Permanently Delete. By default, this option is set to Move to Expired Items.

  Moving a vApp to Expired Items makes the vApp disappear from the end user's My Cloud homepage view but keeps it in the organization administrator's Expired Items view. The organization admin can reset the lease so the end user can access the vApp again or can delete it permanently. When a vApp goes to Expired Items, it's never automatically deleted. Therefore, the organization admin must be conscious of this setting to avoid overconsuming storage resources. On the other hand, if this option is set to Permanently Delete, then the vApp is unrecoverable unless a backup has been saved elsewhere.

- The *vApp template maximum storage lease* follows the same rules as the vApp maximum storage lease but only applies to vApps in catalogs. The default setting is to have the vApp template expire in 90 days and be moved to Expired Items.

The storage leases designed for an organization depend on what the cloud provider has defined as their standard or what the provider and tenant have agreed on. The amount of consumable resources must also take into account dormant VMs when you're designing storage leases.

As pointed out previously, organization administrators need to keep up with expired items. The cloud provider organization, on the other hand, should have different settings. The vApp template storage lease should be set to Never Expire. The reason for this is that if you as the cloud provider create a public catalog with default vApp templates and have only a single

Windows 2008 R2 image, then after 90 days no organization can provision that vApp template. The Never Expire setting makes sure templates aren't moved to a different location and doesn't interfere with daily operations.

Correlating Organizational Networks to Design

When an organization deploys a vApp, it needs some type of networking to communicate. The type of networking the vApp requires depends on the use case. Three types of Org vDC networks are available to facilitate many needs.

One of the simpler types of networks, called an *organization internal network*, is completely isolated. The only communication that takes place on this network is between the vApps on the network. There is no connection from this network to the outside world, as shown in Figure 12.25. Such a network can be used for applications when you need to make sure nothing can be compromised. For example, an internal network can be created for internal projects to ensure that they don't interfere with anything and potentially disrupt production workloads.

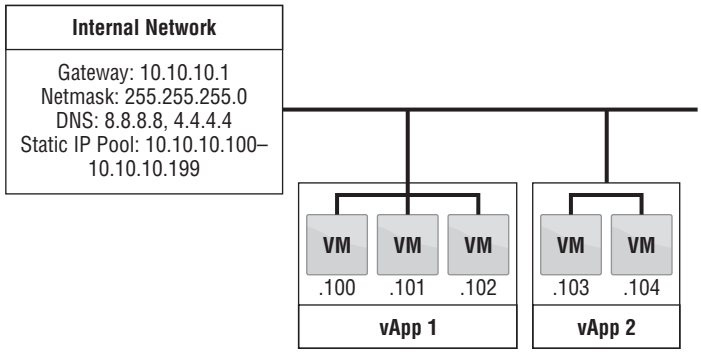


FIGURE 12.25 Organization internal networks are isolated from communication to any external networks. Only vApps connected to this network can communicate with each other.

Another type of networking, called an *organization direct-connect external network*, directly maps to external networks and correlates to a vSphere port group. When a vApp is placed on this type of network, it grabs an IP address and the personality of a normal functioning port group, as shown in Figure 12.26. There is no masking of IP addresses or firewalling.

A vApp is placed on this type of network when it needs to directly communicate to the outside world, such as a web server that needs to be accessible from the Internet. This type of networking is suitable for many initial adoptions of vCloud Director because it doesn't interrupt current processes. For instance, existing processes for provisioning VMs may include being added to an Active Directory domain, added to a configuration management database (CMDB), or probed on the network for adding to a patch and maintenance cycle. Many of these scenarios come into play when IT needs to maintain control of every VM.

If IT can't probe the network to discover new VMs and add them into a scheduled patch routine, then how can the VMs be managed? In this scenario, the VM is placed on a vSphere port group that is no different than many current processes. The OS team is responsible for patches and maintenance, and the network team is responsible for network security at the firewall and the router. The downside is that you aren't getting all the benefits that vCloud Director offers in terms of multitenancy and isolated Layer 2 networks. The IT staff can eventually learn to pick up new tricks to allow management to happen in the next type of networking we'll explore.

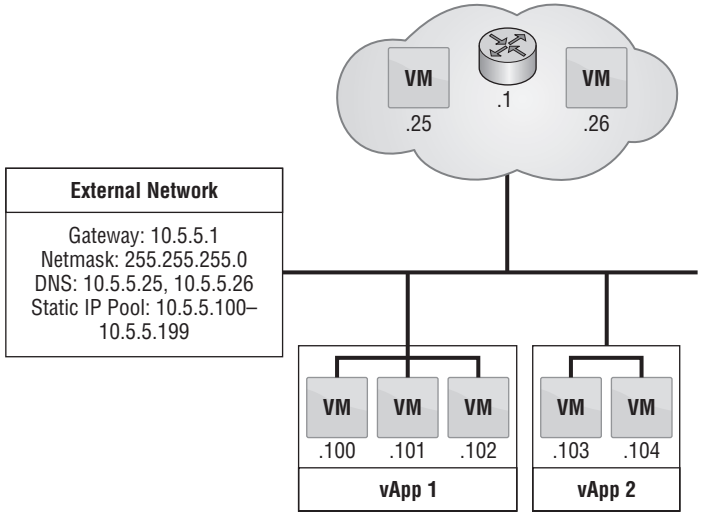


FIGURE 12.26
Connecting a vApp to an organization direct-connect external network allows the vApp to maintain the characterization of a vSphere port group while still being managed by vCloud Director.

The final type of network is called an *organization routed external network*. This type of networking allows for multiple Layer 2 networks to be isolated in multitenant scenarios. vCloud Director 5.1 brought some new enhancements to this type of networking that changes the architecture just a bit.

The vCNS Manager server is responsible for deploying an Edge (formerly vShield Edge) gateway appliance. This device is deployed into an Org vDC that is provisioned for an organization when needed. The Edge gateway is responsible for everything going in and out of an organization routed external network.

The Edge gateway can be considered a mini-firewall. It's the hub of multiple networks to facilitate many IP connections from both inside and outside. It can facilitate multiple external network connections as well as multiple organization routed external networks, up to a maximum of 10 different connections total.

Figure 12.27 shows where the Edge gateway connects to two external networks. The Edge gateway consumes one IP address from the static IP pool of the external networks during creation. The diagram also shows that the Edge has two different organization routed external networks attached to it as well. During the creation of the organization routed external networks, the Edge appliance assumes the role of the gateway for this network. During the configuration of the Edge gateway, one of the external networks must be selected as the default gateway for all traffic of the organization routed external networks.

The Edge gateway plays a few more significant roles as well, such as 5-tuple firewall ruling, static routing, NATing, DHCP, VPN, load balancing, and network throttling (features depend on the level of vCNS licensing purchased). If one of the organization routed external networks must communicate with an external network, the Edge gateway must be configured with a source NAT rule to translate the internal IP addresses and a firewall rule that allows traffic to exit; and if the default gateway isn't the destination, then a static routing rule must be in place.

In Figure 12.27, virtual machines on organization routed external network 1 and virtual machines on organization routed external network 2 won't be able to communicate with one another unless a firewall rule is put into place.

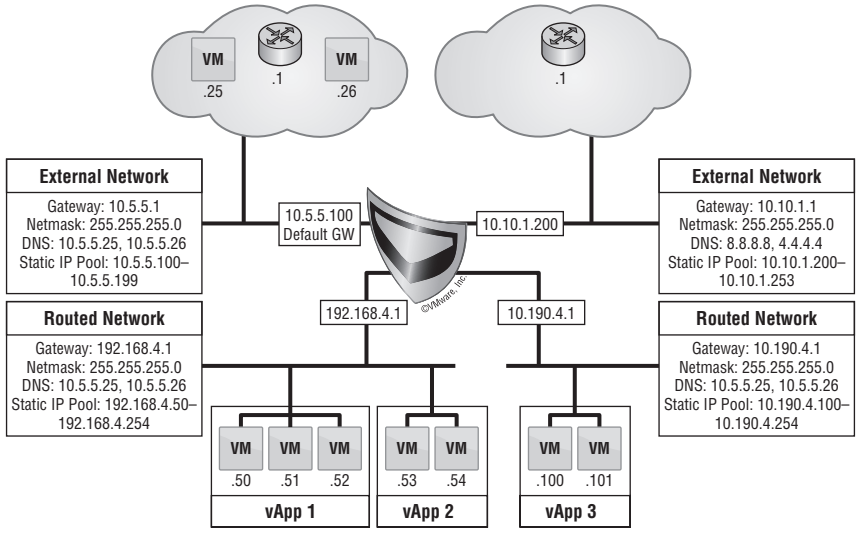


FIGURE 12.27 Organization routed external networks use an Edge gateway device to allow Layer 2 isolated networks to access an external network through NAT and firewall rules.

This type of networking is where we really get into what vCloud Director was intended for when we talk about multitenancy. A single organization can have multiple networks. The number of networks given to an organization by a vCloud administrator will vary depending on the use cases. The networks can be categorized as production or test, based on different applications, or everything can go on one organization routed external network.

When you're designing organization routed external networks, you must be careful not to confuse the end user. Keep the networking simple and familiar. Naming a network Org-Routed-50 means nothing to an end user. But if you name the network something like ACME Production versus ACME Test, then it's clearly defined, even though the end user is unaware that these two networks could be internal, routed, or external. For initial implementations, you may only have only a single network available for vApp provisioning, because bringing too many networks into a single offering can be confusing to end users.

Figure 12.28 depicts two organizations, ACME and EMCA. Each organization uses the same external network to access the Internet and uses the cloud provider's DNS servers. Each organization has a Edge gateway appliance that creates a separate Layer 2 organization routed external network. The organization routed external network is clearly identified to the end user during provisioning of vApps. Both networks use the same IP subnet, 192.168.4.0/24, yet neither has a clue about the other's existence. The Edge gateway is responsible for source NAT to the external network and masking internal IPs. Therefore, the two organizations are logically separated.

End Users and vApp Networking

What do you do if end users want to make sure the vApp they're provisioning is segregated out onto a network they can control? This means they aren't using up network resources on what has been provisioned by the vCloud administrator. Here is where vApp networking takes over.

vApp networks are created by the consumers of the cloud and are used to aggregate VMs together in a vApp. The same types of organization networks apply in vApp networks. Designing for vApp networks doesn't take any networking knowledge but instead focuses on

resources. Some types of vApp networks consume CPU and memory resources because they require Edge devices. As you build the physical design for Provider vDCs, you need to understand how knowledgeable your tenants are about vCloud networking. If a tenant understands that they can automatically create segregated networks on their own, then more resources will be consumed and must be accounted for in the Provider vDC.

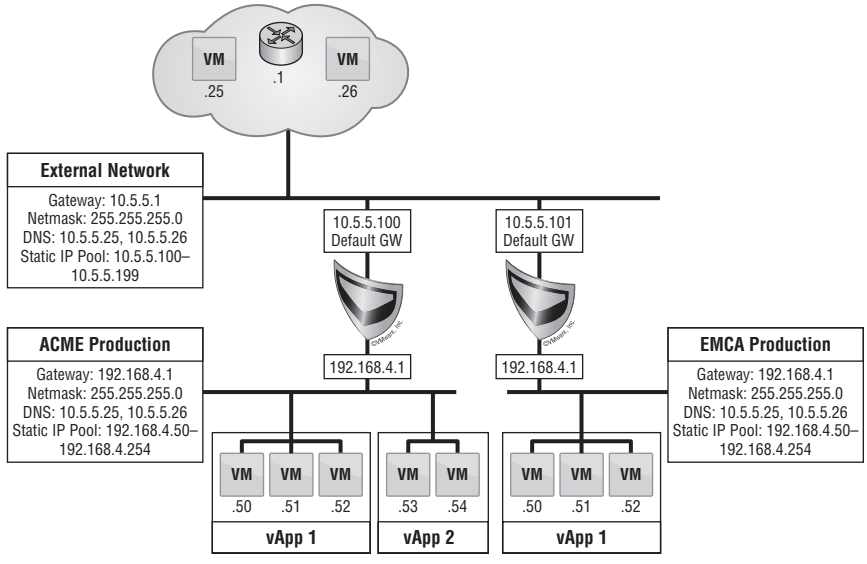


FIGURE 12.28
Two organizations can share the same external network but are logically separated by Edge gateway appliances. This allows multitenancy in the cloud.

The first type of vApp network is called a *direct vApp network*. In this scenario, the end user does nothing. The tenant doesn't configure anything networking related. During the provisioning of a vApp, the end user chooses the type of organization network it will live on. After the vApp is provisioned, the user can turn on the vApp, and it will consume IP addresses from the organization network as shown in Figure 12.29. This is typically what is used in most deployments because the tenants don't know enough about what is available to them. In most cases, this is also an easy way to begin the adoption of vCloud Director. To recap, this is simply provisioning a vApp from a catalog directly onto an organization network that can be an organization internal network, organization external direct network, or organization routed external network.

The second type of vApp network is called a *fenced vApp network*. During vApp provisioning, this scenario requires the end user to check a box that asks whether to fence the vApp. By default, the vApp isn't fenced. If fencing is enabled, an Edge device is deployed and creates a boundary for the vApp. Fencing makes sense when you need to test an application but can't change the IP or MAC addresses associated with the VMs.

This type of networking allows an end user to make identical copies of a vApp without changing a single characteristic. Of course, the use cases could be for developers who write code tied to IP or MAC addresses. In regard to organization internal networks and organization external direct networks, a NAT is necessary for communication and is automatically assigned on the Edge device. If this networking is being deployed on an organization routed external network, then a double NAT must occur because two Edge appliances need to be configured with NAT rules, as shown in Figure 12.30.

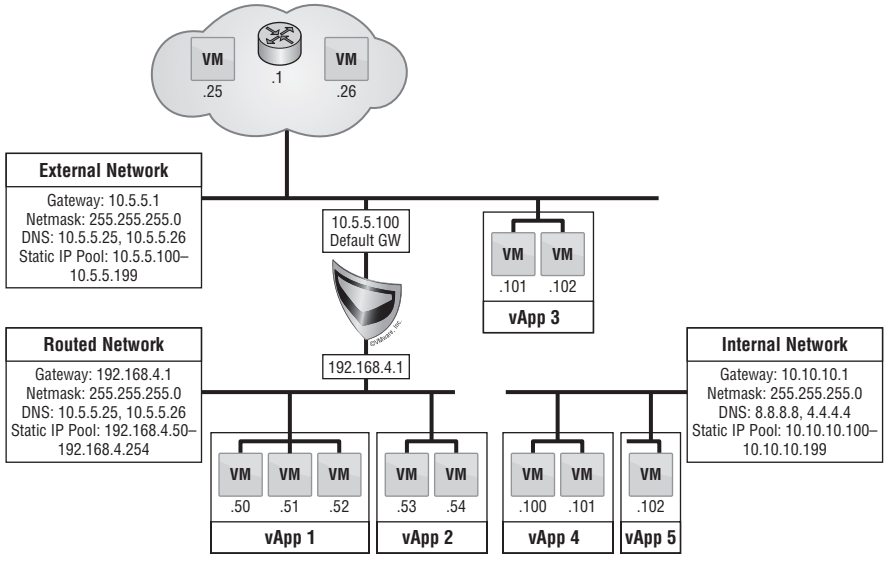


FIGURE 12.29
A direct vApp network consumes ports and IP addresses on the organization network it's connected to.

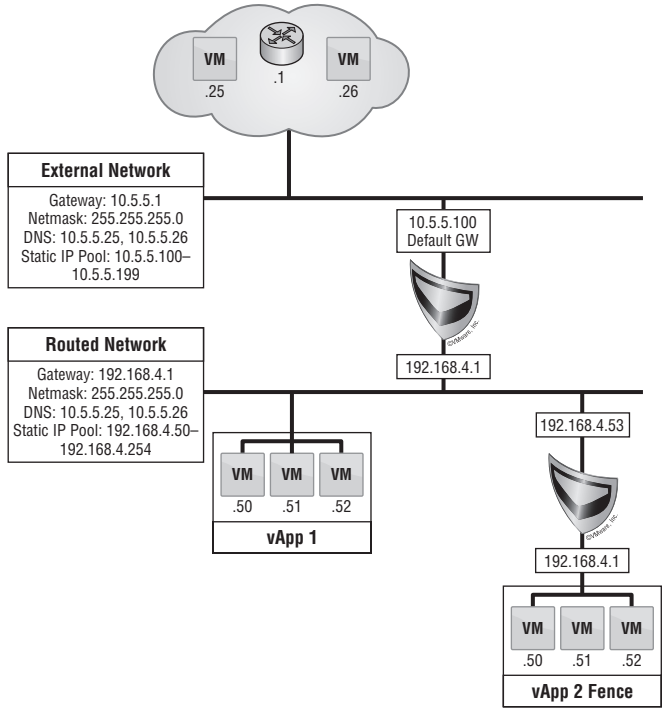


FIGURE 12.30
A fenced vApp creates an identical copy of an existing vApp and inherits the same IP and MAC addresses but is front-ended by an Edge appliance to communicate on a network without experiencing overlapping IP addresses.

The third type is called a *routed vApp network*. This requires the end user to create a brand-new network for communication. The end user is responsible for creating a gateway, a subnet mask, and a static IP pool just as if it were an organization routed external network.

The topology in Figure 12.31 looks very much like deploying a fenced vApp network, except now the end user has the responsibility of maintaining the Edge device that is deployed. The end user can set NAT and firewall rules depending on the vApp requirements.

This scenario is useful in situations where the end user wants to be responsible for networking and making sure only certain VM ports are accessible from the organization network. The routed vApp network creates an extra layer of security that gives the end user total control. This scenario is like an onion. At the cloud provider layer, you create multiple organization networks so every organization can have a place to run their VMs without interfering with other tenants. As you dig into the organization, the end user can spawn Edge devices to run their VMs without interfering with anyone else in their organization.

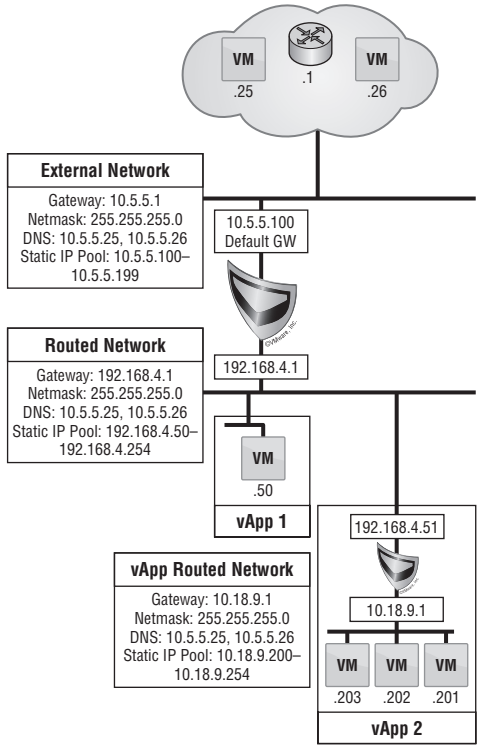


FIGURE 12.31
A routed vApp network uses an Edge device to create an additional layer of security by allowing an end user to maintain firewall rules for their specific vApp when connecting to a particular organization network.

In the last type of network, called an *isolated vApp network*, the end user creates a vApp network that has the same properties as an organization internal network. Anything connected to this network can't talk outside of it, and the VMs on this network can only communicate with one another. These types of networks can be useful to organizations and are most commonly seen in three-tier applications. Organizations that need to analyze how an application will react to a virus can also benefit from this type of network.

Figure 12.32 depicts a scenario with web, application, and database servers in a three-tier vApp. The web server needs to be publicly accessible. The web server can be given multiple virtual interfaces to connect to an organization network while the other interfaces communicate on an isolated network. The web server VM will acquire IP addresses from both networks.

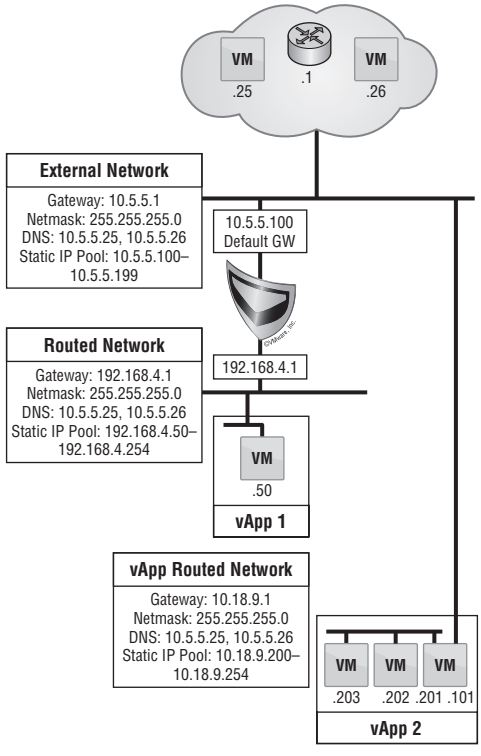


FIGURE 12.32
Isolated vApp networks allow vApps to communicate only with each other without connecting to an organization network. A VM can be given multiple NIC interfaces that can be connected to any type of organization network.

Designing Organization Virtual Datacenters

The last piece of the puzzle is defining the Org vDC. It brings all the earlier parts of this chapter into context:

- A single organization (tenant) can have multiple Org vDCs.
- An Org vDC allocates resources from a single Provider vDC in different allocation models.
- Multiple Org vDCs can consume resources from a single Provider vDC.
- Network pools provide network resources for Org vDCs to consume.

Designing Org vDCs depends on the resources required by the tenant. If you have multiple types of Provider vDCs, and the tenant needs access to each type of resource, then you must create an Org vDC for every type of Provider vDC. In most cases, the Provider vDCs can use

multiple types of storage defined by storage profiles. Therefore, when designing Org vDCs, you need to know how many tenants are subscribed to a single Provider vDC.

For instance, if you have a single large Provider vDC and 20 tenants, then all Org vDCs must consume resources from the same Provider vDC. Here, you can easily predict when resources would be consumed and when they reach a limit. If you had 3 Provider vDCs, you wouldn't want all 20 tenants consuming a single Provider vDC. Therefore, you need to determine the amount and type of resources a tenant will need in an Org vDC and map them to a Provider vDC with appropriate resources.

The cloud administrator is responsible for playing the role of "initial placement DRS" for Org vDCs. Remember, the Org vDC that is being created maps directly to a Provider vDC for resources, so understanding what hardware profile the tenant needs for their apps will dictate the mapping. This is where elastic Provider vDCs play a crucial role. If it's possible to combine all three Provider vDCs without reaching vSphere maximums, doing so will require less administrative management. Yet multiple Provider vDCs may be needed to satisfy different performance and availability characteristics.

In some cases you may have to maintain physical separation of the vSphere resources because of compliance or other regulatory restrictions. In this scenario you can have a Provider vDC be consumed by a single Org vDC, which ensures that only a single organization can use those resources. Of course, this doesn't give you the benefits of being a cloud provider and having multiple tenants consume the same resources while maintaining isolation.

Every Org vDC must be tied to a specific allocation model. The *allocation model* is a mechanism for distributing resources from a Provider vDC to an Org vDC. Such allocation models are necessary for chargeback and showback functionalities to relate costs. You can assign three different allocation models to an Org vDC. They differ in how CPU and memory are allocated in vCloud:

Reservation Pool This pool type guarantees 100% of the resources assigned to it, with their respective reservations and limits defined via a resource pool in a single cluster. If Figure 12.33 represented a cluster with 100% of resources, then a certain amount is dedicated to each tenant.

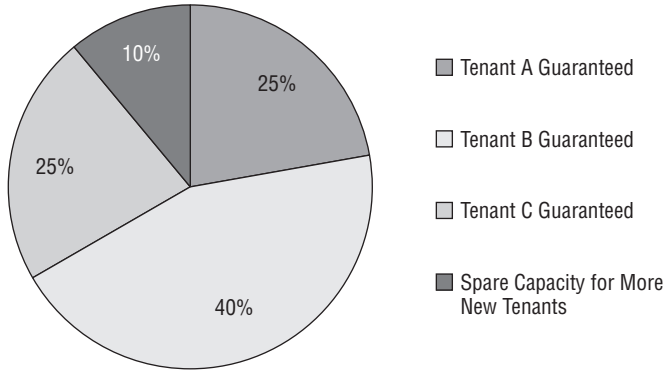


FIGURE 12.33
A reservation pool model guarantees a percentage of resources to the Org vDC from the backing Provider vDC.

Allocation Pool This pool guarantees a certain percentage of resources with a reservation set on a resource pool but no limit defined. In this model, the tenant can burst into more

capacity if needed. The spare burst capacity is a shared set of resources that can be consumed by any organization and isn't guaranteed or allocated, as shown in Figure 12.34.

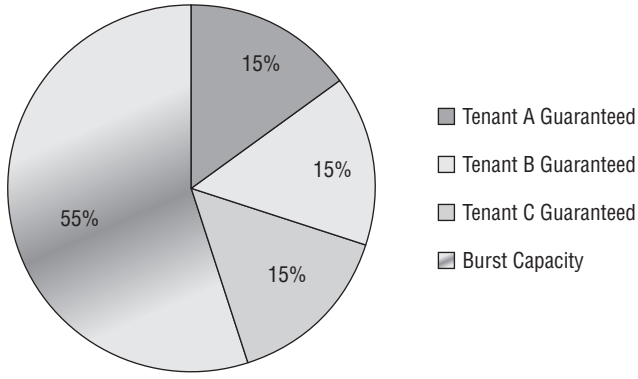


FIGURE 12.34 An allocation pool model will guarantee a percentage of resources to the Org vDC from the backing Provider vDC but also allow consumption of unallocated resources.

Pay-as-You-Go Pool This pool doesn't guarantee any resources via limits or reservations at the resource pool level. The only time resources are committed is during the power-on operations of a vApp, and the reservations and limits are set at the VM level as shown in Figure 12.35. This is similar to cloud-consumption models where you pay per VM.

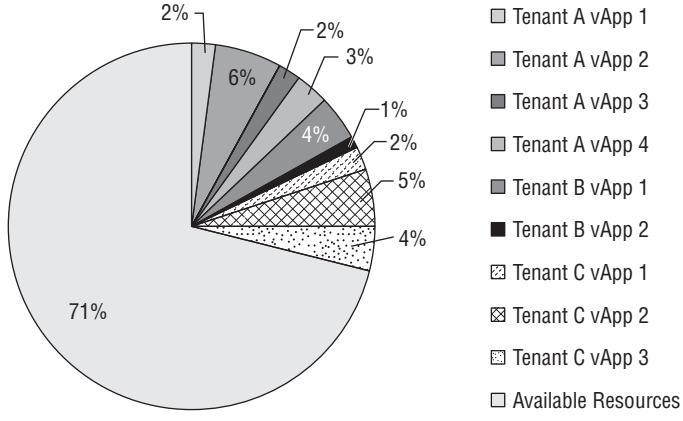


FIGURE 12.35 A pay-as-you-go pool model sets guarantees on the VMs that are provisioned to a Provider vDC in an Org vDC.

Designing for allocation models varies depending on each type of organization and how users will be consuming resources.

The reservation pool makes sense when an organization knows how much it wants to be charged on a monthly basis. This is equivalent to something like budget billing. Every month you're guaranteed a certain set of resources, and you consistently pay the same amount. There will be no surprises. The flip side is that there is no capacity to burst beyond the allocated resources. If you're given 30 GHz and 64 GB of RAM with a reservation pool, then that is all the resources available to be consumed. If you need more capacity, then it must be changed globally, and the changes affect the reservation and limits of the resource pool defined for the Org vDC.

This type of allocation model is useful in enterprises to make sure certain departments don't spin up unnecessary vApps and wildly consume resources. It's an easy way to define how many resources are dedicated to engineering versus finance versus HR versus IT. It's a failsafe to make sure the consumption of resources doesn't impact another tenant. If you're a service provider, you can verify that two organizations, such as ACME and EMCA, won't fight for resources.

The reservation pool model isn't compatible with elastic Provider vDCs and therefore is constrained to a single cluster of hosts or resource pool. This is a big constraint on choosing reservation pools as your allocation model.

The allocation pool is useful when some resources need to be reserved. It's a way to set aside guaranteed resources for provisioning vApps without paying for unused resources. The ability to burst beyond shared resources makes it more flexible than the reservation model. The down side is that the resources that aren't being reserved are consumed on a first-come, first-served basis. Many times, an organization thinks it needs 30 GHz and 64 GB of RAM but really only uses 50%. Making sure there is no overallocation of resources works well in many enterprise environments. A bigger portion of resources can be given to demanding organizations such as engineering and development, and a smaller portion can go to HR and finance. The remaining resources in the Provider vDC can be acquired by any organization if needed. This allocation model will still require the cloud administrator to monitor resource usage among the organizations and make distribution changes as necessary. The allocation pool model is compatible with elastic Provider vDCs.

The final allocation model is pay-as-you-go. There is no reservation of resources in a resource pool in vSphere; instead, the limits and reservations are enforced at the VM level. Reservations and limits are set on the VM during power-on operations. If a VM is powered off, then reservations and limits aren't enforced on the VM.

In this scenario, if there is an unforeseen boot storm, some vApps may not power on because the reservation for the VM can't be met by the Provider vDC. This model creates some unpredictability and uncertainty in a large cloud model when many vApps may be powered off. On the other hand, many enterprises may use this as an initial implementation scenario because it requires little overhead work of administering limits and reservations on a resource pool level and allows a quick way to begin working with vCloud Director. The pay-as-you-go model is compatible with elastic Provider vDCs.

You may have noticed that these allocation models don't include storage. The allocation models focus on CPU and RAM, but allocating storage to an Org vDC is uniform among the three. As described earlier in "Logical Side of Provider Virtual Datacenters," storage profiles are necessary components. During the configuration of an allocation model, the cloud administrator needs to choose the storage profiles available to an Org vDC as well the amount of storage that can be consumed from that profile, as shown in Figure 12.36. The default profile must also be selected. For design, it would be safe to say that you should choose the most economical storage profile. Thin provisioning can be enabled to over-allocate storage as well but isn't enabled by default.

You can't change the type of allocation model once it has been defined for an Org vDC. The only thing you can change is the amount of resources to be set as reservations and limits. If you want a vApp on another allocation model, it will need to be migrated to a new Org vDC.

Fast provisioning is a feature that deploys space-efficient linked clones from a base disk image to quickly provision vApps into an Org vDC. If this option is disabled, then thick clones are created during deployment into an Org vDC. Using fast provisioning with vSphere 5.1, VMFS-5, or NFS with vCloud 5.1 allows for 32-host clusters.

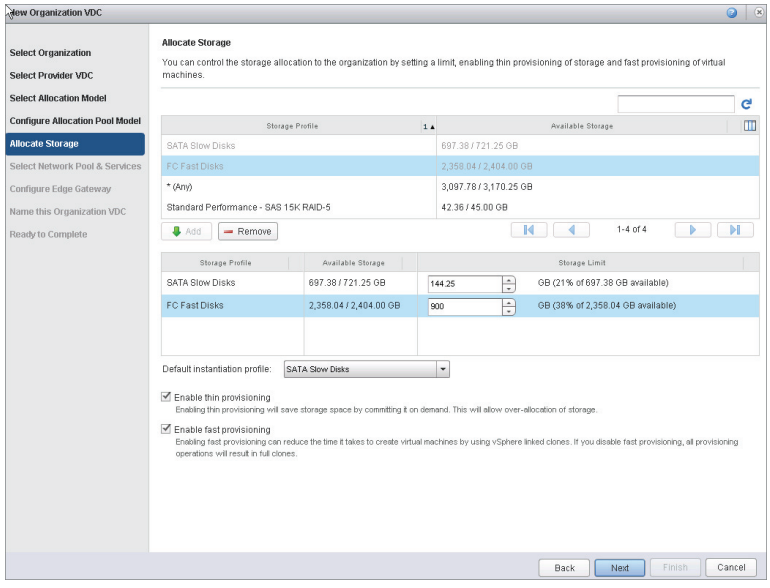


FIGURE 12.36
Storage profiles dictate the class of service or I/O characteristics that can be given to a particular Org vDC. The Org vDC must be allocated a certain amount of storage as well, for provisioning vApps.

Fast-provisioned VMs use *chain lengths* and *shadow VMs*. If the original VM lives in the catalog and a new VM is provisioned, then it can have a chain length of 29 linked clones. The linked clone chain is a clone of the cloned VM. If more than 29 clones are created in the same tree, then the next clone to be provisioned will be a full clone, and a new tree will begin. Before the creation of the clone occurs, vCloud Director determines which datastore has the most available free space and the same storage profile. The datastore with the most available free space will be the target for the next deployed VM. A shadow VM is deployed when the vApp in question must be deployed to another datastore in a different Provider vDC or vCenter instance.

You need to gather storage-capacity requirements for thick- versus fast-provisioned vApps. This brings in a key datastore design consideration. vSphere 5.1 can use 64 TB VMFS-5 datastores supported under vCloud Director. Storage design for vCloud Director should follow the same best practices with regard to vSphere design. Most architects will agree that many smaller datastores are better than a handful of large datastores in a vSphere design, but you must find a good balance between restore time, efficiency of Storage DRS, and IOPS. In terms of vCloud Director, you must be conscious of the maximums. vCloud Director can support a maximum of 1,000 datastores in vCloud 5.1. If you have a very large cloud offering, it may be in your best interest to configure larger datastores.

The next step is defining the network pool to be used in an Org vDC. As discussed earlier, you can choose any type of network pool that has been created. The type of network pool chosen matters only in the context of the actions used on the backend. The end user is unaware of network pools. The cloud administrator is responsible for defining how many networks an Org vDC can create, as shown in Figure 12.37. The number of networks allocated to an Org vDC makes sense only in terms of port-group and VLAN-backed networks. If only 20 VLANs are available in a VLAN-backed network pool, then you have to divvy them up among all

organizations. If you overcommit networks when there are only 20 VLANs, then the creation of new networks will fail. VCD-NI and VXLAN, on the other hand, can create thousands of networks. Figure 12.37 shows a VXLAN example; by default, 1,000 networks can be consumed by a single Org vDC without running into a maximum. Of course, this number may be changed for VCD-NI because it has a maximum of 1,000, so you may want to distribute networks differently.

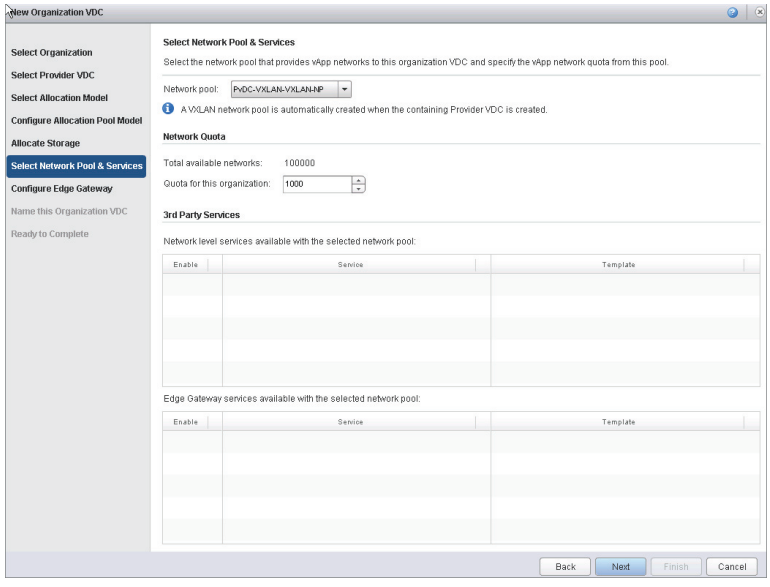


FIGURE 12.37
Network pools must be allocated to an Org vDC for the provisioning of networks in vCloud Director such as organization routed external networks, organization internal networks, vApp routed networks, and vApp isolated networks.

The configuration of an Org vDC relies heavily on networking. During configuration, as shown in Figure 12.38, you have the option to create a new Edge gateway as well as to create an organization routed external network. The Edge gateway appliance has two modes: Compact and Full. The compact version will consume less CPU and memory resources. The full version is needed if an Org vDC grows so large that network latency is experienced due to all the requests. The compact version can be converted to full with minimal downtime if necessary. The Edge gateway can be put in high-availability mode and a secondary Edge gateway appliance provisioned in standby mode. This is useful to overcome a vSphere HA event. If high availability doesn't protect the Edge gateway, then vApps in that Org vDC can't communicate externally until the Edge gateway has been rebooted onto another host in the cluster.

What if you have an Org vDC configured with an allocation pool, but you want a new Org vDC to be configured with pay-as-you-go and to be able to use an existing organization network? Another new feature of vCloud 5.1 is the ability to share organization networks between Org vDCs. During the creation of an organization routed external network, as shown in Figure 12.39, there is a check box for sharing. By default, this option isn't checked, but it's checked in this example.

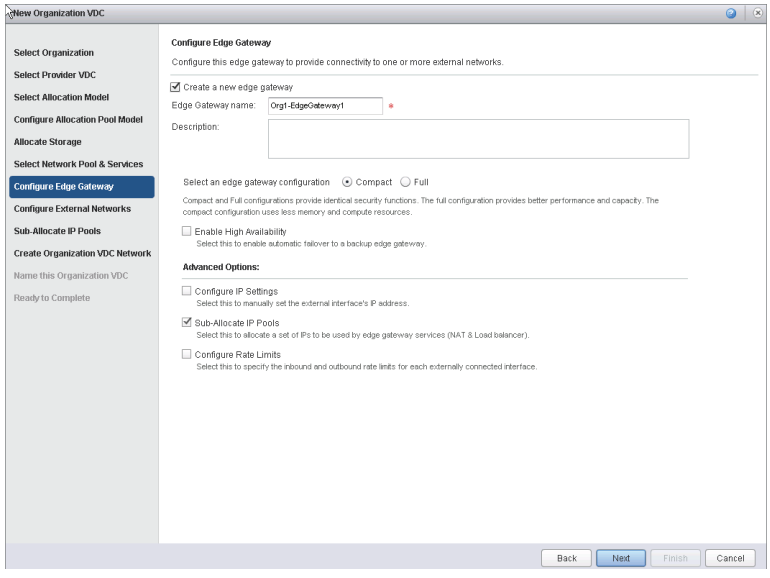


FIGURE 12.38
If you're using organization routed external networks, an Edge gateway device must be configured to allow access to the external networks.

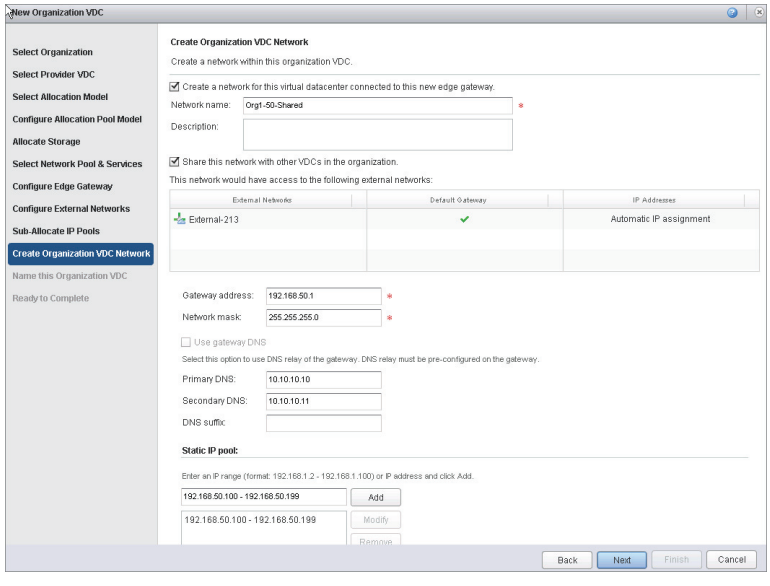


FIGURE 12.39
vCloud 5.1 allows the sharing of organization networks between multiple Org vDCs when backed by a single Provider vDC in an organization.

Multiple Sites

Now that you have an idea about how to architect and build a vCloud environment, let's think about scale. What do you do about bringing multiple datacenters into the mix? vCloud Director was originally intended to be located in a single physical datacenter. As you've seen,

a lot of components make up vCloud Director, from the management stack to the consumable resources. Creating boundaries based on physical location makes implementation easier because resources don't have to travel distances and experience latency. Let's consider an example of bringing multiple cloud environments together to be more seamless.

Suppose you have three locations—Las Vegas, Dallas, and Omaha—tied together through a Multiprotocol Label Switching (MPLS) network. There is a 10 MBps connection between all three sites. You want to create a centralized cloud where a user can request a VM. Because you have only a 10 MBps connection, you don't want the user to access their VM over the WAN; you want the VM local to their site.

This scenario can have a vCloud Director instance in each site. Using a load balancer, you can create a single DNS record that accepts traffic at the load balancer; depending on the originating requester's IP address, they will be directed to the vCloud Director instance nearest to them. Another option is to create a custom-built portal where the user can choose the datacenter where they would like the vApp deployed. The scenario also plays well with the vCloud management infrastructure because each site can be localized for everything while the custom portal takes care of API calls.

Now suppose each office has a 1 Gbps connection with less than 10 ms of latency between them. Can you create one big cloud? It depends. 1 Gbps is plenty of bandwidth to satisfy many operations, such as a console requests when a user in Dallas provisions their VM in Las Vegas. The amount of bandwidth will also play a major role in the process of copying vApps and shadow VMs between cells and datastores in different geographies.

The vCloud management infrastructure is a bit different in this scenario because there are a few options. Depending on bandwidth and latency, all vCloud cells and vCenter Servers can live in a single management domain. For instance, the vCenter Servers hosted in Dallas can control the vSphere hosts residing in Las Vegas and Omaha. Perhaps the distribution of management is deemed more suitable because it can overcome a link failure between sites. vCenter Servers can be localized to their vSphere hosts this way.

vCloud Director cells can be spread out among sites and front-ended by a load balancer to satisfy incoming connections. Less than 20 ms of round-trip time (RTT) latency is required for the communication of vCD cells to vCenter Servers and databases.

Remember, a single database is used by all vCloud Director cells, so traffic must traverse the WAN if cells live in disparate datacenters. This scenario also poses a problem in the way communication is handled by a requestor. There is no guarantee that a vCloud cell won't cross WAN links. If a user in Las Vegas requests console access to a VM in Las Vegas, it's not possible to control which vCloud cell acts as the proxy for the vCenter in Las Vegas. Therefore, a user may end up traveling between sites to access a remote console. This solution is best implemented in a scenario where bandwidth and latency aren't concerns, such as a MAN or campus.

Backup and Disaster Recovery

The backup and disaster-recovery process of vCloud Director 5.1 is tricky. The infrastructure management VMs work seamlessly with existing backup and disaster-recovery processes because these VMs live at a vSphere level and can follow traditional vSphere policies. The vApps that live in vCloud Director, on the other hand, have certain characteristics that make them unique and troublesome.

Backing up VMs in vCloud Director is the easy part; the hard part is restoration. You may have 50 instances of a Windows 2008 R2 server deployed, with the only differentiator being the

unique set of trailing characters. Of course, you can back up every single one of these images, but if you need to restore something, how do you know which VM to use? It's a manual process to grab a VM's universally unique identifier (UUID) to make sure you're restoring the correct VM. When it comes time to restore a backup, backup software doesn't have the notion of Org vDCs and doesn't know where to place the VM. Therefore, it's replaced into a vCloud consumable resource outside of vCloud Director. It's then the administrator's responsibility to take that VM and import it into vCloud Director, placing it in an Org vDC or catalog that is relevant to the tenant. This is why you may want to use agent-based backups that understand the personality of the VM in question and can restore directly to the VM. This isn't an optimal solution by any means. Many backup vendors are hard at work figuring out alternative ways to back up VMs in the cloud.

Disaster recovery is an even tougher situation. The infrastructure management VMs can be protected today with SRM 5.1 and can seamlessly fail over to a DR site. The vApps that live in vCloud Director, on the other hand, can't, due to the UUID situation. When vCloud Director is configured, datastores in Provider vDCs are mapped with specific UUIDs, and the translation of the vApp to the datastore relies on those UUIDs to tell vCloud Director where they live. If SRM were to fail over vApps from vCloud Director to another vSphere infrastructure, then vCloud Director would be unaware of the UUIDs of the datastores at this new vSphere instance. This is the same reason you can't forklift a vCloud Director instance to another vSphere infrastructure.

A process is available from the VMware Professional Services organization to use native array replication with a series of scripts to perform a successful disaster recovery of the cloud. If you want to learn more about disaster recovery of the cloud, visit `www.vmware.com`. In addition, many service providers offer the ability to do disaster recovery to the cloud. As the technology matures, SRM will be able to satisfy automated disaster-recovery efforts for vCloud Director.

Summary

In this chapter, we looked at how many of the technologies previously discussed in this book, such as server profiling, networking design, storage design, and high availability, apply to a vCloud Director design. Let's summarize some of the key points.

Your cloud architecture has two key components, the management infrastructure and the consumable cloud resources. The growth of one directly impacts the other's design. The size of the management infrastructure depends on the size of the cloud deployment and the ecosystem of products used to create a cloud offering. The cloud resources can be any kind of vSphere design, but you need to define SLAs to create a level of service for Provider vDCs. Use user-friendly storage profiles for the tenants of your cloud.

We determined good use cases for external networks where end users need communication. The types of network pools are determined by feature availability of vSphere licensing as well as physical network capability.

The type of organization network you deploy for tenants of your cloud should directly reflect the needs and knowledge of your end users. Don't rush too much change with routed networks if your company isn't ready. If they're ready, more complex vApp networks are available to your end users and will let them maintain control.

Finally, remember that multiple sites can work in a single cloud, given the right amount of network resources, but a suggested implementation is to create isolated clouds.

Index

Note to the Reader: Throughout this index **boldfaced** page numbers indicate primary discussions of a topic. *Italicized* page numbers indicate illustrations.

D

E

F

I

J

K

L

M

O

P

Q

R

S

T

U

V

W

Z