

"This book continues the very high standard we have come to expect from ServiceTech Press. The book provides well-explained vendor-agnostic patterns to the challenges of providing or using cloud solutions from PaaS to SaaS. The book is not only a great patterns reference, but also worth reading from cover to cover as the patterns are thought-provoking, drawing out points that you should consider and ask of a potential vendor if you're adopting a cloud solution."

—Phil Wilkins,
Enterprise Integration Architect, Specsavers

"Thomas Erl's text provides a unique and comprehensive perspective on cloud design patterns that is clearly and concisely explained for the technical professional and layman alike. It is an informative, knowledgeable, and powerful insight that may guide cloud experts in achieving extraordinary results based on extraordinary expertise identified in this text. I will use this text as a resource in future cloud designs and architectural considerations."

—Dr. Nancy M. Landreville,
CEO/CISO, NML Computer Consulting

Cloud Computing Design Patterns

Co-authored and Edited by Thomas Erl,
World's Top-Selling Service Technology Author

Co-authored by Robert Cope and Amin Naserpour

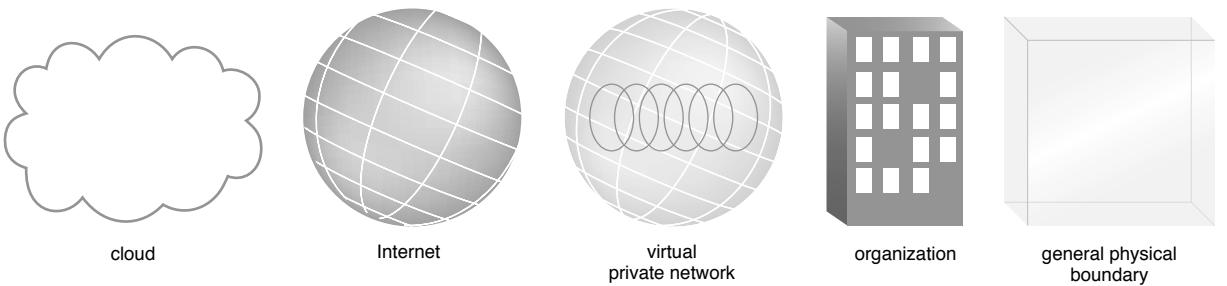
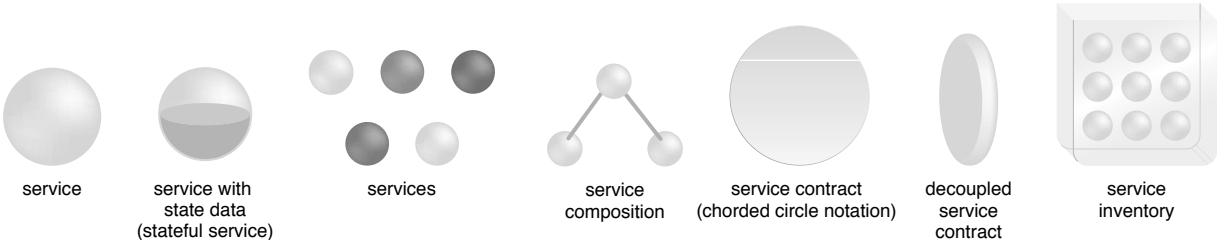
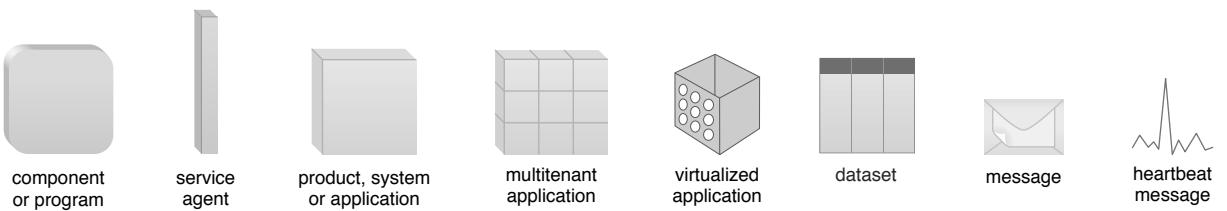
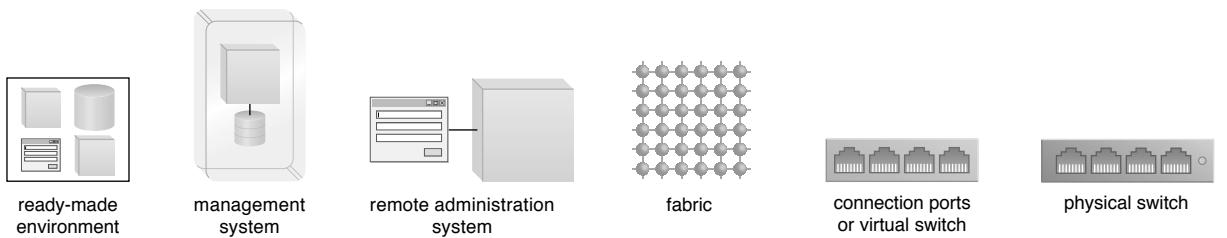
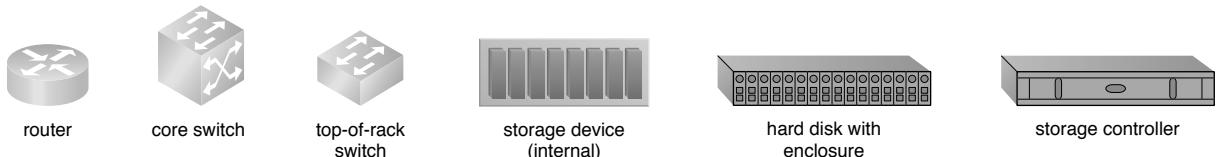
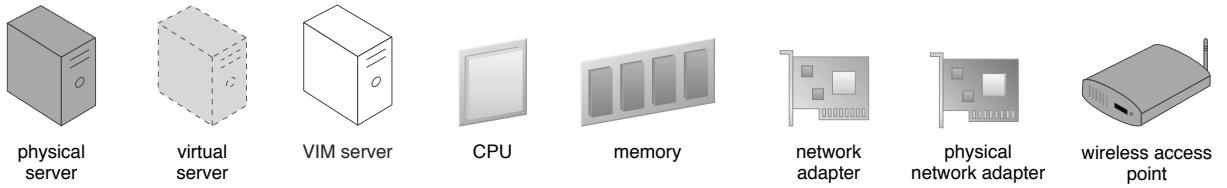
 PRENTICE HALL

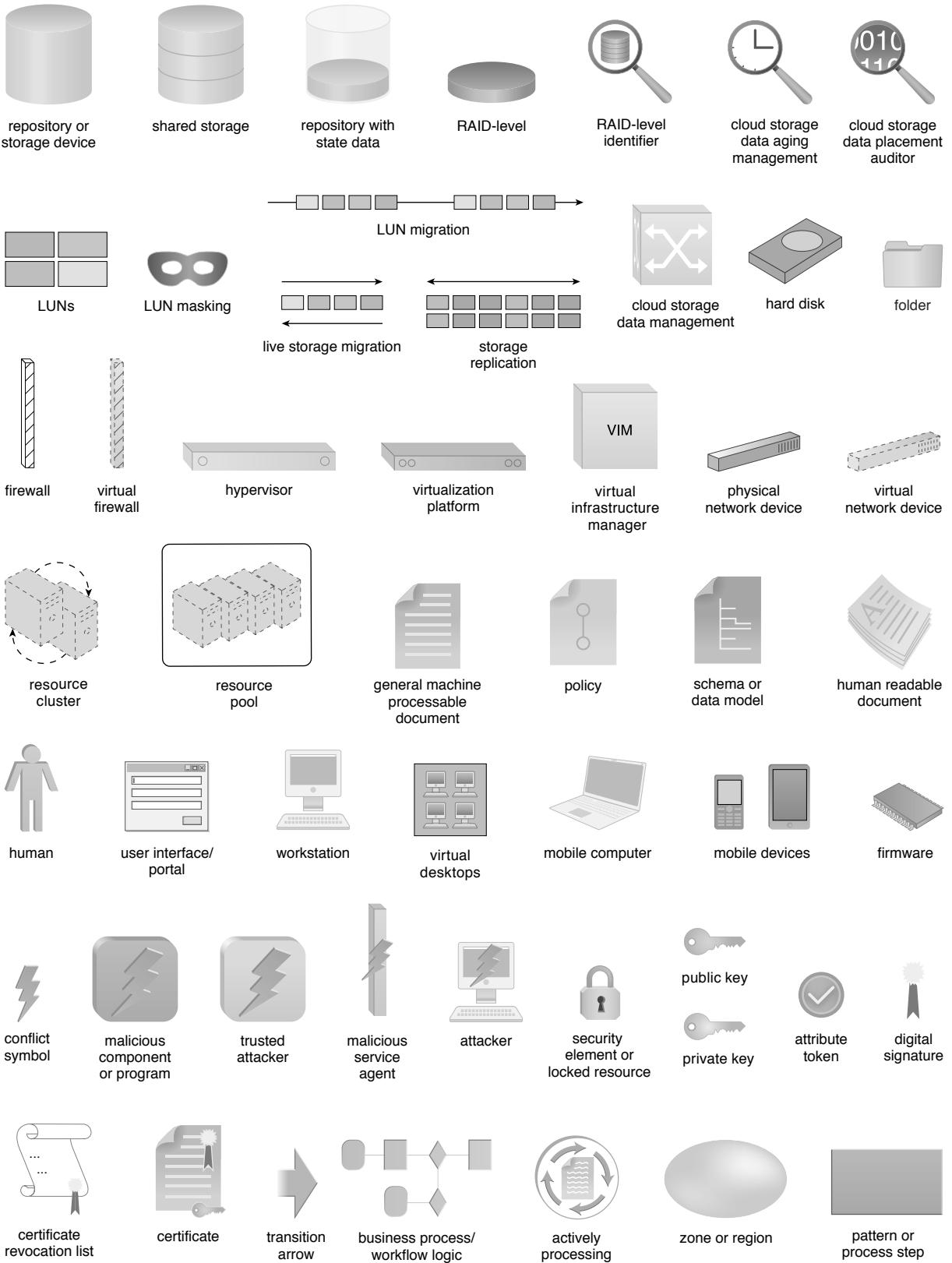
ServiceTech
O O C PRESS

FREE SAMPLE CHAPTER

SHARE WITH OTHERS







Praise for This Book

“Thomas Erl’s text provides a unique and comprehensive perspective on cloud design patterns that is clearly and concisely explained for the technical professional and layman alike. It is an informative, knowledgeable, and powerful insight that may guide cloud experts in achieving extraordinary results based on extraordinary expertise identified in this text. I will use this text as a resource in future cloud designs and architectural considerations.”

—*Dr. Nancy M. Landreville, CEO/CISO, NML Computer Consulting*

“This book continues the very high standard we have come to expect from ServiceTech Press. The book provides well-explained vendor-agnostic patterns to the challenges of providing or using cloud solutions from PaaS to SaaS. The book is not only a great patterns reference, but also worth reading from cover to cover as the patterns are thought-provoking, drawing out points that you should consider and ask of a potential vendor if you’re adopting a cloud solution.”

—*Phil Wilkins, Enterprise Integration Architect, Specsavers*

“This book provides an excellent read for anyone wanting to grasp the fundamentals and advanced concepts of cloud computing. The easy-to-understand format provides the reader with a clear direction on how to enable a more robust, dynamic, and efficient cloud environment while also providing vital information on how to effectively secure core components of the cloud. The reader, who might not have a full understanding of cybersecurity implications as they relate to cloud, will have the foundational knowledge to build out secure cloud environments. I would recommend this book to anyone serious about cloud security.”

—*Sean Cope, CISSP CEH CNDA, FedRAMP Assessment Lead,
Homeland Security Consultants*

“A very well written book, providing details of how to achieve the characteristics of a cloud and hence enable businesses to achieve its benefits.”

—*Kumail Morawala, CCP Certified Trainer*

“Cloud Computing Design Patterns is an excellent book to use when building or maintaining your cloud. The book is vendor neutral, which ensures that there are no conflicts of interest as far as the authors and publisher go. I think that the diagrams and illustrations are particularly helpful since some people seem challenged with trying to visualize virtual machines.”

—Laura Taylor, Relevant Technologies

“Cloud Computing Design Patterns takes a disciplined approach to categorizing cloud design building blocks and simplifying inherent technology complexities. It explains, in a lucid manner, why a particular design pattern is needed and how to approach a pertinent solution. I found the security patterns sections more versatile in covering examples, such as hypervisor attack vectors, threat mitigation strategies, and mobile device management security. Written in a catalog style, this book takes you through a journey of development that is intuitive as well comprehensive enough.”

—Anant Mahajan

“Readers will find it easy to read, comprehend, and apply the cloud pattern principles in practice that have already been adopted by the industry.”

—Matt Lorrain, Greg Ponto, and Michael E. Young,
Security Standards & Architecture team, Esri

“The models seem to be consistent and thorough, which should make them approachable and of value in scoping the design of reliable implementations. Overall, this is a good basis for progressing a common understanding of the vision of cloud practice—well done.”

—Tom Cleary, Australian Computer Society (ACS)

Cloud Computing Design Patterns

Thomas Erl,
Robert Cope,
Amin Naserpour



PRENTICE HALL
NEW YORK • BOSTON • INDIANAPOLIS • SAN FRANCISCO
TORONTO • MONTREAL • LONDON • MUNICH • PARIS • MADRID
CAPE TOWN • SYDNEY • TOKYO • SINGAPORE • MEXICO CITY

ServiceTech
○●◎ PRESS

Many of the designations used by manufacturers and sellers to distinguish their products are claimed as trademarks. Where those designations appear in this book, and the publisher was aware of a trademark claim, the designations have been printed with initial capital letters or in all capitals.

The authors and publisher have taken care in the preparation of this book, but make no expressed or implied warranty of any kind and assume no responsibility for errors or omissions. No liability is assumed for incidental or consequential damages in connection with or arising out of the use of the information or programs contained herein.

For information about buying this title in bulk quantities, or for special sales opportunities (which may include electronic versions; custom cover designs; and content particular to your business, training goals, marketing focus, or branding interests), please contact our corporate sales department at corpsales@pearsoned.com or (800) 382-3419.

For government sales inquiries, please contact
governmentsales@pearsoned.com.

For questions about sales outside the U.S., please contact
international@pearsoned.com.

Visit us on the Web: informati.com/ph

Library of Congress Control Number: 2015935087

Copyright © 2015 Arcitura Education Inc.

All rights reserved. Printed in the United States of America. This publication is protected by copyright, and permission must be obtained from the publisher prior to any prohibited reproduction, storage in a retrieval system, or transmission in any form or by any means, electronic, mechanical, photocopying, recording, or likewise. To obtain permission to use material from this work, please submit a written request to Pearson Education, Inc., Permissions Department, 200 Old Tappan Road, Old Tappan, New Jersey 07675, or you may fax your request to (201) 236-3290.

ISBN-13: 978-0-13-385856-3

ISBN-10: 0-13-385856-1

Text printed in the United States on recycled paper at Courier in Westford, Massachusetts.

First printing: June 2015

Editor-in-Chief

Mark L. Taub

Senior Acquisitions Editor

Trina MacDonald

Development Editors

Natalie Gitt

Maria Lee

Managing Editor

Kristy Hart

Senior Project Editor

Betsy Gratner

Copy Editors

Natalie Gitt

Maria Lee

Senior Indexer

Cheryl Lenser

Proofreaders

Natalie Gitt

Maria Lee

Debbie Williams

Publishing Coordinator

Olivia Basegio

Cover Designer

Thomas Erl

Compositor

Bumpy Design

Graphics

Jasper Paladino

Educational Content Development

Arcitura Education Inc.

To our new addition.

—Thomas Erl

*To my wife, Erin, for her patient support
and our sons, Sean and Troy.*

—Robert Cope

This page intentionally left blank

Contents at a Glance

CHAPTER 1: Introduction	1
CHAPTER 2: Understanding Design Patterns.....	9
CHAPTER 3: Sharing, Scaling and Elasticity Patterns	15
CHAPTER 4: Reliability, Resiliency and Recovery Patterns	97
CHAPTER 5: Data Management and Storage Device Patterns	167
CHAPTER 6: Virtual Server and Hypervisor Connectivity and Management Patterns	221
CHAPTER 7: Monitoring, Provisioning and Administration Patterns.....	283
CHAPTER 8: Cloud Service and Storage Security Patterns.....	335
CHAPTER 9: Network Security, Identity & Access Management and Trust Assurance Patterns	395
CHAPTER 10: Common Compound Patterns	471
APPENDIX A: Cloud Computing Mechanisms Glossary	511
APPENDIX B: Alphabetical Design Patterns Reference	535
About the Authors	541
Index	543

This page intentionally left blank

Contents

CHAPTER 1: Introduction	1
Objective of This Book	2
What This Book Does Not Cover	2
Who This Book Is For	2
Origin of This Book	3
Recommended Reading	3
How This Book Is Organized	3
Chapter 3: Sharing, Scaling and Elasticity Patterns	4
Chapter 4: Reliability, Resiliency and Recovery Patterns	4
Chapter 5: Data Management and Storage Device Patterns	4
Chapter 6: Virtual Server and Hypervisor Connectivity and Management Patterns	4
Chapter 7: Monitoring, Provisioning and Administration Patterns	4
Chapter 8: Cloud Service and Storage Security Patterns	4
Chapter 9: Network Security, Identity & Access Management and Trust Assurance Patterns	4
Chapter 10: Common Compound Patterns	5
Appendix A: Cloud Computing Mechanisms Glossary	5
Appendix B: Alphabetical Design Patterns Reference	5
Additional Information	5
Symbol Legend	5
Pattern Documentation Conventions	5
Updates, Errata, and Resources (www.servicetechbooks.com)	6
Cloud Computing Design Patterns (www.cloudpatterns.org)	6
What Is Cloud? (www.whatiscloud.com)	6
Referenced Specifications (www.servicetechspecs.com)	6
The Service Technology Magazine (www.servicetechmag.com)	6
CloudSchool.com™ Certified Cloud (CCP) Professional (www.cloudschool.com)	6
Social Media and Notification	7

CHAPTER 2: Understanding Design Patterns 9

About Pattern Profiles	11
Requirement	11
Icon	11
Problem	11
Solution	12
Application	12
Mechanisms	12
About Compound Patterns	12
Design Pattern Notation.....	13
Capitalization	13
Page Number References	13
Measures of Design Pattern Application	13
Working with This Catalog	14

CHAPTER 3: Sharing, Scaling and Elasticity Patterns 15

Shared Resources.....	17
Problem	17
Solution	18
Application	19
Mechanisms	21
Workload Distribution	22
Problem	22
Solution	22
Application	22
Mechanisms	24
Dynamic Scalability	25
Problem	25
Solution	27
Application	28
Mechanisms	31

Service Load Balancing	32
Problem	32
Solution	33
Application	34
Mechanisms.....	36
Elastic Resource Capacity	37
Problem	37
Solution	37
Application	38
Mechanisms.....	40
Elastic Network Capacity	42
Problem	42
Solution	43
Application	43
Mechanisms.....	43
Elastic Disk Provisioning	45
Problem	45
Solution	46
Application	48
Mechanisms.....	49
Load Balanced Virtual Server Instances.....	51
Problem	51
Solution	52
Application	53
Mechanisms.....	55
Load Balanced Virtual Switches	57
Problem	57
Solution	58
Application	58
Mechanisms.....	60
Service State Management	61
Problem	61
Solution	61
Application	62
Mechanisms.....	63

Storage Workload Management	64
Problem	64
Solution	64
Application	66
Mechanisms	69
Dynamic Data Normalization	71
Problem	71
Solution	72
Application	72
Mechanisms	73
Cross-Storage Device Vertical Tiering	74
Problem	74
Solution	76
Application	76
Mechanisms	79
Intra-Storage Device Vertical Data Tiering	81
Problem	81
Solution	81
Application	82
Mechanisms	85
Memory Over-Committing	86
Problem	86
Solution	87
Application	88
Mechanisms	89
NIC Teaming	90
Problem	90
Solution	90
Application	91
Mechanisms	92
Broad Access	93
Problem	93
Solution	93
Application	94
Mechanisms	94

CHAPTER 4: Reliability, Resiliency and Recovery Patterns	97
Resource Pooling	99
Problem	99
Solution	99
Application	100
Mechanisms.....	103
Resource Reservation.....	106
Problem	106
Solution	107
Application	107
Mechanisms.....	110
Hypervisor Clustering.....	112
Problem	112
Solution	112
Application	114
Mechanisms.....	117
Redundant Storage	119
Problem	119
Solution	121
Application	121
Mechanisms.....	122
Dynamic Failure Detection and Recovery.....	123
Problem	123
Solution	123
Application	123
Mechanisms.....	126
Multipath Resource Access	127
Problem	127
Solution	128
Application	129
Mechanisms.....	131

Redundant Physical Connection for Virtual Servers	132
Problem	132
Solution	133
Application	134
Mechanisms	136
Synchronized Operating State	138
Problem	138
Solution	138
Application	139
Mechanisms	142
Zero Downtime	143
Problem	143
Solution	143
Application	144
Mechanisms	144
Storage Maintenance Window	147
Problem	147
Solution	148
Application	148
Mechanisms	154
Virtual Server Auto Crash Recovery	155
Problem	155
Solution	156
Application	157
Mechanisms	158
Non-Disruptive Service Relocation	159
Problem	159
Solution	160
Application	160
Mechanisms	164

CHAPTER 5: Data Management and Storage	
Device Patterns	167
Direct I/O Access.....	169
Problem	169
Solution	169
Application	169
Mechanisms.....	171
Direct LUN Access	173
Problem	173
Solution	174
Application	174
Mechanisms.....	176
Single Root I/O Virtualization	178
Problem	178
Solution	179
Application	179
Mechanisms.....	180
Cloud Storage Data at Rest Encryption	181
Problem	181
Solution	182
Application	182
Mechanisms.....	183
Cloud Storage Data Lifecycle Management	184
Problem	184
Solution	185
Application	185
Mechanisms.....	186
Cloud Storage Data Management	187
Problem	187
Solution	188
Application	188
Mechanisms.....	189

Cloud Storage Data Placement Compliance Check	190
Problem	190
Solution	191
Application	191
Mechanisms	192
Cloud Storage Device Masking	194
Problem	194
Solution	194
Application	195
Mechanisms	197
Cloud Storage Device Path Masking	198
Problem	198
Solution	198
Application	199
Mechanisms	200
Cloud Storage Device Performance Enforcement	201
Problem	201
Solution	202
Application	202
Mechanisms	203
Virtual Disk Splitting	204
Problem	204
Solution	205
Application	206
Mechanisms	209
Sub-LUN Tiering	210
Problem	210
Solution	210
Application	211
Mechanisms	213
RAID-Based Data Placement	214
Problem	214
Solution	214
Application	215
Mechanisms	217

IP Storage Isolation	218
Problem	218
Solution	218
Application	218
Mechanisms	220
CHAPTER 6: Virtual Server and Hypervisor Connectivity and Management Patterns	221
 Virtual Server Folder Migration	223
Problem	223
Solution	225
Application	225
Mechanisms	226
 Persistent Virtual Network Configuration	227
Problem	227
Solution	227
Application	228
Mechanisms	229
 Virtual Server Connectivity Isolation	231
Problem	231
Solution	232
Application	233
Mechanisms	234
 Virtual Switch Isolation	235
Problem	235
Solution	236
Application	236
Mechanisms	238
 Virtual Server NAT Connectivity	240
Problem	240
Solution	240
Application	240
Mechanisms	243

External Virtual Server Accessibility	244
Problem	244
Solution	245
Application	245
Mechanisms	246
Cross-Hypervisor Workload Mobility	247
Problem	247
Solution	248
Application	250
Mechanisms	250
Virtual Server-to-Host Affinity	252
Problem	252
Solution	253
Application	254
Mechanisms	257
Virtual Server-to-Host Anti-Affinity	258
Problem	258
Solution	261
Application	261
Mechanisms	264
Virtual Server-to-Host Connectivity	265
Problem	265
Solution	266
Application	266
Mechanisms	266
Virtual Server-to-Virtual Server Affinity	267
Problem	267
Solution	269
Application	269
Mechanisms	271
Virtual Server-to-Virtual Server Anti-Affinity	272
Problem	272
Solution	275
Application	275
Mechanisms	277

Stateless Hypervisor	278
Problem	278
Solution	278
Application	279
Mechanisms.....	282
CHAPTER 7: Monitoring, Provisioning and Administration Patterns.....	283
 Usage Monitoring	285
Problem	285
Solution	285
Application	286
Mechanisms.....	287
 Pay-as-You-Go.....	288
Problem	288
Solution	288
Application	289
Mechanisms.....	291
 Realtime Resource Availability	292
Problem	292
Solution	292
Application	293
Mechanisms.....	294
 Rapid Provisioning	295
Problem	295
Solution	296
Application	296
Mechanisms.....	299
 Platform Provisioning	301
Problem	301
Solution	301
Application	302
Mechanisms.....	304

Bare-Metal Provisioning	305
Problem	305
Solution	305
Application	305
Mechanisms	308
Automated Administration	310
Problem	310
Solution	310
Application	311
Mechanisms	314
Centralized Remote Administration	315
Problem	315
Solution	317
Application	317
Mechanisms	318
Resource Management	320
Problem	320
Solution	320
Application	321
Mechanisms	323
Self-Provisioning	324
Problem	324
Solution	325
Application	325
Mechanisms	329
Power Consumption Reduction	330
Problem	330
Solution	330
Application	331
Mechanisms	334

CHAPTER 8: Cloud Service and Storage Security Patterns	335
Trusted Platform BIOS	337
Problem	337
Solution	338
Application	339
Mechanisms	340
Geotagging	341
Problem	341
Solution	341
Application	342
Mechanisms	343
Hypervisor Protection	344
Problem	344
Solution	346
Application	347
Mechanisms	349
Cloud VM Platform Encryption	350
Problem	350
Solution	350
Application	352
Mechanisms	353
Trusted Cloud Resource Pools	354
Problem	354
Solution	354
Application	356
Mechanisms	358
Secure Cloud Interfaces and APIs	360
Problem	360
Solution	361
Application	361
Mechanisms	363

Cloud Resource Access Control	364
Problem	364
Solution	366
Application	368
Mechanisms	368
Detecting and Mitigating User-Installed VMs	369
Problem	369
Solution	371
Application	372
Mechanisms	374
Mobile BYOD Security	376
Problem	376
Solution	378
Application	380
Mechanisms	381
Cloud Data Breach Protection	382
Problem	382
Solution	384
Application	384
Mechanisms	386
Permanent Data Loss Protection	387
Problem	387
Solution	388
Application	389
Mechanisms	390
In-Transit Cloud Data Encryption	391
Problem	391
Solution	391
Application	392
Mechanisms	394

CHAPTER 9: Network Security, Identity & Access Management and Trust Assurance Patterns	395
Secure On-Premise Internet Access	397
Problem	397
Solution	398
Application	400
Mechanisms	403
Secure External Cloud Connection	404
Problem	404
Solution	404
Application	405
Mechanisms	408
Secure Connection for Scaled VMs	409
Problem	409
Solution	412
Application	414
Mechanisms	415
Cloud Denial-of-Service Protection	416
Problem	416
Solution	418
Application	419
Mechanisms	420
Cloud Traffic Hijacking Protection	421
Problem	421
Solution	423
Application	423
Mechanisms	424
Automatically Defined Perimeter	425
Problem	425
Solution	426
Application	427
Mechanisms	429

Cloud Authentication Gateway	430
Problem	430
Solution	431
Application	432
Mechanisms	435
Federated Cloud Authentication	436
Problem	436
Solution	438
Application	439
Mechanisms	443
Cloud Key Management	444
Problem	444
Solution	445
Application	446
Mechanisms	447
Trust Attestation Service	448
Problem	448
Solution	449
Application	449
Mechanisms	451
Collaborative Monitoring and Logging	452
Problem	452
Solution	455
Application	455
Mechanisms	459
Independent Cloud Auditing	460
Problem	460
Solution	461
Application	463
Mechanisms	464
Threat Intelligence Processing	465
Problem	465
Solution	466
Application	468
Mechanisms	469

CHAPTER 10: Common Compound Patterns	471
“Compound Pattern” vs. “Composite Pattern”	472
Compound Pattern Members	472
Joint Application vs. Coexistent Application	472
Private Cloud	474
Public Cloud	476
Software-as-a-Service (SaaS)	478
Platform-as-a-Service (PaaS)	480
Infrastructure-as-a-Service (IaaS)	482
Elastic Environment	484
Multitenant Environment	486
Resilient Environment	490
Cloud Bursting	492
Burst Out to Private Cloud	493
Burst Out to Public Cloud	496
Burst In	499
Secure Burst Out to Private Cloud/Public Cloud	501
Cloud Balancing	503
Cloud Authentication	505
Resource Workload Management	506
Isolated Trust Boundary	508
APPENDIX A: Cloud Computing Mechanisms Glossary ..	511
APPENDIX B: Alphabetical Design Patterns Reference	535
About the Authors	541
Index	543

This page intentionally left blank

Acknowledgments

- Khaja Ahmed, Amazon
- Wayne Armour, Armoured Networks
- Khalid Asad, IBM
- Alenka Brown, Mcclure Brown
- Antonio Bruno, Arcitura Certified Trainer
- Tom Cleary, Australian Computer Society
- Sean Cope, Homeland Security Consultants
- Damian Crosby, RMS
- Michael Dance Jr., BAH
- Michael Fulton, Proctor & Gamble
- Leszek Jaskierny, HP
- Clint Johnson, SJRB
- Ernest Kim, MITRE
- Michael J. Kristan, MITRE
- Vivek Kumar, Yahoo
- Dr. Nancy M. Landreville, NML Computer Consulting
- Matt Lorrain, Esri
- Kathleen Lynch
- Anant Mahajan
- Ahmad Manzoor, Advanced Global Communications Networks
- Kumail Morawala, CCP Certified Trainer
- Vasudevan Narayanan, IBM
- Bob Natale, MITRE
- Sharon Orser-Jackson, MITRE
- Greg Ponto, Esri
- Yves Roycie, keepmomentum
- Scott Rush, HP
- Vijay Srinivasan, Cognizant
- Umit Tacay, silverplatypus
- Laura Taylor, Relevant Technologies
- Katy Warren, MITRE
- Phil Wilkins, Specsavers
- Michael E. Young, Esri

Special thanks to the Arcitura Education CloudSchool.com research and development team that produced the Cloud Certified Professional (CCP) course modules upon which this book is based.

This page intentionally left blank

This page intentionally left blank

Chapter 4



Reliability, Resiliency and Recovery Patterns

Resource Pooling
Resource Reservation
Hypervisor Clustering
Redundant Storage
Dynamic Failure Detection and Recovery
Multipath Resource Access

Redundant Physical Connection for Virtual Servers
Synchronized Operating State
Zero Downtime
Storage Maintenance Window
Virtual Server Auto Crash Recovery
Non-Disruptive Service Relocation

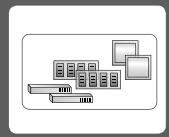
Contingency planning efforts for continuity of operations and disaster recovery are concerned with designing and implementing cloud architectures that provide run-time reliability, operational resiliency, and automated recovery when interruptions are encountered, regardless of origin.

The patterns in this chapter address different aspects of these requirements. Starting with foundational patterns, such as Resource Pooling (99), Resource Reservation (106), Hypervisor Clustering (112), and Redundant Storage (119), which address basic failover and availability demands, the chapter continues with more specialized and complex patterns, such as Dynamic Failure Detection and Recovery (123) and Zero Downtime (143), which establish resilient cloud architectures that act as pillars for enterprise cloud solutions.

It is also worth noting that this set of patterns establishes and contributes to the availability leg of the security triad of confidentiality, integrity, and availability and is further complemented by several cloud security patterns in Chapters 8 and 9 in maximizing the reliability and resiliency potential by protecting against attacks that can compromise the availability of an organization's cloud-hosted IT resources.

Resource Pooling

How can IT resources be organized to support dynamic sharing?



Problem	When sharing identical IT resources for scalability purposes, it can be error-prone and burdensome to keep them fully synchronized on an on-going basis.
Solution	An automated synchronization system is provided to group identical IT resources into pools and to maintain their synchronicity.
Application	Resource pools can be created at different sizes and further organized into hierarchies to provide parent and child pools.
Mechanisms	Audit Monitor, Cloud Storage Device, Cloud Usage Monitor, Hypervisor, Logical Network Perimeter, Pay-Per-Use Monitor, Remote Administration System, Resource Management System, Resource Replication, Virtual CPU, Virtual Infrastructure Manager (VIM), Virtual RAM, Virtual Server

Problem

When assembling identical IT resources for sharing and scalability purposes (such as when applying Shared Resources (17) and Dynamic Scalability (25)), the IT resources need to carefully be kept synchronized so that no one IT resource differs from another.

Manually establishing and maintaining the level of required synchronicity across collections of shared IT resources is challenging, effort-intensive and, most importantly, error-prone. Variances or disparity between shared IT resources can lead to inconsistent runtime behavior and cause numerous types of runtime exceptions.

Solution

Identical IT resources are grouped into resource pools and maintained by a system that automatically ensures they remain synchronized (Figure 4.1). The following items are commonly pooled:

- physical servers
- virtual servers
- cloud storage devices

- internetwork and networking devices
- CPUs
- memory (RAM)

Dedicated pools can be created for each of these items, or respective pools can be further grouped into a larger pool (in which case each individual pool becomes a sub-pool).

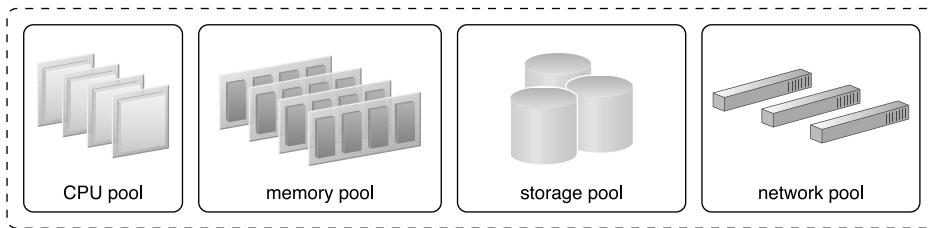


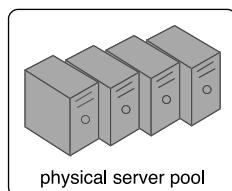
Figure 4.1

A sample resource pool comprised of four sub-pools of CPUs, memory, cloud storage devices, and virtual network devices.

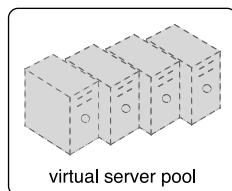
Application

As stated previously, this pattern is primarily applied in support of Shared Resources (17) and Dynamic Scalability (25) in order to establish a reliable system of shared IT resource synchronization. The Resource Pooling pattern itself can be further supported by the application of Resource Reservation (106).

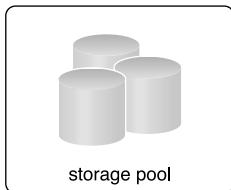
Provided here are common examples of resource pools:



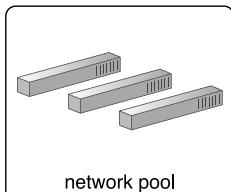
Physical server pools composed of ready-to-go, networked servers installed with operating systems and any other necessary programs or applications.



Virtual server pools are usually configured using templates that cloud consumers can choose from, such as a pool of mid-tier Windows servers with 4 GBs of RAM or a pool of low-tier Ubuntu servers with 2 GBs of RAM.



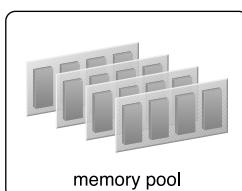
Storage pools (or cloud storage device pools) that consist of file-based or block-based storage structures. Storage pools can contain empty or filled cloud storage devices. Often storage resource pools will take advantage of LUNs.



Network pools (or interconnect pools) are composed of different, preconfigured network connectivity devices. For example, a pool of virtual firewall devices or physical network switches can be created for redundant connectivity, load balancing, or link aggregation.



CPU pools are ready to be allocated to virtual servers. These are often broken down into individual processing cores (as opposed to pooling entire CPUs).



Pools of physical RAM that can be used in newly provisioned physical servers or to vertically scale physical servers.

Resource pools can grow to become complex, with multiple pools created for specific cloud consumers or applications. To help with the organization of diverse resource pools, a hierarchical structure can be established to create parent, sibling, and nested pools.

Sibling resource pools are normally drawn from the same collection of physical IT resources (as opposed to IT resources spread out over different data centers) and are isolated from one another so that each cloud consumer is only provided access to its respective pool (Figure 4.2).

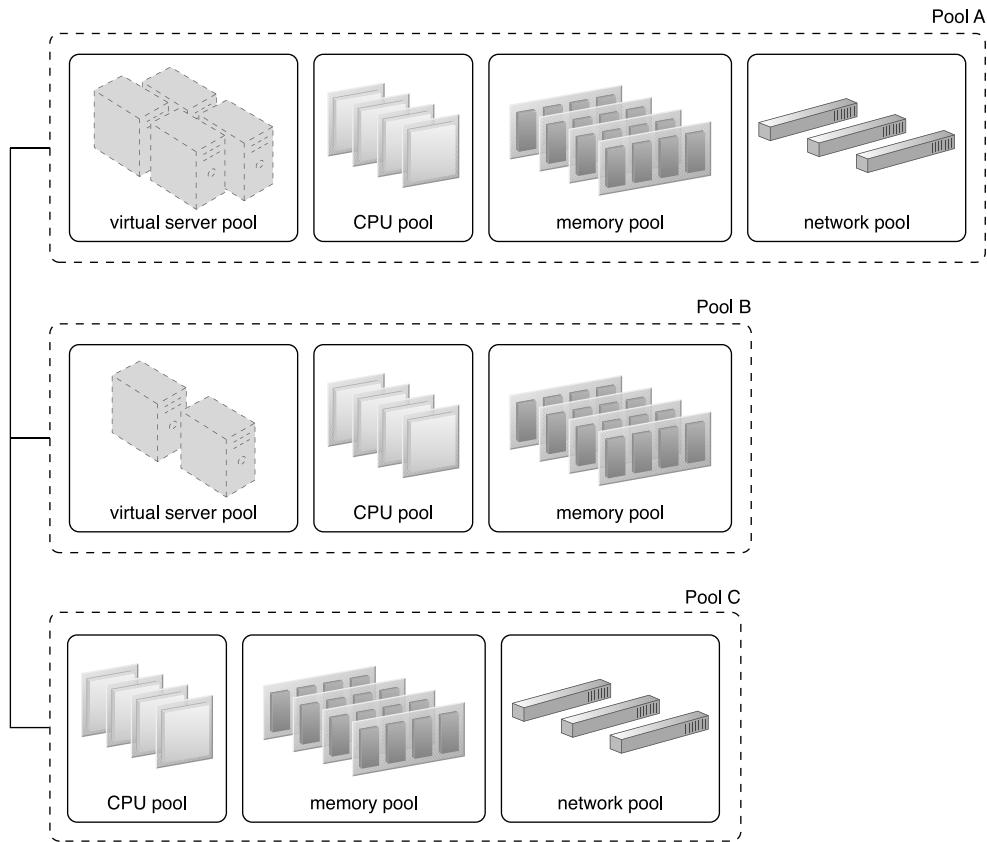


Figure 4.2

Pools B and C are sibling pools taken from the larger Pool A that has been allocated to a cloud consumer. This is an alternative to taking the IT resources for Pool B and Pool C from a general reserve of IT resources that is shared throughout the cloud.

In the nested pool model, larger pools are divided into smaller pools of the same kind (Figure 4.3). Nested pools can be used to assign resource pools to different departments or groups within the same cloud consumer organization.

After resources pools have been defined, multiple instances of IT resources from each pool can be created to provide an in-memory pool of “live” IT resources.

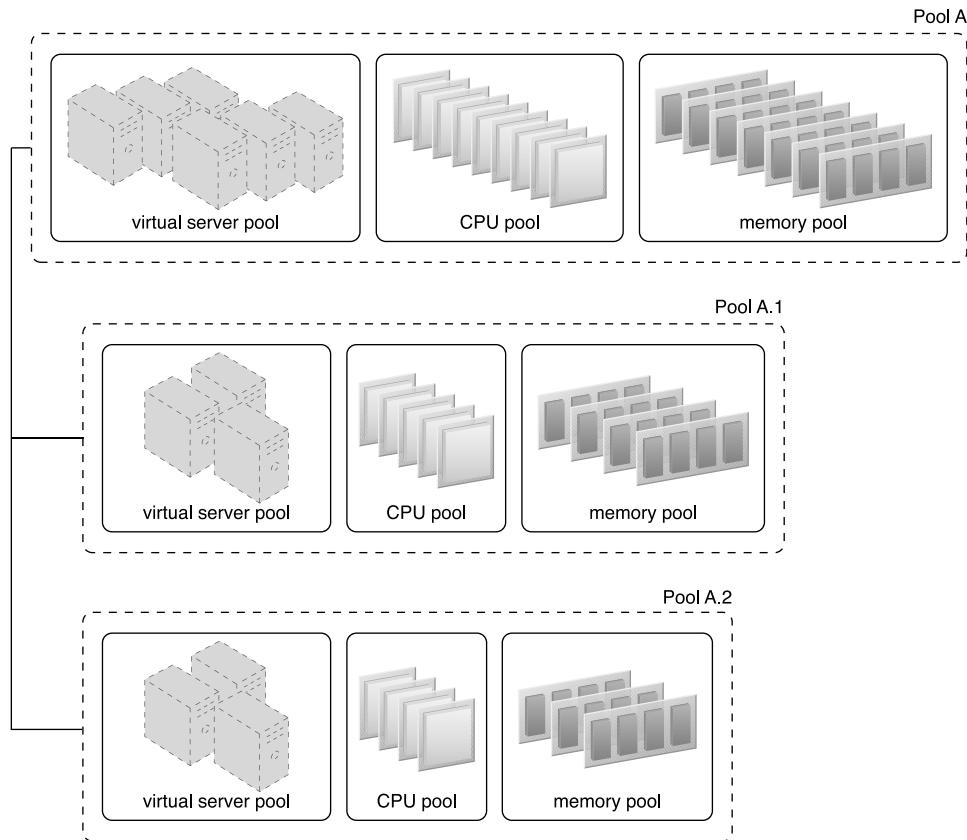


Figure 4.3

Nested Pools A.1 and A.2 are comprised of the same IT resources as Pool A, but in different quantities. Nested pools are generally used to provision cloud services that are rapidly instantiated using the same kind of IT resources with the same configuration settings.

Mechanisms

- *Audit Monitor* – This mechanism monitors resource pool usage to ensure compliance with privacy and regulation requirements, especially when pools include cloud storage devices or data loaded into memory.
- *Cloud Storage Device* – Cloud storage devices are commonly pooled as a result of the application of this pattern.

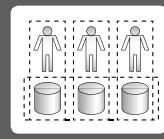
- *Cloud Usage Monitor* – Various cloud usage monitors can be involved with the run-time tracking and synchronization required by IT resources within pools and by the systems managing the resource pools themselves.
- *Hypervisor* – The hypervisor mechanism is responsible for providing virtual servers with access to resource pools, and hosting virtual servers and sometimes the resource pools themselves. Hypervisors further can distribute physical computing capacity between the virtual servers based on each virtual server's configuration and priority.
- *Logical Network Perimeter* – The logical network perimeter can be used to logically organize and isolate the resource pools.
- *Pay-Per-Use Monitor* – The pay-per-use monitor collects usage and billing information in relation to how individual cloud consumers use and are allocated IT resources from various pools.
- *Remote Administration System* – This mechanism is commonly used to interface with backend systems and programs in order to provide resource pool administration features via a front-end portal.
- *Resource Management System* – The resource management system mechanism supplies cloud consumers with the tools and permission management options to administer resource pools.
- *Resource Replication* – This mechanism can be used to generate new instances of IT resources for a given resource pool.
- *Virtual CPU* – This mechanism is used to allocate CPU to virtual servers, and also helps to determine whether a hypervisor's physical CPU is being over-utilized or a virtual server requires more CPU capacity. When a system has more than one CPU or when hypervisors belong to the same cluster, their total CPU capacity can be aggregated into a pool and leveraged by virtual servers.
- *Virtual Infrastructure Manager (VIM)* – This mechanism enables pools of resources to be created on individual hypervisors, and can also aggregate the capacity of multiple hypervisors into a pool from where virtual CPU and memory resources can be assigned to virtual servers.
- *Virtual RAM* – This mechanism is used to allocate memory to virtual servers, and to measure the memory utilization of hypervisors and virtual servers. When more than one hypervisor is present, a pool encompassing the combined memory

capacity of the hypervisors can be created. This mechanism is also used to identify whether more memory should be added to a virtual server.

- *Virtual Server* – This mechanism is associated with the Resource Pooling pattern in how virtual server hosted IT resources are provisioned and consumed by resource pools that are assigned to cloud consumers. Virtual servers themselves may also be pooled.

Resource Reservation

How can shared IT resources be protected from conflicts that can arise from concurrent access?



Problem	When two or more cloud service consumers attempt to instantiate the same shared IT resource, runtime conflicts can occur, including resource constraints due to lack of capacity.
Solution	A system is established whereby a portion of an IT resource (or one or more IT resources) is set aside exclusively for a given cloud service consumer.
Application	The resource management system is used to define IT resource thresholds and to restrict access to reserved IT resources.
Mechanisms	Audit Monitor, Cloud Storage Device, Cloud Usage Monitor, Hypervisor, Logical Network Perimeter, Remote Administration System, Resource Management System, Resource Replication, Virtual CPU, Virtual Infrastructure Manager (VIM), Virtual RAM, Virtual Server

Problem

When applying Shared Resources (17) and Resource Pooling (99) we can give multiple cloud service consumers access to the same IT resources. Depending on how IT resources are designed for shared usage and depending on their available levels of capacity, concurrent access can lead to a runtime exception condition called *resource constraint*.

A resource constraint is a condition that occurs when two or more cloud consumers have been allocated to share an IT resource that does not have the capacity to accommodate the entire processing requirements of the cloud consumers. As a result, one or more of the consumers will encounter degraded performance or be rejected altogether. The cloud service itself may go down, resulting in all cloud consumers being rejected.

Other types of runtime conflicts can occur when an IT resource (especially one not specifically designed to accommodate sharing) is concurrently accessed by different cloud service consumers. For example, nested and sibling resource pools introduce the notion of *resource borrowing*, whereby one pool can temporarily borrow IT resources from other pools. A runtime conflict can be triggered when the borrowed IT resource is not returned due to prolonged usage by the cloud service consumer that is borrowing it. This can inevitably lead back to the occurrence of resource constraints.

Solution

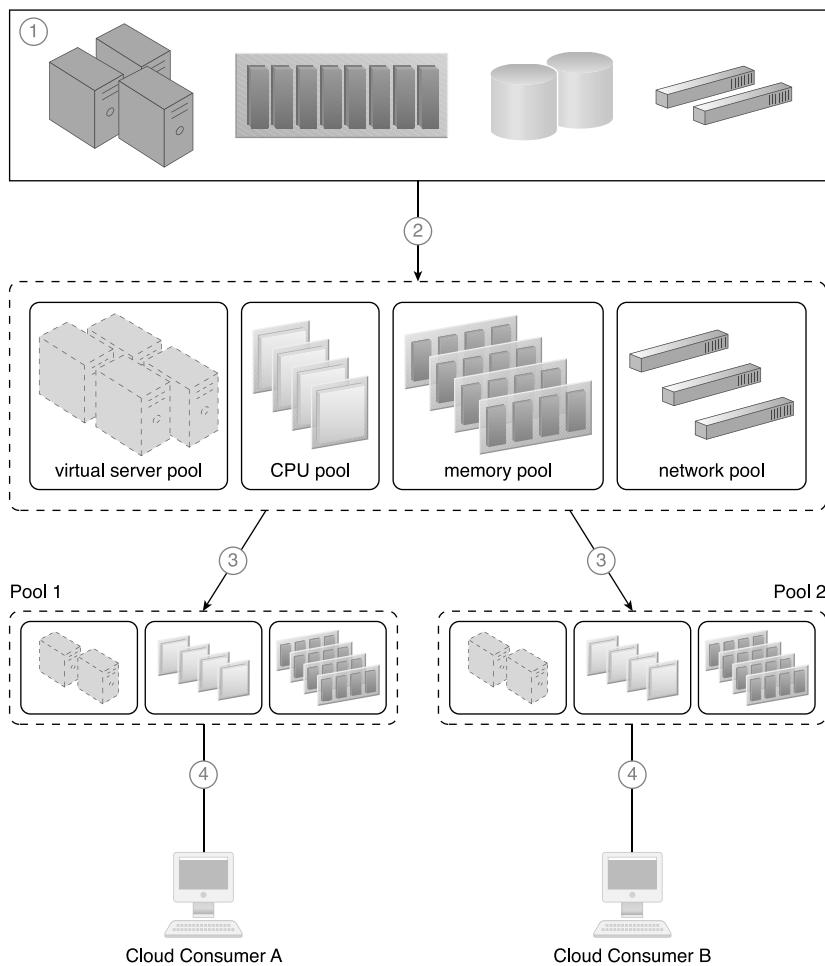
An IT resource reservation system is established to protect cloud service consumers (“tenants”) sharing the same underlying IT resources from each other. This system essentially guarantees a minimum amount of an IT resource for each cloud consumer by putting it aside and making it exclusively available only to the designated cloud service consumer. Potential conflicts, such as resource constraints and resource borrowing, are avoided because the reserved IT resources are never actually shared.

Application

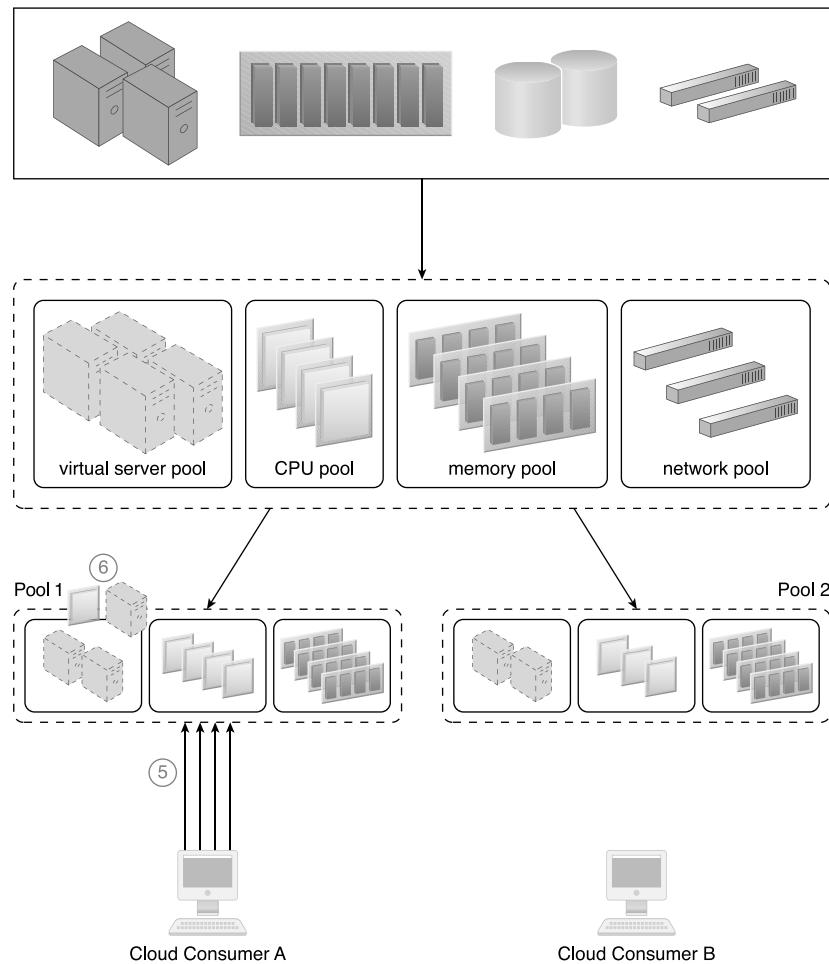
Creating an IT resource reservation system requires the involvement of the resource management system mechanism that can be used to define IT resource usage thresholds for individual IT resources and resource pools. Reservations are created to lock the amount of IT resources that each pool must keep. The balance of IT resources within a pool can still be shared (and borrowed).

The following steps are shown in Figures 4.4 to 4.6:

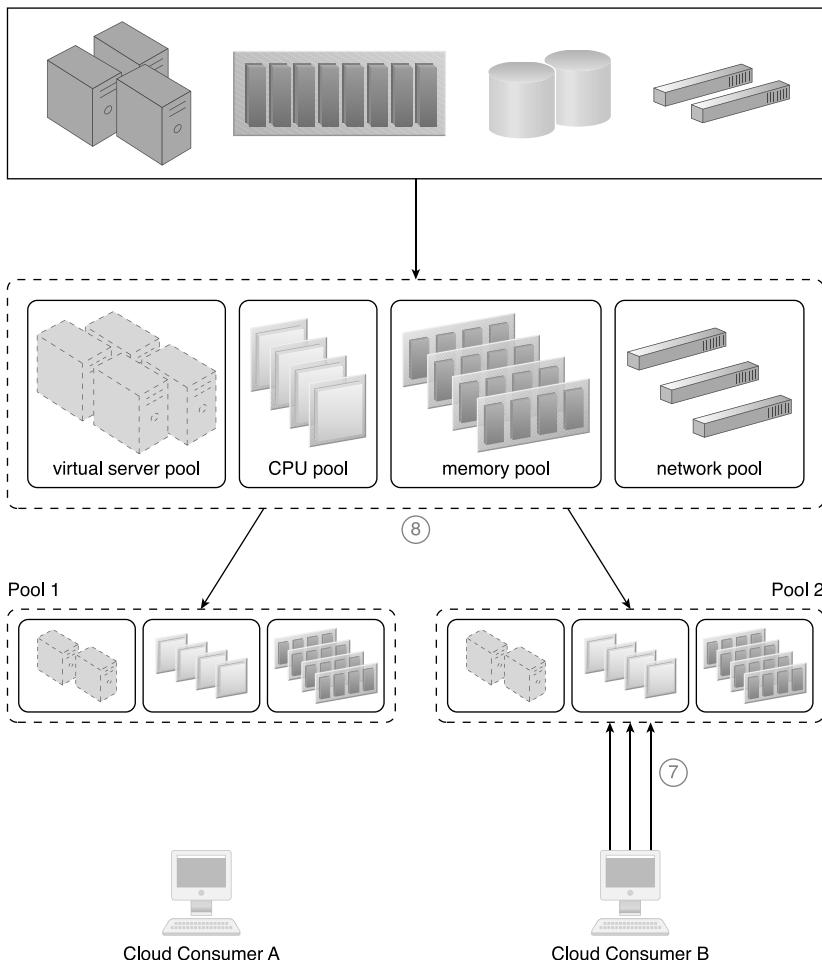
1. A physical resource group is created.
2. A parent resource pool is created from the physical resource group by applying Resource Pooling (99).
3. Two smaller child pools are created from the parent resource pool, and resource limits are defined using the resource management system mechanism.
4. Cloud consumers are provided with access to their own exclusive resource pools.
5. There is an increase in requests from Cloud Consumer A, resulting in more IT resources being allocated.
6. Consequently, some IT resources are borrowed from Pool 2. The amount of borrowed resources, however, is pre-determined by the resource limit defined in Step 3, which ensures that Cloud Consumer B will not face resource constraints.
7. Cloud Consumer B now imposes more requests and usage demands and may soon need to utilize all available IT resources in the pool.
8. The resource management system forces Pool 1 to release the IT resources and move them back to Pool 2 to become available for Cloud Consumer B.

**Figure 4.4**

A resource pool hierarchy to which an IT resource reservation system is applied (Part I).

**Figure 4.5**

A resource pool hierarchy to which an IT resource reservation system is applied (Part II).

**Figure 4.6**

A resource pool hierarchy to which an IT resource reservation system is applied (Part III).

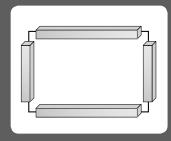
Mechanisms

- *Audit Monitor* – The audit monitor may be responsible for ensuring that the resource reservation system is acting in compliance with cloud consumer auditing, privacy, and other regulatory requirements. The audited information may also pertain to the geographical location of the reserved IT resources.
- *Cloud Storage Device* – This mechanism may be an IT resource reserved by this system.

- *Cloud Usage Monitor* – A cloud usage monitor may be involved with monitoring thresholds that trigger the allocation of reserved IT resources.
- *Hypervisor* – The hypervisor mechanism is associated with this pattern in its commitment to applying reservations for different cloud consumers in order to ensure that they are allocated the guaranteed amounts of IT resources. In support of this, it is responsible for locking, and pre-allocating the virtual servers' reserved computing capacity based on their configurations.
- *Logical Network Perimeter* – This mechanism ensures that reserved IT resources are made exclusively available to the appropriate cloud consumers.
- *Remote Administration System* – This system provides the tools necessary for custom front-ends to provide the administration controls necessary for cloud consumers to create and manage reserved IT resource allocations.
- *Resource Management System* – The resource management system provides essential features for managing IT resource reservations. These features may be encapsulated by a portal produced via the remote administration system mechanism.
- *Resource Replication* – The resource replication mechanism needs to stay updated on any cloud consumer IT resource consumption limitations to determine when new IT resource instances need to be replicated and provisioned.
- *Virtual CPU* – This mechanism is used to define the computing capacity a virtual server needs to have reserved. When this mechanism is used, a specific amount of CPU is explicitly allocated to each virtual server and not shared with other virtual servers.
- *Virtual Infrastructure Manager (VIM)* – This mechanism is used to configure the resources that are reserved for each virtual server.
- *Virtual RAM* – This mechanism is used to configure the memory that virtual servers need reserved and guaranteed. The memory that is reserved for each virtual server is not allocated to or shared with other virtual servers.
- *Virtual Server* – This mechanism hosts the reserved IT resources that are allocated.

Hypervisor Clustering

How can a virtual server survive the failure of its hosting hypervisor or physical server?



Problem	The failure of a hypervisor or its underlying physical server cascades to all hosted virtual servers further causing their hosted IT resources to fail.
Solution	Hypervisors are clustered across multiple physical servers, so that if one fails, active virtual servers are transferred to another.
Application	Heartbeat messages are passed between clustered hypervisors and a central VIM to maintain status monitoring. Shared storage is provided for the clustered hypervisors and further used to store virtual server disks.
Mechanisms	Cloud Storage Device, Hypervisor, Logical Network Perimeter, Resource Cluster, Resource Replication, Virtual Infrastructure Manager (VIM), Virtual Server, Virtual Switch, Virtualization Monitor

Problem

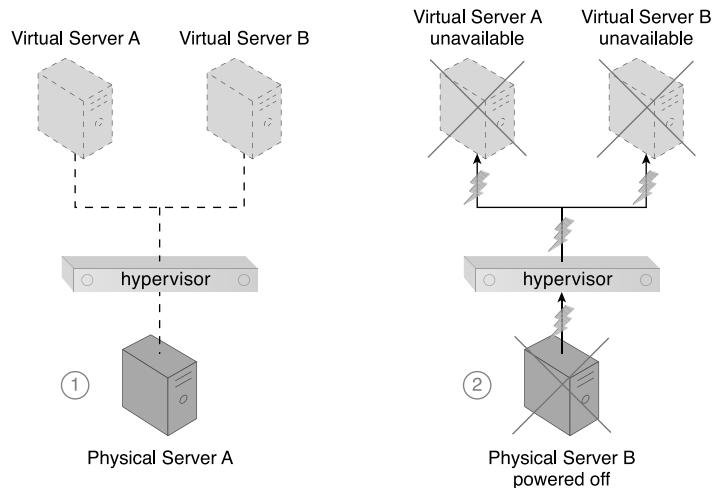
Virtual servers run on a hypervisor, and hardware resources are emulated for the virtual servers via the hypervisors. If the hypervisor fails or if the underlying physical server fails (thereby causing the hypervisor to fail), the failure condition cascades to all of its hosted virtual servers.

The following steps are shown in Figure 4.7:

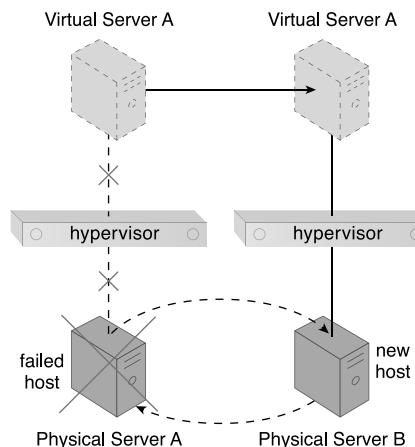
1. Physical Server A hosts a hypervisor that hosts Virtual Servers A and B.
2. When Physical Server A fails, the hypervisor and the two virtual servers fail as well.

Solution

A high-availability hypervisor cluster is created to establish a group of hypervisors that span physical servers. As a result, if a given physical server or hypervisor becomes unavailable, hosted virtual servers can be moved to another physical server or hypervisor (Figure 4.8).

**Figure 4.7**

Two virtual servers experience failure after their host physical server goes down.

**Figure 4.8**

Physical Server A becomes unavailable, thereby bringing down its hypervisor. Because the hypervisor is part of a cluster, Virtual Server A is migrated to a different host (Physical Server B), which has another hypervisor that is part of the same cluster.

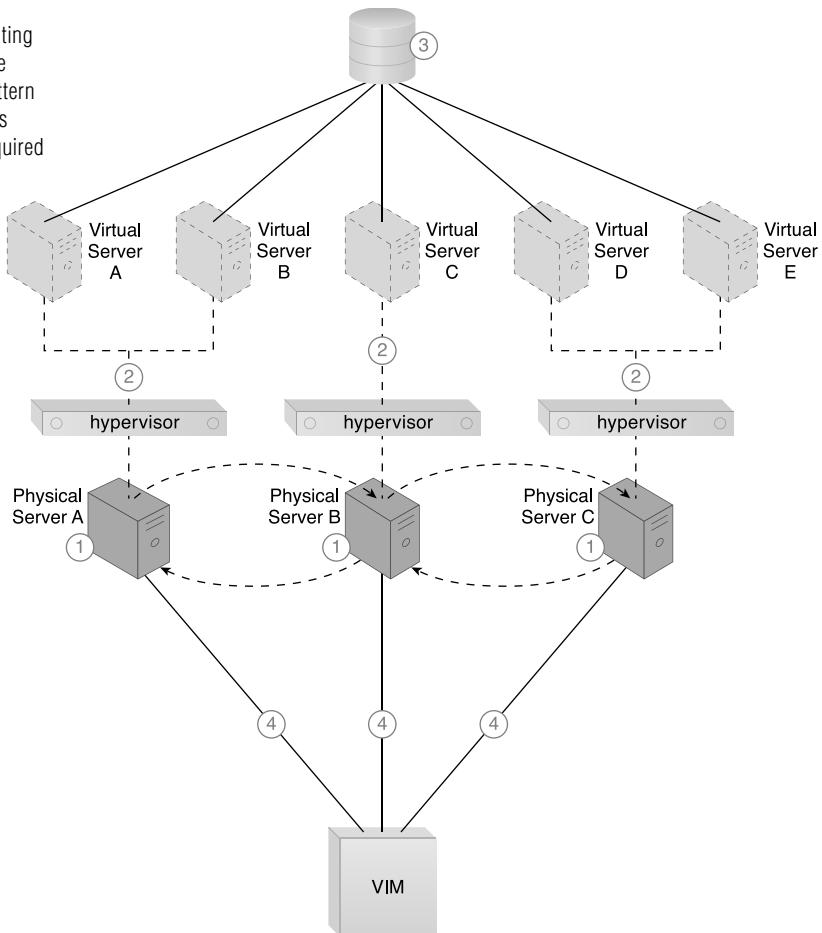
Application

A hypervisor cluster architecture is established and controlled via a central VIM, which sends regular heartbeat messages to the hypervisors to confirm that they are up and running. Any heartbeat messages that are not successfully acknowledged can lead the VIM to initiate the live VM migration program in order to dynamically move affected virtual servers to a new host. The hypervisor cluster utilizes a shared cloud storage device, which is used during the live migration of virtual servers by different hypervisors in the cluster.

Figures 4.9 to 4.12 provide examples of the results of applying the Hypervisor Clustering pattern, accompanied by numbered steps.

Figure 4.9

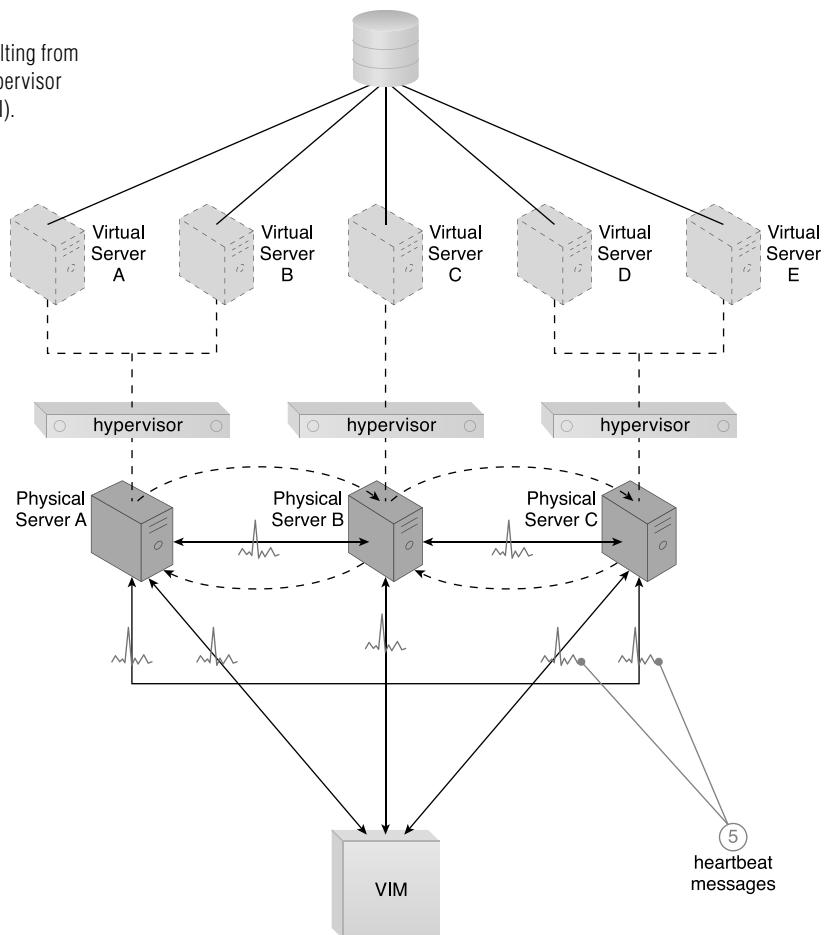
A cloud architecture resulting from the application of the Hypervisor Clustering pattern (Part I). These initial steps detail the assembly of required components.

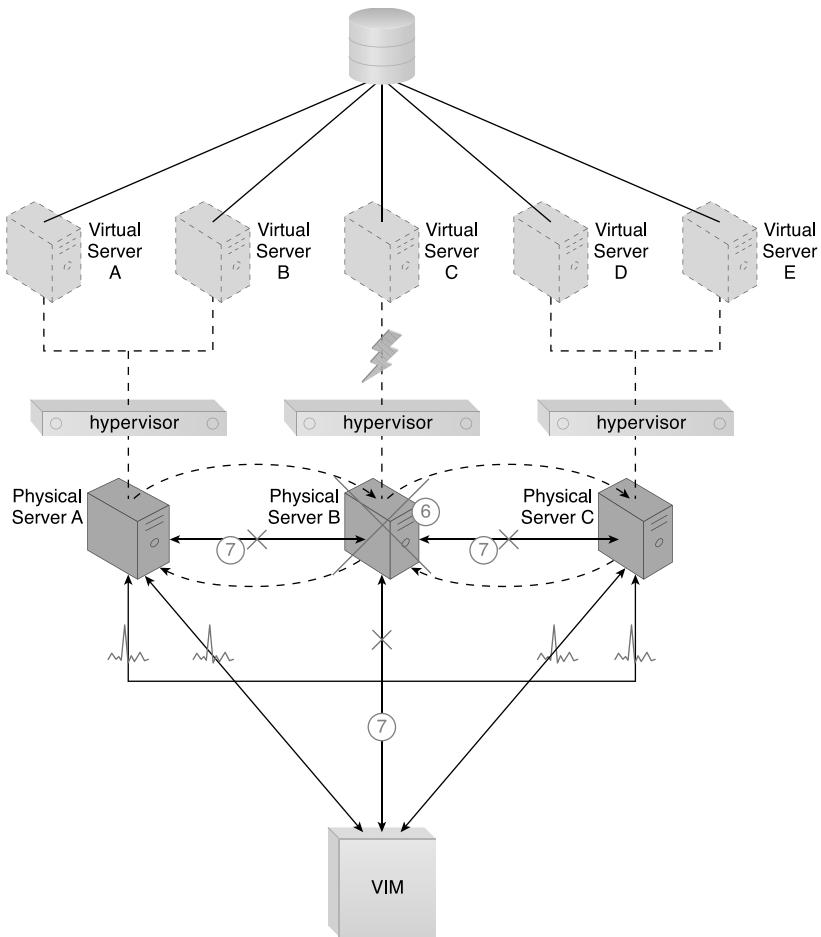


1. Hypervisors are installed on the three physical servers.
2. Virtual servers are created by the hypervisors.
3. A shared cloud storage device containing virtual server configuration files is positioned so that all hypervisors have access to it.
4. The hypervisor cluster is enabled on the three physical server hosts via a central VIM.
5. The physical servers exchange heartbeat messages with each other and the VIM, based on a predefined schedule.

Figure 4.10

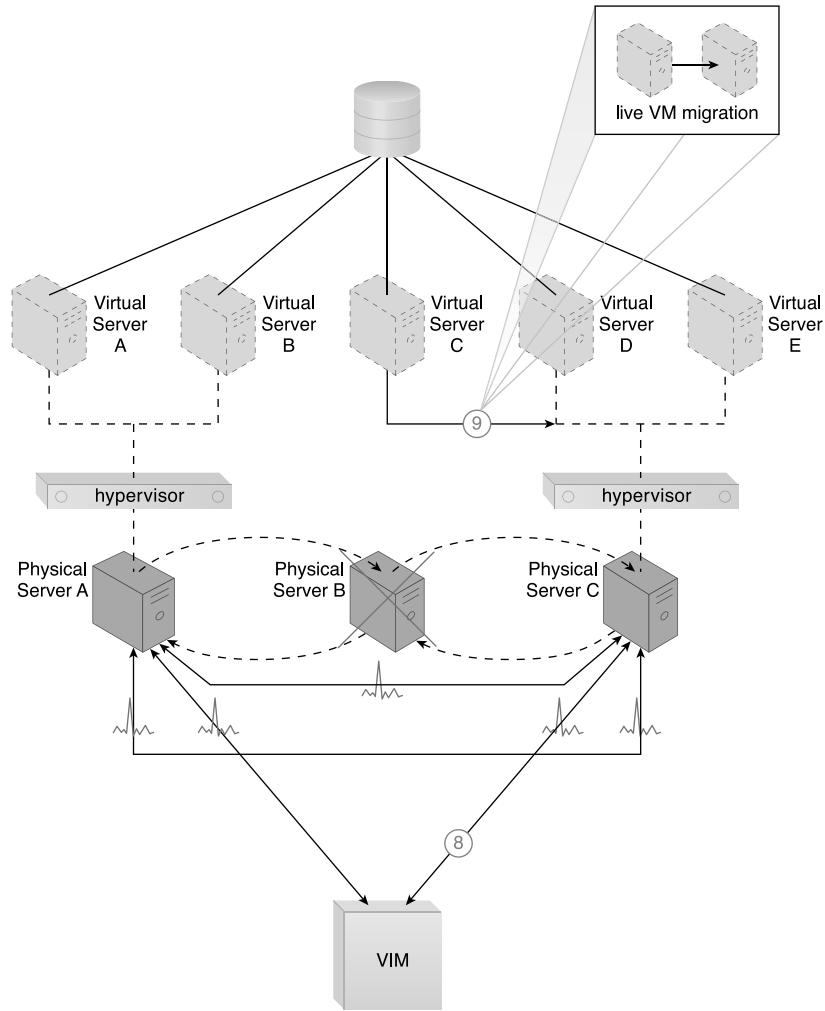
A cloud architecture resulting from the application of the Hypervisor Clustering pattern (Part II).



**Figure 4.11**

A cloud architecture resulting from the application of the Hypervisor Clustering pattern (Part III).

6. Physical Server B fails and becomes unavailable, jeopardizing Virtual Server C.
7. The VIM and the other physical servers stop receiving heartbeat messages from Physical Server B.
8. Based on the available capacity of other hypervisors in the cluster, the VIM chooses Physical Server C as the new host to take ownership of Virtual Server C.
9. Virtual Server C is live-migrated to the hypervisor running on Physical Server C, where it may need to be restarted before continuing to operate normally.

**Figure 4.12**

A cloud architecture resulting from the application of the Hypervisor Clustering pattern (Part IV).

Mechanisms

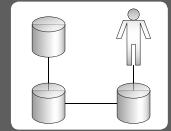
- *Cloud Storage Device* – The cloud storage device mechanism acts as a central repository that hosts the virtual server folders, so that the folders and virtual server configurations are accessible to all of the hypervisors participating in the cluster.
- *Hypervisor* – The hypervisor is the primary mechanism by which this pattern is applied. It acts as a member of the cluster and hosts the virtual servers. If a

hypervisor fails, one of the other available hypervisors restarts its virtual machine to recover the hosted virtual servers from failure.

- *Logical Network Perimeter* – This mechanism creates logical boundaries that ensure that none of the hypervisors of other cloud consumers are accidentally included in a given cluster.
- *Resource Cluster* – The resource cluster is the fundamental mechanism used to create and initiate hypervisor clusters.
- *Resource Replication* – Each hypervisor informs others in the cluster about its status and availability. When a part of cluster configuration needs to be changed, for instance when a virtual switch is created, deleted, or modified, then this update may need to be replicated to all hypervisors via the VIM.
- *Virtual Infrastructure Manager (VIM)* – This mechanism is used to create and configure the hypervisor cluster, add cluster members to the cluster, and cascade the cluster configuration to cluster members.
- *Virtual Server* – Virtual servers represent the type of mechanism that is protected by the application of this pattern.
- *Virtual Switch* – This mechanism is used to ensure that any virtual servers retrieved from hypervisor failure will be accessible to cloud consumers.
- *Virtualization Monitor* – This mechanism is responsible for actively monitoring the hypervisors, and sending alerts whenever one of the hypervisors in the cluster fails.

Redundant Storage

How can the reliability and availability of cloud storage devices survive failure conditions?



Problem	When cloud storage devices fail or become inaccessible, cloud consumers are unable to access data and cloud services relying on access to the device may also fail.
Solution	A failsafe system comprised of redundant cloud storage devices is established so that when the primary device fails, the redundant secondary device takes its place.
Application	Data is replicated from the primary storage to the secondary storage device. A storage service gateway is used to redirect data access requests to the secondary storage device, when necessary.
Mechanisms	Cloud Storage Device, Failover System, Resource Replication

Problem

Cloud storage devices are subject to failure and disruption due to a variety of causes, including network connectivity issues, controller failures, general hardware failure, and security breaches. When the reliability of a cloud storage device is compromised, it can have a ripple effect, causing impact failure across any cloud services, cloud-based applications, and cloud infrastructure program and components that rely on its presence and availability.

The following steps are shown in Figure 4.13:

1. The cloud storage device is installed and configured.
2. Four LUNs are created, one for each cloud consumer.
3. Each cloud consumer sends a request to access its own LUN.
4. The cloud storage device receives the requests and forwards them to the respective LUN.
5. The cloud storage device fails and cloud consumers lose access to their LUNs. This may be due to the loss of the device controller (5.1) or loss of connectivity (5.2).

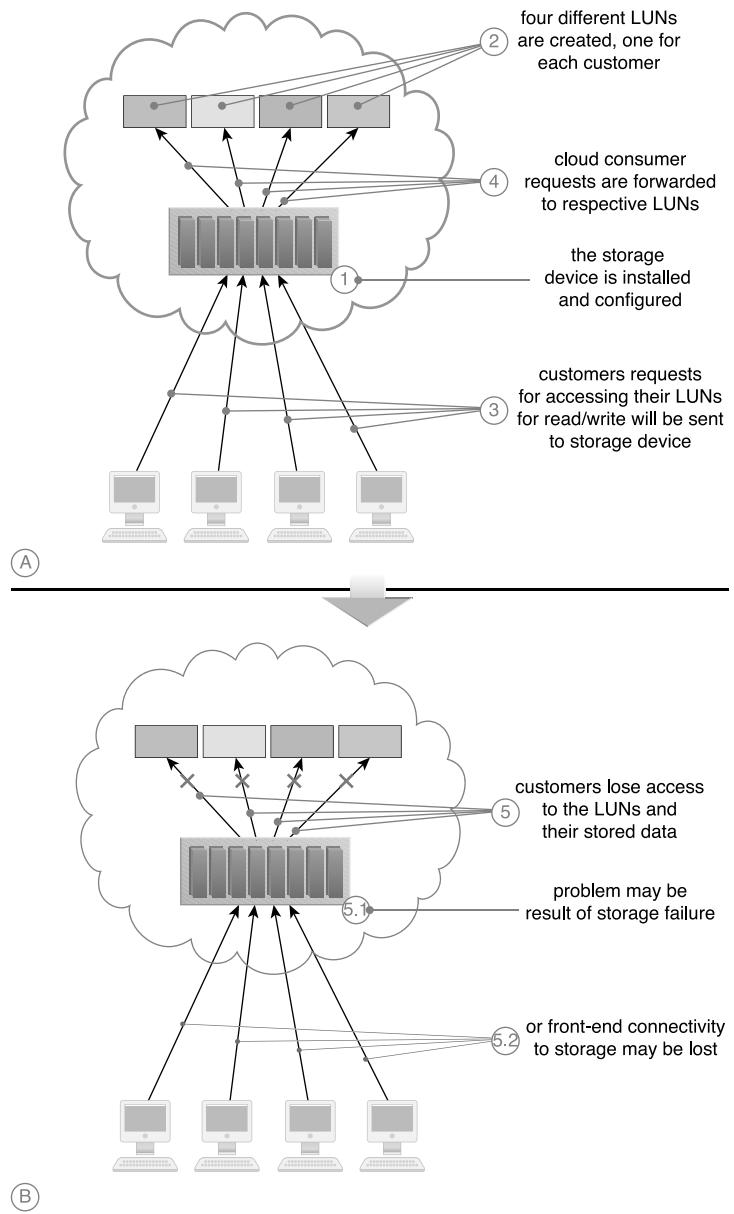


Figure 4.13

A sample scenario that demonstrates the effects of a failed cloud storage device.

Solution

A secondary redundant cloud storage device is incorporated into a system that synchronizes its data with the data in the primary cloud storage device. When the primary device fails, a storage service gateway diverts requests to the secondary device.

The following steps are shown in Figure 4.14:

1. The primary cloud storage device is replicated to the secondary cloud storage device on a regular basis.
2. The primary storage becomes unavailable and the storage service gateway forwards the cloud consumer requests to the secondary storage device.
3. The secondary storage forwards the requests to the LUNs, allowing cloud consumers to continue to access to their data.

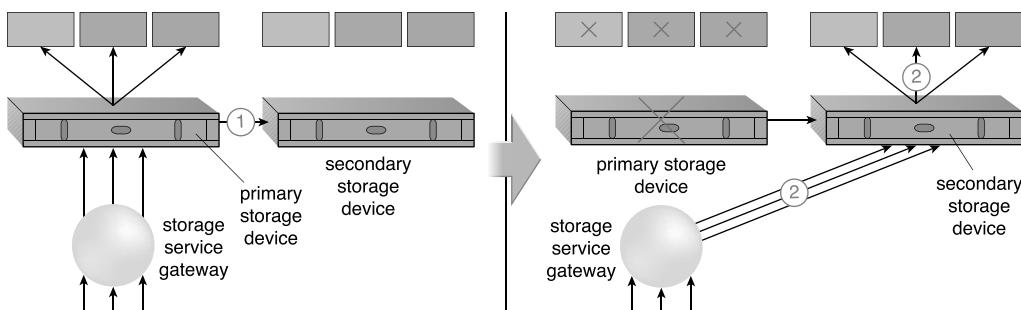


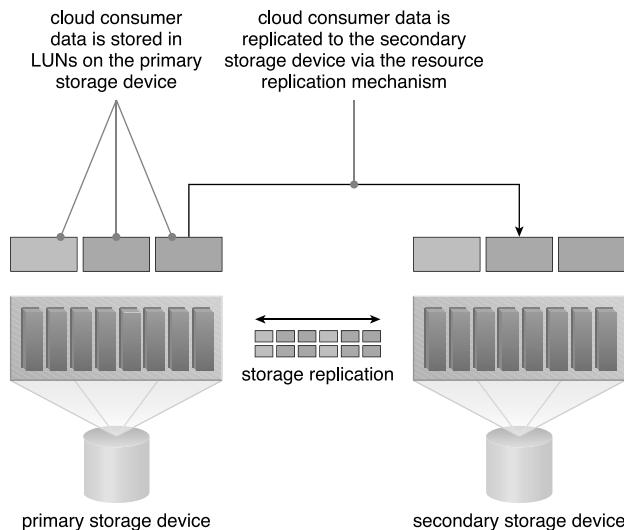
Figure 4.14

A simple scenario demonstrating the failover of redundant storage.

Application

This pattern fundamentally relies on the resource replication mechanism to keep the primary cloud storage device synchronized with any additional duplicate secondary cloud storage devices that comprise the failover system (Figure 4.15).

Cloud providers may locate secondary cloud storage devices in a different geographical region than the primary cloud storage device, usually for economic reasons. For some types of data, this may introduce legal concerns. The location of the secondary cloud storage device can dictate the protocol and method used for synchronization because some replication transport protocols have distance restrictions.

**Figure 4.15**

Storage replication is used to keep the redundant storage device synchronized.

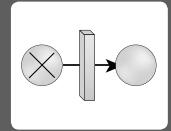
Some cloud providers use storage devices with dual array and storage controllers to improve device redundancy. They may place the secondary storage device in a different physical location for cloud balancing and disaster recovery purposes. In this case, cloud providers may need to lease a network connection via a third-party cloud provider, to establish replication between two devices.

Mechanisms

- *Cloud Storage Device* – This is the mechanism to which the pattern is primarily applied.
- *Failover System* – The application of the Redundant Storage pattern results in a specialized failover system based on the use of duplicate storage devices and a storage service gateway.
- *Resource Replication* – The failover system created by the application of this pattern relies on this mechanism to keep cloud storage devices synchronized.

Dynamic Failure Detection and Recovery

How can the notification and recovery of IT resource failure be automated?



Problem	When cloud-based IT resources fail, manual intervention may be unacceptably inefficient.
Solution	A watchdog system is established to monitor IT resource status and perform notifications and/or recovery attempts during failure conditions.
Application	Different intelligent monitoring and recovery technologies can be used to establish the automation of failure detection and recovery tasks with a focus on watching, deciding upon, acting upon, reporting, and escalating IT resource failure conditions.
Mechanisms	Audit Monitor, Cloud Usage Monitor, Failover System, SLA Management System, SLA Monitor

Problem

Cloud environments can be comprised of vast quantities of IT resources being accessed by numerous cloud consumers. Any of those IT resources can experience predictable failure conditions that require intervention to resolve. Manually administering and solving standard IT resource failures in cloud environments is generally inefficient and impractical.

Solution

A resilient watchdog system is established to monitor and respond to a wide range of pre-defined failure scenarios. This system is further able to notify and escalate certain failure conditions that it cannot automatically solve itself.

Application

The resilient watchdog system relies on a specialized cloud usage monitor (that can be referred to as the intelligent watchdog monitor) to actively monitor IT resources and take pre-defined actions in response to pre-defined events (Figures 4.16 and 4.17).

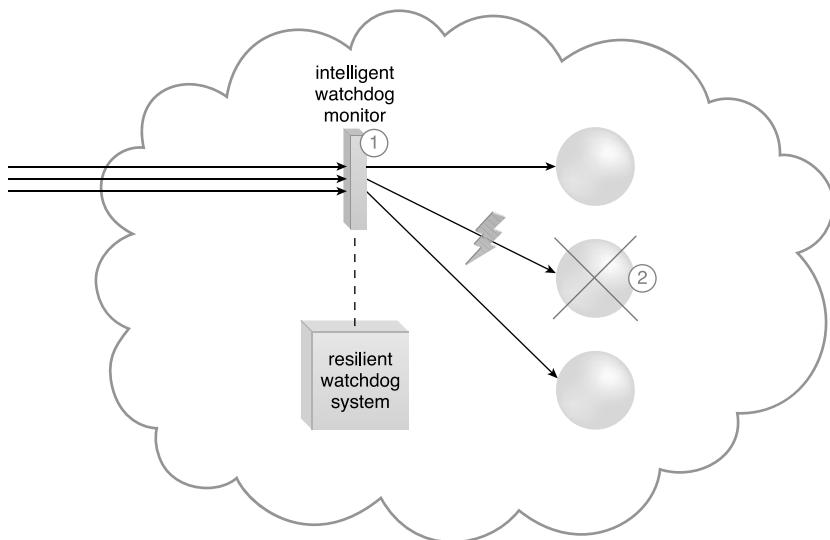


Figure 4.16

The intelligent watchdog monitor keeps track of cloud consumer requests (1) and detects that a cloud service has failed (2).

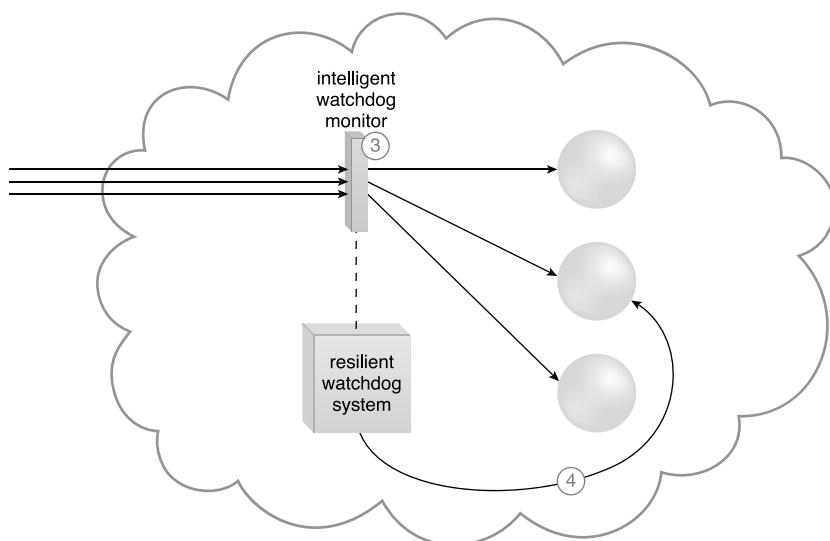


Figure 4.17

The intelligent watchdog monitor notifies the watchdog system (3), which restores the cloud service based on predefined policies (4).

The resilient watchdog system, together with the intelligent watchdog monitor, performs the following five core functions:

- watching
- deciding upon an event
- acting upon an event
- reporting
- escalating

Sequential recovery policies can be defined for each IT resource to determine how the intelligent watchdog monitor should behave when encountering a failure condition (Figure 4.18). For example, a recovery policy may state that before issuing a notification, one recovery attempt should be carried out automatically.

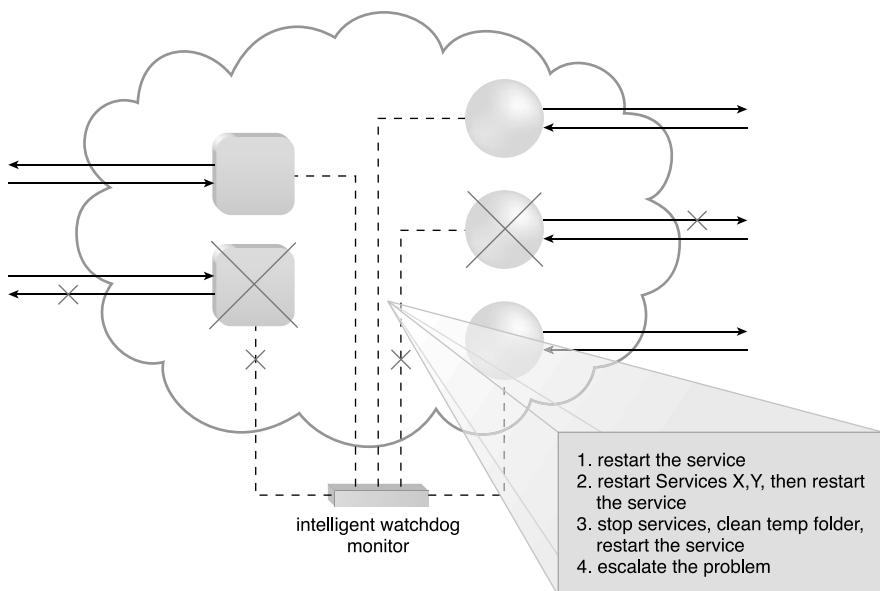


Figure 4.18

In the event of any failures, the active monitor refers to its predefined policies to recover the service step by step, escalating the processes as the problem proves to be deeper than expected.

When the intelligent watchdog monitor escalates an issue, there are common types of actions it may take, such as:

- running a batch file
- sending a console message
- sending a text message
- sending an email message
- sending an SNMP trap
- logging a ticket in a ticketing and event monitoring system

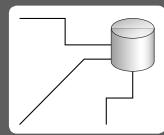
There are varieties of programs and products that can act as an intelligent watchdog monitor. Most can be integrated with standard ticketing and event management systems.

Mechanisms

- *Audit Monitor* – This mechanism may be required to ensure that the manner in which this pattern is carried out at runtime is in compliance with any related legal or policy requirements.
- *Cloud Usage Monitor* – Various specialized cloud usage monitors may be involved with monitoring and collecting IT resource usage data as part of failure conditions and recovery, notification, and escalation activity.
- *Failover System* – Failover is fundamental to the application of this pattern, as the failover system mechanism is generally utilized during the initial attempts to recover failed IT resources.
- *SLA Management System* and *SLA Monitor* – The functionality introduced by the application of the Dynamic Failure Detection and Recovery pattern is closely associated with SLA guarantees and therefore commonly relies on the information managed and processed by these mechanisms.

Multipath Resource Access

How can an IT resource be accessed when its pre-defined path is lost or becomes unavailable?



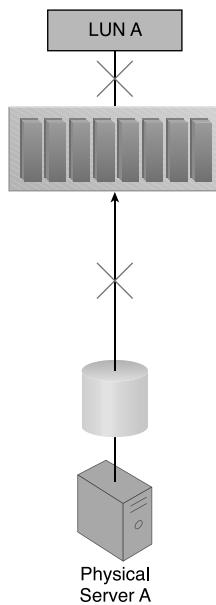
Problem	When the path to an IT resource is lost or becomes unavailable, the IT resource becomes inaccessible. This can jeopardize the stability of an entire cloud-based solution until the cloud provider is able to supply the cloud consumer with the lost or updated path.
Solution	Alternative paths to IT resources are provided to give cloud consumers a means of programmatically or manually overcoming path failures.
Application	A multipathing system that resides on the server or hypervisor is established to provide multiple alternative paths to the same, unique IT resource, while ensuring that the IT resource is viewed identically via each alternative path.
Mechanisms	Cloud Storage Device, Hypervisor, Logical Network Perimeter, Resource Replication, Virtual Server

Problem

Certain IT resources can only be accessed using an assigned path (hyperlink) that leads to the location of the IT resources. The path can be inadvertently lost or incorrectly defined by the cloud consumer or changed by the cloud provider. When a cloud consumer no longer possesses the correct and exclusive path to an IT resource, this IT resource becomes inaccessible and unavailable (Figure 4.19). When this unavailability occurs without warning at runtime, exception conditions can result that compromise the stability of larger cloud solutions that depend on the IT resource's availability.

Figure 4.19

Physical Server A is connected to LUN A via a single fiber channel, and uses the LUN to store different types of data. The fiber channel connection becomes unavailable due to an HBA card failure and invalidates the path used by Physical Server A, which has now lost access to LUN A and all of its stored data.

**Solution**

A multipathing system is established to provide alternative paths to IT resources providing cloud consumers with a means of programmatically or manually overcoming path failures (Figure 4.20).

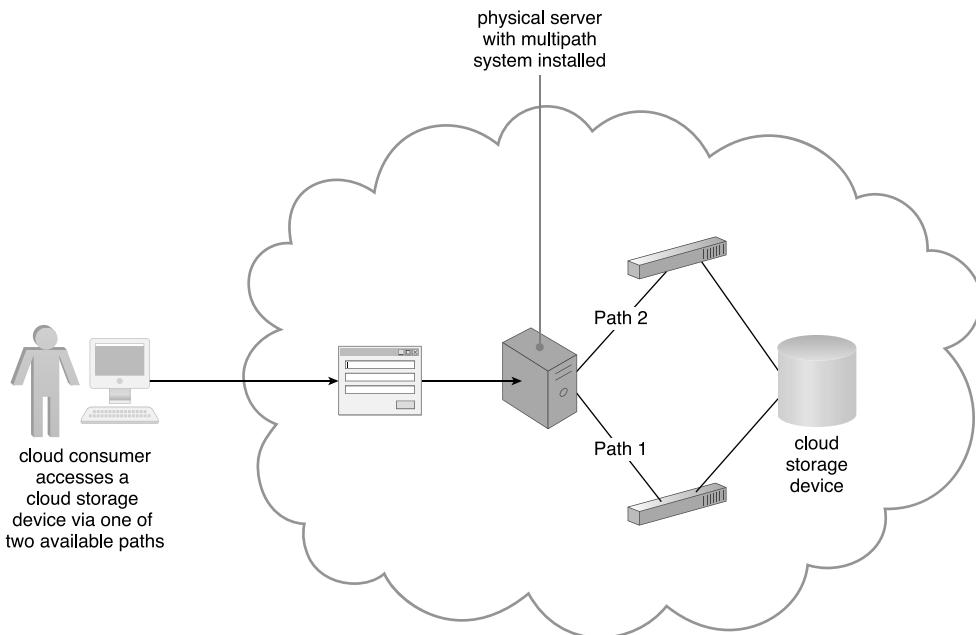


Figure 4.20

A multipathing system providing alternative paths to a cloud storage device.

Application

The application of this pattern requires the use of a multipathing system and the creation of alternative paths (or hyperlinks) that are assigned to specific IT resources. The alternative paths may be physical or virtual. The multipathing system resides on the server or hypervisor, and ensures that each IT resource can be seen via each alternative path identically.

The following steps are shown in Figure 4.21:

1. Physical Server A is connected to the LUN A storage device via two different paths.
2. LUN A is seen as different LUNs from each of the two paths.
3. The multipathing system is put in place and configured.
4. LUN A is seen as one identical LUN from both paths.

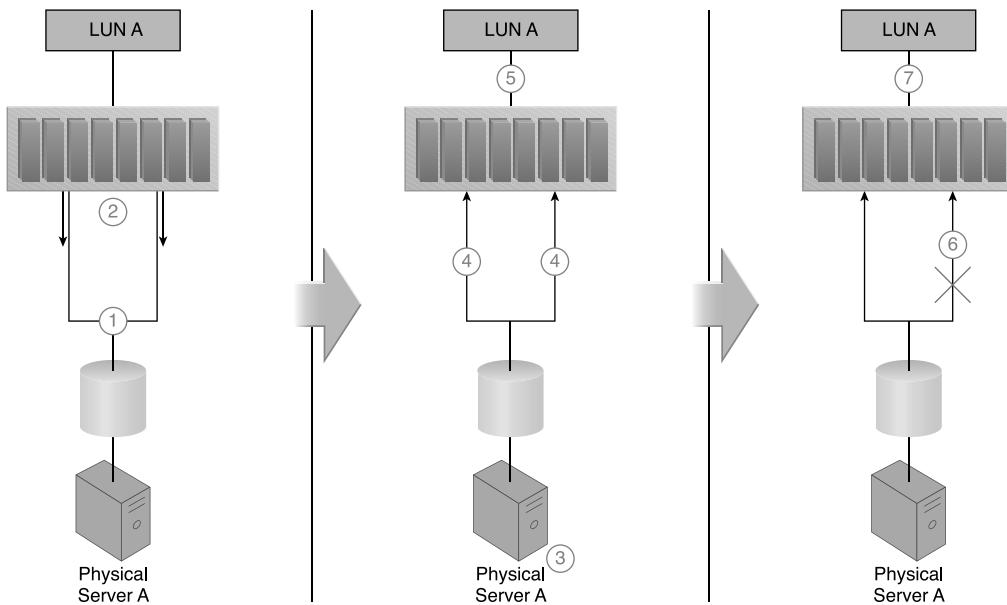


Figure 4.21

An example of a multipathing system.

5. Physical Server A has access to LUN A from two different paths.
6. A link failure occurs and one of the paths becomes unavailable.
7. Physical Server A can still use LUN A because the other link remains active.

In some cases, a specific driver is required by the operating system to ensure that it understands the redundant paths and does view two paths leading to the same IT resource as two separate IT resources, as shown in Figure 4.22.

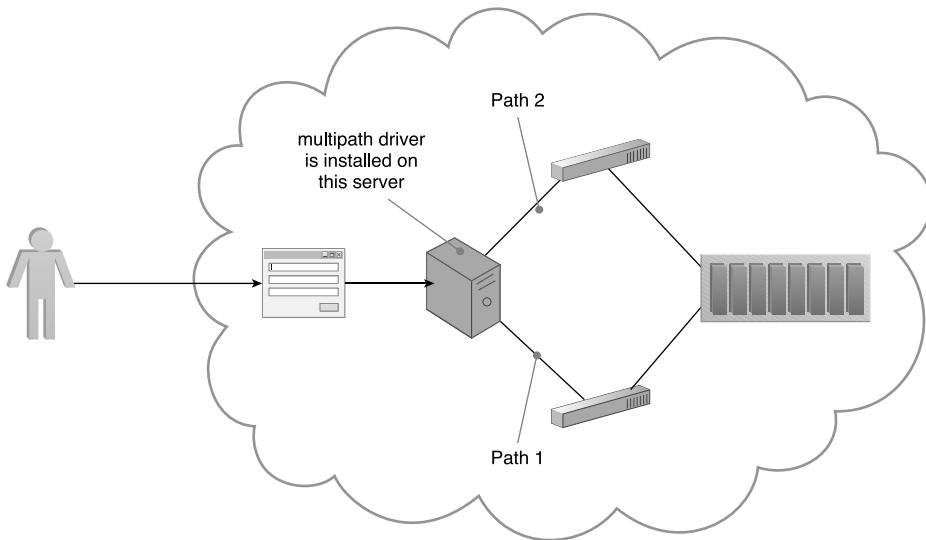


Figure 4.22

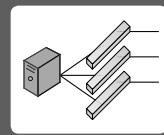
A multipath driver is installed on a server to ensure that the operating system understands the redundant paths and views two paths leading to the same IT resource as two separate IT resources.

Mechanisms

- *Cloud Storage Device* – The cloud storage device is a common IT resource that may require the creation of an alternative path in order to remain accessible by solutions that rely on data access.
- *Hypervisor* – An alternate path to a hypervisor is required to have a redundant link to the hosted virtual servers.
- *Logical Network Perimeter* – This mechanism guarantees that the privacy of cloud consumers is upheld even when multiple paths to the same IT resource are created.
- *Resource Replication* – The resource replication mechanism is required when it is necessary to create a new instance of an IT resource in order to generate the alternative path.
- *Virtual Server* – This mechanism is associated with Multipath Resource Access in how it hosts services that have multipath access via different links or virtual switches. In some cases, the hypervisor itself provides multipath access to the virtual server.

Redundant Physical Connection for Virtual Servers

How can a virtual server be kept connected when its physical connection fails?



Problem	If the virtual switch uplink port used by a virtual server fails, the virtual server becomes isolated and unable to connect to the network or any of its hosted IT resources.
Solution	A redundant, physical backup network connection is established for virtual servers.
Application	A second physical network card is added to the physical host and is configured as a hot standby uplink port for the virtual switch.
Mechanisms	Failover System, Hypervisor, Logical Network Perimeter, Physical Uplink, Resource Replication, Virtual Infrastructure Manager (VIM), Virtual Server, Virtual Switch

Problem

A virtual server is connected to an external network via a virtual switch uplink port. If the uplink fails (due to, for example, cable disconnection or port failure), the virtual server becomes isolated and disconnects from the external network.

The following steps are shown in Figure 4.23:

1. A physical network adapter installed on the physical server host is connected to the physical switch on the network.
2. A virtual switch is created for use by two virtual servers. Because it requires access to the physical external network, the physical network adapter is attached to the virtual switch to be used as an uplink to the network.
3. The virtual servers communicate with the external network via the attached physical uplink network card.
4. A connection failure occurs, either because of a physical link connectivity issue between the physical adapter and the physical switch (4.1), or because of a physical network card failure (4.2).
5. The virtual servers lose access to the physical external network and are no longer accessible by their cloud consumers.

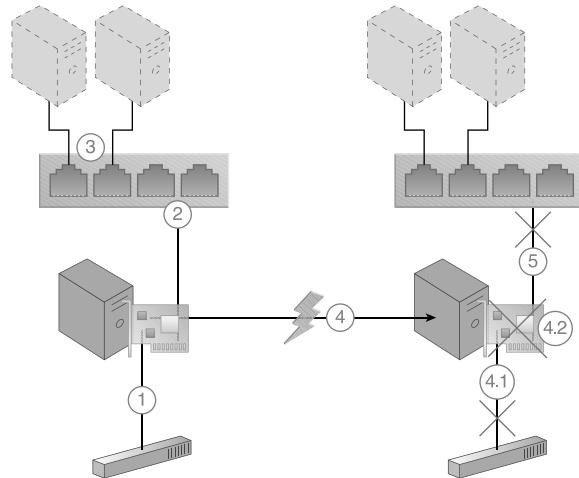


Figure 4.23

The steps that can lead to the separation of virtual servers from their external network connection.

Solution

One or more redundant uplink connections are established and positioned in standby mode. A redundant uplink connection is available to take over as the active uplink connection whenever the primary uplink connection becomes unavailable or experiences failure conditions (Figure 4.24).

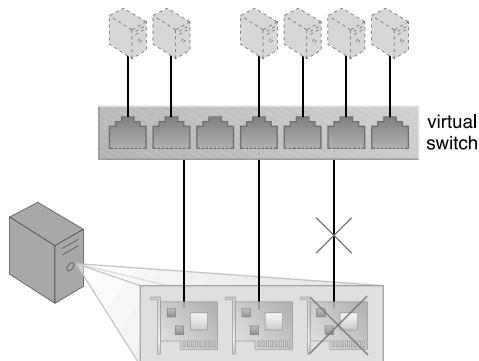


Figure 4.24

Redundant uplinks are installed on a physical server hosting several virtual servers. When one fails, another takes over to maintain the virtual servers' active network connections.

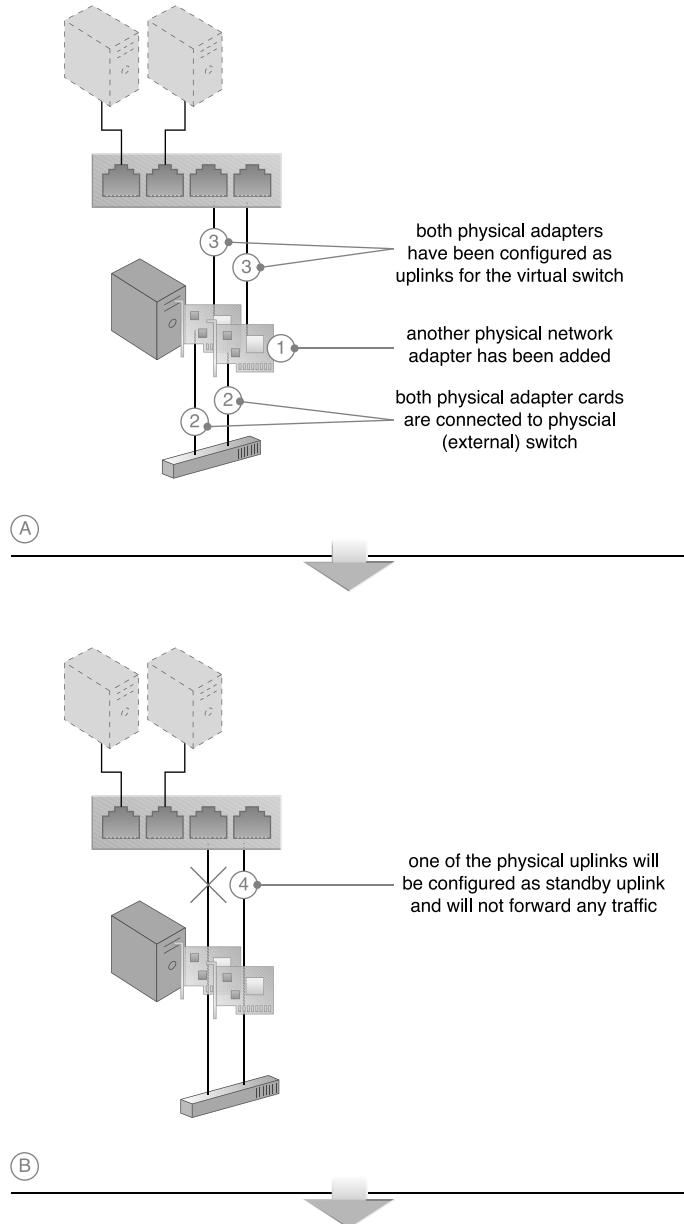
Application

While the main uplink is working, virtual servers connect to the outside via that port. As soon as it fails, the standby uplink will automatically become the active uplink, and the server will send the packets to the outside via the new uplink. This process is also transparent to virtual servers and users.

While the second NIC is connected and receives the virtual server's packets, it is not forwarding any traffic while the primary uplink is alive. If, and when, the primary uplink fails, the secondary uplink starts to forward the packets without any pause or interruption. If the failed uplink happens to come back into operation, it will take over the lead role and the second NIC goes into standby mode again.

The following steps are shown in Figures 4.25 and 4.26:

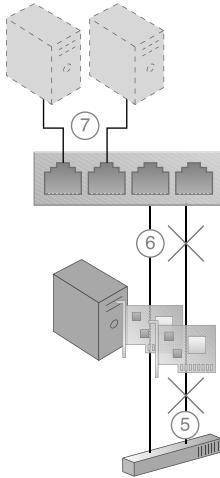
1. A new network adapter is added to support a redundant uplink.
2. Both network cards are connected to the physical external switch.
3. Both physical network adapters are configured to be used as uplink adapters for the virtual switch.
4. One physical network adapter is designated as the primary adapter, whereas the other is designated as the secondary adapter providing the standby uplink. The secondary adapter does not forward any packets.
5. The primary uplink forwards packets to the external network until it becomes unavailable.
6. When required, the secondary standby uplink automatically becomes the primary uplink and uses the virtual switch to forward the virtual servers' packets to the external network.
7. The virtual servers stay connected to the external physical network, without interruptions.

**Figure 4.25**

An example scenario of the utilization of a redundant uplink (Part I).

Figure 4.26

An example scenario of the utilization of a redundant uplink (Part II).



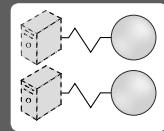
Mechanisms

- *Failover System* – The failover system is utilized to perform the failover of an unavailable uplink to a standby uplink.
- *Hypervisor* – The hypervisor hosts the virtual servers and some of the virtual switches, and provides virtual networks and virtual switches with access to the virtual servers. If a virtual switch's physical uplink becomes unavailable, this mechanism is responsible for forwarding the virtual servers' traffic using another available physical uplink on the virtual switch.
- *Logical Network Perimeter* – Logical network perimeters ensure that the virtual switches that are allocated or defined for each cloud consumer remain isolated.
- *Physical Uplink* – This mechanism is used to establish connectivity between virtual switches and physical switches. Additional physical uplinks can be attached to a virtual switch to improve redundancy.
- *Resource Replication* – Resource replication is used to replicate the current status of the active uplink to a standby uplink, so that the connection remains active without disruption.
- *Virtual Infrastructure Manager (VIM)* – This mechanism is used to configure virtual switches and their uplinks, and performs the configurations on the hypervisors so that they can use another available uplink should an active uplink fail.

- *Virtual Server* – This pattern is primarily applied in support of maintaining the network connections for virtual servers.
- *Virtual Switch* – This mechanism uses the attached physical uplinks to establish physical connection redundancy that allows virtual servers to be redundantly connected to cloud consumers and the physical network.

Synchronized Operating State

How can the availability and reliability of virtual servers be ensured when high availability and clustering technology is unavailable?



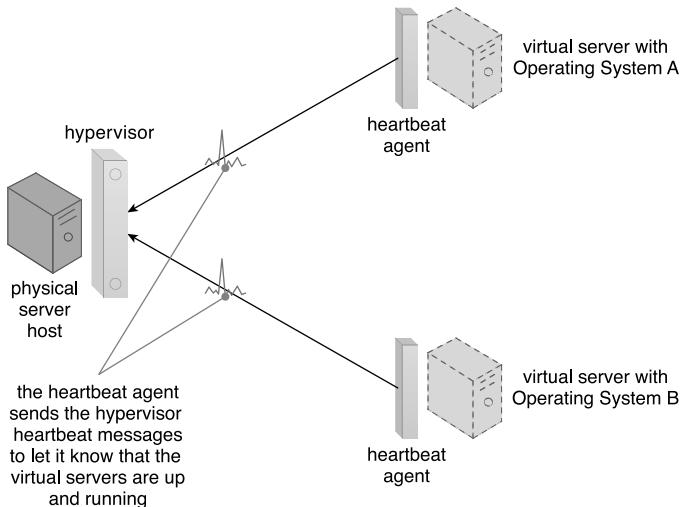
Problem	A cloud consumer may be prevented from utilizing high availability and clustering technology for its virtual servers or operating systems, thereby making them more vulnerable to failure.
Solution	A composite failover system is created to not rely on clustering or high availability features but instead use heartbeat messages to synchronize virtual servers.
Application	The heartbeat messages are processed by a specialized service agent and are exchanged between hypervisors, the hypervisor and virtual server, and the hypervisor and VIM.
Mechanisms	Cloud Storage Device, Failover System, Hypervisor, Resource Replication, State Management Database, Virtual Server

Problem

Technical restrictions, licensing restrictions, or other reasons may prevent a cloud consumer from taking advantage of clustering and high availability technology and products. This can seriously jeopardize the availability and scalability of its cloud services and applications.

Solution

A system comprised of a set of mechanisms and relying on the use of heartbeat messages is established to emulate select features of clustering and high availability IT resources (Figure 4.27).

**Figure 4.27**

Special heartbeat agents are employed to monitor heartbeat messages exchanged between the servers.

Application

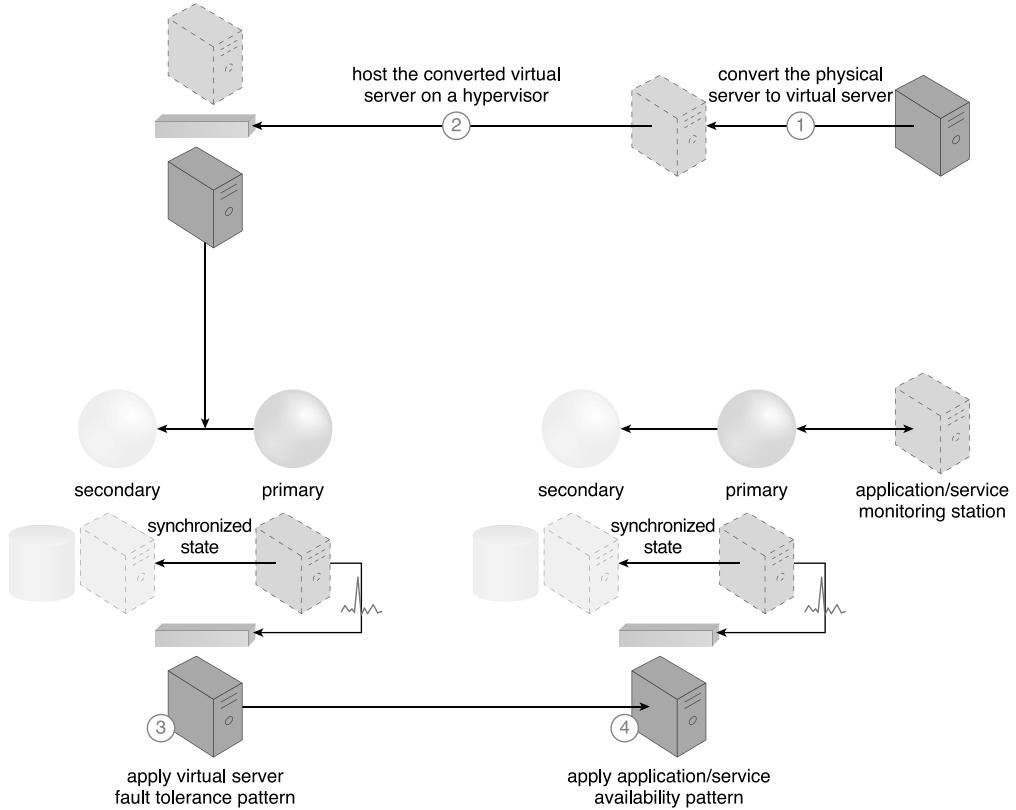
Heartbeat messages are processed by a heartbeat monitor agent and are exchanged between:

- hypervisors
- each hypervisor and each virtual server
- each hypervisor and the central VIM

If an operating system is placed on a physical server, it needs to be converted into a virtual server prior to the issuance of heartbeat messages.

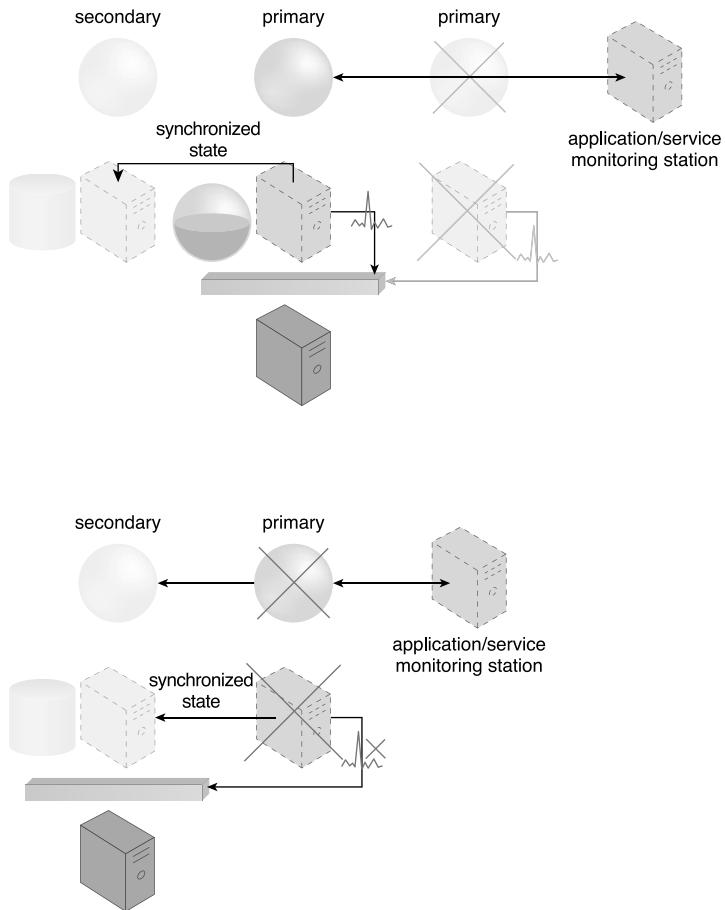
The following steps are shown in Figure 4.28:

1. A virtual server is created from the physical server.
2. The hypervisor proceeds to host the virtual server.
3. The primary virtual server is equipped with fault tolerance and maintains a synchronized state via the use of heartbeat messages.
4. The secondary server that shares the synchronized state is available in case the primary virtual server fails.

**Figure 4.28**

The cloud architecture resulting from the application of this pattern.

The application/service monitoring station monitors the servers and cloud services. In the event of failure, this station attempts recovery based on sequential pre-defined policies. If the primary server's operating system fails, procedures are in place to avoid downtime (Figure 4.29).

**Figure 4.29**

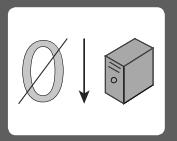
When the primary virtual server fails, along with its hosted cloud service, heartbeat messages are no longer transmitted. As a result, the hypervisor recognizes the failure and switches activity to the secondary virtual server that maintains the synchronized state. After the primary virtual server is back online, the hypervisor creates a new secondary for the new primary, and proceeds to save it as a synchronized non-active state.

Mechanisms

- *Cloud Storage Device* – Cloud storage devices may be used to host the primary and secondary (shadow) copies of virtual server data and cloud service instances.
- *Failover System* – The failover system is responsible for providing failsafe logic in support of switch cloud consumer requests from a primary virtual server to a secondary virtual server.
- *Hypervisor* – The hypervisor hosts the primary and secondary (shadow) state data, in addition to providing the features that resource replication needs to replicate the primary state.
- *Resource Replication* – Resource replication performs the replication of the primary virtual server state to a secondary (shadow) copy.
- *State Management Database* – The state management database actively stores and restores secondary operating state data in support of primary virtual server failure and recovery.
- *Virtual Server* – Virtual servers are the primary mechanism to which this pattern is applied.

Zero Downtime

How can downtime of virtual servers be avoided or eliminated?



Problem	It is challenging to provide zero downtime guarantees when a physical host acts as a single point of failure for virtual servers.
Solution	A fault tolerance system is established so that when a physical server fails, virtual servers are migrated to another physical server.
Application	A combination of virtual server fault tolerance, replication, clustering, and load balancing are applied and all virtual servers are stored in a shared volume allowing different physical hosts to access their files.
Mechanisms	Audit Monitor, Cloud Storage Device, Cloud Usage Monitor, Failover System, Hypervisor, Live VM Migration, Logical Network Perimeter, Physical Uplink, Resource Cluster, Resource Replication, Virtual CPU, Virtual Disk, Virtual Infrastructure Manager (VIM), Virtual Network, Virtual RAM, Virtual Server, Virtual Switch, Virtualization Agent, Virtualization Monitor

Problem

A physical server naturally acts as a single point of failure for the virtual servers it hosts. As a result, when the physical server fails or is compromised, the availability of any (or all) hosted virtual servers can be affected. This makes the issuance of zero downtime guarantees by a cloud provider to cloud consumers challenging.

Solution

A failover system is established so that virtual servers are dynamically moved to different physical server hosts in the event that their original physical server host fails. For example, in Figure 4.30, Virtual Server A is dynamically moved to another physical server host.

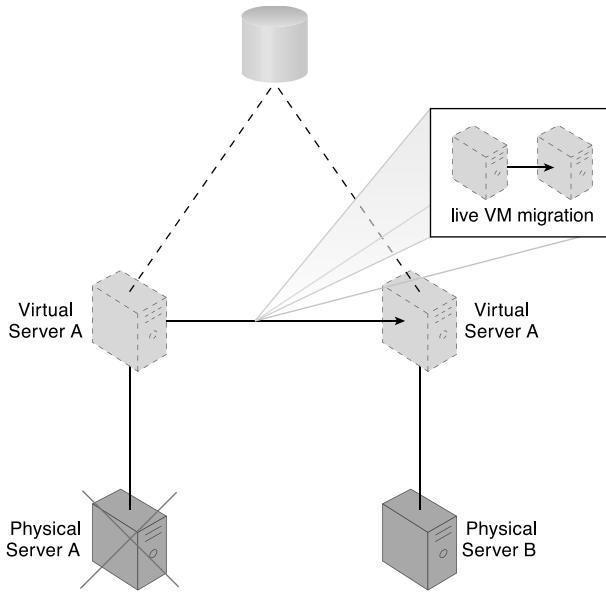


Figure 4.30

Physical Server A fails, triggering the live VM migration program to dynamically move Virtual Server A to Physical Server B.

Application

Multiple physical servers are assembled into a group that is controlled by a fault tolerance system capable of switching activity from one physical server to another, without interruption. Resource cluster and live VM migration components are commonly part of this form of high availability cloud architecture.

The resulting fault tolerance assures that, in case of physical server failure, hosted virtual servers will be migrated to a secondary physical server. All virtual servers are stored on a shared volume (as per Persistent Virtual Network Configuration (227)) so that other physical server hosts in the same group can access their files.

Live storage replication can further be utilized to guarantee that virtual server files and hard disks remain available via secondary storage devices.

Mechanisms

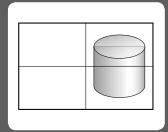
- *Audit Monitor* – This mechanism may be required to ensure that the relocation of virtual servers does not relocate hosted data to prohibited locations.

- *Cloud Storage Device* – A cloud storage device is used to store virtual server network configuration data shared by the physical servers. It stores virtual servers and virtual disks in a central repository so that other available hypervisors can access the files and power on the failed virtual servers in case one of the hypervisors fails.
- *Cloud Usage Monitor* – Incarnations of this mechanism are used to monitor the actual IT resource usage of cloud consumers to help ensure that virtual server capacities are not exceeded.
- *Failover System* – The failover system can be used to switch from a failed primary physical server to a secondary physical server.
- *Hypervisor* – The hypervisor of each affected physical server hosts the affected virtual servers.
- *Live VM Migration* – When multiple instances of the same service or virtual server are provisioned for the purpose of redundancy and availability, this mechanism is used to seamlessly distribute different instances of the same service between different hypervisors to make sure one hypervisor will not become a single point of failure.
- *Logical Network Perimeter* – Logical network perimeters provide and maintain the isolation that is required to ensure that each cloud consumer remains within its own logical boundary subsequent to virtual server relocation.
- *Physical Uplink* – Physical uplinks are used and deployed in a redundant model, so that the virtual servers and services will not lose their connectivity to the cloud service consumers if a physical uplink fails or becomes disconnected.
- *Resource Cluster* – The resource cluster mechanism is applied to create different types of active/active cluster groups that collaboratively improve the availability of virtual server-hosted IT resources.
- *Resource Replication* – This mechanism can create new virtual server and cloud service instances upon primary virtual server failure.
- *Virtual CPU* – The virtual CPU mechanism is used to provide CPU cycling, scheduling, and processing capabilities to the virtual servers.
- *Virtual Disk* – This mechanism is used to allocate local storage space to the hosted virtual servers.

- *Virtual Infrastructure Manager (VIM)* – This mechanism is used to control the availability and redundancy of the virtual servers and services, and initiates proper command when rebalancing the environment or recreating a new instance of a service or virtual server is required.
- *Virtual Network* – This mechanism is used to connect virtual servers and the services hosted on top of them.
- *Virtual RAM* – This mechanism is used to establish access for the virtual servers and applications to the physical memory installed on the physical server.
- *Virtual Server* – This is the mechanism to which this pattern primarily applied.
- *Virtual Switch* – This mechanism is used to connect hosted virtual servers to the physical network and external cloud service consumers using physical uplinks.
- *Virtualization Agent* – Virtual servers use this mechanism to send regular heartbeat messages to the hypervisor. A recovery process is initiated if the hypervisor does not receive heartbeats after an extended period of time.
- *Virtualization Monitor* – This mechanism is used to monitor the virtual servers' availability and operational status.

Storage Maintenance Window

How can access to data in a cloud storage device be preserved during a maintenance outage?



Problem	Hardware maintenance on cloud storage devices can require shutting down the device, resulting in loss of data access and disruption of service.
Solution	An outage prevention system is created to temporarily move the data without interruption during maintenance and other types of outages.
Application	LUN migration is applied to temporarily transfer data to a separate cloud storage device during the maintenance window.
Mechanisms	Cloud Storage Device, Failover System, Resource Replication

Problem

Cloud storage devices subject to maintenance and administrative tasks may need to be temporarily shut down, thereby causing an outage to cloud service consumers and IT resources that require access to the devices and the data they host (Figure 4.31).

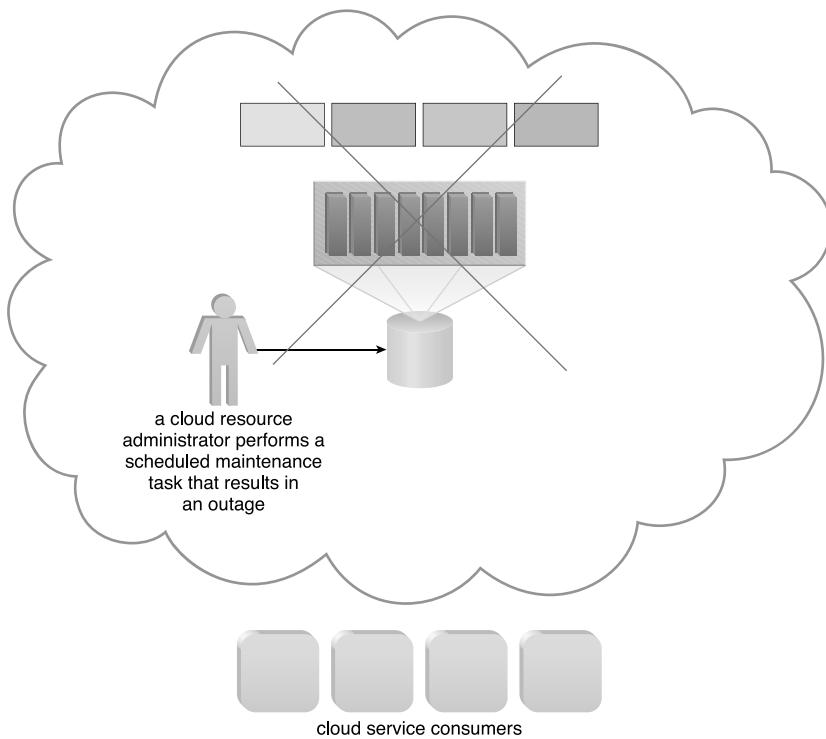


Figure 4.31

The maintenance task carried out by a cloud resource administrator causes an outage for the cloud storage device. Resultantly, the cloud storage device becomes unavailable to cloud service consumers.

Solution

Prior to a cloud storage device undergoing a maintenance outage, its data can be temporarily moved to a duplicate, secondary cloud storage device. Cloud service consumers are automatically and transparently redirected to the secondary cloud storage device and are unaware that the primary cloud storage device has been taken offline.

Application

Live storage migration is used to convert the data as a whole into an isolated mode and move it to the secondary cloud storage device, as shown in Figures 4.32 to 4.37.

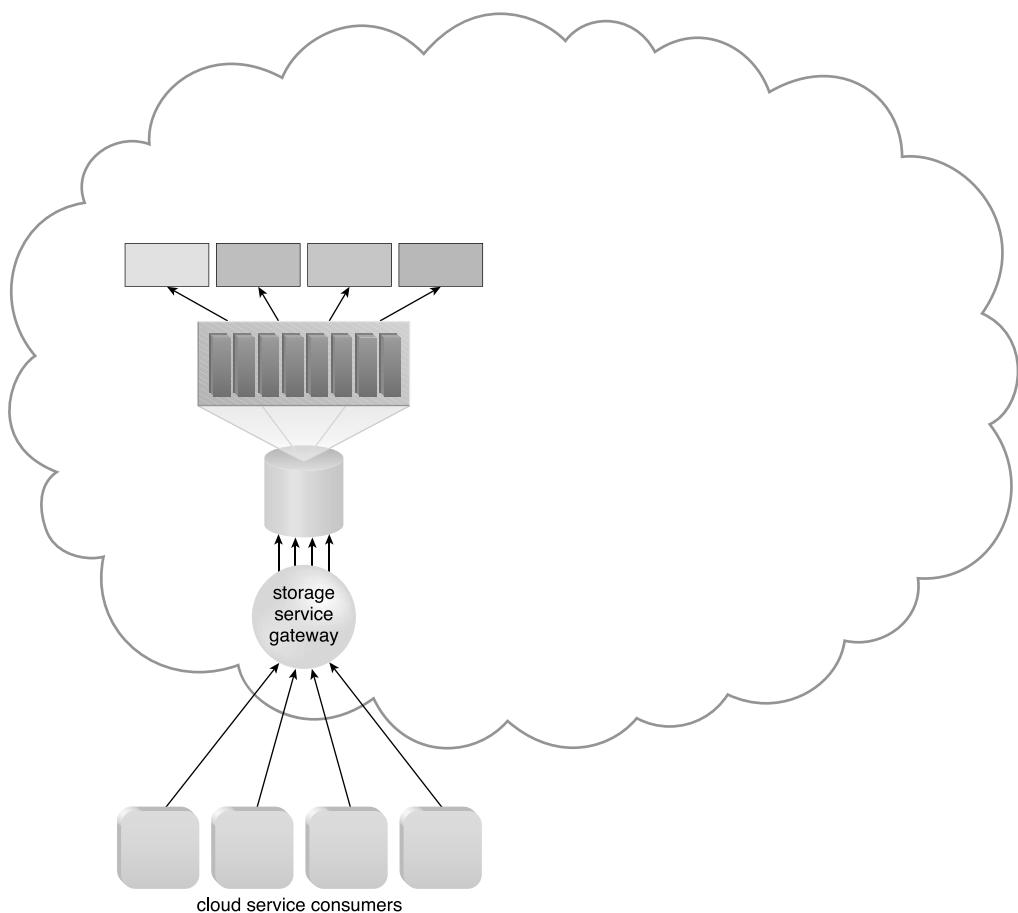
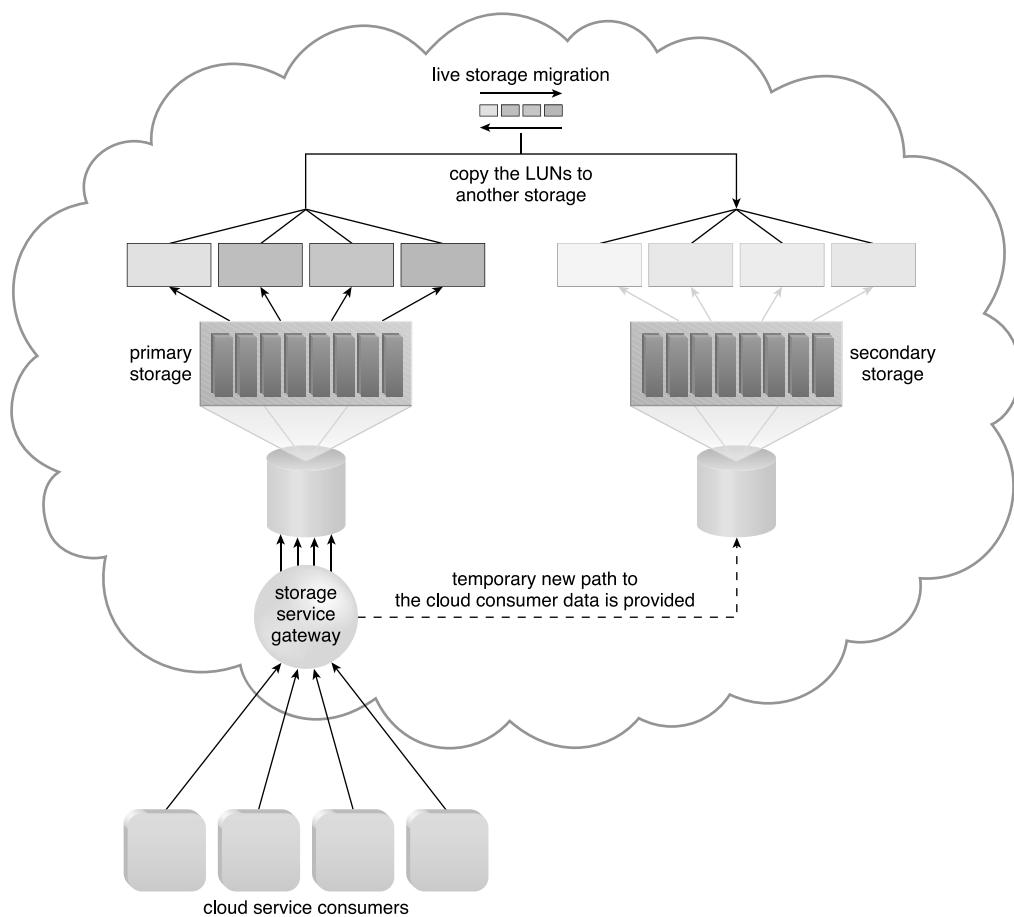
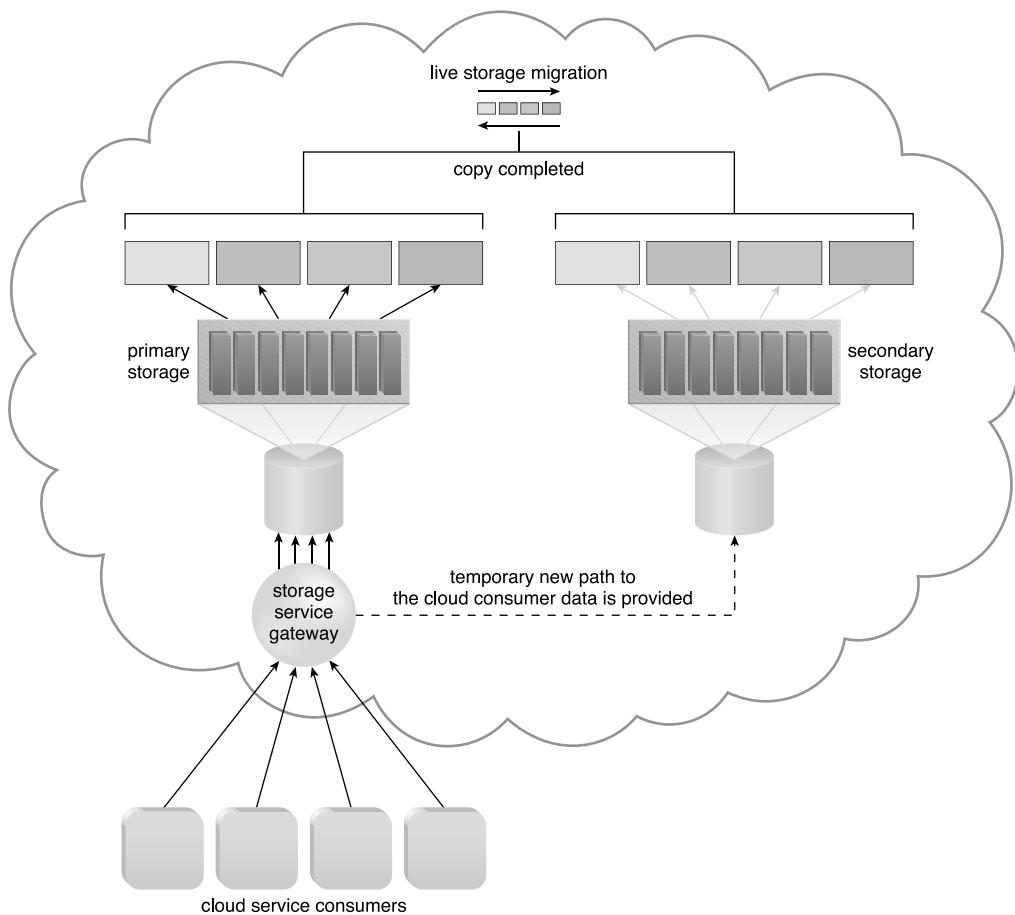


Figure 4.32

The cloud storage device is scheduled to undergo a maintenance outage.

**Figure 4.33**

Live storage migration moves the LUNs from the primary storage device to a secondary storage device.

**Figure 4.34**

When the LUN's data has been migrated, requests for the data are forwarded to the duplicate LUNs on the secondary storage device.

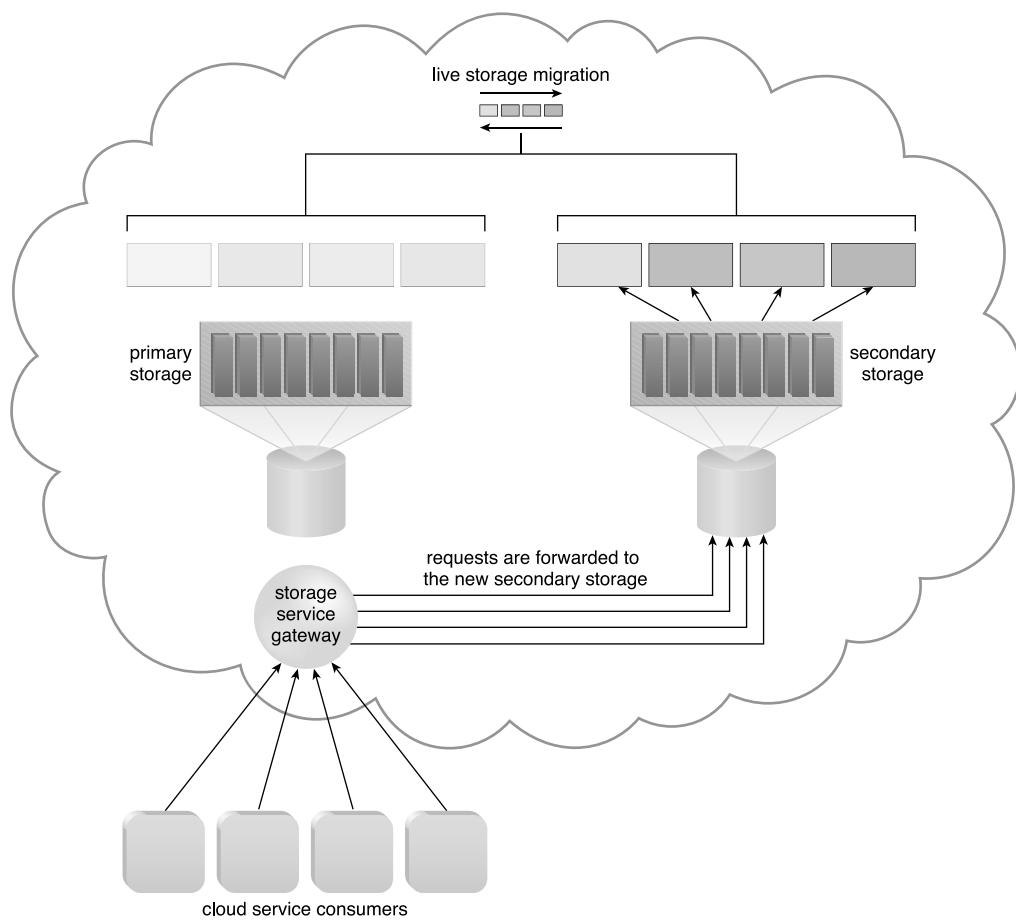
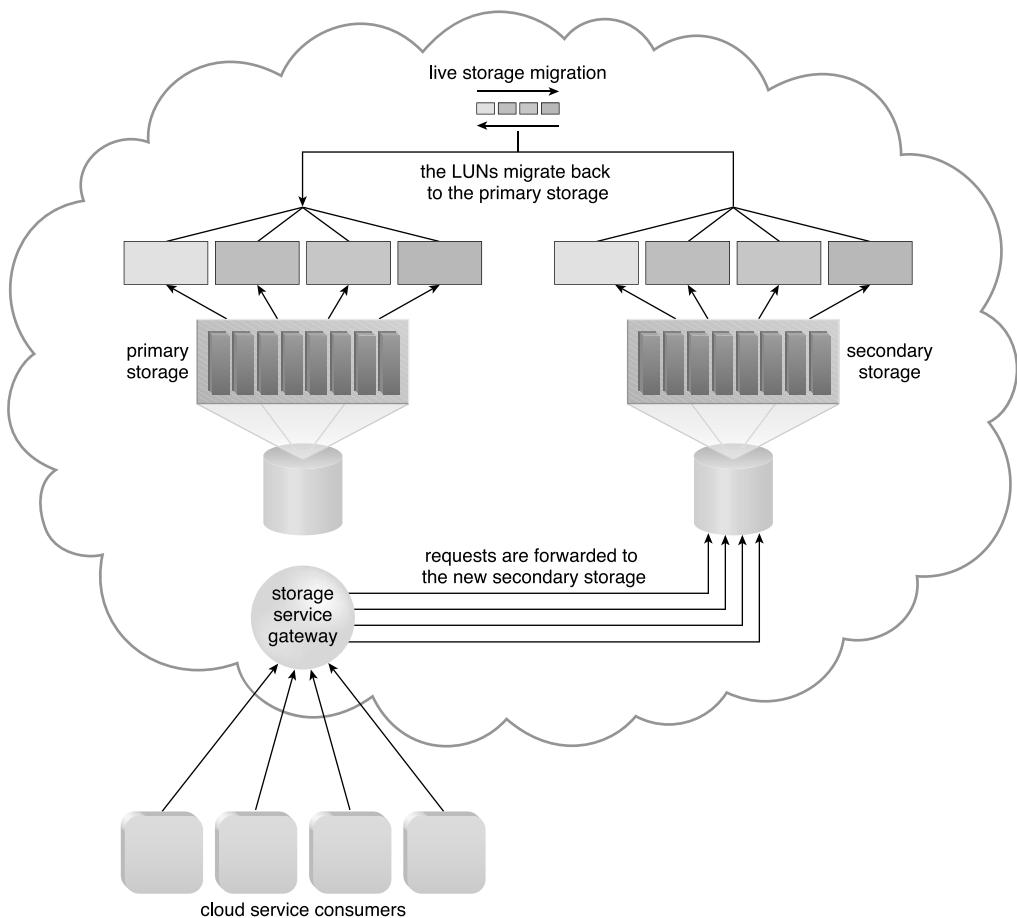


Figure 4.35

The primary storage is powered off for maintenance.

**Figure 4.36**

When it is confirmed that the maintenance task on the primary storage device has been completed, the primary storage is brought back online. Live storage migration subsequently restores the LUN data from the secondary storage device to the primary storage device.

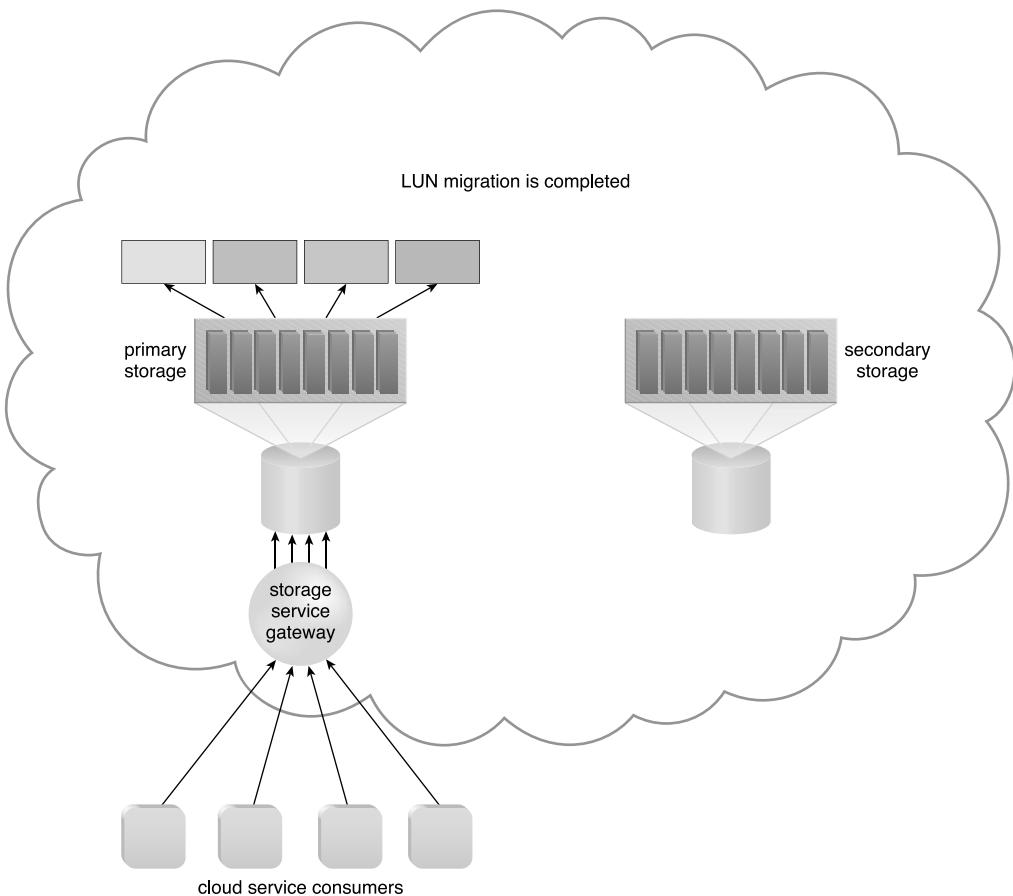


Figure 4.37

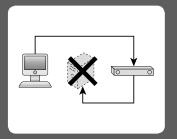
When the LUN migration is completed, all data access requests are forwarded back to the primary storage device.

Mechanisms

- *Cloud Storage Device* – This is the primary mechanism to which this pattern is applied.
- *Failover System* – Although the migration is often pre-scheduled when this pattern is applied, both manually and automatically initiated failover can be incorporated into this cloud architecture.
- *Resource Replication* – The resource replication mechanism is used to keep the primary and secondary storage devices synchronized.

Virtual Server Auto Crash Recovery

In the event that a virtual server's operating system crashes, how can the hosted cloud services be automatically recovered?



Problem	A virtual server whose operating system suddenly fails needs to be able to have its hosted cloud services automatically recovered.
Solution	The virtual server's activity is constantly monitored and traced for recovery, in the event of an operating system failure.
Application	Applying this pattern involves specific techniques and mechanisms that are used by the hypervisor to check the operational status of the virtual server.
Mechanisms	Hypervisor, Virtualization Agent

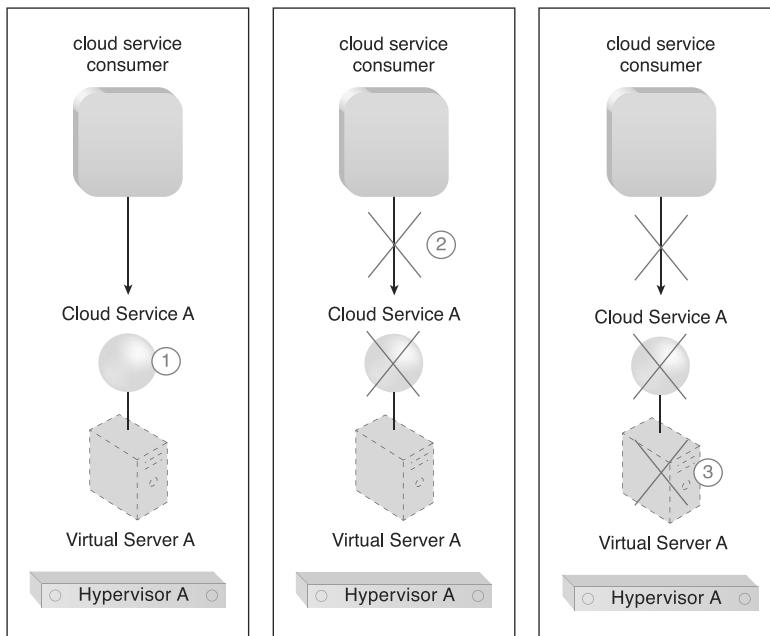
Problem

When the operating system of a virtual server fails or crashes, the cloud services that are hosted on this virtual server also become unavailable. This can in turn cause an outage or even an SLA breach, since some organizations have little to no tolerance for outages.

The following steps are shown in Figure 4.38:

1. Cloud Service A is running on Virtual Server A.
2. Cloud consumers suddenly cannot access the service.
3. An investigation shows that Hypervisor A is working fine and has been allocating resources to Virtual Server A. However, Virtual Server A's resource usage is zero. Further investigation reveals that its operating system has crashed, which is why Cloud Service A is not working.

The system administrator has to manually reboot Virtual Server A in order to bring it back into operation.

**Figure 4.38**

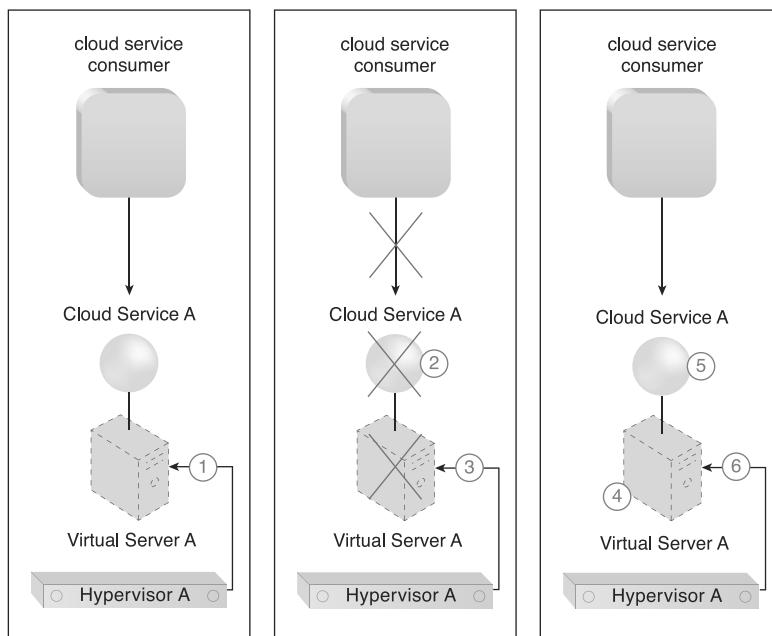
Cloud Service A becomes suddenly inaccessible to a cloud consumer.

Solution

Applying this pattern ensures that the operational status of a given virtual server is always being checked by the hypervisor on a routine basis. If the virtual server is not running or shows no signs of operation after a certain length of time, then the hypervisor takes action and restarts the virtual server automatically, to recover the virtual server from a crash.

The scenario that results is illustrated in Figure 4.39 in the following steps:

1. Hypervisor A is monitoring Virtual Server A's operational status.
2. Cloud Service A becomes unavailable due to an operating system failure on Virtual Server A.
3. Hypervisor A becomes aware of the nonoperational status of Virtual Server A immediately.
4. Hypervisor A restarts Virtual Server A.

**Figure 4.39**

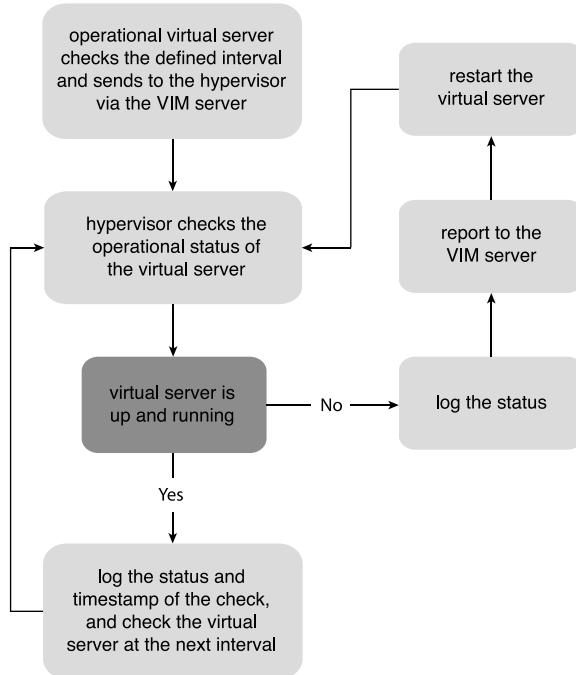
After the crash, Cloud Service A becomes available again.

5. Service resumes without requiring human interaction and cloud consumers can access the virtual server.
6. Hypervisor A continues to monitor the operational status of Virtual Server A.

Application

This pattern can be applied in different ways, depending on the brand and model of the hypervisor and the mechanism used to track the resource utilization of the virtual servers. The following chart in Figure 4.40 illustrates the steps involved in applying the pattern.

Different methods and mechanisms can be used to check the virtual server's operational status, such as a mechanism that can install an agent inside the virtual server that reports back to the hypervisor. Another mechanism is a hypervisor that checks the resource usage of the virtual server, including memory and CPU usage, at pre-defined intervals. A different method is to check the virtual server's network traffic and storage traffic for communication over the network and whether it is accessing or requesting any storage.

**Figure 4.40**

The steps involved in applying this pattern are shown.

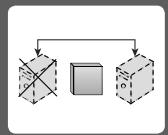
While this pattern ensures that virtual servers, applications, and services are operational and can be automatically recovered in the case of an operating system failure, this pattern may also restart the virtual server as a result of a “false positive.”

Mechanisms

- *Hypervisor* – The hypervisor mechanism hosts the virtual servers and is responsible for making sure the virtual servers are up and running. Any failed or crashed virtual servers are restarted by this mechanism.
- *Virtualization Agent* – This mechanism establishes one-way communication via specialized messages that are sent by the virtual servers to the host hypervisor at frequent and regular intervals to confirm virtual server operation.

Non-Disruptive Service Relocation

How can cloud service activity be temporarily or permanently relocated without causing service interruption?



Problem	There are circumstances under which redirecting cloud service activity or relocating an entire cloud service implementation is required or preferable. However, diverting service activity or relocating a cloud service implementation can cause outage, thereby disrupting the availability of the cloud service.
Solution	A system can be established whereby cloud service redirection or relocation is carried out at runtime by temporarily creating a duplicate implementation before the original implementation is deactivated or removed.
Application	Virtualization technology is used by the system to enable the duplication and migration of the cloud service implementation across different locations in realtime.
Mechanisms	Cloud Storage Device, Cloud Usage Monitor, Hypervisor, Live VM Migration, Pay-Per-Use Monitor, Resource Replication, SLA Management System, SLA Monitor, Virtual Infrastructure Manager (VIM), Virtual Server, Virtual Switch

Problem

A cloud service can become unavailable due to a number of reasons, such as:

- The cloud service encounters more runtime usage demand than it has processing capacity to handle.
- The cloud service implementation needs to undergo a maintenance update that mandates a temporary outage.
- The cloud service implementation needs to be permanently migrated to a new physical server host.

Cloud service consumer requests are rejected if a cloud service becomes unavailable, which can potentially result in exception conditions. Rendering the cloud service temporarily unavailable to cloud consumers is not preferred even if the outage is planned.

Solution

A system is established by which a pre-defined event triggers the duplication or migration of a cloud service implementation at runtime, thereby avoiding any disruption in service for cloud consumers.

An alternative to scaling cloud services in or out with redundant implementations, cloud service activity can be temporarily diverted to another hosting environment at runtime by adding a duplicate implementation onto a new host. Cloud service consumer requests can similarly be temporarily redirected to a duplicate implementation when the original implementation needs to undergo a maintenance outage. The relocation of the cloud service implementation and any cloud service activity can also be permanent to accommodate cloud service migrations to new physical server hosts.

Application

A key aspect to the underlying architecture is that the system ensures that the new cloud service implementation is successfully receiving and responding to cloud service consumer requests *before* the original cloud service implementation is deactivated or removed.

A common approach is to employ the live VM migration component to move the entire virtual server instance hosting the cloud service. The automated scaling listener and/or the load balancer mechanisms can be used to trigger a temporary redirection of cloud service consumer requests in response to scaling and workload distribution requirements. In this case either mechanism can contact the VIM to initiate the live VM migration process.

The steps involved in applying the Non-Disruptive Service Relocation pattern are illustrated in Figures 4.41 through 4.43.

1. The automated scaling listener monitors the workload for a cloud service.
2. As the workload increases, a pre-defined threshold within the cloud service is reached.
3. The automated scaling listener signals the VIM to initiate the relocation.
4. The VIM signals both the origin and destination hypervisors to carry out a runtime relocation via the use of a live VM migration program.
5. A second copy of the virtual server and its hosted cloud service are created via the destination hypervisor on Physical Server B.

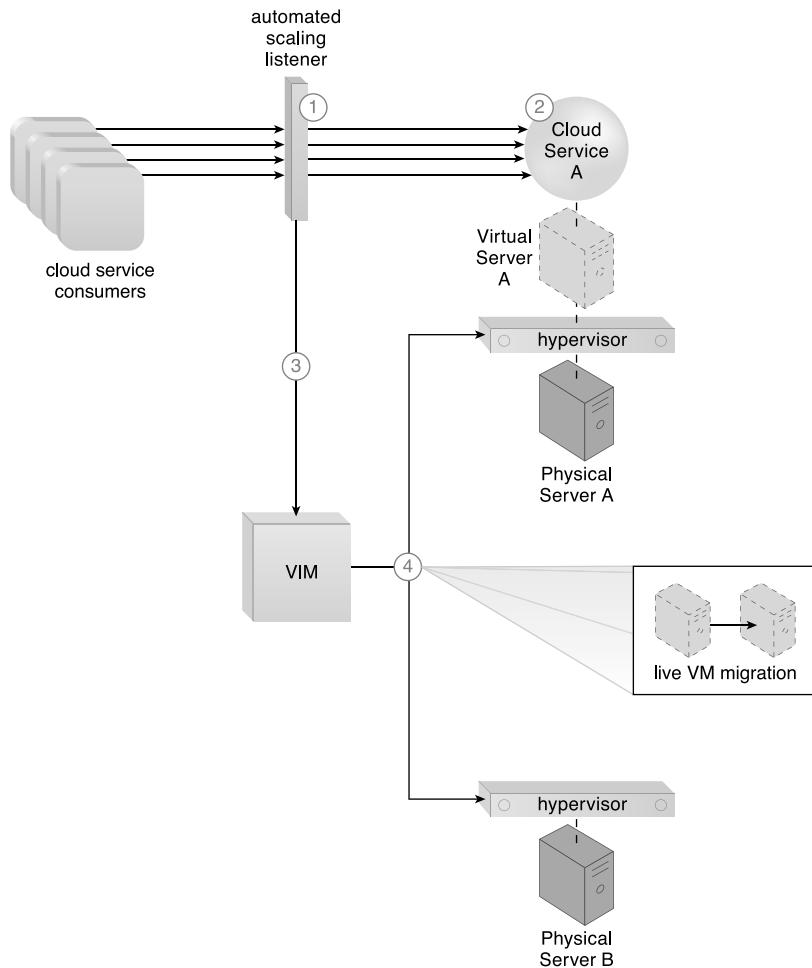


Figure 4.41

An example of a scaling-based application of the Non-Disruptive Service Relocation pattern (Part I).

6. The state of both virtual server instances is synchronized.
7. The first virtual server instance is removed from Physical Server A after it is confirmed that cloud service consumer requests are being successfully exchanged with the cloud service on Physical Server B.
8. Cloud service consumer requests are only sent to the cloud service on Physical Server B from hereon.

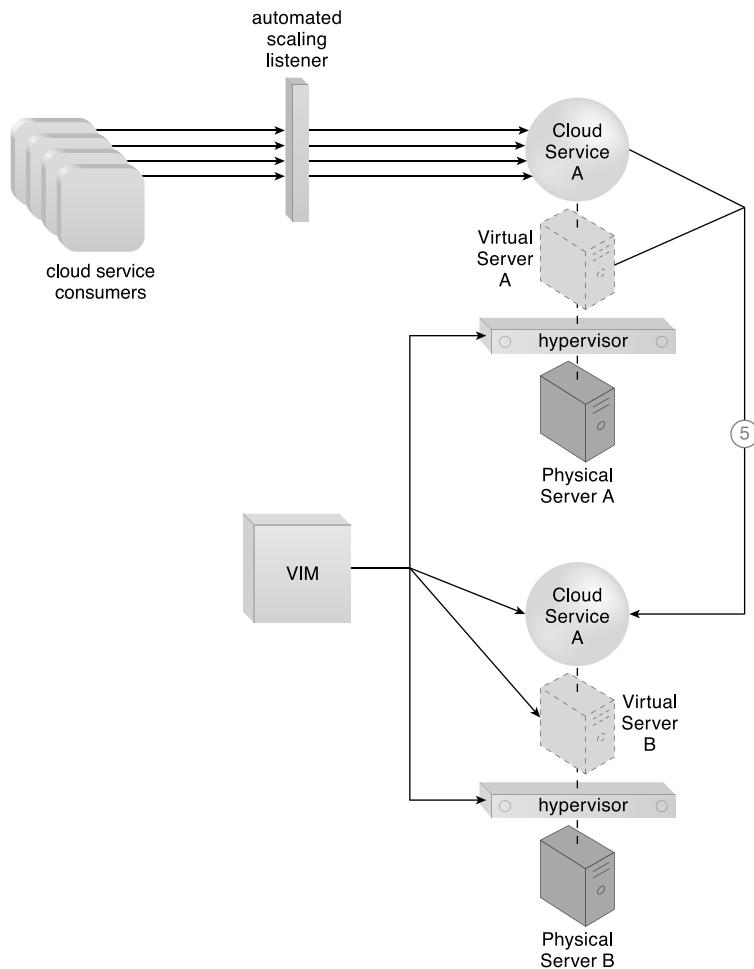
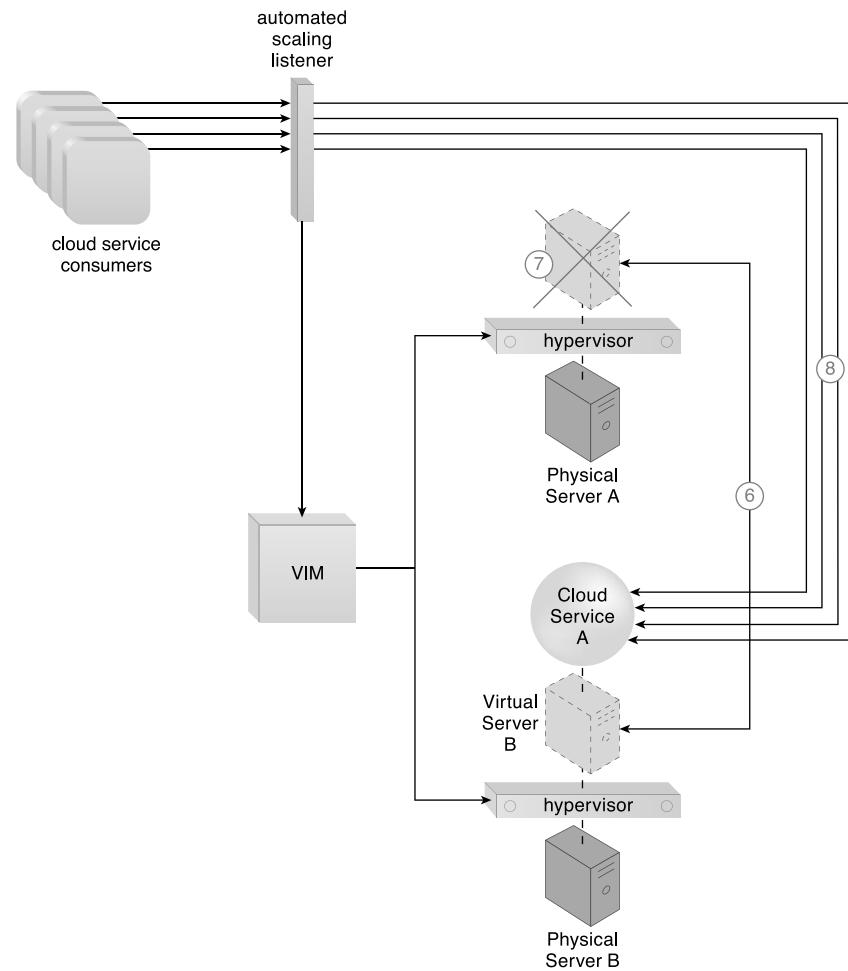


Figure 4.42

An example of a scaling-based application of the Non-Disruptive Service Relocation pattern (Part II).

**Figure 4.43**

An example of a scaling-based application of the Non-Disruptive Service Relocation pattern (Part III).

Depending on the location of the virtual server's disks and configuration, this migration can happen in one of two ways:

- If the virtual server disks are stored on a local storage device or on non-shared remote storage devices attached to the source host, then a copy of the virtual server disks is created on the destination host (either on a local or remote shared/non-shared storage device). After the copy has been created, both virtual server instances are synchronized and virtual server files are subsequently removed from the origin host.
- If the virtual server's files are stored on a remote storage device shared between origin and destination hosts, there is no need to create the copy of virtual server disks. In this case, the ownership of the virtual server is simply transferred from the origin to the destination physical server host, and the virtual server's state is automatically synchronized.

Note that this pattern conflicts and cannot be applied together with Direct I/O Access (169). A virtual server with direct I/O access is locked into its physical server host and cannot be moved to other hosts in this fashion.

Furthermore, Persistent Virtual Network Configuration (227) may need to be applied in support of this pattern so that by moving the virtual server, its defined network configuration is not inadvertently lost, which would prevent cloud service consumers from being able to connect to the virtual server.

Mechanisms

- *Cloud Storage Device* – This mechanism is fundamental to the Non-Disruptive Service Relocation pattern in how it provides the storage required to host data pertaining to the virtual servers in a central location.
- *Cloud Usage Monitor* – Cloud usage monitors are used to continuously track IT resource usage and activity of the system established by the Non-Disruptive Service Relocation pattern.
- *Hypervisor* – The hypervisor is associated with this pattern in how it is used to host the virtual servers that are hosting the cloud services that need to be relocated. It is further used to transfer a virtual server's ownership and runtime, including CPU and memory state, from one hypervisor to another.

- *Live VM Migration* – This mechanism is responsible for transferring the ownership and runtime information of a virtual server from one hypervisor to another.
- *Pay-Per-Use Monitor* – The pay-per-use monitor is used to continuously collect the service usage costs of the IT resources at both their source and destination locations.
- *Resource Replication* – The resource replication mechanism is used to instantiate the shadow copy of the cloud service at its destination.
- *SLA Management System* – The SLA management system is responsible for acquiring SLA information from the SLA monitor, in order to obtain cloud service availability assurances both during and after the cloud service has been copied or relocated.
- *SLA Monitor* – This monitoring mechanism collects the aforementioned information required by the SLA management system.
- *Virtual Infrastructure Manager (VIM)* – This mechanism is used to initiate relocation, which can be automated in response to a threshold being reached or monitoring event.
- *Virtual Server* – Virtual servers generally host the cloud services at the source and destination locations.
- *Virtual Switch* – The virtual switch mechanism keeps virtual servers connected to and accessible over the network.

This page intentionally left blank

About the Authors

Thomas Erl

Thomas Erl is a top-selling IT author, founder of Arcitura Education Inc., and series editor of the *Prentice Hall Service Technology Series from Thomas Erl*. With more than 200,000 copies in print worldwide, his books have become international bestsellers and have been formally endorsed by senior members of major IT organizations, such as IBM, Microsoft, Oracle, Intel, Accenture, IEEE, HL7, MITRE, SAP, CISCO, HP, and many others. As CEO of Arcitura Education Inc., Thomas has led the development of curricula for the internationally recognized Big Data Science Certified Professional (BDSCP), Cloud Certified Professional (CCP), and SOA Certified Professional (SOACP) accreditation programs, which have established a series of formal, vendor-neutral industry certifications obtained by thousands of IT professionals around the world. Thomas has toured more than 20 countries as a speaker and instructor. More than 100 articles and interviews by Thomas have been published in numerous publications, including *The Wall Street Journal* and *CIO Magazine*.

Robert Cope

Robert Cope has more than 25 years of experience in mission-critical systems development, spanning all aspects of the software system engineering lifecycle from architectural development, experimentation and prototyping, requirements development, design, implementation, and operations to acquisition program management for large systems. With more than 10 years in research, development, and implementation of security architecture, Public Key Infrastructure (PKI) security technology, and security services for large organizations, he has vast experience in information assurance, identity management deployment, operations, and maintenance of large-scale high assurance identity management enclaves.

Robert is the CEO of Homeland Security Consultants, a Federal Risk and Authorization Management Program (FedRAMP)-approved Third Party Assessment Organization (3PAO) for certifying cloud services. He led the development of the virtualization and cloud computing architecture for a large organization and was the chief architect responsible for the development of an enterprise authentication service, leading a team to integrate the organization's identity and access management service architecture using Model Based System Engineering (MBSE) and the System Modeling Language (SysML).

Robert is a Certified Trainer for Arcitura's Cloud School and SOA School. He has been a contributing member of the National Institute of Standards and Technology (NIST) Cloud-adapted Risk Management Framework (CRMF) and a contributing member of the Organization for the Advancement of Structured Information Standards (OASIS) IdCloud Technical Committee. He is also a member of the International Council on Systems Engineering (INCOSE).

Amin Naserpour

A certified IT professional with more than 14 years of experience in solution architecture and design, engineering, and consultation, Amin Naserpour specializes in designing medium to enterprise-level complex solutions for partially to fully virtualized front-end infrastructures. His portfolio includes clients such as VMware, Microsoft, and Citrix, and his work consists of integrating front-ends with back-end infrastructure-layer solutions. Amin designed a unified, vendor-independent cloud computing framework that he presented at the 5th International SOA, Cloud + Service Technology Symposium in 2012. Certified in cloud computing, virtualization, and storage, Amin currently holds Technical Consultant and Cloud Operations Lead positions for Hewlett-Packard, Australia.

Index

A

ABAC. *See* attribute based access control (ABAC)
access control, 513
ADC. *See* application delivery controller (ADC)
ADP. *See* automatically defined perimeter (ADP) controller
AGS. *See* authentication gateway service (AGS)
application delivery controller (ADC), 403
 defined, 512
application layer attacks, 417
attestation service, 190, 192, 358, 451
 defined, 512
attribute authority, 368
 defined, 513
attribute based access control (ABAC), 368
 defined, 513
attribute store. *See* attribute authority
audit monitor, 21, 24, 69, 79, 103, 110, 126, 144, 287, 294, 318
 defined, 513
authentication gateway service (AGS), 363, 366-368, 435
 defined, 513
Automated Administration design pattern, 38, 284, 302, 475, 477, 479, 481, 483, 493, 499
 profile, 310-314

automated scaling listener, 22, 28, 31, 40, 43, 55, 69, 79, 85, 213, 287, 314
 defined, 514
automatically defined perimeter (ADP) controller, 429
 defined, 514
Automatically Defined Perimeter design pattern, 510
 profile, 425-429

B

bandwidth, 43
Bare-Metal Provisioning design pattern, 475, 477, 483
 profile, 305-309
bare metal virtualization, 344
basic input/output system (BIOS), 337
BGP (Border Gateway Protocol), 421
billing management system, 291, 318
 defined, 514
BIOS (basic input/output system), 337
BIOS/firmware rootkits, 337
bootkits, 338
Border Gateway Protocol (BGP), 421
botnets, 417
Broad Access design pattern, 16, 318, 475, 477, 479, 481, 483
 profile, 93-95

- Burst In compound pattern**, 473, 492
 profile, 499-500
- Burst Out to Private Cloud compound pattern**, 473, 496-497, 501
 profile, 493-495
- Burst Out to Public Cloud compound pattern**, 473, 492, 501
 profile, 496-498
- C**
- CA.** *See certificate authority (CA)*
- capacity watchdog system**, 52-55
- capitalization in design pattern notation**, 13
- CCG.** *See cloud consumer gateway (CCG)*
- CCP (Cloud Certified Professional)**, 6
- Centralized Remote Administration design pattern**, 284, 293, 321, 475, 477, 479, 481, 483
 profile, 315-319
- certificate**, 358, 435, 443
 defined, 514
- certificate authority (CA)**, 435, 443
 defined, 515
- certificate revocation list (CRL)**, 435, 443
 defined, 515
- certificate trust store**, 430, 435-436, 443
 defined, 515
- certificate validation service (CVS)**, 443
 defined, 515
- CKMS.** *See cryptographic key management system (CKMS)*
- Cloud Authentication compound pattern**, 473
 profile, 505
- Cloud Authentication Gateway design pattern**, 502, 505, 509
 profile, 430-435
- Cloud Balancing compound pattern**, 473
 profile, 503-504
- cloud-based security groups**, 354, 358, 364, 368, 409, 415
 defined, 517
- Cloud Bursting compound pattern**, 473
 profile, 492
- Cloud Computing: Concepts, Technology & Architecture (Erl)**, 2-5, 14
- cloud consumer gateway (CCG)**, 408
 defined, 516
- Cloud Data Breach Protection design pattern**, profile, 382-385
- Cloud Denial-of-Service Protection design pattern**, profile, 416-420
- Cloud Key Management design pattern**, 406, 509
 profile, 444-447
- Cloud Resource Access Control design pattern**, 502, 509
 profile, 364-368
- cloud service types**, 453
- cloud storage data aging management**, 186
- Cloud Storage Data at Rest Encryption design pattern**, profile, 181-183
- Cloud Storage Data Lifecycle Management design pattern**, profile, 184-186
- Cloud Storage Data Management design pattern**, profile, 187-189
- cloud storage data placement auditor**, 192
 defined, 516
- Cloud Storage Data Placement Compliance Check design pattern**, profile, 190-193
- cloud storage device**, 21, 24, 31, 49, 55, 63, 69, 73, 79, 85, 103, 110, 117, 122, 131, 142, 145, 154, 164, 171, 176, 183, 186, 189, 192, 197, 209, 213, 217, 226, 250, 282, 299, 308, 314, 394
 defined, 516
- Cloud Storage Device Masking design pattern**, profile, 194-197
- Cloud Storage Device Path Masking design pattern**, profile, 198
- Cloud Storage Device Performance Enforcement design pattern**, profile, 201-203
- cloud storage device performance monitor**, 201-203
 defined, 516

- cloud storage device pools**, 101
cloud storage management portal, 189, 217, 220
 defined, 517
Cloud Traffic Hijacking Protection design pattern, profile, 421-424
cloud usage monitor, 21, 24, 31, 36, 40, 43, 48-49, 55, 60, 63, 69, 80, 85, 104, 111, 126, 145, 164, 171, 176, 287, 291, 294, 314, 318
 defined, 517
Cloud VM Platform Encryption design pattern, profile, 509
 profile, 350-353
cloud workload scheduler, 357-358
 defined, 517
coexistent application
 of compound patterns, 473
 defined, 13
Collaborative Monitoring and Logging design pattern, 509
 profile, 452-459
community clouds, 453
compound patterns
 Burst In, 473, 492
 profile, 499-500
 Burst Out to Private Cloud, 473, 492, 496-497, 501
 profile, 493-495
 Burst Out to Public Cloud, 473, 492, 501
 profile, 496-498
 Cloud Authentication, 473
 profile, 505
 Cloud Balancing, 473
 profile, 503-504
 Cloud Bursting, 473
 profile, 492
 coexistent application of, 473
 composite patterns versus, 472
 defined, 12-13
 design patterns as members, 472
 Elastic Environment, 473-477
 profile, 484-485
 Infrastructure-as-a-Service, 20, 473
 profile, 482-483
 Isolated Trust Boundary, 477, 481, 486-487
 profile, 508-510
 joint application of, 472
 Multitenant Environment, 473-483, 494-496
 profile, 486-489
 Platform-as-a-Service, 473, 486
 profile, 480-481
 Private Cloud, 473, 476, 482-486, 490
 profile, 474-475
 Public Cloud, 20, 473-474, 482-486, 490
 profile, 476-477
 Resilient Environment, 473-477
 profile, 490-491
 Resource Workload Management, 473
 profile, 506
 Secure Burst Out to Private Cloud/Public Cloud, 473
 profile, 501-502
 Software-as-a-Service, 20, 473, 486
 profile, 478-479
CPU pools, 101
CRL. *See certificate revocation list (CRL)*
Cross-Hypervisor Workload Mobility design pattern, profile, 247-251
Cross-Storage Device Vertical Tiering design pattern, 485, 494, 499
 profile, 74-80
cryptographic key management system (CKMS), 183, 197, 353, 386, 390, 394, 424, 447
 defined, 517
custom reporter (Usage Monitoring design pattern), 287
custom scripts (Rapid Provisioning design pattern), 296
CVS. *See certificate validation service (CVS)*

D

data normalization, 71-73
data source loader, 290
data transport mechanism, 186
denial-of-service (DoS) attacks, 416-420
deployment agent, 306
deployment component, 306
deployment data store, 296
design patterns. *See also compound patterns*
 benefits of, 10
 defined, 2
 as members of compound patterns, 472
 list of, 536
 notation for
 capitalization, 13
 page number references, 13
 profile format, 11-12
 Web site, 6, 14
Design Patterns: Elements of Reusable Object-Oriented Software (Gamma, et al), 3
Detecting and Mitigating User-Installed VMs design pattern, profile, 369-374
digital certificates. *See certificate*
digital signature, 340, 349, 359, 394, 415, 451
 defined, 518
DIL procedures, 443
Direct I/O Access design pattern, 43, 164, 178-179, 485
 profile, 169-172
Direct LUN Access design pattern, 485
 profile, 173-177
discovery agent, 306
distributed denial-of-service (DDoS) attacks, 416
distributed reflector denial-of-service (DRDoS) attacks, 416
DNS reflection attacks, 416
domain name service (DNS), 403, 420
 defined, 518

driver rootkits, 338

Dynamic Data Normalization design pattern, 16, 485
 profile, 71-73
Dynamic Failure Detection and Recovery design pattern, 98, 490
 profile, 123-126
dynamic horizontal scaling, 28-31
dynamic relocation, 29
Dynamic Scalability design pattern, 38, 99-100, 479, 481, 483, 493
 profile, 25-31
dynamic storage provisioning, 46
dynamic vertical scaling, 29

E

Elastic Disk Provisioning design pattern, 485
 profile, 45-50
Elastic Environment compound pattern, 473, 475, 477
 profile, 484-485
Elastic Network Capacity design pattern, 485
 profile, 42-44
Elastic Resource Capacity design pattern, 485, 493, 504
 profile, 37-41
EMM system. *See enterprise mobility management (EMM) system*
encryption, 181-183, 197, 390, 394, 424
 defined, 518
endpoint threat detection and response (ETDR) system, 469
 defined, 518
enterprise mobility management (EMM) system, 381
 defined, 519
External Virtual Server Accessibility design pattern, 242
 profile, 244-246

F

- failover system**, 122, 126, 136, 142, 145, 154
 - defined, 519
- Federated Cloud Authentication design pattern**, 505, 510
 - profile, 436-443
- federation of users**, 443
- firewalls (Secure Connection for Scaled VMs design pattern)**, 409-415
- fixed-disk storage allocation**, 45

G

- gateway**. *See* **cloud consumer gateway (CCG)**
- Geotagging design pattern**, 502, 510
 - profile, 341-343
- geotags**, 192, 343, 359
 - defined, 519

H

- hardened virtual server images**, defined, 519
- hardware-based VM discovery system**, 374
 - defined, 520
- hardware security module (HSM)**, 340, 447
 - defined, 520
- honeypots**, 469
 - defined, 520
- host-based security system (HBSS)**, 375
 - defined, 521
- hosted virtualization**, 345
- HSM**. *See* **hardware security module (HSM)**
- hybrid clouds**, 453
- hypervisor**
 - defined, 521
 - purpose of, 222
- Hypervisor Clustering design pattern**, 98, 269, 491, 503
 - profile, 112-118
- Hypervisor Protection design pattern**, 509
 - profile, 344-349

I

- IaaS**. *See* **Infrastructure-as-a-Service compound pattern; Infrastructure-as-a-Service environments**
- icons in pattern profiles**, 11
- identity and access management (IAM) system**, 189, 363, 366-368
 - defined, 521
- IDPS**. *See* **intrusion detection and prevention system (IDPS)**
- Independent Cloud Auditing design pattern**, 510
 - profile, 460-464
- Infrastructure-as-a-Service compound pattern**, 20, 473
 - profile, 482-483
- Infrastructure-as-a-Service environments**, flexibility in, 222
- intelligent automation engine**, 43, 311-314
- intelligent watchdog monitor**, 125-126
- interconnect pools**, 101
- In-Transit Cloud Data Encryption design pattern**, 510
 - profile, 391-394
- Intra-Storage Device Vertical Data Tiering design pattern**, 485
 - profile, 81-85
- intrusion detection and prevention system (IDPS)**, 403, 469
 - defined, 522
- IP Storage Isolation design pattern**, profile, 218-220
- Isolated Trust Boundary compound pattern**, 477, 481, 486-487
 - profile, 508-510
- IT resources**
 - dynamic scaling, 27
 - horizontal scaling, 22
 - sharing, risks and challenges, 20

J-K**joint application**

- of compound patterns, 472
- defined, 13

kernel rootkits, 338

Key Management design pattern. *See* **Cloud Key Management design pattern**

L**live VM migration**, 40, 56, 145, 165, 251, 257, 264, 271, 277, 334, 415

- defined, 522

Load Balanced Virtual Server Instances

- design pattern**, 254, 261, 269, 276, 331, 503, 506

- limitations of, 253

- profile, 51-56

Load Balanced Virtual Switches design

- pattern**, 237, 245, 491, 506

- profile, 57-60

load balancer, 22-24, 33-36, 56, 60, 70, 287

- defined, 522

logical network perimeter, 21, 24, 43, 56, 60,

- 70, 104, 111, 118, 131, 136, 145, 171, 229,

- 308, 318,

- defined, 523

LUN masking, 197, 220

- defined, 523

M**malware hashes**, 469

- defined, 523

management loader, 306**management portal**. *See* **cloud storage**

- management portal**

measured boot, 339**mechanisms in pattern profiles**, 12**Memory Over-Committing design**

- pattern**, 16

- profile, 86-89

Mobile BYOD Security design pattern,

- profile, 376-381

multi-device broker, 94, 318

- defined, 523

Multipath Resource Access design

- pattern**, 482

- profile, 127-131

multitenancy, virtualization versus, 487**Multitenant Environment compound**

- pattern**, 473-475, 477, 479, 481, 483, 494, 496

- profile, 486-489

N**nested resource pools**, 102**network bandwidth**, 43**network forensics monitor (NFM)**

- defined, 524

NIC Teaming design pattern, 16

- profile, 90-92

Non-Disruptive Service Relocation design

- pattern**, 180, 260, 475, 477, 479, 481

- profile, 159-165

normalization (Dynamic Data

- Normalization design pattern**), 71-73

notification service for this book series, 7**O****Open Virtualization Format (OVF)**,

- converting virtual servers to, 248

operating system baseline (Rapid

- Provisioning design pattern**), 296

orchestration engine, 451

- defined, 524

O/S boot load bootkits, 338**OVF (Open Virtualization Format)**,

- converting virtual servers to, 248

P-Q**PaaS**. *See* **Platform-as-a-Service compound**

- pattern**; **Platform-as-a-Service**

- environments**

page number references in design pattern

- notation**, 13

- parent resource pools, 101
pattern languages, defined, 11
Pattern-Oriented Software Architecture (Buschmann, et al), 3
pattern profile format, 11-12
patterns, defined, 10. *See also* compound patterns; design patterns
Patterns of Enterprise Application Architecture (Fowler), 3
Pay-as-You-Go design pattern, 284-285, 475, 477, 479, 481, 483
 profile, 288-291
pay-per-use monitor, 31, 41, 43, 50, 63, 80, 85, 104, 165, 171, 176, 287, 291, 318
 defined, 524
Permanent Data Loss Protection design pattern, profile, 387-390
Persistent Virtual Network Configuration design pattern, 144, 164, 234, 493, 504
 profile, 227-230
physical RAM pools, 101
physical server pools, 100
physical uplink, 60, 92, 136, 145, 229, 234, 238, 243, 246, 251
 defined, 524
PKI. *See* public key infrastructure (PKI)
Platform-as-a-Service compound pattern, 473, 486
 profile, 480-481
Platform-as-a-Service environments, networking interfaces, 222
Platform Provisioning design pattern, 284, 481
 profile, 301-304
platform trust policy, 343, 359
 defined, 524
PNIC hardware devices, functionality, 179
pools. *See* Resource Pooling design pattern
Power Consumption Reduction design pattern, profile, 330-334
Prentice Hall Service Technology Series from Thomas Erl, 2-6
pre-signed validations, 442
Private Cloud compound pattern, 473, 476, 482, 484, 486, 490
 profile, 474-475
private clouds, 453
problems in pattern profiles, 11
protocol attacks, 417
Public Cloud compound pattern, 20, 473-474, 482, 484, 486, 490
 profile, 476-477
public clouds, 453
public key certificates. *See* certificate
public key infrastructure (PKI), 435, 443
 defined, 525
- R**
- RAID-Based Data Placement design pattern, profile, 214-217
RAID-level identifier, 217, 220
 defined, 525
Rapid Provisioning design pattern, 284, 302, 475, 477, 479, 481, 483, 485, 491, 493-494, 503
 profile, 295-300
ready-made environment, 304
 defined, 525
Realtime Resource Availability design pattern, 284, 319, 475, 477, 479, 481, 483
 profile, 292-294
Redundant Physical Connection for Virtual Servers design pattern, 245, 490, 504
 profile, 132-137
Redundant Storage design pattern, 98, 485, 490, 493, 504
 profile, 119-122
remote administration system, 104, 111, 319
 defined, 525
requirements in pattern profiles, 11
Resilient Environment compound pattern, 473, 475, 477
 profile, 490-491

- resilient watchdog system**, 123-125
 - resource borrowing**, 106
 - resource cluster**, 24, 36, 56, 118, 145
 - defined, 526
 - resource constraints**, 106
 - Resource Management design pattern**, 475, 477, 479, 481, 483
 - profile, 320
 - resource management system**, 104, 111, 304, 308, 319
 - defined, 526
 - Resource Pooling design pattern**, 20, 28, 38, 98, 106-107, 475, 477, 481, 483, 485-487, 494, 496, 499
 - profile, 99-105
 - resource replication**, 21, 24, 31, 36, 41, 44, 50, 56, 60, 63, 104, 111, 118, 122, 131, 136, 142, 145, 154, 165, 171, 177, 193, 229, 300, 304, 309, 314
 - defined, 526
 - Resource Reservation design pattern**, 20, 88, 98, 100, 477, 479, 481, 485-487, 494, 496
 - profile, 106-111
 - Resource Workload Management compound pattern**, 473
 - profile, 506
 - rootkits**, types of, 337
- S**
- SaaS**. *See Software-as-a-Service compound pattern; Software-as-a-Service environments*
 - sandbox**, 469
 - defined, 526
 - Secure Burst Out to Private Cloud/ Public Cloud compound pattern**, 473
 - profile, 501-502
 - Secure Cloud Interfaces and APIs design pattern**, 510
 - profile, 360-363
 - Secure Connection for Scaled VMs design pattern**, 502, 510
 - profile, 409-415
 - Secure External Cloud Connection design pattern, profile**, 404-408
 - secure firmware boot**, 339
 - Secure On-Premise Internet Access design pattern, profile**, 397-403
 - secure token service (STS)**, 368, 435
 - defined, 526
 - security information and event management (SIEM) system**, 403, 459, 464, 469
 - defined, 526
 - Self-Provisioning design pattern**, 297, 302, 318, 475, 477, 479, 481, 483
 - profile, 324-329
 - sequence logger**, 296
 - sequence manager**, 296
 - server groups**, 35
 - server images**, 296
 - server templates**, 296
 - Service Load Balancing design pattern**, 485, 491, 494, 503
 - profile, 32-36
 - Service State Management design pattern**, 481
 - profile, 61-63
 - Shared Resources design pattern**, 16, 99-100, 106, 475, 477, 479, 481, 483, 486-487
 - profile, 17-21
 - sibling resource pools**, 101
 - SIEM**. *See security information and event management (SIEM) system*
 - Single Root I/O Virtualization design pattern**, profile, 178-180
 - single sign-on (SSO)**, defined, 527
 - SLA management system**, 126, 165, 294, 309
 - defined, 527
 - SLA monitor**, 126, 165, 287, 294, 319
 - defined, 527
 - SOA Design Patterns (Erl)**, 3
 - Software-as-a-Service compound pattern**, 20, 473, 486
 - profile, 478-479

Software-as-a-Service environments, networking interfaces, 222
solutions in pattern profiles, 12
SSO (single sign-on), defined, 527
statefulness, 61-63
Stateless Hypervisor design pattern, profile, 278-282
state management database, 63, 142 defined, 527
Storage Maintenance Window design pattern, 491 profile, 147-154
storage path masking, 220 defined, 528
storage pools, 101
Storage Workload Management design pattern, 485, 504, 506 profile, 64-70
STS. *See* secure token service (STS)
sub-LUN migration, 213 defined, 528
Sub-LUN Tiering design pattern, profile, 210-213
symbols, legend, 5
Synchronized Operating State design pattern, 491 profile, 138-142

T

thin provisioning, 46-48
Threat Intelligence Processing design pattern, profile, 465-469
threat intelligence system, 386, 469 defined, 528
TPM (trusted platform module), 193, 339-340, 343, 349, 359, 451, 529
traffic filter, 420 defined, 528
traffic monitor, 420, 424 defined, 529
trust attestation service. *See* attestation service

Trust Attestation Service design pattern, 502, 510 profile, 448-451
trusted boot, 339
Trusted Cloud Resource Pools design pattern, 502, 510 profile, 354-359
Trusted Platform BIOS design pattern, 502, 510 profile, 337-340
trusted platform module (TPM), 193, 339-340, 343, 349, 359, 451, 529
trust models for CVS, 442

U

usage database, 286
Usage Monitoring design pattern, 284, 289, 475, 477, 479, 481, 483, 485, 491, 493, 499 profile, 285-287
usage monitoring station, 286
usage reporter, 287

V

vCPU. *See* virtual CPU (vCPU)
vDisk. *See* virtual disk (vDisk)
VIM. *See* virtual infrastructure manager (VIM)
virtual appliance Cross-Hypervisor Workload Mobility design pattern, 251 defined, 529
virtual CPU (vCPU) Cross-Hypervisor Workload Mobility design pattern, 21, 41, 56, 104, 111, 145, 251 defined, 529
Virtual Disk Splitting design pattern, profile, 209
virtual disk (vDisk), 145, 209, 251 defined, 530
virtual firewall, 234, 415 defined, 530

- virtual infrastructure manager (VIM),** 21, 41, 56, 60, 89, 92, 104, 111, 118, 136, 146, 165, 172, 177, 209, 226, 230, 234, 238, 246, 251, 257, 264, 271, 277, 282, 334, 375
defined, 530
- virtual machines (VMs).** *See* **virtual server**
- virtual network,** 146, 243, 251, 239
defined, 530
- virtual private cloud (VPC),** 408
defined, 531
- virtual private network (VPN),** 403, 408, 429, 435
defined, 531
- virtual private network (VPN) cloud hub.**
See **VPN cloud hub**
- virtual RAM (vRAM),** 21, 41, 56, 89, 104, 111, 146
defined, 531
- virtual server,** 21, 24, 31, 41, 44, 51-56, 60, 63, 105, 111, 118, 131-137, 142, 146, 165, 172, 177, 230, 248, 300, 304, 314, 353, 369-374, 522
defined, 531
- Virtual Server Auto Crash Recovery design pattern, profile,** 155-158
- Virtual Server Connectivity Isolation design pattern, profile,** 231-234
- Virtual Server Folder Migration design pattern, profile,** 223-226
- Virtual Server NAT Connectivity design pattern, profile,** 240-243
- virtual server pools,** 100
- virtual server snapshot,** 251
defined, 532
- virtual server state manager,** 251
defined, 532
- Virtual Server-to-Host Affinity design pattern, profile,** 252-257
- Virtual Server-to-Host Anti-Affinity design pattern, profile,** 258-264
- Virtual Server-to-Host Connectivity design pattern, profile,** 265-266
- Virtual Server-to-Virtual Server Affinity design pattern, profile,** 234
profile, 267-271
- Virtual Server-to-Virtual Server Anti-Affinity design pattern, profile,** 272-277
- Virtual Switch Isolation design pattern, profile,** 235-239
- virtual switches,** 56-60, 92, 118, 137, 146, 165, 230, 234, 239, 243, 246, 251, 266
defined, 532
- virtualization,** 19, 487
types of, 344
- virtualization agent,** 89, 146, 158
defined, 532
- virtualization monitor,** 56, 89, 118, 146, 209, 334
defined, 533
- VMs (virtual machines).** *See* **virtual server**
- volume-based attacks,** 416-417
- VPC.** *See* **virtual private cloud (VPC)**
- VPN.** *See* **virtual private network (VPN)**
- VPN cloud hub,** 408
defined, 533
- vRAM.** *See* **virtual RAM (vRAM)**

W

Web sites

- www.cloudpatterns.org, 6, 14
- www.cloudschool.com, 6
- www.servicetechbooks.com, 2, 6-7
- www.servicetechmag.com, 6
- www.servicetechspecs.com, 6
- www.whatiscloud.com, 6, 14

Workload Distribution design pattern, 475, 477, 479, 481-482, 485, 491, 493, 504
profile, 22-24

workloads, defined, 517

X-Y-Z

X.509 certificates. *See* **certificate**

Zero Downtime design pattern, 98, 491
profile, 143-146