



Stephen Cole, Gareth Digby, Christopher Fitch,
Steve Friedberg, Shaun Qualheim, Jerry Rhoads,
Michael Roth, Blaine Sundrud

AWS Certified SysOps Administrator

OFFICIAL STUDY GUIDE

ASSOCIATE EXAM

A photograph of a white lighthouse with a black lantern room, situated on a rocky island or coastline. In the foreground, there's a large, craggy rock formation. To the left of the lighthouse is a white house with a red roof. The background shows a calm sea under a clear sky.

Covers exam objectives, including monitoring and metrics, high availability, analysis, deployment and provisioning, data management, security, networking, and much more...

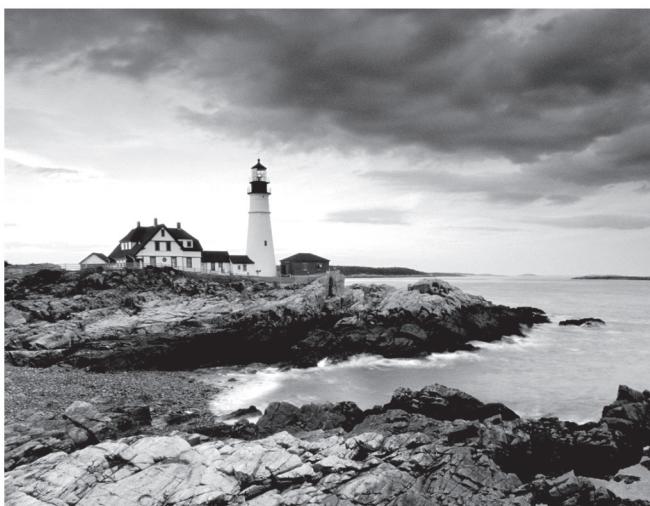
Includes an interactive online learning environment and study tools with:

- + 2 custom practice exams
- + 100 electronic flashcards
- + Searchable key term glossary

AWS

Certified SysOps Administrator Official

Study Guide - Associate Exam



Stephen Cole, Gareth Digby, Christopher Fitch,
Steve Friedberg, Shaun Qualheim, Jerry Rhoads,
Michael Roth, Blaine Sundrud



Senior Acquisitions Editor: Kenyon Brown
Development Editor: Gary Schwartz
Production Editor: Rebecca Anderson
Copy Editor: Kezia Endsley
Editorial Manager: Mary Beth Wakefield
Production Manager: Kathleen Wisor
Executive Editor: Jim Minatel
Book Designers: Judy Fung and Bill Gibson
Proofreader: Nancy Carrasco
Indexer: Robert Swanson
Project Coordinator, Cover: Brent Savage
Cover Designer: Wiley
Cover Image: © Getty Images Inc./Jeremy Woodhouse

Copyright © 2018 by Amazon Web Services, Inc.

Published by John Wiley & Sons, Inc. Indianapolis, Indiana

Published simultaneously in Canada

ISBN: 978-1-119-37742-9

ISBN: 978-1-119-37744-3 (ebk.)

ISBN: 978-1-119-37743-6 (ebk.)

Manufactured in the United States of America

No part of this publication may be reproduced, stored in a retrieval system or transmitted in any form or by any means, electronic, mechanical, photocopying, recording, scanning or otherwise, except as permitted under Sections 107 or 108 of the 1976 United States Copyright Act, without either the prior written permission of the Publisher, or authorization through payment of the appropriate per-copy fee to the Copyright Clearance Center, 222 Rosewood Drive, Danvers, MA 01923, (978) 750-8400, fax (978) 646-8600. Requests to the Publisher for permission should be addressed to the Permissions Department, John Wiley & Sons, Inc., 111 River Street, Hoboken, NJ 07030, (201) 748-6011, fax (201) 748-6008, or online at <http://www.wiley.com/go/permissions>.

Limit of Liability/Disclaimer of Warranty: The publisher and the author make no representations or warranties with respect to the accuracy or completeness of the contents of this work and specifically disclaim all warranties, including without limitation warranties of fitness for a particular purpose. No warranty may be created or extended by sales or promotional materials. The advice and strategies contained herein may not be suitable for every situation. This work is sold with the understanding that the publisher is not engaged in rendering legal, accounting, or other professional services. If professional assistance is required, the services of a competent professional person should be sought. Neither the publisher nor the author shall be liable for damages arising herefrom. The fact that an organization or Web site is referred to in this work as a citation and/or a potential source of further information does not mean that the author or the publisher endorses the information the organization or Web site may provide or recommendations it may make. Further, readers should be aware that Internet Web sites listed in this work may have changed or disappeared between when this work was written and when it is read.

For general information on our other products and services or to obtain technical support, please contact our Customer Care Department within the U.S. at (877) 762-2974, outside the U.S. at (317) 572-3993 or fax (317) 572-4002.

Wiley publishes in a variety of print and electronic formats and by print-on-demand. Some material included with standard print versions of this book may not be included in e-books or in print-on-demand. If this book refers to media such as a CD or DVD that is not included in the version you purchased, you may download this material at <http://booksupport.wiley.com>. For more information about Wiley products, visit www.wiley.com.

Library of Congress Control Number: 2017947567

TRADEMARKS: Wiley, the Wiley logo, and the Sybex logo are trademarks or registered trademarks of John Wiley & Sons, Inc. and/or its affiliates, in the United States and other countries, and may not be used without written permission. AWS is a registered trademark of Amazon Technologies, Inc. All other trademarks are the property of their respective owners. John Wiley & Sons, Inc. is not associated with any product or vendor mentioned in this book.

10 9 8 7 6 5 4 3 2 1

For our customers (whom we are always obsessing over). May this book find you well in your pursuit of becoming an AWS Certified Systems Operator.

Acknowledgments

The authors would like to thank a few people who helped us develop and write the *AWS Certified SysOps Administrator Official Study Guide – Associate Exam*.

First and foremost, a very big thank you to all of our friends and families who put up with us spending weekends, evenings, and vacations creating content, writing questions, and reviewing each other's chapters. Their flexibility, patience, and support made this book a reality.

Thank you to Nathan Bower and Victoria Steidel, the technical writers at AWS who reviewed and edited every single piece of content. They were always willing to review content, and their due diligence kept us on schedule. Their wisdom made us all better writers.

We could not have written this book without the help of our friends at Wiley. Gary Schwartz, Project Editor, provided valuable advice that kept us on track with our deliverables. Additionally, we were guided by Kezia Endsley, Copy Editor, who further refined our content to make the various chapters written by the many authors, flow into one cohesive piece of work.

A special thanks to Eli Schilling, Biff (yes that's his real name) Gaut, and Brian Wagner. Eli gathered the group of authors to write this book, Biff provided us much needed foresight, as he had co-authored the *AWS Certified Solutions Architect Official Study Guide*, and Brian helped us write some last-minute questions for the online practice exams.

Lastly, we want to thank all the Solutions Architects and Technical Trainers at AWS who participated in certification blueprint development, question writing, review sessions, and the development of a world-class certification program for cloud practitioners that is setting the standard for our industry. #LearnAndBeCurious

About the Authors



Stephen Cole is a Technical Trainer with AWS, having joined the Training and Certification team in 2016. He received his Bachelor of Arts degree from Indiana University of Pennsylvania (IUP) in 1991 and, in 2015, earned a Master of Arts in Organizational Leadership from Gonzaga University. Currently, he has two AWS certifications: Solutions Architect Associate and SysOps Administrator Associate. Stephen would like to express his gratitude and appreciation for his wife, Laura, and son, Eli, as they were both very patient while sacrificing significant family time for this book.



Gareth Digby, Technical Trainer for AWS, delivers training on AWS services to students throughout North America. Gareth holds a B.Sc. and Ph.D. in Electrical and Electronic Engineering from the University of Swansea. Gareth has held full time faculty posts in the Electrical and Electronic Engineering Department, at University of Swansea and at the School of Electrical and Electronic Engineering, University of Birmingham. He has taught as adjunct faculty in the Department of Computer Science at University of Oxford and the Penn State Great Valley School of Graduate Professional Studies. Prior to joining AWS, in addition to his academic posts, Gareth has held systems engineering and system architecture roles on a variety of public sector projects. Gareth wants to thank Enfield Grammar School for introducing him to computers, the Electrical and Electronic Engineering Department and the Computer Science Department at University of Wales, Swansea for inspiring him to teach about computers, and his family for allowing him to pursue these passions for far too many years.



Christopher Fitch is a Technical Trainer with AWS. He has over 15 years' experience in various engineering, administration, and architectural positions. His experience brings with it a combination of academic and hands-on knowledge that's provided a diverse and well-rounded set of skills. Prior to working with AWS, he spent most of his career working with the DoD. Christopher holds a Bachelor's of Science in Technical Management from DeVry University, a Master of Science in Information Systems, and a Master of Science in Network and Communications Management from the Keller Graduate School. Chris is a geek at heart. He is a native Floridian and Seattle transplant who is passionate about design, photography, and biking.



Steve Friedberg has been an educator for 40 years, teaching in ten different countries. He has been a course developer and instructor for IBM, Oracle, DEC, Cisco, Microsoft, Marconi, and TIBCO, as well as an adjunct professor at Grace College in Winona Lake, IN. He has been with AWS as a Technical Trainer for over a year, and he holds all three AWS Associate certifications. Steve's formal training includes a Bachelor of Science in Engineering from Cornell University and a Master of Arts in Education from Ball State University. He lives with his wife in Indiana near his children and grandchildren. His real passion is teaching and developing curriculum about the Old Testament feasts, holidays, and prophecies.



Shaun Qualheim has been with AWS since September 2014. He is currently serving customers as a Senior Solutions Architect. In previous lives, Shaun was a Linux Systems Administrator at companies ranging from a leading computational fluid dynamics (CFD) company to one of the largest educational assessment companies in the world. Shaun is the proud father of an amazing 9-year old son, Jackson, who loves to come to the NYC AWS office and socialize with everyone. He wishes to thank his team for their patience while he worked on this book. Shaun would like to dedicate his portion of this book to his father, who taught him the value of never wavering in doing what's right for the customer and whose example continues to show him the value of diligent work. Without that guidance, Shaun wouldn't be where he is today.



Jerry Rhoads has been with AWS since May 2014. Jerry started off as a Solutions Architect, and he recently joined the Training and Certification Team as a Technical Trainer. Jerry holds a Bachelor's of Science in Computer Science and a Master of Science in Information Systems Technology from the George Washington University, as well as all five AWS certifications. Jerry would like to give special thanks to Dr. Marjorie Battaglia, who inspired him to be a better writer; Reggie Carreker, who provided him with a passion for teaching; his wife, Linda, and his four children (+ one on the way), Ashley, Harry, Tinsley, and Liam for their much-needed patience and inspiration.



Michael Roth is a Technical Trainer with AWS, having joined Amazon in 2015. He is a Certified Cisco Network Academy Instructor and has taught Linux. Michael graduated from the University of Michigan with a Bachelor of Science in Zoology and a Bachelor of Arts in Urban Planning. He also has a Master of Science Degree in Telecommunications Management from Golden Gate University. Michael would like to thank his co-workers in the AWS Technical Training Organization—he is very proud to be a part of this amazing group of people. Finally, he would like to thank his spouse, Betsy, and son, Robert. Without their support and love, this book would not have been possible.



Blaine Sundrud began his teaching career at the Show Low Arizona High School before becoming a product evangelist for Digital Technology International. At DTI, Blaine worked with newspapers from around the world helping them improve their publishing platforms, until he realized that supporting the print newspaper industry was not a long-term employment option. Blaine now works in the Training and Certification department at AWS, where he holds all five certifications. His current focus is on leveraging brain science to improve classroom learning through the NeuroEducate program that he developed at AWS. Blaine wants to thank his three children: Kelly, Hunter, and Dessa for their resiliency, as well as his wife, Diana, for her high availability.

Contents at a Glance

| | | |
|------------------------|---|-----|
| <i>Foreword</i> | <i>xix</i> | |
| <i>Introduction</i> | <i>xxi</i> | |
| <i>Assessment Test</i> | <i>xxvi</i> | |
| Chapter 1 | Introduction to Systems Operations on AWS | 1 |
| Chapter 2 | Working with AWS Cloud Services | 23 |
| Chapter 3 | Security and AWS Identity and Access Management (IAM) | 41 |
| Chapter 4 | Compute | 107 |
| Chapter 5 | Networking | 151 |
| Chapter 6 | Storage Systems | 207 |
| Chapter 7 | Databases | 249 |
| Chapter 8 | Application Deployment and Management | 313 |
| Chapter 9 | Monitoring and Metrics | 363 |
| Chapter 10 | High Availability | 441 |
| Appendix | Answers to the Review Questions | 481 |
| <i>Index</i> | | 499 |

Contents

| | | |
|--|--|-----------|
| <i>Foreword</i> | <i>xix</i> | |
| <i>Introduction</i> | <i>xxi</i> | |
| <i>Assessment Test</i> | <i>xxvi</i> | |
| Chapter 1 | Introduction to Systems Operations on AWS | 1 |
| Systems Operators | 2 | |
| Deploying Systems | 2 | |
| Monitoring Systems | 2 | |
| Optimizing Systems | 3 | |
| Fortifying Systems | 3 | |
| Securing Systems | 3 | |
| AWS Certified SysOps Administrator - Associate | 4 | |
| Which AWS Services Should You Study? | 4 | |
| Reference Architecture: The Three-Tier Design | 5 | |
| Introduction to the Three-Tier Design | 5 | |
| Sample Scenario | 6 | |
| Reference Architecture: The Serverless Design | 14 | |
| Key Product: Serverless Design | 17 | |
| Summary | 18 | |
| Exam Essentials | 18 | |
| Key Pieces to Study | 19 | |
| Review Questions | 20 | |
| Chapter 2 | Working with AWS Cloud Services | 23 |
| Introduction to AWS Cloud Services | 24 | |
| Systems Operations Using the AWS Toolset | 24 | |
| AWS Software Development Kits (SDKs) | 30 | |
| AWS Internet of Things (IoT) and Mobile Software Development Kits (SDKs) | 33 | |
| Summary | 34 | |
| Exam Essentials | 35 | |
| Resources to Review | 35 | |
| Exercises | 35 | |
| Review Questions | 38 | |
| Chapter 3 | Security and AWS Identity and Access Management (IAM) | 41 |
| Security on AWS | 43 | |
| Shared Responsibility Model | 43 | |
| AWS Security Responsibilities | 43 | |

| | | |
|------------------|---|------------|
| | Customer Security Responsibilities | 44 |
| | AWS Global Infrastructure Security | 44 |
| | Physical and Environmental Security | 46 |
| | Business Continuity Management | 47 |
| | Network Security | 48 |
| | Network Monitoring and Protection | 49 |
| | AWS Compliance Program | 50 |
| | Securing Your AWS Account with AWS Identity and Access Management (IAM) | 51 |
| | IAM User | 52 |
| | IAM Groups | 56 |
| | IAM Policies | 56 |
| | IAM Roles | 57 |
| | Best Practices for Securing Your AWS Account | 58 |
| | Securing Your AWS Cloud Services | 59 |
| | Key Pairs | 59 |
| | Monitoring to Enhance Security | 62 |
| | AWS CloudTrail | 62 |
| | Amazon Virtual Private Cloud (Amazon VPC) Flow Logs | 62 |
| | Amazon CloudWatch | 63 |
| | AWS Config | 63 |
| | Amazon Inspector | 64 |
| | AWS Certificate Manager | 64 |
| | AWS Web Application Firewall (AWS WAF) | 64 |
| | AWS Trusted Advisor | 64 |
| | AWS Cloud Service-Specific Security | 65 |
| | Compute Services | 65 |
| | Networking | 69 |
| | Storage | 75 |
| | AWS Storage Gateway Security | 80 |
| | Database | 80 |
| | Application Services | 88 |
| | Analytics Services | 89 |
| | Deployment and Management Services | 91 |
| | Mobile Services | 92 |
| | Applications | 94 |
| | Summary | 95 |
| | Exam Essentials | 96 |
| | Exercises | 98 |
| | Review Questions | 103 |
| Chapter 4 | Compute | 107 |
| | Introduction to AWS Compute Services | 109 |
| | Amazon Elastic Compute Cloud (Amazon EC2) | 111 |
| | Implementation | 111 |
| | Management | 117 |
| | Security | 122 |

| | | |
|--|-------------------|------------|
| Amazon EC2 Container Service (Amazon ECS) | 123 | |
| Implementation | 124 | |
| Management | 124 | |
| Security | 125 | |
| AWS Elastic Beanstalk | 125 | |
| Languages Supported in AWS Elastic Beanstalk | 126 | |
| Services that AWS Elastic Beanstalk Deploys | 126 | |
| Management | 126 | |
| Security | 127 | |
| AWS Lambda | 128 | |
| Implementation | 128 | |
| Management | 130 | |
| Security | 130 | |
| Amazon Lightsail | 130 | |
| Implementation | 131 | |
| Management | 131 | |
| Security | 133 | |
| AWS Batch | 133 | |
| Implementation | 133 | |
| Management | 135 | |
| Security | 135 | |
| Summary | 135 | |
| Exam Essentials | 136 | |
| Resources to Review | 139 | |
| Exercises | 140 | |
| Review Questions | 146 | |
| Chapter 5 | Networking | 151 |
| Introduction to Networking on AWS | 153 | |
| Amazon Virtual Private Cloud (Amazon VPC) | 154 | |
| Amazon VPC Implementation | 154 | |
| Amazon VPC Management | 164 | |
| AWS Direct Connect | 166 | |
| AWS Direct Connect Implementation | 167 | |
| AWS Direct Connect Management | 169 | |
| AWS Direct Connect Security | 170 | |
| Load Balancing | 171 | |
| Load Balancing Implementation | 172 | |
| Load Balancing Management | 176 | |
| Load Balancing Security | 178 | |
| Virtual Private Network (VPN) | 178 | |
| VPN Installation | 178 | |
| VPN Management | 179 | |
| Amazon Route 53 | 179 | |
| Amazon Route 53 Implementation | 180 | |
| Amazon Route 53 Management | 185 | |

| | |
|---|------------|
| Amazon CloudFront | 185 |
| Amazon CloudFront Implementation | 186 |
| Amazon CloudFront Management | 194 |
| Amazon CloudFront Security | 194 |
| Summary | 195 |
| Resources to Review | 195 |
| Exam Essentials | 196 |
| Exercises | 198 |
| Review Questions | 201 |
| Chapter 6 Storage Systems | 207 |
| Understanding Different Storage Options | 209 |
| Block Storage vs. Object Storage | 209 |
| Block Storage Basics | 210 |
| Object Storage Basics | 210 |
| Retrieval Times (Hot vs. Cold Storage) | 211 |
| Cost Efficiency | 211 |
| Block Storage on AWS | 212 |
| Amazon Elastic Block Store (Amazon EBS) | 212 |
| Instance Store | 221 |
| Amazon Elastic File System (Amazon EFS) | 222 |
| Object Storage on AWS | 224 |
| Amazon Simple Storage Service (Amazon S3) | 224 |
| Amazon Glacier | 230 |
| Systems Operator Scenario: The Newspaper | 232 |
| Storage Needs | 233 |
| Solution Breakdown | 233 |
| Additional Storage Solutions | 234 |
| Amazon CloudFront | 234 |
| AWS Storage Gateway | 235 |
| AWS Snowball | 235 |
| Summary | 236 |
| Resources to Review | 236 |
| Exam Essentials | 237 |
| Exercises | 239 |
| Review Questions | 244 |
| Chapter 7 Databases | 249 |
| Introduction to AWS Databases | 250 |
| SQL vs. NoSQL | 251 |
| Relational Databases Overview | 252 |
| Relational Database Design | 252 |
| Non-Relational Database Overview | 253 |
| Amazon RDS Features and Benefits | 254 |
| Amazon Aurora | 256 |

| | |
|--|------------|
| Monitoring Amazon RDS | 278 |
| Monitoring Tools | 278 |
| Amazon RDS Pricing | 282 |
| Non-Relational Databases | 283 |
| Amazon DynamoDB | 283 |
| Amazon DynamoDB Core Components | 284 |
| Amazon Redshift | 292 |
| Cluster Management | 293 |
| Cluster Access and Security | 293 |
| Databases | 294 |
| Monitoring Clusters | 295 |
| Amazon ElastiCache | 296 |
| Summary | 298 |
| Resources to Review | 298 |
| Exam Essentials | 299 |
| Exercises | 300 |
| Review Questions | 307 |
| Chapter 8 Application Deployment and Management | 313 |
| Introduction to Application Deployment and Management | 314 |
| Deployment Strategies | 314 |
| Provisioning Infrastructure | 314 |
| Deploying Applications | 315 |
| Configuration Management | 315 |
| Scalability Capabilities | 318 |
| Monitoring Resources | 318 |
| Continuous Deployment | 319 |
| Deployment Services | 322 |
| AWS Elastic Beanstalk | 323 |
| Amazon EC2 Container Service | 325 |
| AWS OpsWorks Stacks | 328 |
| AWS CloudFormation | 330 |
| AWS Command Line Interface (AWS CLI) | 345 |
| Summary | 346 |
| Resources to Review | 347 |
| Exam Essentials | 347 |
| Exercises | 349 |
| Review Questions | 358 |
| Chapter 9 Monitoring and Metrics | 363 |
| Introduction to Monitoring and Metrics | 364 |
| An Overview of Monitoring | 364 |
| Why Monitor? | 364 |
| Amazon CloudWatch | 365 |
| AWS CloudTrail | 365 |

| | |
|---|-----|
| AWS Config | 365 |
| AWS Trusted Advisor | 366 |
| AWS Service Health Dashboard | 366 |
| AWS Personal Health Dashboard | 367 |
| Amazon CloudWatch | 367 |
| Metrics | 369 |
| Custom Metrics | 369 |
| Amazon CloudWatch Metrics Retention | 370 |
| Namespaces | 371 |
| Dimensions | 372 |
| Statistics | 373 |
| Units | 374 |
| Periods | 374 |
| Aggregation | 375 |
| Dashboards | 376 |
| Percentiles | 376 |
| Monitoring Baselines | 377 |
| Amazon EC2 Status Checks | 378 |
| Authentication and Access Control | 379 |
| AWS Cloud Services Integration | 382 |
| Amazon CloudWatch Limits | 382 |
| Amazon CloudWatch Alarms | 384 |
| Alarms and Thresholds | 384 |
| Missing Data Points | 386 |
| Common Amazon CloudWatch Metrics | 386 |
| Amazon CloudWatch Events | 395 |
| Events | 396 |
| Rules | 397 |
| Targets | 397 |
| Metrics and Dimensions | 398 |
| Amazon CloudWatch Logs | 399 |
| Archived Data | 400 |
| Log Monitoring | 400 |
| Amazon CloudWatch Logs: Agents and IAM | 401 |
| Searching and Filtering Log Data | 403 |
| Monitoring AWS Charges | 406 |
| Detailed Billing | 407 |
| Cost Explorer | 409 |
| AWS Billing and Cost Management Metrics and Dimensions | 410 |
| AWS CloudTrail | 411 |
| What Are Trails? | 411 |
| Types of Trails | 411 |
| Multiple Trails per Region | 412 |
| Encryption | 412 |

| | |
|--|------------|
| AWS CloudTrail Log Delivery | 412 |
| Overview: Creating a Trail | 413 |
| Monitoring with AWS CloudTrail | 413 |
| AWS CloudTrail vs. Amazon CloudWatch | 414 |
| AWS CloudTrail: Trail Naming Requirements | 414 |
| Getting and Viewing AWS CloudTrail Log Files | 414 |
| AWS Config | 417 |
| Ways to Use AWS Config | 418 |
| AWS Config Rules | 419 |
| AWS Config and AWS CloudTrail | 420 |
| Pricing | 421 |
| Summary | 421 |
| Resources to Review | 422 |
| Exam Essentials | 423 |
| Exercises | 425 |
| Review Questions | 438 |
| Chapter 10 High Availability | 441 |
| Introduction to High Availability | 443 |
| Amazon Simple Queue Service | 444 |
| Using Amazon Simple Queue Service to Decouple an Application | 444 |
| Standard Queues | 448 |
| First-In, First-Out Queues | 448 |
| Dead Letter Queues | 449 |
| Shared Queues | 449 |
| Amazon Simple Notification Service | 450 |
| Mobile Push Messaging | 451 |
| Amazon SNS Fan-Out Scenario | 451 |
| Highly Available Architectures | 452 |
| Network Address Translation (NAT) Gateways | 453 |
| Elastic Load Balancing | 453 |
| Auto Scaling | 454 |
| Session State Management | 455 |
| Amazon Elastic Compute Cloud Auto Recovery | 455 |
| Scaling Your Amazon Relational Database | |
| Service Deployment | 456 |
| Multi-Region High Availability | 457 |
| Amazon Simple Storage Service | 457 |
| Amazon DynamoDB | 457 |
| Amazon Route 53 | 457 |
| Highly Available Connectivity Options | 463 |
| Redundant Active-Active VPN Connections | 463 |
| Redundant Active-Active AWS Direct Connect | |
| Connections | 465 |
| AWS Direct Connect with Backup VPN Connection | 466 |

| | | |
|-----------------|--|------------|
| | Disaster Recovery | 467 |
| | Backup and Restore Method | 467 |
| | Pilot Light Method | 468 |
| | Warm-Standby Method | 470 |
| | Multi-Site Solution Method | 470 |
| | Failing Back from a Disaster | 471 |
| | Summary | 472 |
| | Resources to Review | 473 |
| | Exam Essentials | 473 |
| | Exercises | 474 |
| | Review Questions | 478 |
| Appendix | Answers to the Review Questions | 481 |
| | Chapter 1: Introduction to Systems Operations on AWS | 482 |
| | Chapter 2: Working with AWS Cloud Services | 483 |
| | Chapter 3: Security and AWS Identity and Access Management (IAM) | 483 |
| | Chapter 4: Compute | 485 |
| | Chapter 5: Networking | 486 |
| | Chapter 6: Storage Systems | 488 |
| | Chapter 7: Databases | 490 |
| | Chapter 8: Application Deployment and Management | 492 |
| | Chapter 9: Monitoring and Metrics | 494 |
| | Chapter 10: High Availability | 496 |
| <i>Index</i> | | 499 |

Table of Exercises

| | | | |
|-----------------|------------|--|-----|
| Exercise | 2.1 | Install and Configure AWS CLI on Linux or Mac | 36 |
| Exercise | 2.2 | Install and Configure AWS CLI on Windows with MSI | 36 |
| Exercise | 3.1 | Creating AWS Identity and Access Management (IAM) Users | 99 |
| Exercise | 3.2 | Create IAM Credentials | 99 |
| Exercise | 3.3 | Create IAM Groups | 100 |
| Exercise | 3.4 | Working with IAM Policies | 101 |
| Exercise | 3.5 | Working with IAM Roles | 101 |
| Exercise | 4.1 | Create a Linux Instance via the AWS Management Console | 141 |
| Exercise | 4.2 | Create a Windows Instance via the AWS Management Console | 142 |
| Exercise | 4.3 | Create a Linux Instance via the AWS CLI | 142 |
| Exercise | 4.4 | Create a Windows Instance via the AWS CLI | 143 |
| Exercise | 4.5 | Inspect the AWS Service Health Dashboards | 143 |
| Exercise | 4.6 | Use the Elastic IP Addresses | 144 |
| Exercise | 4.7 | Work with Metadata | 144 |
| Exercise | 4.8 | Attach an AWS IAM Role to an Instance | 145 |
| Exercise | 5.1 | Create an Elastic IP (EIP) | 198 |
| Exercise | 5.2 | Create an Amazon VPC | 198 |
| Exercise | 5.3 | Tag Your Amazon VPC and Subnets | 199 |
| Exercise | 5.4 | Create an Elastic Network Interface (ENI) | 199 |
| Exercise | 5.5 | Associate the ENI | 200 |
| Exercise | 5.6 | Test Your ENI | 200 |
| Exercise | 5.7 | Delete VPC | 200 |
| Exercise | 6.1 | Create an Encrypted Amazon EBS Volume | 240 |
| Exercise | 6.2 | Monitor Amazon EBS Using Amazon CloudWatch | 240 |
| Exercise | 6.3 | Create and Attach an Amazon EFS Volume | 240 |
| Exercise | 6.4 | Create and Use an Amazon S3 Bucket | 241 |
| Exercise | 6.5 | Enable Amazon S3 Versioning | 242 |
| Exercise | 6.6 | Enable Cross-Region Replication | 242 |
| Exercise | 6.7 | Create an Amazon Glacier Vault | 242 |
| Exercise | 6.8 | Enable Lifecycle Rules | 243 |
| Exercise | 7.1 | Create a New Option Group Using the Console | 300 |
| Exercise | 7.2 | Create an Amazon DynamoDB Table from the AWS CLI | 301 |

| | | | |
|-----------------|-------------|--|-----|
| Exercise | 7.3 | Add Items to the Amazon DynamoDB Table MusicCollection Using the AWS CLI | 302 |
| Exercise | 7.4 | Create a MySQL Amazon RDS DB Instance | 303 |
| Exercise | 8.1 | Create an AWS Elastic Beanstalk Environment..... | 349 |
| Exercise | 8.2 | Manage Application Versions with AWS Elastic Beanstalk..... | 349 |
| Exercise | 8.3 | Perform a Blue/Green Deployment with AWS Elastic Beanstalk | 350 |
| Exercise | 8.4 | Create an Amazon ECS Cluster..... | 350 |
| Exercise | 8.5 | Launch an Amazon EC2 Instance Optimized for Amazon ECS | 351 |
| Exercise | 8.6 | Use Amazon ECR..... | 352 |
| Exercise | 8.7 | Work with Amazon ECS Task Definitions..... | 352 |
| Exercise | 8.8 | Work with Amazon ECS Services | 354 |
| Exercise | 8.9 | Create an AWS OpsWorks Stack..... | 355 |
| Exercise | 8.10 | Make a Layer in AWS OpsWorks Stacks..... | 355 |
| Exercise | 8.11 | Add an Amazon EC2 Instance to an AWS OpsWorks Stacks Layer | 356 |
| Exercise | 8.12 | Add an Application to AWS OpsWorks Stacks | 356 |
| Exercise | 8.13 | Create an AWS CloudFormation Stack..... | 357 |
| Exercise | 8.14 | Delete an AWS CloudFormation Stack..... | 357 |
| Exercise | 9.1 | Search for Available Metrics | 425 |
| Exercise | 9.2 | View Available Metrics for Running Amazon EC2 Instances by Namespace and Dimension Using the Amazon CloudWatch Console | 426 |
| Exercise | 9.3 | View Available Metrics by Namespace, Dimension, or Metric Using the AWS CLI..... | 429 |
| Exercise | 9.4 | List All Available Metrics for a Specific Resource..... | 430 |
| Exercise | 9.5 | List all Resources that Use a Single Metric | 430 |
| Exercise | 9.6 | Get Statistics for a Specific Resource | 430 |
| Exercise | 9.7 | Get CPU Utilization for a Single Amazon EC2 Instance from the Command Line | 433 |
| Exercise | 9.8 | Create a Billing Alert..... | 435 |
| Exercise | 9.9 | Create a Billing Alarm..... | 435 |
| Exercise | 9.10 | Create an Amazon CloudWatch Dashboard..... | 436 |
| Exercise | 10.1 | Create an Amazon SNS Topic | 475 |
| Exercise | 10.2 | Create a Subscription to Your Topic | 475 |
| Exercise | 10.3 | Publish to Your Topic | 475 |
| Exercise | 10.4 | Create an Amazon Simple Queue Service (Amazon SQS)..... | 476 |
| Exercise | 10.5 | Subscribe the Queue to Your Amazon SNS Topic | 476 |
| Exercise | 10.6 | Deploy Amazon RDS in a Multi-AZ Configuration | 477 |

Foreword

I entered college in 1978, and I immediately found a second home at the computer lab on campus. This lab was home to an IBM mainframe and a roomful of noisy keypunch machines. I punched my code onto a stack of cards, and I handed the stack to a system operator. The operator loaded the cards into the reader, and my job was queued for processing. If things went well and the mainframe was not too busy, I would have my cards and my output back within four hours or so. The operator managed the work queue for the mainframe, adjusting the balance of jobs and priorities, looking for hot spots and slowdowns, and keeping the monolithic mainframe as busy and as productive as possible at all times.

As a young, curious student, I always wondered what was happening behind the scenes. As a young, impoverished student, in the days before the Internet, information was not always easy to come by. I found a rack of manuals in the lab, figured out how to order others for free, and even scavenged the trash cans for operating system “builds” to study. That thirst for knowledge, with a focus on understanding how things work at the most fundamental level, has worked really well for me over the intervening four decades.

A little over a decade ago, I wrote blog posts to announce the launches of Amazon Simple Storage Service (Amazon S3) and Amazon Elastic Compute Cloud (Amazon EC2). Those early launches set the tone for what was to come, introducing services that emerged with a minimal feature set that would be enhanced over time in response to customer feedback. At that time, aspiring AWS developers and architects did not need to make very many choices when they set out to build an AWS-powered system. There was one instance type, a couple of Availability Zones in a single Region, and simple access via the AWS CLI and the API.

Back in my mainframe days, operations was a hands-on affair. There was little in the way of tooling or automation; the operator was expected to watch the console, check on status, and to deal with issues as they arose. Today, many routine operations are handled automatically. Fault tolerance, automatic scaling, load balancing, and other high-level facilities take on many chores that were once described in detailed run books. With this change, systems operations comes into play much earlier in the system-building process, with the goal of setting up the system for success and high availability. At the same time, the operations role now spans a wider range of tasks and technologies including networking, security, and optimization. With pay-as-you-go services now the norm, people who once focused on technology can now add business skills to their repertoire.

If you are about to read this book, I am sure that you know that AWS is far more complex than it was a decade ago. On the Amazon EC2 side alone, there are now dozens of instance types, multiple types of Amazon Elastic Block Storage (Amazon EBS) volumes, and far more moving parts. There are now close to 100 services, each of which can be a valuable addition to your toolbox. The vocabulary itself has changed, with new terms such as containers, microservices, serverless computing, infrastructure as code, and so forth now commonplace.

You now face many choices when you set out to design and implement a new system. This book is designed to provide you with detailed information on many aspects of AWS, coupled with the practical knowledge needed to put your new knowledge to use and to earn your AWS certification. Within its chapters, you will find service overviews, sample scenarios, test-taking tips, and exercises. After setting up your AWS tools, you will learn about security, compute services, storage services, networking, databases, and more. Towards the end of the book, you will wrap up by learning about monitoring, metrics, and high availability. As you will soon see, the authors have packed it with the insights that they have gained while putting AWS to use in a wide variety of customer environments. There are no better teachers than those who have actually put their theory into practice.

You can choose to study the chapters individually, or you can digest the entire book as-written. Either way, I know that you will be well-prepared to build great systems and to pass your certification exams. I strongly encourage you to get hands-on experience with each service by working through the scenarios and the exercises.

I believe in the principle of life-long learning, especially when it comes to technology. The half-life of knowledge is shorter than ever before, and keeping up is far better than catching up. So dive deep and keep on learning!

— Jeff Barr, *Chief Evangelist, AWS*

Introduction

Preparing to take and pass any certification is a studious process. The *AWS Certified SysOps Administrator Official Study Guide - Associate Exam* was written to align with the exam blueprint to enable you to study for the exam, perform exercises, and answer review questions to enable you to become a skilled systems operator on the AWS cloud and to take and pass the AWS Certified SysOps Administrator – Associate exam with confidence.

This study guide presents the set of topics needed to round out a systems operator/systems administrator’s hands-on experiences with AWS by covering the relevant AWS cloud services and concepts within the scope of the AWS Certified SysOps Administrator – Associate exam. This study guide begins with an introduction to Systems Operations on AWS, which is then followed by chapters on specific domains covered in the exam. In addition to the material covered on the exam, the chapters go deep into the actual technology. The authors go deep on topics that will serve you in preparing for the exam and the book should make a good desktop reference on AWS systems operations.

Each chapter includes specific information on the service or topic covered, followed by an Exam Essentials section that contains key information needed in your exam preparation. The Exam Essentials section is followed by a Test Taking Tip to help you prepare for what you will experience on the exam or at the testing center.

Next, each chapter includes an Exercise section with activities designed to help reinforce the topic of the chapter with hands-on learning. Each chapter then contains sample Review Questions to get you accustomed to answering questions about how to use and administer AWS cloud services.

Following this up-front section, the book contains a self-assessment exam with 25 questions. Two practice exams with 50 questions each are also available to help you gauge your readiness to take the exam, and flashcards are provided to help you learn and retain key facts needed to prepare for the exam.

If you are looking for a targeted book, created by technical trainers and solutions architects who wrote, reviewed, and developed the AWS Certified SysOps Administrator – Associate exam, then this is the book for you.

What Does this Book Cover?

This book covers topics that you need to know to prepare for the Amazon Web Services (AWS) Certified SysOps Administrator – Associate exam:

Chapter 1: Introduction to Systems Operations on AWS This chapter provides an introduction to System Operations on AWS. It provides an overview of the AWS cloud services covered on the AWS Certified SysOps Administrator – Associate exam.

Chapter 2: Working with AWS Cloud Services This chapter shows you how to configure your workstation to work with AWS cloud services. You will install the AWS Command

Line Interface (AWS CLI). Topics include AWS CLI, jmespath (a query language for JSON, <http://jmespath.org>), and the Boto software development kit (SDK).

Chapter 3: Security and AWS Identity and Access Management (IAM) In this chapter, you will learn about the Shared Responsibility Model and the different layers of security. You will learn how to secure your systems with services such as AWS Key Management Service (AWS KMS), AWS Hard Security Module (AWS HSM), Security Groups, and Network Access Control Lists (nACLs). Furthermore, the chapter covers AWS Identity and Access Management (IAM) and Security Best Practices.

Chapter 4: Compute This chapter describes how to use the compute stack on AWS. The topics covered are Amazon Elastic Compute Cloud (Amazon EC2), AWS Lambda, AWS Beanstalk, Amazon Elastic Container Service (Amazon ECS), Amazon Lightsail, and AWS Batch. You will provision an Amazon EC2 instance, assign an Amazon EC2 Role, and work with instance metadata.

Chapter 5: Networking In this chapter, you will learn how to deploy Amazon Virtual Private Cloud (Amazon VPC) and the various methods to connect to your Amazon VPC. Additionally, you will learn how to use the Elastic Load Balancing service, Amazon Route 53, and Amazon CloudFront.

Chapter 6: Storage Systems This chapter covers deploying and using the various storage options on AWS. The services covered include: Amazon Simple Storage Service (Amazon S3), Amazon Elastic File Service (Amazon EFS), Amazon Elastic Block Service (Amazon EBS), the Amazon EC2 instance store Volumes, Amazon Glacier, AWS Snowball, and AWS Snowmobile.

Chapter 7: Databases This chapter covers the use of AWS managed database services: Amazon Relational Database Service (Amazon RDS), Amazon DynamoDB, Amazon Redshift, and Amazon ElastiCache. You will learn how these managed services simplify the setup and operation of relational databases, NoSQL databases, data warehouses, and in-memory caches.

Chapter 8: Application Deployment and Management This chapter focuses on the various methods of deployment of applications and infrastructure; for example, blue/green and rolling deployments. You will learn about AWS OpsWorks, AWS Elastic Beanstalk, Amazon EC2 Container Service, and AWS CloudFormation.

Chapter 9: Monitoring and Metrics In this chapter, you will learn about how to monitor your environment with Amazon CloudWatch, AWS CloudTrail, AWS Config, AWS Trusted Advisor, and AWS Service Health Dashboard.

Chapter 10: High Availability This chapter covers high availability on AWS. You will be introduced to decoupling strategies using Amazon Simple Queue Service (Amazon SQS) and Amazon Simple Notification Service (Amazon SNS). The chapter covers deploying your application to multiple Availability Zones and Multiple AWS Regions. Other high availability topics include Auto Scaling, failover with Amazon Route 53, and redundant VPN and AWS Direct Connect connections.

Interactive Online Learning Environment and Test Bank

The authors have worked hard to provide you with some really great tools to help you with your certification process. The interactive online learning environment that accompanies the *AWS Certified SysOps Administrator Official Study Guide: Associate Exam* provides a test bank with study tools to help you prepare for the certification exam. This will help you increase your chances of passing it the first time! The test bank includes the following:

Sample Tests All the questions in the book are provided in the form of review questions that are located at the end of each chapter. There is a 25-question assessment at the end of this introductory section. In addition, there are two practice exams with 50 questions each. Use these questions to test your knowledge of the study guide material. The online test bank runs on multiple devices.

Flashcards The online test banks include 100 flashcards specifically written to quiz your knowledge of operations on AWS. After completing all of the exercises, review questions, practice exams, and flashcards, you should be more than ready to take the exam. The flashcard questions are provided in a digital flashcard format (a question followed by a single correct answer with URL links for additional information). You can use the flashcards to reinforce your learning and provide last-minute test prep before the exam.

Glossary A glossary of key terms from this book is available as a fully searchable PDF.



Go to <http://www.wiley.com/go/sybextestprep> to register and gain access to this interactive online learning environment and test bank with study tools.

Exam Objectives

The AWS Certified SysOps Administrator – Associate exam validates technical expertise in deployment, management, and operations on the AWS platform. Exam concepts that you should understand for this exam include the following:

- Deploying, managing, and operating scalable, highly available, and fault tolerant systems on AWS
- Migrating an existing on-premises application to AWS
- Implementing and controlling the flow of data to and from AWS
- Selecting the appropriate AWS service based on compute, data, or security requirements

- Identifying appropriate use of AWS operational best practices
 - Estimating AWS usage costs and identifying operational cost control mechanisms
- In general, certification candidates should have the following:
- One or more years of hands-on experience operating AWS-based applications
 - Experience provisioning, operating, and maintaining systems running on AWS
 - Ability to identify and gather requirements to define a solution to be built and operated on AWS
 - Capabilities to provide AWS operations and deployment guidance and best practices throughout the lifecycle of a project

The exam covers seven different domains, with each domain broken down into objectives and subobjectives.

Objective Map

The following table lists each domain and its weighting in the exam, along with the chapters in the book where that domain's objectives and subobjectives are covered.

| Domain | Percentage of Exam | Chapter |
|---|--------------------|----------------|
| Domain 1.0 Monitoring and Metrics | 15% | |
| 1.1 Demonstrate ability to monitor availability and performance | | 3, 5, 7, 9, 10 |
| 1.2 Demonstrate ability to monitor and manage billing and cost optimization processes | | 7, 9 |
| Domain 2.0: High Availability | 15% | |
| 2.1 Implement scalability and elasticity based on scenario | | 4, 7, 8, 10 |
| 2.2 Ensure level of fault tolerance based on business needs | | 4, 5, 7, 8, 10 |
| Domain 3.0: Analysis | 15% | |
| 3.1 Optimize the environment to ensure maximum performance | | 5, 9 |

| Domain | Percentage of Exam | Chapter |
|---|--------------------|------------------|
| 3.2 Identify performance bottlenecks and implement remedies | 9 | |
| 3.3 Identify potential issues on a given application deployment | 9 | |
| Domain 4.0: Deployment and Provisioning | 15% | |
| 4.1 Demonstrate the ability to build the environment to conform with the architected design | | 1, 4, 6, 7, 8 |
| 4.2 Demonstrate the ability to provision cloud resources and manage implementation automation | | 1, 2, 4, 6, 7, 8 |
| Domain 5.0: Data Management | 12% | |
| 5.1 Demonstrate ability to create backups for different services | | 6, 7 |
| 5.2 Demonstrate ability to enforce compliance requirements | | 6 |
| 5.3 Manage backup and disaster recovery processes | | 7, 10 |
| Domain 6.0: Security | 15% | |
| 6.1 Implement and manage security policies | | 3, 5, 7 |
| 6.2 Ensure data integrity and access controls when using the AWS platform | | 1, 3, 6, 7, 9 |
| 6.3 Demonstrate understanding of the shared responsibility model | | 3, 4, 7 |
| 6.4 Demonstrate ability to prepare for security assessment use of AWS | | 3, 9 |
| Domain 7.0: Networking | 13% | |
| 7.1 Demonstrate ability to implement networking features of AWS | | 1, 5, 10 |
| 7.2 Demonstrate ability to implement connectivity features of AWS | | 5, 7, 10 |

Assessment Test

1. You notice in the AWS Management Console that your Amazon Elastic Compute Cloud (Amazon EC2) Instance State is *Failed*. What would cause this?
 - A. Loss of network connectivity
 - B. Loss of System Power
 - C. Incompatible kernel
 - D. Software issues on the physical host
2. What is the difference between a Public Subnet and a Private Subnet in a VPC?
 - A. The Route Table in the Private Subnet has a route to the Network Address Translation (NAT), while the Route Table in a Public Subnet does not.
 - B. The Route Table in the Public Subnet has a route to the Internet Gateway (IGW), while the Route Table in a Private Subnet does not.
 - C. The Public Subnet has NAT server, while a Private Subnet does not.
 - D. Only Elastic Load Balancers are allowed in the Public Subnet.
3. You have deployed eight Amazon Elastic Compute Cloud (Amazon EC2) instances in the us-west-1a Availability Zone and two Amazon EC2 instances in us-west-1b Availability Zone. You noticed that the two Amazon EC2 instances in us-west-1b received the same amount of traffic that is load balanced between the other eight Amazon EC2 instances located in the us-west-1a Availability Zone. How can you fix this from the load balancer?
 - A. Enable cross-load balancing on your load balancer.
 - B. Create an Auto Scaling group, and configure it to balance out the instances between the Availability Zones.
 - C. Create three instances in us-west-1b, and terminate three instances in us-west-1a.
 - D. Migrate to an Application load balancer.
4. You have launched an Amazon Relational Database Service (Amazon RDS) database instance running MySQL. When you created the Amazon RDS instance, you did not specify a maintenance window, and now you need to update the instance size from micro to large. If you request to have the update happen inside the maintenance window, what will occur?
 - A. Nothing. The command will be ignored until you create and apply a maintenance window.
 - B. Nothing. It is not possible to change the DB size using Amazon RDS.
 - C. AWS will select and use a default maintenance window if one is not provided.
 - D. AWS will prompt you to provide a maintenance window when you make the request.

5. Which of the following is the customer's responsibility in the Shared Responsibility Model?
 - A. Restricting access to Amazon Elastic Compute Cloud (Amazon EC2) using Security Groups
 - B. Restricting physical access to AWS datacenters
 - C. Destroying physical media used in AWS datacenters
 - D. Managing updates to the Hypervisors on which instances run
6. You are tasked with storing 200 GB of archival images that are requested infrequently, averaging one or two requests per image each day. Which is the most cost effective storage option for the images?
 - A. Amazon Elastic Block Store (Amazon EBS) io1
 - B. Amazon EBS gp2
 - C. Amazon Simple Storage Service (Amazon S3)
 - D. Amazon Elastic File System (Amazon EFS)
7. You need storage for your production MySQL database. The database is 19 TB in size, and you will need to have approximately 10,000 IOPS—mostly writes. Without considering price, which storage option satisfies the requirements?
 - A. Provisioned Amazon Elastic File System (Amazon EFS) 20 TB volume with 10,000 IOPS
 - B. Two provisioned Amazon EFS 10 TB volumes with 5,000 IOPS per volume and RAID0 striping
 - C. Provisioned Amazon Elastic Block Store (Amazon EBS) (io1) 20 TB volume with 10,000 IOPS
 - D. Two Provisioned Amazon EBS (io1) 10 TB volumes with 5,000 IOPS per volume and RAID0 striping
8. What is the purpose of Amazon Elastic Compute Cloud (Amazon EC2) user data?
 - A. To install software on the Amazon EC2 instance at boot
 - B. To list any public keys associated with the instance
 - C. To show a Public IP address to an Amazon EC2 instance
 - D. To show the localhost name for the instance
9. You have created an Amazon Virtual Private Cloud (Amazon VPC) with the CIDR of 10.0.0.0/16. You now need to divide that VPC into a Public Subnet and a Private Subnet. Which one below is a valid combination?
 - A. Public 10.1.0.0/24
Private 10.2.0.0/24
 - B. Public 10.0.0.1/24
Private 10.0.0.2/24
 - C. Public 10.0.1.0/24
Private 10.0.2.0/24
 - D. Public 10.0.1.0/16
Private 10.0.2.0/16

- 10.** You have created an Auto Scaling group with a minimum of two Amazon Elastic Compute Cloud (Amazon EC2) instances, a maximum of six instances, and a desired capacity of four instances. Your instances take 20 minutes to launch, and they take three minutes to start once built. How can you configure autoscaling to start and stop instances versus launching new instances from Amazon Machine Instances (AMIs)?
- A.** Create a new Auto Scaling launch configuration, and configure the Auto Scaling group to start the instances.
 - B.** Edit the Auto Scaling group's launch configuration to start instances.
 - C.** This is not possible, as Auto Scaling cannot stop and start instances.
 - D.** Configure the Auto Scaling group to use the Amazon EC2 recovery service.
- 11.** You have a Multi-AZ Amazon Relational Database Service (Amazon RDS) database running MySQL. During a planned outage, how does AWS ensure that, when switching from the primary DB to the standby, it will not affect your application servers?
- A.** Amazon RDS uses Elastic IP addresses that are detached from the primary database and then attached to the standby instance. This promotes the standby to be the primary.
 - B.** Amazon RDS uses the Elastic Queue Service to process requests from application servers and send them to database engines. Since this is done at the Hypervisor, no user intervention is required.
 - C.** Amazon RDS runs both database instances independently, and each has their own connection string. You will have to update the code on your application servers because AWS has no visibility above the Hypervisor.
 - D.** Amazon RDS uses Amazon Route 53 to create connection strings and will automatically update the IP address to point at the standby instance.
- 12.** When attaching an Amazon Elastic Block Store (Amazon EBS) volume to an Amazon Elastic Compute Cloud (Amazon EC2) instance, what conditions must be true?
- A.** The Amazon EBS volume must be in the same Availability Zone (AZ) as the instance.
 - B.** The Amazon EBS volume must be in the same account as the instance.
 - C.** The Amazon EBS volume must be assigned to an AMI ID.
 - D.** The Amazon EBS volume must have the same security group as the instance.
- 13.** You've been asked to migrate a busy Amazon Relational Database Service (Amazon RDS) for MySQL database to Amazon Aurora. You need to do so with little downtime and with no lost data. What is the best way to meet the above requirements?
- A.** Take a snapshot of the MySQL Amazon RDS instance. Use that snapshot to create an Amazon Aurora Read Replica of the Amazon RDS for MySQL database. Once replication catches up, make the Aurora Read Replica into a standalone Amazon Aurora DB cluster, and point the application to the new Amazon Aurora DB cluster.
 - B.** Create an Amazon Simple Storage Service (Amazon S3) bucket, and upload the Amazon RDS database as a flat file dump into the bucket. Restore from the dump to a new Amazon Aurora database.

- C. Restore the most recent Amazon RDS automated backup to a new Amazon Aurora instance. Stop the application, point the application at the new Amazon Aurora DB instance, and start the application.
 - D. Take a snapshot. Restore the snapshot to a new Amazon Aurora instance. Point the application to the new Amazon Aurora DB instance, and start the application.
14. In case of a failure of the primary node of an Amazon Relational Database Service (Amazon RDS) instance with an RDS Multi-AZ deployment, you must do the following to recover:
- A. Nothing. The node will automatically fail over to each of the three included read replicas in alternative regions.
 - B. Nothing. The node will automatically fail over to the standby instance; a short amount of downtime may occur.
 - C. Manually stand up a new instance by restoring from the most recent automatic backup.
 - D. Manually initiate the failover using the AWS CLI `initialize-rds-failover` command.
15. As part of an application requirement that you've been given, you must deploy a new Amazon DynamoDB database. You must do so in a highly available manner. How do you deploy this database?
- A. Deploy the Amazon DynamoDB database in a single Availability Zone (AZ). Set up an automatic backup job to Amazon S3 and an automatic restore job from S3 to a DynamoDB database in a second AZ.
 - B. Use the Amazon DynamoDB Local version in two AZs.
 - C. You can't use Amazon DynamoDB for HA requirements.
 - D. Deploy an Amazon DynamoDB database in the desired region.
16. Your application has a database that has been reported as being slow by your end users. Upon investigation, you find that the database is seeing an extraordinarily high volume of read activity. What is one way to overcome this constraint?
- A. Place an Amazon CloudFront distribution between the application layer and the database.
 - B. Use AWS Shield to protect against too many reads from the application layer to the database.
 - C. Use Amazon ElastiCache to provide a cache for frequent reads between the application layer and database.
 - D. Use AWS Route53 Latency Based Routing to direct traffic to the least latent database node.

- 17.** You have assigned an Elastic IP to an Amazon Elastic Compute Cloud (Amazon EC2) instance. You then terminate that instance. What happens to that Elastic IP?
- A.** The Elastic IP remains associated with the account, and you start getting charged for it.
 - B.** The Elastic IP remains associated with the account.
 - C.** The Elastic IP is deleted along with the Amazon EC2 instance.
 - D.** You cannot delete an Amazon EC2 instance with an Elastic IP associated with it. You must remove or delete the Elastic IP first.
- 18.** You are using Amazon CloudFront to serve static content to your users. What would be the best way to control access to the content?
- A.** Create an Amazon Simple Service (Amazon S3) bucket policy using AWS Identity and Management (IAM) as the mechanism to control access.
 - B.** Have your application create and distribute either signed URLs or signed cookies.
 - C.** Use the LIST Distribution API within Amazon CloudFront.
 - D.** Use Origin Access Identity (OAI) to secure access to content in Amazon CloudFront.
- 19.** You are looking to encrypt your Amazon DynamoDB table. How would you do this?
- A.** In the Amazon DynamoDB console, turn on server-side encryption.
 - B.** Via the AWS CLI, turn on server-side encryption.
 - C.** Use client-side encryption, as Amazon DynamoDB does not support server-side encryption.
 - D.** Enable Transparent Data Encryption (TDE).
- 20.** You are part of a team which is rebuilding your company's monolithic web application. The team plans on using a tiered architecture. One of the primary goals is to be able to use Auto Scaling to add and remove Amazon Elastic Compute Cloud (Amazon EC2) instances on demand. To this end, you need to get user state data off of individual instances. Which of the following AWS cloud services will provide you with a shared data store that is highly durable and has low latency?
- A.** Amazon DynamoDB
 - B.** Amazon EC2 Instance Storage
 - C.** Amazon Relational Database Service (Amazon RDS)
 - D.** Amazon Simple Storage Service (Amazon S3)
- 21.** Your company maintains an application that has a home-grown messaging cluster. You want to avoid maintaining this legacy cluster, and you need to migrate to an AWS service that provides this functionality. What service do you set up?
- A.** AWS X-Ray
 - B.** Amazon CloudFront
 - C.** Amazon Elasticsearch
 - D.** Amazon Simple Queue Service (Amazon SQS)

- 22.** You need to create an Amazon Virtual Private Cloud (Amazon VPC) that will allow you to use AWS Direct Connect. Which of the following combinations will allow you to use AWS Direct Connect but also prevent connectivity to the Internet?
- A.** You are not able to do so. If you have an AWS Direct Connect connection, by default, you have a connection to the Internet.
 - B.** Create a VPC with both an Internet Gateway (IGW) and a VPN Gateway.
 - C.** Create a VPC with an AWS Direct Connect Gateway.
 - D.** Create a VPC with a VPN Gateway.
- 23.** You have 10 Amazon Elastic Compute Cloud (Amazon EC2) instances behind a classic load balancer. What do you need to do to ensure that traffic is routed only to healthy instances?
- A.** Terminate the unhealthy instances.
 - B.** Enable cross-zone load balancing on your load balancer.
 - C.** Turn on health checks, and the load balancer will send traffic to the healthy instances.
 - D.** Nothing. The load balancer will terminate the unhealthy instances.
- 24.** You have noticed that your Auto Scaling group has scaled up to its maximum size. How can you be notified when your Auto Scaling group scales out and scales in?
- A.** Have your Auto Scaling group send messages to Amazon Simple Queue Service (Amazon SQS). Periodically check the queue for your Auto Scaling messages.
 - B.** Configure an Amazon Simple Notification Service (Amazon SNS) topic with an SMS subscription to your phone number.
 - C.** Configure an Amazon Simple Notification Service (Amazon SNS) topic with an AWS Lambda function that sends an email.
 - D.** Periodically query the Auto Scaling group to check the desired capacity.
- 25.** You are given a project to implement a High Performance Computing (HPC) workload for your R&D department. The workload takes tasks one-by-one, and it is tolerant of a node in the cluster failing. Each task runs for approximately one hour. Which of the following AWS cloud services is best suited for your workload from a cost-effectiveness standpoint?
- A.** Amazon Elastic Compute Cloud (Amazon EC2) Spot
 - B.** Amazon EC2 on-demand instances
 - C.** Amazon Elastic Compute Cloud (Amazon EC2) reserved instances
 - D.** AWS Lambda

Answers to the Assessment Test

1. C. Instance Status monitors the software and hardware of your individual instance. The other items listed are issues that affect the underlying AWS hardware.
2. B. The route table in the Public Subnet has a route pointing to the IGW. IGWs are associated with VPCs, not with subnets. Elastic Load Balancers can be in both the Public Subnet and the Private Subnet.
3. A. By default, the load balancer distributes traffic evenly across the Availability Zones that you enable for your load balancer. To distribute traffic evenly across all registered instances in all enabled Availability Zones, enable cross-zone load balancing on your load balancer. However, it is still recommended that you maintain approximately equivalent numbers of instances in each Availability Zone for better fault tolerance.
4. C. If you don't specify a preferred maintenance window when you create the DB instance or DB cluster, then Amazon RDS assigns a 30-minute maintenance window on a randomly selected day of the week.
5. A. You are responsible for security in the cloud, which includes configuring and applying Security Groups to resources running within the customer's account.
6. C. Amazon Simple Storage Service (Amazon S3) provides a low-cost method of storing objects in a highly available and durable manner.
7. D. You can choose up to a maximum of 16 TB per Amazon EBS volume, so you must create a RAID array of multiple volumes to achieve the IOPS being sought.
8. A. User data is run at boot time and can be used to install software. The other answers listed are examples of metadata, which is associated with the Amazon EC2 instance and can be accessed via the Amazon EC2 metadata service.
9. C. Subnets of a VPC have to be in the same address space as the VPC itself.
10. C. You can use scaling policies to increase or decrease the number of running Amazon EC2 instances in your group automatically to meet changing conditions. When the scaling policy is in effect, the Auto Scaling group adjusts the desired capacity of the group and launches or terminates the instances as needed. If you manually scale or scale on a schedule, you must adjust the desired capacity of the group in order for the changes to take effect.
11. D. Amazon RDS connection strings are based on Amazon Route 53 DNS. Inside Amazon RDS, they are referred to as endpoints. Endpoints include both the DNS name and the port number for the database instance.
12. A. Amazon EBS volumes must reside within the same Availability Zone (AZ) as the instance to which you are attaching the volume.
13. A. You can create an Amazon Aurora Read Replica to sync data from an Amazon RDS for MySQL source. By failing over to the replica, you can efficiently migrate between databases. Use a manual snapshot to pre-populate the read replica.

14. B. An Amazon RDS instance in a Multi-AZ deployment will automatically fail from a failed primary node to the standby node.
15. D. An Amazon DynamoDB database is replicated across three facilities in an AWS Region automatically.
16. C. Amazon ElastiCache provides an in-memory cache that can cache frequently read data and alleviate common read queries from hitting your database layer.
17. A. Elastic IPs are associated with the account, not the Amazon EC2 instance. However, unassigned Elastic IPs incur a charge. This is to discourage hoarding of IP addresses.
18. B. IAM should not be used because content will be accessed by individuals who do not have an IAM account. The LIST Distribution API just lists distributions; it does not control access. Origin Access Identity (OAI) is how you control access to content in an Amazon S3 object, not an Amazon CloudFront Distribution.
19. C. Amazon DynamoDB does not support the AWS Key Management Service (AWS KMS) nor server-side encryption. You can use customer-side encryption to store encrypted data in Amazon DynamoDB.
20. A. Amazon DynamoDB is a fast and flexible NoSQL database service for all applications that need consistent, single-digit millisecond latency at any scale. It is fully managed, and it supports both document and key-value store models.
21. D. Amazon SQS provides a scalable message queueing service, which allows the operator to avoid the undifferentiated heavy lifting associated with running a traditional message queuing application.
22. D. AWS Direct Connect uses the VPN Gateway as the gateway to the VPC. You can create a VPC that only has a VPN Gateway attached to it.
23. C. You can configure health checks, which are used to monitor the health of the registered instances so that the load balancer can send requests only to the healthy instances.
24. B. When you use Auto Scaling to scale your applications automatically, it is useful to know when Auto Scaling is launching or terminating the Amazon EC2 instances in your Auto Scaling group. Amazon SNS coordinates and manages the delivery or sending of notifications to subscribing clients or endpoints. You can configure Auto Scaling to send a SNS notification whenever your Auto Scaling group scales. AWS Lambda blocks port 25, the SMTP port, therefore it is not possible to send emails with Lambda.
25. A. Spot instances provide you with access to unused Amazon EC2 capacity at steep discounts relative to On-Demand prices. The Spot price fluctuates based on the supply and demand of available unused EC2 capacity.

Chapter 1



Introduction to Systems Operations on AWS

**THE AWS CERTIFIED SYSOPS
ADMINISTRATOR - ASSOCIATE EXAM
TOPICS COVERED IN THIS CHAPTER MAY
INCLUDE, BUT ARE NOT LIMITED TO, THE
FOLLOWING:**

Domain 4.0: Deployment and Provisioning

- ✓ 4.1 Demonstrate ability to build the environment to conform with the architectural design
- ✓ 4.2 Demonstrate ability to provision cloud resources and manage implementation automation

Content may include the following:

- How to deploy cloud services
- Familiarity with three-tier architectures
- Deploying serverless architectures

Domain 6.0: Security

- ✓ 6.1 Ensure data integrity and access controls when using the AWS platform

Content may include the following:

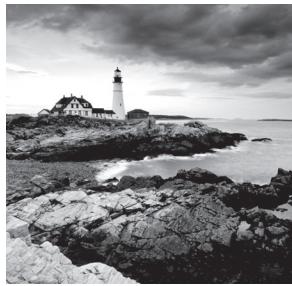
- AWS shared responsibility model
- AWS CloudTrail
- Amazon EC2 Security Groups
- Network access control lists (ACLs)

Domain 7.0: Networking

- ✓ 7.1 Demonstrate the ability to implement networking features on AWS

Content may include the following:

- Amazon Virtual Private Cloud (Amazon VPC)



Systems Operators

You are a *systems operator*, and it is your job to keep your application environments running at maximum performance at all times. Just as a pit crew enables the racecar driver to win a race, systems operators are the pit crew—they help end users function successfully in their day-to-day jobs. You are an AWS systems operator, and this book will help you obtain the AWS Certified SysOps Administrator - Associate certification.

Deploying Systems

You might find yourself manually installing common, off-the-shelf packages on standalone instances. You might be coordinating an enterprise-wide effort to embrace fully-automated continuous deployment/continuous integration. Wherever you are on that spectrum, the responsibility to get it running in the first place falls on your shoulders.

However, deployment comprises much more than initializing systems. As enterprises evolve from monolithic application servers to container services, micro services, and serverless architectures, keeping up with the continuous stream of service updates requires attention and automation that you must manage.

Monitoring Systems

You might have a wall of monitors, all rendering real-time data on the environments in your care. You might have fully-automated alert functions that respond to changes in behavior, repairing or replacing failing parts and keeping you informed of these adjustments.

Nonetheless, you are monitoring much more than just network latency or CPU consumption. You have analytic engines that trace patterns in user behaviors—both consumers and employees. Your bots constantly review log files, looking for unusual activity and notifying you of anomalies.

Optimizing Systems

As a systems operator, you are your company's best agent for maximizing performance because your analytics help you choose the correct infrastructure configuration, the optimal storage methods, and the best possible customer outcome.



By 123net - Own work, CC BY-SA 3.0, <https://commons.wikimedia.org/w/index.php?curid=17384917>

However, you do more than optimize for speed; you optimize for cost. By using elastic environments, your environment not only automatically scales out during peak demand to minimize latency, but it also automatically scales in later to minimize spend. You manage an environment that is highly utilized every hour of every day.

Fortifying Systems

Things break and systems go offline, but you don't let that keep you up at night. You maintain highly available architectures: systems that detect failed components and automatically switch over, replacing and restoring as needed without interruption of service to your consumers.

But your availability methods cover more than single regions and multiple Availability Zones. Systems operations on AWS involves using multi-region and hybrid methods when needed to ensure continuity of operations no matter what Mother Nature throws at you.

Securing Systems

The combination of security groups, access control lists, and private networks in concert with native tools such as Amazon CloudFront and AWS Shield, help your environment stand up to the most sinister of attacks.

Threats don't always come from the outside, however. You know that the most dangerous vector is the internal attack. That's why you have meticulously employed a policy of compartmentalized, restricted privilege sets so that no one can step into unauthorized territory, along with detailed Application Programming Interface (API) logging that reports on all actions to provide comprehensive control over your assets.

AWS Certified SysOps Administrator - Associate

As detailed in the introduction to this chapter, AWS systems operators focus on a wide range of responsibilities. The *AWS Certified SysOps Administrator - Associate certification* is engineered to test your knowledge of systems operations domains. This book not only explains the domains on the exam, but it walks you through the different aspects of AWS with which you must be familiar in order to be successful as an AWS systems operator.

The test is organized into seven domains of relatively equal weight:

1. Monitoring and Metrics
2. High Availability
3. Analysis
4. Deployment and Provisioning
5. Data Management
6. Security
7. Networking

As you explore individual AWS architectures and services, it is important to note that many of the AWS products have operational considerations that apply to most, if not all, seven domains.

Which AWS Services Should You Study?

The simple answer is, "all of them."

AWS is constantly evolving and adding new offerings. As of this writing, AWS has more than 90 unique services. Each one has security, data, monitoring, and availability considerations. As an AWS systems operator, you are tasked with understanding those considerations along with how to optimize the service for performance and cost. The next few chapters in this book walk you through the service categories, explain how those services are addressed from an operational perspective, and discuss what you should study.

With more than 90 services and approximately 55 questions, mathematically not every service can be addressed in the certification exam. Commonly used services might appear in many different questions, although services with more specific use cases are much less likely to appear.

For example, when studying the storage products, you must understand the options found in Amazon Simple Storage Service (Amazon S3), Amazon Elastic Block Store (Amazon EBS), and Amazon Glacier. You can expect these services to appear in questions throughout all of the domains. In contrast, AWS Snowmobile could be on the test, but because it is used only in a few specific scenarios, statistically it is unlikely to appear more than once, if at all.

The best rule of thumb is to look at common reference architectures. If you see services in those architectures, plan on them being integral to the test. However, do not discount other services; everything is fair game.

The following section provides specific reference architectures that you can use as you plan on what services to study.

Reference Architecture: The Three-Tier Design

One of the earliest cloud-native architectures used is the three-tier design, which includes the following:

- A front-end web server layer
- An application middle layer
- A database layer

In many cases, the first two layers might be fronted, or decoupled, with elastic load balancers.

Introduction to the Three-Tier Design

The model of a three-tier architecture was introduced in the late 1990s. It was an evolution from a two-tier architecture (client/server), which was an evolution from a monolithic (mainframe-based) architecture. One of the original drivers for a three-tier architecture was the desire to implement a web-based interface to existing applications, which were currently being accessed via a command-line interface (CLI).

The focus of this model is on application architecture. Each application has its own unique architecture, which exists independently of any other application.

Web Tier

The *Web Tier* is the front end to the application. It accepts the request from the user and passes that request to the Application Tier. It takes the response from the Application

Tier and presents it back to the user. The format of the response is controlled at this tier, whether it is an HTML document, a CSV file, a PDF file, or some other format.

This tier has no direct access to the Database Tier, and it should be decoupled from any processes happening in the Application Tier or the Database Tier.

Application Tier

The *Application Tier* is a middleware tier where the internal business logic resides. It responds to requests from the Web Tier and communicates directly with the Database Tier. The Application Tier operates and scales independently of the other tiers.

Database Tier

The *Database Tier* is a back-end tier where the databases manage the state of the application. This tier should only be accessed by the Application Tier. It processes requests from the Application Tier and provides responses back to the Application Tier.

Sample Scenario

To better prepare you for the exam, this book references a few sample architectures. These are provided to give a framework to the discussions. Although the problem we might be addressing is specific, the services we use are universal to most architectures on AWS.



Real World Scenario

Three-Tier Architecture

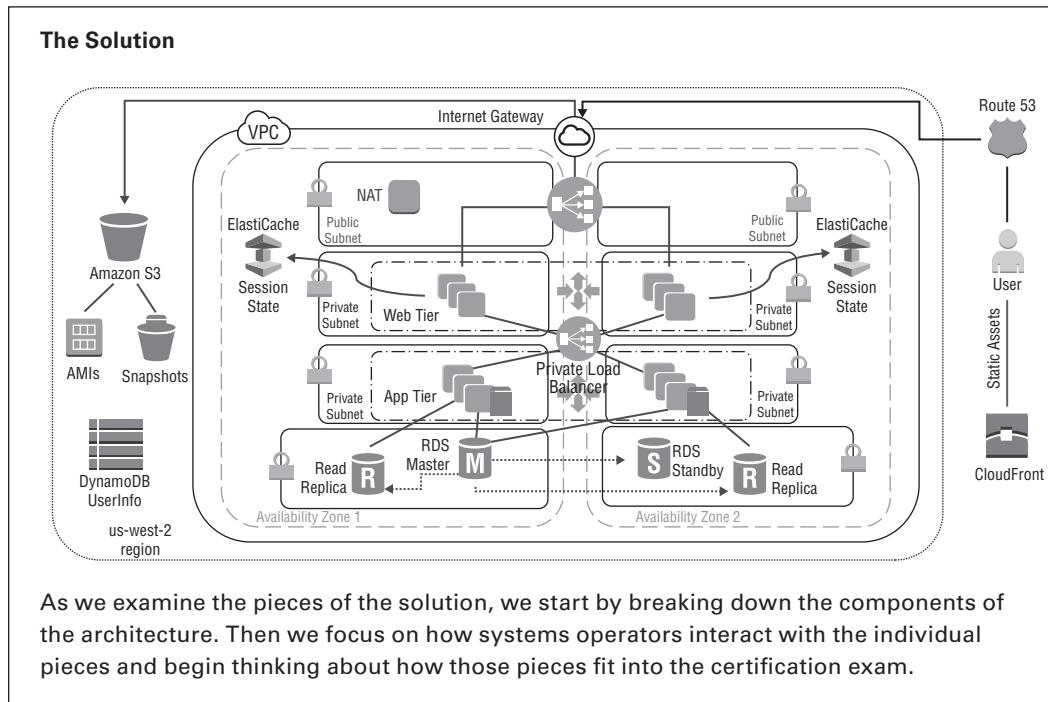
The Challenge

An application runs an order management system for a global company. The application will manage inventory, customer records, and orders in an integrated system.

Some of the system requirements include flexibility to adjust to changing needs. It must be scalable to handle variable customer loads. It must have separate front-end and processing layers to allow User Interface (UI) development to be isolated from business logic programming.

It must be cost effective. In addition to scalable web and application instances, it should leverage native, cost-effective services such as elastic load balancing and Amazon S3.

The environment must be secure. Steps should be taken to ensure that all traffic is properly protected in transit and at rest. All access must be controlled and monitored at all times. All critical data must be stored in durable, highly-available systems, protected against node failure.



Environment

Architectures live inside *AWS Regions*; in this scenario, in us-west-2 (Oregon, United States). Regions are made up of multiple *Availability Zones*, which provide the foundation for highly available architectures. Although this is a systems operation exam, it is critical to understand the nature of AWS Regions and Availability Zones.



Each AWS Region is a separate geographic area. Each AWS Region has multiple, isolated locations known as *Availability Zones*. *AWS Regions* and *Availability Zones* are discussed in Chapter 5, “Networking.”

Networking

Networking components start inside the AWS Region with Amazon Virtual Private Cloud (Amazon VPC). *Amazon VPC* is a private network in the AWS Region that isolates all traffic from the millions of other applications running in AWS. A deep dive into Amazon VPC (and the rest of its components) is found in Chapter 5.

Amazon VPC is divided into *subnets*; all assets running in your Amazon VPC are assigned to a subnet. Unlike on-premises subnetting decisions that can affect latency between servers, Amazon VPC subnets only affect access. Access between subnets is

controlled through *network Access Control Lists (nACLs)*, and access in and out of Amazon VPC is controlled through attached gateways. In this scenario, the only gateway is the *Internet Gateway (IGW)*, and it allows traffic to and from external (public IP) sources.

By granting route table access to the gateway only to specific subnets, ingress and egress can be tightly controlled. In this scenario, public subnets indicate IGW access. Without IGW access, the subnets become private; that is, they are accessible only to private IP networks.



To learn about the other gateways that could be leveraged to create hybrid or other private architectures, refer to Chapter 5.

Security groups are often part of the networking discussion. They provide stateful firewalls that operate at the Hypervisor levels for all individual *Amazon Elastic Compute Cloud (Amazon EC2)* instances and other Amazon VPC objects. In this scenario, we potentially have seven different security groups:

Public Elastic Load Balancing The only security group that allows full public access

Web Tier Amazon EC2 This accepts traffic only from public Elastic Load Balancing.

Private Elastic Load Balancing This accepts traffic only from Web Tier Amazon EC2.

Application Tier Amazon EC2 This accepts traffic only from private Elastic Load Balancing.

Amazon ElastiCache This accepts traffic only from Application Tier Amazon EC2.

Amazon Relational Database Service (Amazon RDS) This accepts traffic only from Application Tier Amazon EC2.

Network Address Translation (NAT) This is used only for internally initiated outbound traffic.

By specifically stacking security groups in this manner, you can provide layers of network security that surround the database portion of the three-tier design.

Compute

In this scenario, you use traditional compute methods, such as Linux servers running on Amazon EC2. Amazon EC2 comes in many sizes (how many CPUs, how much memory, how much network capacity, and so on), known as *instances*. Based on the Amazon Machine Image (AMI), each Amazon EC2 instance can run a wide range of Linux- or Windows-based operating systems as well as preinstalled software packages. Amazon EC2 instances also support runtime configuration as required.

The requirements for the scenario include scalable solutions. AWS provides Auto Scaling as an engine that can take predefined launch configurations and dynamically add or remove instances from the web or the Application Tier based on metrics.



Details on Amazon EC2, Auto Scaling, and other compute resources are found in Chapter 4, "Compute."

Database

Amazon RDS runs in your Amazon VPC on Amazon EC2. You select the database engine and version (MySQL, Oracle, Postgres, and so forth) and the configuration (the size of the Amazon EC2 instance, which subnets to use, how often to take backups, and so on). Amazon RDS takes care of the infrastructure of the instances and the engine; your database administrator (DBA) takes care of the database schema and data.

This scenario also includes *Amazon DynamoDB*, a native NoSQL engine optimized for consistent low latency, high availability, and strongly consistent reads and writes. Unlike Amazon RDS (or do-it-yourself databases running on Amazon EC2), Amazon DynamoDB operates at the regional level through API access only.



For details on how Amazon DynamoDB and other databases function, refer to Chapter 7, "Databases."

Storage

This scenario looks at storage in three different areas: the block storage used by the Amazon EC2 instances, the object storage keeping all of the media as well as backups and AMIs, and the caching storage used by Amazon CloudFront.

Amazon EBS is durable, persistent block storage used by most Amazon EC2 and Amazon RDS instances. It provides drive space for boot volumes and data volumes. Additionally, AWS provides ephemeral storage for many Amazon EC2 instance types through instance storage. Deciding which one to use becomes an operational value judgment, one that compares speed, persistence, and cost.

Object storage is provided by *Amazon S3*. *Amazon S3*, like Amazon DynamoDB, operates at the regional level outside Amazon VPC. It is only accessed through API commands that your operations team controls with fine-grained precision. Highly cost-effective and massively durable, *Amazon S3* provides web-enabled storage for content as well as protected storage for database backups and AMI storage.

Amazon CloudFront is the AWS *content delivery network service (CDN)*. This application leverages *Amazon CloudFront* to cache content close to consumers in order to improve performance (reduce latency) and reduce costs.



Storage systems, including shared file systems, the Amazon Elastic File System (*Amazon EFS*), and cold storage via *Amazon Glacier*, are discussed in Chapter 6, "Storage."

User Management

Although not drawn in the sample three-tier architecture diagram, user management becomes one of the critical elements of the AWS operational design. Operator access is controlled through *AWS Identity and Access Management (IAM)*. IAM maintains control over validating authentication methods (passwords, access keys, and so on) and then grants access to authenticated operators.

Because everything in AWS is accessed through APIs, IAM becomes a comprehensive tool for controlling all permissions to AWS services and resources.

For established enterprise customers, IAM can be integrated with existing directory systems via AWS Directory Service.



AWS IAM controls access to AWS services and resources. It does not control access to the Amazon EC2 operating system or application-level authentication. For more details, refer to the shared responsibility model in Chapter 3, "Security and AWS Identity and Access Management (IAM)."

Security, Monitoring, and Deployment

Security is integral to every part of the AWS platform. This means that security is part of each piece of the architecture.



There are some specific AWS security tools, such as Amazon Inspector, Amazon VPC Flow Logs, Amazon CloudWatch Logs, and others which provide a more focused toolset that the AWS operations team can leverage to ensure the security profile of the AWS application. These and many other tools are discussed in Chapter 3.

Monitoring of critical systems is provided by *Amazon CloudWatch*, which provides visibility into metrics that happen on the Customer side of the shared responsibility model. Thousands of metrics across more than 90 services keep track of everything from CPU consumption to latency, queue depths, and so on.

AWS CloudTrail records every API call in the AWS system, including:

- Who made the API call
- When the API call was performed
- Where the API call originated
- The result of the API call

These records and other log files are processed through Amazon CloudWatch Logs, which analyze text data for patterns that trigger alerts and corresponding actions.

Automated deployment methods ensure that human error does not disrupt rollouts or updates to production or sandbox environments. *AWS CloudFormation* turns infrastructure plans into code, allowing your operations team to build and tear down entire systems in a single action. Refer to Chapter 8, "Application Deployment and Management," for more details.

Key Products: Three-Tier Design

As described above, the three-tier architecture consists of a web front end, an application layer, and database layer. In addition to the compute, storage, and database resources, additional AWS infrastructure may need to be deployed. Refer to Table 1.1 for a list of key products.

TABLE 1.1 Key Products: Three-Tier Architecture

| Tools to Enable Hybrid Cloud Architectures | Description |
|--|---|
| AWS Regions and Availability Zones | Amazon EC2 is hosted in multiple locations world-wide. These locations are composed of regions and Availability Zones. Each region is a separate geographic area. Amazon EC2 provides you with the ability to place resources, such as instances and data, in multiple locations. |
| Availability Zones | Within a region are two or more Availability Zones. The Availability Zone is isolated, but the Availability Zones in a region are connected through low-latency links. |
| Edge Locations | Edge Locations = AWS Lambda@Edge Amazon CloudFront, Amazon Route 53, AWS Shield, and AWS WAF services that are offered at AWS Edge Locations. |
| Hybrid cloud architecture | Integration of on-premises resources with cloud resources |
| Amazon Route 53 | Amazon Route 53 is a highly available and scalable cloud Domain Name System (DNS) web service. |
| Amazon CloudFront | Amazon CloudFront is a web service that speeds up distribution of your static and dynamic web content to your users. CloudFront delivers your content through a worldwide network of data centers called <i>edge locations</i> . |
| Amazon VPC | A virtual private cloud (VPC) is a virtual network dedicated to your AWS account. It is logically isolated from other virtual networks in the AWS cloud. You can launch your AWS resources, such as Amazon EC2 instances, into your VPC. |
| Internet Gateways | An Internet gateway is a horizontally-scaled, redundant, and highly available VPC component that allows communication between instances in your VPC and the Internet. |
| Subnets | A subnetwork or subnet is a logical subdivision of an IP network. |
| Route tables | A route table is a set of rules that is used to determine where data packets traveling over an Internet Protocol (IP) network will be directed. |

TABLE 1.1 Key Products: Three-Tier Architecture (*continued*)

| Tools to Enable Hybrid Cloud Architectures | Description |
|---|--|
| Amazon EC2 Security groups | A security group acts as a virtual firewall that controls the traffic for one or more instances. |
| AWS Elastic Load Balancing | Elastic Load Balancing automatically distributes your incoming application traffic across multiple targets, such as Amazon EC2 instances. It monitors the health of registered targets and routes traffic only to the healthy targets. Elastic Load Balancing supports two types of load balancers: Application Load Balancers and Classic Load Balancers. |
| Amazon Elastic Compute Cloud (Amazon EC2) | Amazon Elastic Compute Cloud (Amazon EC2) is a web service that provides secure, resizable compute capacity in the cloud. |
| Auto Scaling | Auto Scaling helps you maintain application availability and allows you to scale your Amazon EC2 capacity up or down automatically according to conditions you define. |
| Amazon Relational Database Service (Amazon RDS) | Amazon Relational Database Service (Amazon RDS) is a web service that makes it easier to set up, operate, and scale a relational database in the cloud. |
| Amazon DynamoDB | Amazon DynamoDB is a fully-managed NoSQL database service that provides fast and predictable performance with seamless scalability. |
| Amazon ElastiCache | Amazon ElastiCache is a web service that makes it easy to deploy, operate, and scale an in-memory data store or cache in the cloud. |
| Amazon Simple Storage Service (Amazon S3) | Amazon S3 is object storage with a simple web service interface to store and retrieve any amount of data from anywhere on the web. |
| Amazon Elastic Block Store (Amazon EBS) | Amazon EBS provides persistent block storage volumes for use with Amazon EC2 instances in the AWS Cloud. |
| Amazon Elastic File System (Amazon EFS) | Amazon EFS is a file storage service for use with Amazon EC2. Amazon EFS provides a file system interface, file system access semantics (such as strong consistency and file locking), and concurrently-accessible storage for up to thousands of Amazon EC2 instances. |
| Amazon Glacier | Amazon Glacier is an extremely low-cost storage service that provides secure, durable, and flexible storage for data backup and archival. |

| Tools to Enable Hybrid Cloud Architectures | Description |
|--|---|
| AWS Identity and Access Management (IAM) | AWS IAM is a web service that helps you securely control access to AWS resources for your users. You use IAM to control who can use your AWS resources (authentication) and what resources they can use and in what ways (authorization). |
| Active Directory Connector | AD Connector is designed to give you an easy way to establish a trusted relationship between your Active Directory and AWS. |
| Web identity federation | AWS IAM supports identity federation for delegated access to the AWS Management Console or AWS APIs. With identity federation, external identities (federated users) are granted secure access to resources in your AWS account without having to create IAM users. |
| Amazon CloudWatch | Amazon CloudWatch is a monitoring service for AWS cloud resources and the applications you run on AWS. You can use Amazon CloudWatch to collect and track metrics, collect and monitor log files, set alarms, and automatically react to changes in your AWS resources. |
| Amazon CloudWatch Logs | You can use Amazon CloudWatch Logs to monitor, store, and access your log files from Amazon Elastic Compute Cloud (Amazon EC2) instances, AWS CloudTrail, and other sources. You can then retrieve the associated log data from CloudWatch Logs. |
| Amazon VPC Flow Logs | VPC Flow Logs is a feature that enables you to capture information about the IP traffic going to and from network interfaces in your VPC. Flow log data is stored using Amazon CloudWatch Logs. |
| Amazon Inspector | Amazon Inspector is an automated security assessment service that helps improve the security and compliance of applications deployed on AWS. Amazon Inspector automatically assesses applications for vulnerabilities or deviations from best practices. |
| Amazon S3 Access Logs | In order to track requests for access to your bucket, you can enable access logging. Each access log record provides details about a single access request, such as the requester, bucket name, request time, request action, response status, and error code, if any. |
| AWS CloudTrail | AWS CloudTrail is a web service that records API calls made on your account and delivers log files to your Amazon S3 bucket. |
| AWS CloudFormation | AWS CloudFormation is a service that gives developers and businesses an easy way to create a collection of related AWS resources and provision them in an orderly and predictable fashion. |

TABLE 1.1 Key Products: Three-Tier Architecture (*continued*)

| Tools to Enable Hybrid Cloud Architectures | Description |
|--|---|
| AWS Elastic Beanstalk | AWS Elastic Beanstalk makes it even easier for developers to deploy and manage applications in the AWS Cloud quickly. Developers simply upload their application, and Elastic Beanstalk automatically handles the deployment details of capacity provisioning, load balancing, Auto Scaling, and application health monitoring. |
| AWS OpsWorks Stacks | AWS OpsWorks Stacks lets you manage applications and servers on AWS and on-premises. With OpsWorks Stacks, you can model your application as a stack containing different layers, such as load balancing, database, and application server. |

It may seem like a daunting list, but this represents the core services (the toolset) that all AWS systems operators need to understand fully. As with any craft, it is important to use the right tool for the right job. You could use a torque wrench to smooth wet concrete, but of course there are much more appropriate tools for that task. Knowing the wide variety of AWS tools available to you is just as important.

Reference Architecture: The Serverless Design

As application design continues to evolve, individual instances are replaced with *container services*. Container services eventually are replaced by the final abstraction: *serverless architectures*.

There are many variations of serverless architectures. Rather than assume a generic use case, let's look at a specific scenario that might be used by your operations team.

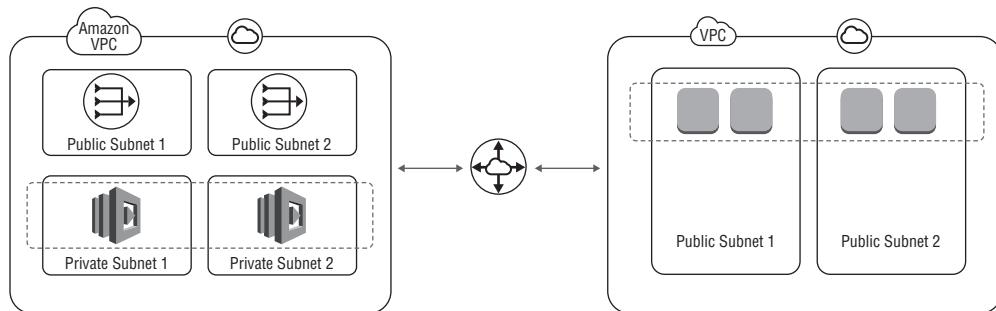
 **Real World Scenario**

Serverless Architectures

The Challenge

In this scenario, we want to find a better way to track the number of outstanding security updates on our production fleet. A serverless solution would be ideal, because we would not be adding any servers to maintain and we would only be paying for the compute time of the AWS Lambda functions.

The Solution



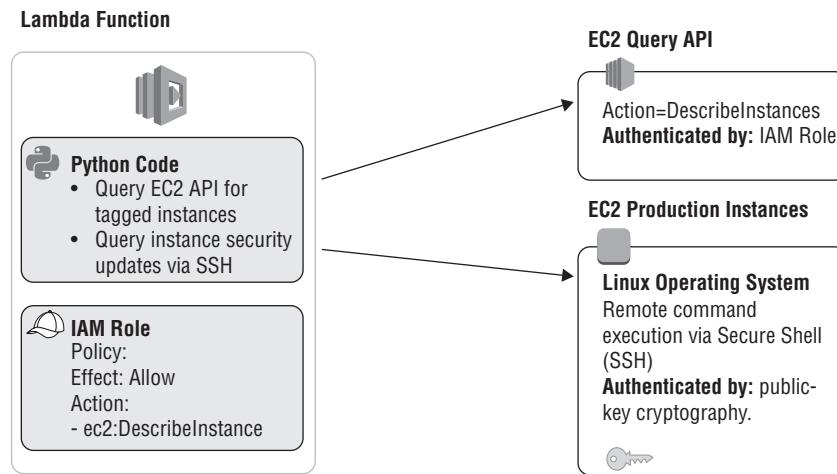
Python code executing in AWS Lambda on a regular schedule will use the Secure Shell (SSH) protocol to query for outstanding security updates on production instances. Python code (running anywhere) can use the AWS Boto Software Development Kit (SDK) to query Amazon EC2 for a list of specially tagged instances. The Python code establishes an SSH connection to the instances, and it executes a small script to find the number of required security updates. After you have this information, you can present it to the systems operations team as a tag on the instances, again using the AWS Boto SDK.

Networking

The AWS *Lambda functions* run in their own Amazon VPC. We establish Amazon VPC peering between the two Amazon VPCs to allow network connections between the AWS Lambda function and the production Amazon EC2 instances. This requires the creation of routing tables to direct the traffic between the two Amazon VPCs.

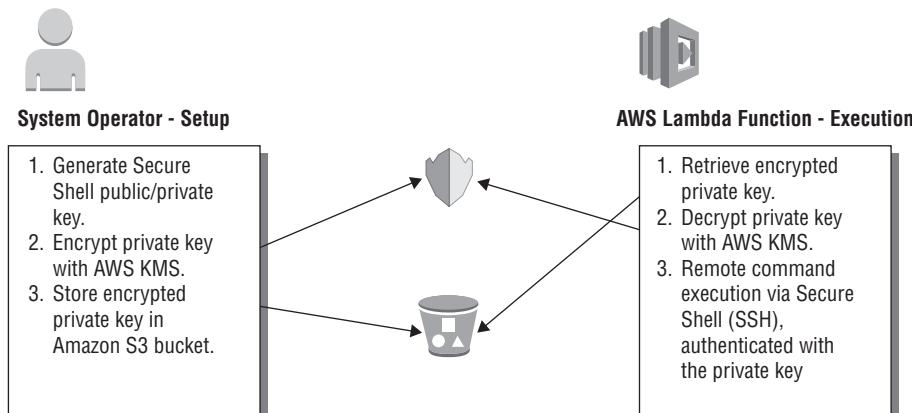
Security and Authentication

The AWS Lambda function must authenticate at two different levels: when the function queries the Amazon EC2 APIs via the Boto SDK and when the function establishes an SSH connection to the operating system on the production instances. AWS Lambda functions are configured with an IAM role and policy, which grants access to query the Amazon EC2 APIs. SSH authentication uses a Rivest-Shamir-Adleman (RSA) public/private key authentication. The AWS Lambda function has the private portion on the key. The Linux operating system on the production instances is configured with the public portion of the key. The operating system uses the public key to authenticate the SSH connection being initiated from the AWS Lambda function (see Figure 1.1).

FIGURE 1.1 Lambda function interacting with the Amazon EC2 API and EC2 instances

Lambda supports these runtime versions: Node.js, Java, and .Net Core. For more information, see Chapter 4.

Let's take an extra step to secure the private portion of the SSH key. This key is used by the AWS Lambda function to prove that it is allowed to SSH into the production instances and execute a script—so it is very important to keep secrets secret! The secret key is encrypted using the *AWS Key Management Service (AWS KMS)* and stored in Amazon S3. For the AWS Lambda function to retrieve the key from Amazon S3 and decrypt with AWS KMS, you must update the IAM policy associated with the AWS Lambda function. More information on cryptography is provided in Chapter 3. (See Figure 1.2.)

FIGURE 1.2 AWS KMS operations with Lambda

Who is allowed to access the encrypted private key in Amazon S3? Who is allowed to decrypt it? This is determined by the IAM policies in the AWS application.

Where and how do we apply network firewall type rules? The AWS Lambda function will be communicating to the production Amazon EC2 instances on the SSH port 22. Let's apply the least privilege principle here and ensure that only the AWS Lambda function is able to connect on port 22. We do this by creating security groups for both the production instances and the AWS Lambda function.

Key Product: Serverless Design

Many of the same services used in the three-tier architecture are used in the serverless design. Here are some of the unique services leveraged by this serverless architecture:

TABLE 1.2 Key Products: Serverless Design

| AWS Product | Description |
|--|---|
| AWS Lambda | AWS Lambda lets you run code without provisioning or managing servers. |
| AWS Lambda@Edge | Lambda@Edge, now in Preview, allows you to write functions deployed to the AWS network of Edge locations in response to Amazon CloudFront. |
| AWS Key Management Service (AWS KMS) | AWS KMS is a managed service that makes it easy for you to create and control the encryption keys used to encrypt your data. |
| Amazon S3 web hosting | You can host a static website on Amazon S3. On a static website, individual web pages include static content. They may also contain client-side scripts. By contrast, a dynamic website relies on server-side processing, including server-side scripts such as PHP, JSP, or ASP.NET. Amazon S3 does not support server-side scripting. |
| Amazon API Gateway | Amazon API Gateway is a fully managed service that makes it easy for developers to publish, maintain, monitor, and secure APIs at any scale. |
| Amazon Kinesis | Amazon Kinesis is a platform for streaming data on AWS, offering powerful services to make it easy to load and analyze streaming data and also providing the ability for you to build custom streaming data applications for specialized needs. |
| Amazon Simple Queue Service (Amazon SQS) | Amazon Simple Queue Service (Amazon SQS) is a fully-managed message queuing service for reliably communicating among distributed software components and microservices—at any scale. |

TABLE 1.2 Key Products: Serverless Design (*continued*)

| AWS Product | Description |
|---|---|
| Amazon Simple Notification Service (Amazon SNS) | Amazon SNS is a web service that coordinates and manages the delivery or sending of messages to subscribing endpoints or clients. |
| Amazon Simple Email Service (Amazon SES) | Amazon SES is a low-cost solution for sending automated emails, such as order confirmations, shipping notices, order status updates, policy changes, password resets, and other messages that keep your customers informed. |
| AWS Web Application Firewall (AWS WAF) | AWS WAF gives you control over which traffic to allow or block to your web applications by defining customizable web security rules. |
| AWS Shield | AWS Shield is a managed Distributed Denial of Service (DDoS) protection service that safeguards web applications running on AWS. |

Summary

Preparing for the certification exam requires comfort with a wide range of AWS services. One of the best ways to get comfortable is to use the AWS services themselves. Make sure that as part of your study, you take the time to create an account on AWS, log in to the AWS Management Console, launch the products, and get used to managing the various options. Practice builds the mental muscle memory that will give you the confidence in your answers.

Now that you know what types of architectures you will be dealing with and which products deserve the majority of your focus, let's start looking through the various service families covered throughout the AWS Certified SysOps Administrator - Associate exam.

Exam Essentials

Each chapter in this book ends with a list of important concepts to study. This list is not comprehensive, as the material is covered in the chapter itself, but the concepts are a good place to do a quick review of important testing areas. Every chapter ends with a useful tip from AWS trainers who specialize in helping people pass their certification exams. Look to these tips for good test-taking strategies that complement your core AWS knowledge.

Key Pieces to Study

Understand how AWS Regions and Availability Zones work to provide geographic distribution of services. Know how to deploy your environment across multiple Availability Zones and how to use Amazon CloudFront to take advantage of AWS edge locations.

Understand the shared responsibility model and that it is foundational to understanding how to secure your environment in AWS. Know which parts of any given service are managed by AWS and which parts you are responsible for securing.

Understand how the IAM engine separates the authentication layer from the authorization process. Be familiar with the way that credentials are presented to AWS when an API is called.

Test Taking Tips

Time management is key for this exam. You only have 80 minutes—don't waste them all on a question that has you stumped. Mark it for later review and move on. You will often be surprised that, when you come back to it later, the answer will be clear.

There is no penalty for wrong guesses. Make sure that you enter an answer for every question, even if you have no idea what the right answer might be. You won't pass the exam if you guess every question, but it never hurts to try on the few that you might not know.

The AWS Certified SysOps Administrator - Associate exam is not designed to give you trick questions. If one answer seems obviously right, but another answer might be correct under special circumstances, go with the obvious answer. Dr. Theodore Woodward's aphorism for his University of Maryland medical interns applies here: "If you hear hoof beats, think of horses not zebras."

Multiple-choice questions require all answers to be correct; there is no partial credit for getting a portion correct. Pay extra attention to those questions when doing your review.

Plan on leaving time at the end of the exam for review. Even if you think you know an answer, you can mark it and return to it when you are done with the exam. Go through each one of those marked questions to make sure that you are still confident with those answers. Just be careful not to overthink your answer (remember "horses not zebras").

Many questions have answer sets that are combinations of two pairs of answers. In AWS, everything is an API. In the next chapter, you will learn how to work with APIs and SDKs. So let's start our engines and get on with the nitty gritty of working with AWS Services!

Review Questions

1. Which AWS Cloud service allows you to gain system-wide visibility into resource utilization, application performance, and operational health?
 - A. Amazon CloudWatch
 - B. AWS OpsWorks
 - C. AWS Identity and Management (IAM)
 - D. AWS CloudTrail
2. Which AWS Cloud service enables you to capture information about the IP traffic going to and from network interfaces in your VPC?
 - A. Amazon CloudWatch
 - B. AWS OpsWorks
 - C. AWS CloudFormation
 - D. Amazon VPC Flow Logs
3. Which AWS Cloud service enables governance, compliance, operational auditing, and risk auditing of your AWS account?
 - A. Amazon CloudWatch
 - B. AWS CloudTrail
 - C. Amazon Simple Storage Service (Amazon S3) Access Logs
 - D. Amazon Elastic Compute Cloud (Amazon EC2) Security Groups
4. What is the term used for an environment that extends an existing on-premises infrastructure into the cloud to connect cloud resources to internal systems?
 - A. Scatter architecture
 - B. Multi-location architecture
 - C. Hybrid cloud architecture
 - D. There isn't a term for this type of architecture.
5. Which of the following services acts as a virtual firewall that controls the traffic for one or more instances?
 - A. Network Access Control Lists (nACLs)
 - B. Security Groups
 - C. Availability Zones
 - D. Amazon Virtual Private Cloud (Amazon VPC)
6. A three-tier architecture is comprised of which of the following layers? (Choose three.)
 - A. Database layer
 - B. Front-end web server layer
 - C. Security layer
 - D. Application layer

7. Each AWS region is composed of two or more locations that provide you with the ability to introduce high availability, fault tolerance, and/or scale to your applications. What are these locations called?
 - A. Data centers
 - B. Edge locations
 - C. Compute centers
 - D. Availability Zones
8. What AWS Cloud service is designed to give you an easy way to establish a trusted relationship between your Active Directory and AWS?
 - A. Amazon Elastic Compute Cloud (Amazon EC2)
 - B. AWS Key Management Service (AWS KMS)
 - C. Amazon Virtual Private Cloud (Amazon VPC)
 - D. Active Directory Connector
9. What AWS Cloud service provides a logically isolated section of the AWS Cloud where systems operators can launch AWS resources into a virtual network they defined?
 - A. Amazon Virtual Private Cloud (Amazon VPC)
 - B. Amazon Route 53
 - C. Availability Zones
 - D. Security Groups
10. You manage a fleet of web servers hosted on Amazon Elastic Compute Cloud (Amazon EC2). Most, if not all, of the websites are static in nature. What AWS Cloud service can host a static website, thus replacing servers?
 - A. Amazon Elastic Compute Cloud (Amazon EC2)
 - B. Amazon Simple Storage Service (Amazon S3)
 - C. Amazon Route 53
 - D. Amazon API Gateway

Chapter 2

Working with AWS Cloud Services

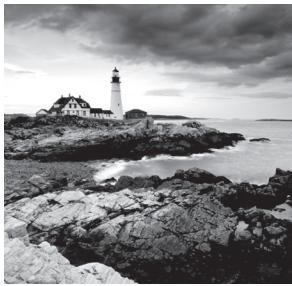
**THE AWS CERTIFIED SYSOPS
ADMINISTRATOR - ASSOCIATE EXAM
TOPICS COVERED IN THIS CHAPTER MAY
INCLUDE, BUT ARE NOT LIMITED TO, THE
FOLLOWING:**

Domain 4.0 Deployment and Provisioning

- ✓ 4.2 Demonstrate the ability to provision cloud resources and manage implementation automation.

Content may include the following:

- How to configure your workstation to manage and deploy AWS resources



Introduction to AWS Cloud Services

As a qualified candidate for the AWS Certified SysOps Administrator – Associate certification, it's not enough to read the guide—you need to get your hands dirty by digging in. This chapter provides you with a starting point for using several AWS tools that will help you be successful as you learn how to use the cloud in a more effective manner.

Systems Operations Using the AWS Toolset

It's likely that you are familiar with the *AWS Management Console*, the web-based interface to AWS Cloud services. In this study guide, we won't spend much time instructing you on the basics of the AWS Management Console. You've probably been using it already, and we believe there is more value in instructing you, the systems operator, in the tools that will allow you to integrate AWS functionality into the scripting environments in which you are already an expert.

There are several AWS-provided tools available for customers to create, maintain, and delete AWS resources at the command line or in code: the *AWS Command Line Interface (AWS CLI)*, *AWS Tools for PowerShell*, and *AWS Software Development Kits (SDKs)*. Understanding these tools is an essential part of an effective cloud operations team's automation and scripting toolkit.

Installing the AWS CLI

To find instructions on how to install the latest version of the AWS CLI, navigate to <http://aws.amazon.com/cli> in a web browser. For Windows, you'll download and install the 32-bit or 64-bit installer that is appropriate for your computer. If you're using Mac or Linux and have Python and pip installed, installing the latest version of the AWS CLI is as simple as running **pip install awscli**.

Upgrading the AWS CLI

Upgrading the AWS CLI on a Linux or Mac computer is as simple as running `pip install --upgrade awscli`. For Windows users, you'll have to download the latest installer and install the latest version.



You should follow the AWS Security Bulletins page at <https://aws.amazon.com/security/security-bulletins/> to stay aware of security notifications about the AWS CLI.

Configuration

After installing the AWS CLI, run `aws configure` to configure it with your credentials. Specifically, you will need an *access key* and *secret key* created for your AWS Identity and Access Management (IAM) user. Optionally, you can set a *region* (for example, `us-east-1`) and a default output format (for example, `JSON`) after entering your access key and secret key. The `aws configure` Command Options are shown in Table 2.1.

TABLE 2.1 The `aws configure` Command Options

| Variable | Description | Options |
|-----------------------|--|-----------------------------|
| AWS access key ID | Required. The access key obtained for your user from the IAM service | None |
| AWS secret access key | Required. The secret key obtained for your user from the IAM service | None |
| Default region name | Required. The shortened AWS Region name you want to use by default. This default region can be overridden on each execution using the <code>--region</code> parameter. | Any valid short region name |
| Default output format | The format in which you want to have the AWS CLI output by default | "JSON," "text," or "table" |

<http://docs.aws.amazon.com/cli/latest/reference/configure/index.html>



Safeguard your access key and secret key credentials as you would a user name and password for the AWS Management Console. Safeguarding these credentials is crucial to help prevent unauthorized access to your AWS infrastructure.

If you ever believe that your credentials are compromised, you should deactivate them immediately.

You can also create multiple profiles by appending `--profile profile-name` to the `aws configure` command. This can be handy in a number of different situations. You may want to have separate profiles with separate privileges for development, testing, and production environments. You could also create unique profiles for multiple accounts that you need to access. Creating different profiles will allow you to execute commands using different configurations for each.

After you've run `aws configure`, your credentials are stored in `~/.aws/credentials` on Mac or Linux, or in `%UserProfile%\aws\credentials` on Windows. Your other configuration parameters are stored in `~/.aws/config` on Mac or Linux, or in `%UserProfile%\aws\config` on Windows. The AWS CLI will look in these locations for the credentials and configuration information each time it is called to execute a command.



This chapter has only started covering the configuration options for the AWS CLI. AWS provides you with the ability to specify a Multi-Factor Authentication (MFA) device to use with your credentials, an *Amazon Resource Name (ARN)* corresponding to a role that you want to assume for cross-account access, and more. Find out more details on the configuration options available by running `aws help config-vars`.

Environment Variables

You can specify configuration parameters using environment variables as well, as listed in Table 2.2. This ability can come in handy for making swift changes in scripts or on a temporary basis from the command line.

TABLE 2.2 Environment Variables

| Environment Variable | Description |
|-----------------------|---|
| AWS_DEFAULT_PROFILE | Default profile name |
| AWS_DEFAULT_REGION | Default AWS Region |
| AWS_CONFIG_FILE | Alternate location of config |
| AWS_DEFAULT_OUTPUT | Default output style |
| AWS_CA_BUNDLE | Certificate authority (CA) certificate bundle |
| AWS_ACCESS_KEY_ID | AWS access key |
| AWS_SECRET_ACCESS_KEY | AWS secret key |
| AWS_SESSION_TOKEN | AWS token (temp credentials) |

How you change the variable depends on the shell you are using. In the bash shell, which is most commonly the default on Linux and Mac systems, you use the format `export environment_variable=option` to set the new variable.

Getting Help on the AWS CLI

You can add the option `help` to the end of nearly every AWS CLI command to determine a list of available options. For example, executing `aws help` will return a list of all of the services available to use as options. Running `aws s3 help` will return a list of the valid parameters to pass as part of a command-line call to Amazon Simple Storage Service (Amazon S3).

Autocompletion

Support for *tab completion*—the ability to start typing a command and have a list of valid options to complete your command appear when you press tab—is a feature built into the AWS CLI but not enabled by default. You can enable autocompletion for the bash shell (Linux or Mac) by typing `complete -C aws_completer aws`.

Source Code

AWS makes the AWS CLI source code available within the terms of the Apache License, Version 2.0. If you remain within the license, you can review the code before using it or adapt it into a new tool for your own project. There is an active community involved with the source code in which you are encouraged to participate. Find the code and more information on the user community at <https://github.com/aws/aws-cli>.



AWS publishes code on GitHub, not only for the AWS CLI, but also for the AWS SDKs and many other tools. Doing so helps give customers access to the code behind these tools to help them incorporate the code into their projects and spend more time innovating rather than creating building blocks. Take a look at some of the tools available at <https://github.com/aws/>.

Working with Services

Executing an AWS CLI command is as simple as typing `aws` and then a command string followed by a list of options.

The format of your command will generally take the form of the following:

```
aws service parameter1 parameter2 ... parameterN
```

For example, `aws ec2 describe-instances` will return a list of your Amazon Elastic Compute Cloud (Amazon EC2) instances, along with their properties, running in your configured region. `aws s3 ls s3://mycertification/` will return an object listing of an Amazon S3 bucket you own named mycertification.

Output Types

In the Configuration section, we mentioned that you can represent the data retrieved using the AWS CLI in three output formats: “JSON,” “text,” or “table.” Each format can provide a number of benefits to the user depending on the use case in question.

JSON is the default format, and it provides data in a form that is easily parsed and ingested by applications. This format is commonly used in other AWS Cloud services (for example, AWS CloudFormation), and it is a standard in which operations personnel should become well versed if they want to excel. Text output allows the operator to output data in a tab-delimited format that can be parsed by tools like grep and other text parsers. (If you happen to be a Linux systems administrator, you’re likely very familiar with this tool.) Table format is often more easily human readable than JSON or text.

Avoiding Unwieldy Lines

As you gain more experience using the AWS CLI, you will find that your command lines can become increasingly difficult to manage effectively as your parameters become more complex. There are several strategies to deal with this problem.

First, in Linux or Mac, you can use the backslash character to separate a command into several lines. For example, this command:

```
aws rds download-db-log-file-portion --db-instance-identifier awstest1  
--log-file-name "error/postgres.log"
```

is equivalent to the following command, parsed with backslashes:

```
aws rds \  
download-db-log-file-portion \  
--db-instance-identifier awstest1 \  
--log-file-name "error/postgres.log"
```

Using backslashes makes the command more easily comprehensible to a human reader, thus assisting with troubleshooting when errors occur.

Next, some AWS CLI commands take a JSON-formatted string as part of the input. For example, the **aws ec2 create-security-group** command has a parameter **--cli-input-json** that takes a JSON-formatted string as an input. As an alternative to entering the string via the command line, you can refer to a local file as follows:

```
aws ec2 create-security-group --cli-input-json file://filename.json
```

where `filename.json` is the file containing the JSON string.

Additionally, you can store the JSON string as an object in Amazon S3 or another web-hosted location and access the file as a URL:

```
aws ec2 create-security-group \  
--cli-input-json \  
https://s3.amazonaws.com/cheeeeessssseeee/filename.json
```

This gives you the ability to reuse more easily the JSON string that you’ve created for one environment in another.

Using `query` to Filter Results

As you explore using the AWS CLI, you will find that there is a wealth of information about your AWS environment that can be retrieved using the tool. Command-line output is comprehensive. Running the command `aws ec2 describe-instances` returns dozens of values describing each instance running: InstanceId, PublicDnsName, PrivateDnsName, InstanceType, and much more. There are times when you don't want to return all of those values, though. What do you do if you want to retrieve only a list of the Amazon Machine Image (AMI) IDs that your instances are running so that you can make sure that your fleet is running your preferred image?

That's where the `--query` option comes in. This option allows you to filter results so that only the output with the parameters you specify are returned. `Query` uses the JMESPath query language as its input for filtering to the results you specify.



You can find a tutorial for the JMESPath query language at
<http://jmespath.org/tutorial.html>.

Here are some examples of query in practical use cases. Perhaps you want to obtain the metadata for your Amazon Relational Database Service (Amazon RDS) instances, but only those that are running in the `us-east-1e` Availability Zone:

```
aws rds describe-db-instances \
--query 'DBInstances[?AvailabilityZone==`us-east-1e`]' \
--output text
```

Maybe you want a list of your AWS IoT things that are Intel Edison devices:

```
aws iot list-things --query 'things[?thingTypeName==`IntelEdison`]' --output text
```

Or maybe you've been tasked with identifying a list of the instances with their associated instance type that are running in your environment so that they can be targeted as candidates for upgrades to newer generation types:

```
aws ec2 describe-instances \
--query 'Reservations[*].Instances[*].[InstanceId, LaunchTime, InstanceType]' \
--output text
```

That last one is a bit different than what we've executed in the previous examples. Note that we are working our way down the JSON hierarchy. First we specify that everything under `Reservations` and then everything under `Instances` is in scope for our query (the `*` character works as our wildcard here). In the final set of brackets, we specify what specific fields at that level we want to return—`InstanceId`, `LaunchTime`, and `InstanceType` in this example, allowing us to see only which fields are useful to us for our task.

`Query` can be a powerful tool. However, output can vary among the resources you list using the AWS CLI (differing fields may be present in your output based on a number of variables). Accordingly, it's recommended that you rely on text format for any outputs that you run through `query`; you can see that we've added that `output` parameter to the queries here. Additionally, using text format makes it easier to use tools like `grep` on the output.

AWS Tools for Windows PowerShell

To this point, we've been focusing on the AWS CLI tool in our discussion of how a systems operator can effectively administer a customer's cloud resources from the command line. Because this tool works across operating systems, the AWS CLI provides an effective way to administer across various shells.

There is, however, a notable contingent of IT professionals whose favorite command-line shell is Windows PowerShell. To serve those customers who prefer PowerShell, we have provided a full-featured tool for that environment called *AWS Tools for Windows PowerShell*. Although we will not dive into this tool in this book, if you love PowerShell, you can find more information at <https://aws.amazon.com/powershell/>.

AWS Software Development Kits (SDKs)

AWS provides a number of SDKs for use by programmers. Although we don't expect that a systems operator would use an SDK directly on a regular basis, as a knowledgeable AWS resource, it's important that you understand that the SDKs and the underlying APIs they use exist, and that you have some general knowledge about how they are used.

There are a few reasons for this. For one thing, some of these languages—Python, for example—straddle the lines between programming languages that developers use to compile executable code and scripting languages that administrators use to perform infrastructure tasks. That leads into the next reason why we're talking about SDKs: The line between development and operations is increasingly blurry. As operations and development responsibilities merge into the new world of DevOps, it's important for those in charge of operations to understand the basics of how applications integrate with infrastructure.

AWS Certification Paths

There are three paths that an AWS Certification candidate can take toward Professional status: Architecting, Developing, and the one you're focusing on by reading this book, Operations. It's worth noting that while the Architecting path has its own professional certification (the AWS Certified Solutions Architect – Professional), the Developing and Operations paths share the same professional credential: the AWS Certified DevOps Engineer certification.

As the differentiation between Development and Operations becomes increasingly blurry, it's important for both groups to understand what the other does on a daily basis. Hence, the SysOps and Developer paths merge at the Professional level.

It's through the AWS SDKs and the APIs that underlie them that applications built on AWS can manage infrastructure as code. The concept of infrastructure as code is powerful, disruptive, and sets the cloud apart from the old IT world.

At the time this book was written, AWS SDKs were available for the following programming languages:

- Android
- Browser (JavaScript)
- iOS
- Java
- .NET
- Node.js
- PHP
- Python
- Ruby
- Go
- C++

There are also two purpose-specific SDKs:

- AWS Mobile SDK
- AWS IoT Device SDK

The language-specific SDKs contain APIs that allow you easily to incorporate the connectivity and functionality of the wider range of AWS Cloud services into your code without the difficulty of writing those functions yourself. Extensive documentation accompanies each SDK, giving you guidance as to how to integrate the functions into your code.

We focus on the AWS SDK for Python as our reference SDK for this chapter.

Boto

The AWS SDK for Python is also known as Boto. Like the other AWS SDKs and many of our tools, it is available as an open source project in GitHub for the community to view freely, download, and branch under the terms of its license. There is an active Boto community, including a chat group, which can help answer questions. Let's get started by installing Boto and jump right into using it.

AWS and Open Source

AWS has been committed to the idea of open source software since day one. Open source code allows customers to review code freely and contribute new code that is optimized or corrected. AWS not only uses open source software, such as Xen, SQL, and the Linux operating system, but often contributes improvements to various open source communities.

Installing Boto

Given that Boto is an SDK for Python, it requires Python to be installed prior to its own installation. The method of doing so depends on the operating system involved. You can find more information about installing Python at <http://www.python.org/>. Another prerequisite is pip, a Python tool for installing packages, which can be found at <https://pip.pypa.io/>.

After installing Python and pip, you install Boto using the following command:

```
pip install boto3
```

It's worth noting the `boto3` at the end of the `install` command. The current version of the Boto SDK is 3. Although Boto 2 is still in use, we highly encourage customers to use Boto 3. Throughout the rest of this chapter, when we refer to "Boto," we are referring to Boto 3.

By default, Boto uses the credential files that you established in setting up the AWS CLI as its own credentials for authenticating to the AWS API endpoints.

Features of Boto

Boto contains a variety of APIs that operate at either a high level or a low level. The low-level APIs (*Client APIs*) are mapped to AWS Cloud service-specific APIs. The details of how to use the low-level APIs are found in the Boto 3 documentation at <https://boto3.readthedocs.io/en/latest/guide/clients.html>. Although the low-level APIs can be useful, we suspect that those involved in systems operations will not often need to dig into the specifics of their use.

The higher-level option, *Resource APIs*, allows you to avoid calling the network at the low level and instead provide an object-oriented way to interact with AWS Cloud services. We'll cover the use of Resource APIs in more detail next.

Boto also has a helpful feature called the *waiter*. Waiters provide a structure that allows for code to wait for changes to occur in the cloud. For example, when you create a new Amazon EC2 instance, there is a nominal amount of time until that instance is ready to use. Having your code rely on a waiter to proceed only when the resource is ready can save you time and effort.

There is also support for multithreading in Boto. By importing the `threading` module, you can establish multiple Boto sessions. Those multiple Boto sessions operate independently from one another, allowing you to maintain a level of isolation between the transactions that you're running.

These are just a few of the features Boto offers. For an in-depth look at these features, or to learn more about other features available in this SDK, refer to the Boto General Feature Guides at <https://boto3.readthedocs.io/en/latest/guide/index.html#general-feature-guides>.

Fire It Up

If you want to use Boto in your Python code, you start by importing the Boto SDK:

```
import boto3
```

And, if you're using Interactive mode, you then press Enter.

Using Resource Application Programming Interfaces (APIs)

To start using the Boto class in Python, invoke it by calling `boto3.resource` and passing in a service name in single quotes. For example, if you wanted to perform an action using Amazon EC2, you would execute something similar to the following:

```
ec2 = boto3.resource('ec2')
```

You now have an object called `ec2`, which you can use to act on Amazon EC2 instances in your AWS account. You can then instantiate a method object pointing to a specific instance in your Amazon Virtual Private Cloud (Amazon VPC):

```
myinstance = ec2.Instance('i-0xxxxxxxxxxxxxx')
```

Perhaps you want to stop an Amazon EC2 instance programmatically, possibly at the end of a shift to save money when it's not being used. You can then issue a command to stop that instance:

```
myinstance.stop()
```

You can start the instance back up again automatically prior to the beginning of the next shift:

```
myinstance.start()
```

Acting on infrastructure resources isn't your only option with Boto. You can also retrieve information from the APIs about your resources. For example, perhaps you want to find out what AMI is being used for the instance in question. You can do that by passing the following command:

```
instance.image_id
```

A string is returned for this command containing the AMI ID of the named instance.

AWS Internet of Things (IoT) and AWS Mobile Software Development Kits (SDKs)

We've been covering the language-specific AWS SDKs, which focus on the management of many AWS Cloud services. There are two purpose-specific SDKs as well: The AWS IoT Device SDK and the AWS Mobile SDK. Like the general-usage SDKs that we covered previously, these purpose-specific SDKs also provide developers with the ability to use prebuilt libraries that make it easier for them to focus on innovation in their code, not in how they connect to the infrastructure that runs it.

The IoT and Mobile SDKs are different because they are purpose-built to streamline connecting physical devices, like phones and tablets or sensors and hubs, to the cloud. The SDKs are provided for a variety of languages commonly used on their respective platforms.

At the time this book was written, these SDKs were available for the following languages/platforms:

AWS Mobile SDK

- Android
- iOS
- Unity
- .NET
- Xamarin
- React Native

AWS IoT Device SDK

- Embedded C
- JavaScript
- Arduino Yún
- Java
- Python
- iOS
- Android
- C++

Like the SDKs previously discussed, many of these SDKs provide their source code in GitHub. Each contains extensive documentation and helpful sample code to allow developers to get up and running quickly.

Summary

As a highly-qualified AWS systems operator, you should have a strong understanding of the use of the AWS CLI. It makes your job easier and helps you automate much of the undifferentiated heavy lifting that performing cloud tasks manually would entail. Knowing how to use these AWS tools makes gaining experience to become a qualified candidate for the AWS Certified SysOps Administrator – Associate exam that much easier.

Additionally, you should have enough of an understanding of the AWS SDKs that you can help your development colleagues understand their business value and use your knowledge to help them troubleshoot as connectivity or infrastructure issues arise.

Exam Essentials

Know the basic ways to interact with AWS. AWS Management Console, AWS Software Development Kits, and the AWS Command Line Interface (AWS CLI). On the exam, you will not be asked any code, or commands. However, you must understand there is more than one way to administer your AWS resources.

Understand Access Keys. As the Systems Operator, understand that access keys are not generated when you create an IAM account. For more information on access keys, refer to Chapter 3, “Security and AWS Identity and Access Management (IAM).”

Resources to Review

For assistance in completing the exercises that follow, we recommend these resources:

The AWS CLI User Guide at:

<http://docs.aws.amazon.com/cli/latest/userguide>

AWS Command Line Interface on Microsoft Windows:

<http://docs.aws.amazon.com/cli/latest/userguide/awscli-install-windows.html>

AWS Tools for PowerShell:

<https://aws.amazon.com/powershell/>

JMESPath query language:

<http://jmespath.org/tutorial.html>

BOTO General Feature Guides:

<https://boto3.readthedocs.io/en/latest/guide/index.html#general-feature-guides>

Exercises

Throughout this book, you will be using the AWS CLI. These exercises serve as the instructions to install the CLI on Linux, Mac, and Windows.

By now you have set up an account in AWS. If you haven’t already, now would be the time to do so. It is important to note that these exercises are in your AWS account and thus are not free.

Use the Free Tier when launching resources. The AWS Free Tier applies to participating services across the following AWS regions: US East (Northern Virginia), US West (Oregon), US West (Northern California), Canada (Central), EU (London), EU (Ireland), EU (Frankfurt), Asia Pacific (Singapore), Asia Pacific (Tokyo), Asia Pacific (Sydney), and South America (Sao Paulo). For more information, see https://aws.amazon.com/s/dm/optimization/server-side-test/free-tier/free_np/.

EXERCISE 2.1**Install and Configure AWS CLI on Linux or Mac**

In this exercise, you install the AWS CLI and configure it for use with your account.

1. Begin by logging in to the AWS Management Console with your user name, password, and MFA device (if applicable).
2. Navigate to the IAM service.
3. Select Security, and then click on your user name.
4. Under Security Credentials, click Create Access Key.
5. Before closing the dialog box that appears, save your access key and secret key in a safe place.
6. If Python isn't installed on your system, install it using the directions at <http://www.python.org/>.
7. If pip isn't installed on your system, install it using the directions at <https://pip.pypa.io/en/stable/installing/>.
8. Install the AWS CLI using the command **pip install awscli --upgrade --user**.
9. Configure the AWS CLI by running the command **aws configure** and filling in the access key and secret key that you obtained in Step 5. Optionally (but highly recommended), specify a default region, such as us-east-2. Also optionally, specify a default format type.
10. Test that you have set up the AWS CLI correctly and can connect to the AWS API endpoints by running the command **aws ec2 describe-availability-zones**. The AWS CLI should return a list of the available Availability Zones in your default region.

EXERCISE 2.2**Install and Configure AWS CLI on Windows with MSI**

1. Begin by logging in to the AWS Management Console with your user name, password, and MFA device (if applicable).
2. Navigate to the IAM service.
3. Select Security, and then click on your user name.
4. Under Security Credentials, click Create Access Key.
5. Before closing the dialog box that appears, save your access key and secret key in a safe place.
6. Download the MSI Installer (<https://s3.amazonaws.com/aws-cli/AWSCLI64.msi>).

7. Run the downloaded MSI installer.
 8. Follow the instructions that appear.
 9. Open a command prompt.
 10. Type **aws --version**.
 11. Configure the AWS CLI by running the command **aws configure** and filling in the access key and secret key that you obtained in Step 5. Optionally (but highly recommended), specify a default region, such as us-east-2. Also optionally, specify a default format type.
 12. Test that you have set up the AWS CLI correctly and can connect to the AWS API endpoints by running the command **aws ec2 describe-availability-zones**. The AWS CLI should return a list of the available Availability Zones in your default region.
-

Review Questions

1. Which of the following is a dependency of the AWS CLI for Mac and Linux?
 - A. C++
 - B. Python
 - C. Java
 - D. .NET SDK
2. Which of the following AWS CLI output formats is best geared toward human viewing?
 - A. Table
 - B. JSON
 - C. Text
3. How can one split apart a long AWS CLI command line to make it more easily readable by the operator?
 - A. Use the semi-colon (;) between statements, with a newline after each backslash.
 - B. Press Enter after every three words, followed by the Enter key twice at the end of the statement.
 - C. Use backslashes (\) every few words, with a newline after each backslash.
 - D. Use the tab key between each word.
4. For which languages are AWS SDKs available? (Choose all that apply.)
 - A. Ruby
 - B. Basic
 - C. Perl
 - D. Python
 - E. Pascal
5. Which command can you run to find more information about the proper syntax and options for AWS IoT commands from the CLI?
 - A. aws help
 - B. aws-iot list-commands
 - C. aws iot help
 - D. aws --help
6. What does the waiter allow you to do in Boto?
 - A. Order a pool of Amazon EC2 instances to be delivered to your Auto Scaling group.
 - B. Delete all unused Security Groups in your Amazon Virtual Private Cloud (Amazon VPC).
 - C. Wait for ordered infrastructure to become available before continuing.
 - D. Automatically distribute Amazon S3 data across regions.

7. In which situation would you use the AWS IoT Device SDK?
 - A. To order AWS IoT Buttons
 - B. To create a new AWS account to use for IoT
 - C. As a dashboard for performing analytics upon your IoT messages
 - D. To simplify the process of connecting things to the AWS IoT service
8. For the CLI commands that accept formatted files as input, the input file must be in which of the following formats?
 - A. Text format
 - B. JSON
 - C. Comma Separated Values (CSV)
 - D. XML
 - E. HTML
9. Which option allows you to filter output?
 - A. --filter
 - B. --find
 - C. --sort
 - D. --query
10. In what file are your Access Key and Secret Key stored after executing the `aws configure` command?
 - A. config
 - B. credentials
 - C. profile
 - D. awskeys

Chapter 3



Security and AWS Identity and Access Management (IAM)

**THE AWS CERTIFIED SYSOPS
ADMINISTRATOR - ASSOCIATE EXAM
TOPICS COVERED IN THIS CHAPTER MAY
INCLUDE, BUT ARE NOT LIMITED TO, THE
FOLLOWING:**

Domain 1.0: Monitoring and Metrics

- ✓ 1.1 Demonstrate ability to monitor availability and performance

Domain 6.0: Security

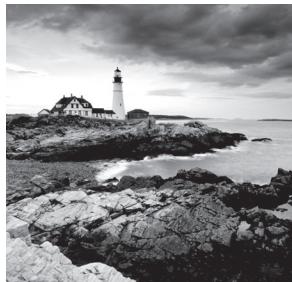
- ✓ 6.1 Implement and manage security policies
- ✓ 6.2 Ensure data integrity and access controls when using the AWS platform
- ✓ 6.3 Demonstrate understanding of the shared responsibility model
- ✓ 6.4 Demonstrate ability to prepare for security assessment use of AWS

Content may include the following:

- AWS platform compliance
- AWS security attributes (customer workloads down to physical layer)
- AWS administration and security services
- AWS Identity and Access Management (IAM)
- Amazon Virtual Private Cloud (Amazon VPC)
- AWS CloudTrail



- Amazon CloudWatch
- AWS Config
- Amazon Inspector
- Ingress vs. egress filtering and which AWS Cloud services and features fit
- Core Amazon Elastic Compute Cloud (Amazon EC2) and Amazon Simple Storage Service (Amazon S3) security feature sets
- Incorporating common conventional security products (firewall, Virtual Private Network [VPN])
- Distributed Denial of Service (DDoS) mitigation
- Encryption solutions (e.g., key services)
- Complex access controls (e.g., sophisticated security groups, Access Control Lists [ACLs])



Security on AWS

AWS delivers a scalable cloud computing platform with high availability and dependability that provides the tools to enable you to run a wide range of applications. These tools assist you in protecting the confidentiality, integrity, and availability of your systems and data.

The AWS Certified SysOps Administrator – Associate exam focuses on how to use the AWS tool set to secure your account and your environment. The Security domain is 15 percent of this exam!

Shared Responsibility Model

Before we go into the details of how AWS secures its resources, we talk about how security in the cloud is different than security in your on-premises datacenters. When you move computer systems and data to the cloud, security responsibilities become shared between you and your Cloud Services Provider (CSP). In this case, AWS is responsible for securing the underlying infrastructure that supports the cloud, and you’re responsible for anything that you put on the cloud or connect to the cloud. This shared responsibility model can reduce your operational burden in many ways, and in some cases, it may even improve your default security posture without any additional action on your part.

The amount of security configuration work you have to do varies depending on which services you select and how you evaluate the sensitivity of your data. However, there are certain security features—such as individual user accounts and credentials, Secure Sockets Layer (SSL)/Transport Layer Security (TLS) for data transmissions to encrypt data in transit, encryption of data at rest, and user activity logging—that you should configure no matter which AWS service you use.

AWS Security Responsibilities

AWS is responsible for protecting the global infrastructure that runs all of the services offered in the AWS Cloud. This infrastructure is composed of the hardware, software, networking, and facilities that run AWS Cloud services. Protecting this infrastructure is AWS number one priority. Although you can’t visit our datacenters or offices to see this protection firsthand, we provide several reports from third-party auditors, which have verified our compliance with a variety of relevant computer security standards and regulations.



In addition to protecting this global infrastructure, AWS is responsible for the security configuration of its products that are considered managed services. Examples of these services include Amazon DynamoDB, Amazon Relational Database Service (Amazon RDS), Amazon Redshift, Amazon EMR, Amazon WorkSpaces, and several other services. These services provide the scalability and flexibility of cloud-based resources with the additional benefit of being managed. For these services, AWS will handle basic security tasks like guest operating system and database patching, firewall configuration, and disaster recovery. For most of these managed services, all you have to do is configure logical access controls for the resources and protect your account credentials. A few of them may require additional tasks, such as setting up database user accounts, but the overall security configuration work is performed by the service.

Customer Security Responsibilities

With the AWS Cloud, you can provision virtual servers, storage, databases, and desktops in minutes instead of weeks. You can also use cloud-based analytics and workflow tools to process your data as you need it, and then store it in your own datacenters or in the cloud. Which AWS Cloud services you use determines how much configuration work you have to perform as part of your security responsibilities. For example, for Amazon Elastic Compute Cloud (Amazon EC2) instances, you're responsible for management of the guest operating system (including updates and security patches), any application software or utilities you install on the instances, and the configuration of the AWS-provided firewall (called a security group) on each instance. These are basically the same security tasks that you're used to performing no matter where your servers are located. AWS managed services like Amazon RDS or Amazon Redshift provide all of the resources you need in order to perform a specific task, but without the configuration work that can come with them. With managed services, you don't have to worry about launching and maintaining instances, patching the guest operating system or database, or replicating databases—AWS handles that for you. But as with all services, you should protect your AWS account credentials, and set up individual user accounts with AWS Identity and Access Management (IAM) so that each of your users has her own credentials, and you can implement segregation of duties. You should consider using Multi-Factor Authentication (MFA) with each account, requiring the use of SSL/TLS to communicate with your AWS resources, and setting up Application Programming Interface (API) and user activity logging with AWS CloudTrail. Figure 3.1 demonstrates the shared responsibility model.

AWS Global Infrastructure Security

AWS operates the global cloud infrastructure that you use to provision a variety of basic computing resources, such as processing and storage. The AWS global infrastructure includes the facilities, network, hardware, and operational software (for example, host operating system, virtualization software) that support the provisioning and use of these resources. The AWS global infrastructure is designed and managed according to security

best practices as well as a variety of security compliance standards. As a systems operator, you can be assured that you're building web architectures on top of some of the most secure computing infrastructure in the world. See Figure 3.2 for a depiction of AWS global infrastructure.

FIGURE 3.1 Shared responsibility model

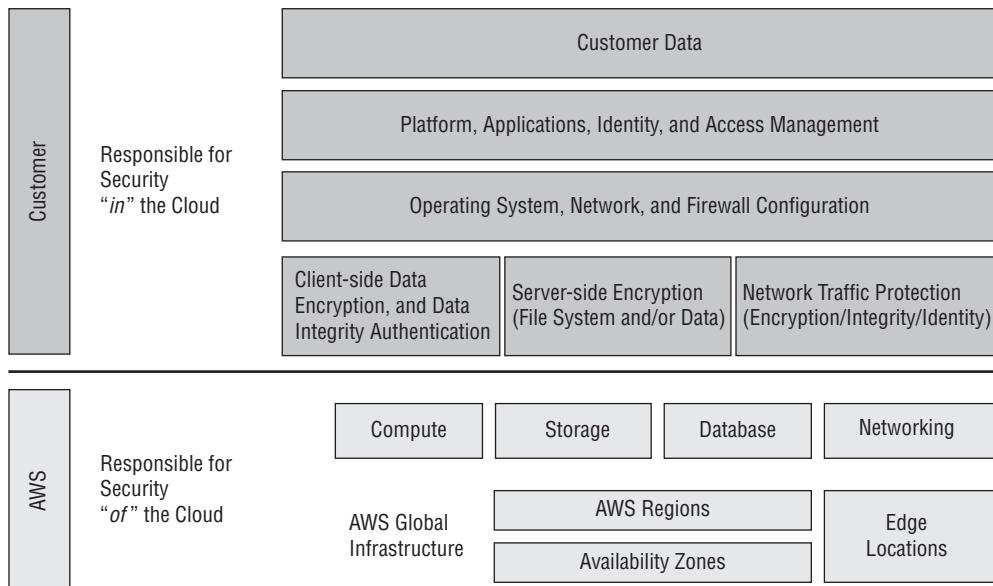


FIGURE 3.2 Amazon Web Services Regions and Availability Zones (as of April 2017)



Physical and Environmental Security

AWS datacenters are state of the art, using innovative architectural and engineering approaches. Amazon has many years of experience in designing, constructing, and operating large-scale datacenters. This experience has been applied to the AWS platform and infrastructure. AWS datacenters are housed in nondescript facilities. Physical access is strictly controlled both at the perimeter and at building ingress points by professional security staff using video surveillance, intrusion detection systems, and other electronic means. Authorized staff must pass two-factor authentication a minimum of two times to access datacenter floors. All visitors and contractors are required to present identification and are signed in and continually escorted by authorized staff.

AWS provides datacenter access and information only to employees and contractors who have a legitimate business need for such privileges. When an employee no longer has a business need for these privileges, his access is immediately revoked, even if he continues to be an employee of Amazon or AWS. All physical access to datacenters by AWS employees is logged and audited routinely.

Fire Detection and Suppression

AWS datacenters have automatic fire detection and suppression equipment to reduce risk. The fire detection system uses smoke detection sensors in all datacenter environments, mechanical and electrical infrastructure spaces, chiller rooms, and generator equipment rooms. These areas are protected by wet-pipe, double-interlocked pre-action or gaseous sprinkler systems.

Power

AWS datacenter electrical power systems are designed to be fully redundant and maintainable without impact to operations. Uninterruptible Power Supply (UPS) units provide backup power in the event of an electrical failure for critical and essential loads in the facility. AWS datacenters use generators to provide backup power for the entire facility.

Climate and Temperature

Climate control is required to maintain a constant operating temperature for servers and other hardware, which prevents overheating and reduces the possibility of service outages. AWS datacenters are built to maintain atmospheric conditions at optimal levels. Personnel and systems monitor and control temperature and humidity at appropriate levels.

Management

AWS monitors electrical, mechanical, and HVAC systems and equipment so that any issues are immediately identified. AWS staff performs preventative maintenance to maintain the continued operability of equipment.

Storage Device Decommissioning

When a storage device has reached the end of its useful life, AWS procedures include a decommissioning process that is designed to prevent customer data from being exposed to unauthorized individuals.

Business Continuity Management

Amazon's infrastructure has a high level of availability and provides customers with the features to deploy a resilient IT architecture. AWS has designed its systems to tolerate system or hardware failures with minimal customer impact. Datacenter business continuity management at AWS is under the direction of the Amazon Infrastructure Group.

Availability

Datacenters are built in clusters in various global regions. All datacenters are online and serving customers; no datacenter is “cold.” In case of failure, automated processes move data traffic away from the affected area. Core applications are deployed in an N+1 configuration so that, in the event of a datacenter failure, there is sufficient capacity to enable traffic to be load-balanced to the remaining sites.

AWS provides its customers with the flexibility to place instances and store data in multiple geographic regions and also across multiple Availability Zones in each region. Each Availability Zone is designed as an independent failure zone. This means that Availability Zones are physically separated in a typical metropolitan region and are located in lower-risk flood plains (specific flood zone categorization varies by region). In addition to having discrete UPS and on-site backup generation facilities, they are each fed via different grids from independent utilities to reduce single points of failure further. Availability Zones are all redundantly connected to multiple tier-1 transit providers.



You should architect your AWS usage to take advantage of multiple regions and Availability Zones. Distributing applications across multiple Availability Zones provides the ability to remain resilient in the face of most failure modes, including natural disasters or system failures.

Incident Response

The Amazon Incident Management Team employs industry-standard diagnostic procedures to drive resolution during business-impacting events. Staff operators provide 24 hours a day, 7 days a week coverage to detect incidents and to manage the impact and resolution.

Communication

AWS has implemented various methods of internal communication at a global level to help employees understand their individual roles and responsibilities and to communicate significant events in a timely manner. These methods include orientation and training programs

for newly hired employees, regular management meetings for updates on business performance and other matters, and electronic means such as video conferencing, electronic mail messages, and the posting of information via the Amazon intranet.

AWS has also implemented various methods of external communication to support its customer base and the community. Mechanisms are in place to allow the Customer Support Team to be notified of operational issues that impact the customer experience. An AWS Service Health Dashboard is available and maintained by the Customer Support Team to alert customers to any issues that may be of broad impact. The AWS Security Center is available to provide customers with security and compliance details about AWS. Customers can also subscribe to AWS Support offerings that include direct communication with the Customer Support Team and proactive alerts to any customer-impacting issues.

Network Security

The AWS network has been architected to permit you to select the level of security and resiliency appropriate for your workload. To enable you to build geographically dispersed, fault-tolerant web architectures with cloud resources, AWS has implemented a world-class network infrastructure that is carefully monitored and managed.

Secure Network Architecture

Network devices, including firewall and other boundary devices, are in place to monitor and control communications at the external boundary of the network and at key internal boundaries in the network. These boundary devices employ rule sets, *Access Control Lists (ACLs)*, and configurations to enforce the flow of information to specific information system services.

ACLs, or traffic flow policies, are established on each managed interface to manage and enforce the flow of traffic. ACL policies are approved by Amazon Information Security. These policies are automatically pushed to ensure that these managed interfaces enforce the most up-to-date ACLs.

Secure Access Points

AWS has strategically placed a limited number of access points to the cloud to allow for a more comprehensive monitoring of inbound and outbound communications and network traffic. These customer access points are called API endpoints, and they permit secure HTTP access (HTTPS), which allows you to establish a secure communication session with your storage or compute instances within AWS. To support customers with Federal Information Processing Standard (FIPS) cryptographic requirements, the SSL-terminating load balancers in AWS GovCloud (US) are FIPS 140-2 compliant.

In addition, AWS has implemented network devices that are dedicated to managing interfacing communications with Internet Service Providers (ISPs). AWS employs a redundant connection to more than one communication service at each Internet-facing edge of the AWS network. These connections each have dedicated network devices.

Transmission Protection

You can connect to an AWS access point via HTTP or HTTPS using SSL, a cryptographic protocol that is designed to protect against eavesdropping, tampering, and message forgery. For customers who require additional layers of network security, AWS offers the Amazon Virtual Private Cloud (Amazon VPC) (as referenced in Chapter 5, “Networking”), which provides a private subnet within the AWS Cloud and the ability to use an IPsec Virtual Private Network (VPN) device to provide an encrypted tunnel between the Amazon VPC and your datacenter.

Network Monitoring and Protection

The AWS network provides significant protection against traditional network security issues, and you can implement further protection. The following are a few examples of the network monitoring and protection services and features that AWS offers.

Distributed Denial of Service (DDoS) attacks AWS API endpoints are hosted on large, Internet-scale, world-class infrastructure that benefits from the same engineering expertise that has built Amazon into the world’s largest online retailer. Proprietary Distributed Denial of Service (DDoS) mitigation techniques are used. Additionally, AWS networks are multi-homed across a number of providers to achieve Internet access diversity.

Man-in-the-Middle (MITM) attacks All of the AWS APIs are available via SSL-protected endpoints that provide server authentication. Amazon EC2 Amazon Machine Images (AMIs) automatically generate new Secure Shell (SSH) host certificates on first boot and log them to the instance’s console. You can then use the secure APIs to call the console and access the host certificates before logging into the instance for the first time. AWS encourages you to use SSL for all of your interactions.

IP spoofing Amazon EC2 instances cannot send spoofed network traffic. The AWS-controlled, host-based firewall infrastructure will not permit an instance to send traffic with a source IP or Machine Access Control (MAC) address other than its own.

Port scanning Unauthorized port scans by Amazon EC2 customers are a violation of the AWS Acceptable Use Policy. Violations of the AWS Acceptable Use Policy are taken seriously, and every reported violation is investigated. Customers can report suspected abuse via the contacts available on the AWS website. When unauthorized port scanning is detected by AWS, it is stopped and blocked. Port scans of Amazon EC2 instances are generally ineffective because, by default, all inbound ports on Amazon EC2 instances are closed and are only opened by the customer. Strict management of security groups can further mitigate the threat of port scans. If you configure the security group to allow traffic from any source to a specific port, then that specific port will be vulnerable to a port scan. In these cases, you must use appropriate security measures to protect listening services that may be essential to their application from being discovered by an unauthorized port scan. For example, a web server must clearly have port 80 (HTTP) open to the world, and the administrator of this server is responsible for the security of the HTTP server software,

such as Apache®. You may request permission to conduct vulnerability scans as required to meet your specific compliance requirements. These scans must be limited to your own instances and must not violate the AWS Acceptable Use Policy. Advance approval for these types of scans can be requested by submitting a request via the AWS website.

Packet sniffing by other tenants Although you can place your interfaces into promiscuous mode, the Hypervisor will not deliver any traffic to them that is not addressed to them. Even two virtual instances that are owned by the same customer located on the same physical host cannot listen to each other's traffic. Although Amazon EC2 does provide ample protection against one customer inadvertently or maliciously attempting to view another customer's data, as a standard practice, you should *encrypt* sensitive traffic.



It is not possible for a virtual instance running in promiscuous mode to receive, or "sniff," traffic that is intended for a different virtual instance.



Attacks such as Address Resolution Protocol (ARP) cache poisoning do not work within Amazon EC2 and Amazon VPC.

AWS Compliance Program

AWS Compliance enables you to understand the robust controls in place at AWS to maintain security and data protection in the cloud. As you build your systems on top of the AWS Cloud infrastructure, compliance responsibilities will be shared. By tying together governance-focused, audit-friendly service features with applicable compliance or audit standards, AWS Compliers build on traditional programs and help customers operate in a secure and controlled environment. The IT infrastructure that AWS provides is designed and managed in alignment with security best practices and a variety of IT security standards, including, but not limited to the following:

- Service Organization Controls (SOC) 1/Statements on Standards for Attestation Engagements (SSAE) 16/International Standard on Assurance Engagements (ISAE) 3402 (formerly Statement on Auditing Standards [SAS] 70), SOC 2, and SOC 3
- Federal Information Security Management Act (FISMA)
- Federal Risk and Authorization Management Program (FedRAMP)
- Department of Defense (DoD) Security Requirements Guide (SRG) Levels 2 and 4
- Payment Card Industry Data Security Standard (PCI DSS) Level 1
- International Organization for Standardization (ISO) 9001, ISO 27001, ISO 27017, and ISO 27018
- International Traffic in Arms Regulations (ITAR)

- FIPS 140-2
- Singapore Multi-Tier Cloud Security Standard (MTCS) Level 3
- Germany Cloud Computing Compliance Controls Catalog (C5)
- United Kingdom Cyber Essentials Plus
- Australia Information Security Registered Assessors Program (IRAP)

In addition, the flexibility and control that the AWS platform provides allow you to deploy solutions that meet several industry-specific standards, including:

- Criminal Justice Information Services (CJIS)
- Cloud Security Alliance (CSA)
- Family Educational Rights and Privacy Act (FERPA)
- Health Insurance Portability and Accountability Act (HIPAA)
- Motion Picture Association of America (MPAA)



AWS provides a wide range of information regarding its IT control environment to customers through whitepapers, reports, certifications, accreditations, and other third-party attestations. Refer to <https://aws.amazon.com/compliance> for more information.



Customers must notify AWS Support prior to starting any vulnerability scanning and penetration testing. Vulnerability scanning and penetration testing are not permitted against the following Amazon EC2 instance types: nano, micro, and small. Refer to <https://aws.amazon.com/security/penetration-testing/> for more information.

Now that we have discussed the shared responsibility model, let's move on to IAM and how to secure your AWS account.

Securing Your AWS Account with AWS Identity and Access Management (IAM)

AWS Identity and Access Management (IAM) provides centralized management of access and authentication of users to the services in an AWS account. The service provides the mechanisms to identify who has access to an AWS account and to control what they can do with the AWS Cloud services in that AWS account. The IAM service is provided at no additional charge.

User, group, and role entities are created in IAM and have policies (JSON policy documents) applied to them to control their access to AWS resources. IAM allows your policies to define how resources can be accessed (for example, launching and terminating Amazon EC2 instances) and what resources can be accessed (such as Amazon S3 buckets and their contents). An IAM best practice is to use the principle of least privilege when setting the policies that control access to AWS resources. IAM allows appropriate credentials to be defined for users.

IAM is also integrated with AWS Marketplace so that you can control who in your organization can subscribe to the software and services offered in AWS Marketplace. Because subscribing to certain software in AWS Marketplace launches an Amazon EC2 instance to run the software, this is an important access control feature. Using IAM to control access to AWS Marketplace also enables AWS account owners to have fine-grained control over usage and software costs.

In this section, we cover IAM users, IAM groups, IAM roles, and IAM policies.

IAM User

An *IAM user* is an entity created in an AWS account that provides a way to interact with the resources in the account. A user can be any individual, system, or application that interacts with AWS resources, either programmatically (using AWS Software Development Kits [SDKs]), through the AWS Management Console, or through the AWS Command Line Interface (AWS CLI). Each user has a unique name within the AWS account and a unique set of security credentials not shared with other users. IAM eliminates the need to share passwords or keys and enables you to minimize the use of your AWS account credentials.

When an AWS account is created, the credentials used at account creation become the root user account ID (for example, the root account user name is the email address specified at account creation). It is important to know that the actions of the root user cannot be restricted. Any entity—a user or application—that has the root user account credentials can undertake any activity in an AWS account. A best practice is not to use the root user account credentials and instead create a separate IAM user with all administrative privileges. Use that IAM user to apply the principle of least privilege by creating additional individual IAM accounts for your various users.

IAM Credentials

There are three ways to use AWS Cloud services and access resources: via the AWS Management Console using a web browser, the AWS CLI, and the AWS SDKs through API calls. These three access mechanisms require credentials.

The AWS Management Console requires either the root user email address and a password, or an IAM user name and a password. A best practice is not to use the root user email address for access to the AWS account. The AWS CLI and AWS SDKs require an access key ID and a secret access key. Create credentials only as needed to maintain the principle of least privilege. See Table 3.1 for descriptions of IAM credentials.

TABLE 3.1 IAM Credentials

| Types of Security Credentials | Description |
|-------------------------------|--|
| Email address and password | Associated with the AWS account root user |
| IAM user name and password | Used to access the AWS Management Console |
| Access keys | Typically used with AWS CLI and programmatic requests like APIs and AWS SDKs |
| Key pairs | Used only for specific AWS Cloud services like Amazon EC2 |
| MFA | Can be enabled for the root user and IAM users as an extra layer of security |

Managing Passwords

Passwords are required to access your AWS account, individual IAM user accounts, AWS Discussion Forums, and the AWS Support Center. You specify the password when you first create the account, and you can change it at any time by going to the Security Credentials page. AWS passwords can be up to 128 characters long and contain special characters, so we encourage you to create a strong password that cannot be easily guessed.

Password Policy Options

The following list describes the options that are available when you configure a password policy for your account:

Require minimum password length. You can specify the minimum number of characters allowed in an IAM user password. You can enter any number from 6 to 128.

Require at least one uppercase letter. You can require that IAM user passwords contain at least one uppercase character from the ISO basic Latin alphabet (A to Z).

Require at least one lowercase letter. You can require that IAM user passwords contain at least one lowercase character from the ISO basic Latin alphabet (a to z).

Require at least one number. You can require that IAM user passwords contain at least one numeric character (0 to 9).

Require at least one non-alphanumeric character. You can require that IAM user passwords contain at least one of the following non-alphanumeric characters:
! @ # \$ % ^ & * () _ + - = [] { } | '.

Allow users to change their own passwords. You can permit all IAM users in your account to use the IAM console to change their own passwords.

Enable password expiration. You can set IAM user passwords to be valid for only the specified number of days. You specify the number of days that passwords remain valid after they are set. You can choose a password expiration period between 1 and 1,095 days, inclusive.

Prevent password reuse. You can prevent IAM users from reusing a specified number of previous passwords. You can set the number of previous passwords from 1 to 24, inclusive.

Password expiration requires administrator reset. You can prevent IAM users from choosing a new password after their current password has expired. If you leave this checkbox cleared and an IAM user allows her password to expire, the user will be required to set a new password before accessing the AWS Management Console.



Alternatively, you can let only some users manage passwords, either for themselves or for others. To do so, you clear the Allow Users to Change Their Own Password checkbox. When you allow your IAM users to change their own passwords, IAM automatically allows them to view the password policy. IAM users need permission to view the account's password policy in order to create a password that complies with the policy.



The AWS Management Console warns IAM users when they are within 15 days of password expiration. IAM users can change their passwords at any time (as long as they have been given permission to do so). When they set a new password, the rotation period for that password starts over. An IAM user can have only one valid password at a time.



Before you enable this option, be sure that your AWS account has more than one user with administrative permissions (that is, permission to reset IAM user passwords), or that your administrators also have access keys that enable them to use the AWS CLI or AWS Tools for Windows PowerShell separately from the AWS Management Console. When this option is enabled and one administrator's password expires, a second administrator is required to sign in to the AWS Management Console to reset the expired password of the first administrator. If the administrator with the expired password has valid access keys, then she can also run an AWS CLI or AWS Tools for Windows PowerShell command to reset her own password. The requirement for a second administrator applies only if a password expires and the first administrator has no access keys.



IAM can be used to look up the last time a user signed into his AWS account. This mechanism can be used to verify if the root account was used to sign in to the account. It is also recommended that AWS CloudTrail be used to monitor logins, as discussed later in this chapter.

Managing IAM Access Keys

AWS requires that all API requests be signed; that is, they must include a digital signature that AWS can use to verify the identity of the requestor. You can calculate the digital signature using a cryptographic hash function. The input to the hash function in this case includes the text of your request and your secret access key. If you use any of the AWS SDKs to generate requests, the digital signature calculation is done for you; otherwise, you can have your application calculate it and include it in your REST or Query requests by following the directions in the AWS documentation.

Not only does the signing process help protect message integrity by preventing tampering with the request while it is in transit, it also helps protect against potential replay attacks. A request must reach AWS within 15 minutes of the timestamp in the request; otherwise, AWS denies the request.

The most recent version of the digital signature calculation process is Signature Version 4, which calculates the signature using the Keyed-Hash Message Authentication Code (HMAC)-Secure Hash Algorithm (SHA) 256 protocol. Version 4 provides an additional measure of protection over previous versions by requiring that you sign the message using a key that is derived from your secret access key instead of using the secret access key itself. In addition, you derive the signing key based on credential scope, which facilitates cryptographic isolation of the signing key.

Because access keys can be misused if they fall into the wrong hands, we encourage you to save them in a safe place and not embed them in your code. For customers with large fleets of elastically scaling Amazon EC2 instances, the use of IAM roles can be a more secure and convenient way to manage the distribution of access keys. IAM roles provide temporary credentials, which are not only automatically loaded to the target instance, but are also automatically rotated multiple times a day.

Access keys comprise two components:

- Access key ID
- Secret access key

The access key is active by default. Each user can have two active access keys. This enables access keys to be rotated without the user temporarily losing access to his AWS account. Users can be given permissions to list, rotate, and manage their own keys.

Access keys can be disabled or deleted to revoke access. Disabling an access key makes it inactive. A deleted access key is removed forever and cannot be retrieved.

A best practice is to rotate access keys regularly for all of the IAM users in an AWS account. Unnecessary credentials should be removed from users who do not need them. IAM can be used to obtain an access key history, which includes the time when the key was last used along with the region and service that was accessed. This information helps when rotating old keys and removing active keys in an AWS account.

The IAM access key can be placed in:

Linux: `~/.aws/credentials` file

Windows: `\%USERPROFILE%\awscredentials`

The root user access keys should not be used. The recommended practice is to delete the root user access keys.

Multi-Factor Authentication (MFA)

MFA adds an additional layer of security when accessing AWS Cloud services. When you enable this optional feature, you will need to provide a six-digit, single-use code in addition to your standard user name and password credentials before access is granted to your AWS account settings or AWS Cloud services and resources (for example, providing something you have [MFA device] and something you know [a password]). You can enable MFA devices for your AWS account and for the users you have created under your AWS account using IAM. In addition, you can add MFA protection for access across AWS accounts for when you want to allow a user you've created under one AWS account to use an IAM role to access resources under another AWS account. You can require the user to use MFA before assuming the role as an additional layer of security.

The IAM service supports the following MFA types:

- Hardware devices (Gemalto)
- Virtual MFA applications (such as Google Authenticator)
- Simple Message Service (SMS) (via mobile devices) can be used for MFA with an AWS account.

Virtual MFA applications are software applications, typically installed on smart phones, that generate authentication codes. These authentication codes need to be compatible with the Time-based One Time Password (TOTP) standard in order to be used with AWS accounts.

SMS MFA uses SMS text messaging to verify an IAM user. When the user signs in to the AWS Management Console, she will receive an authentication code via text message that she will enter into her browser. Users do not need to use a hardware token or a virtual MFA application if they use SMS MFA.

IAM Groups

An *IAM group* is a collection of IAM users. There is no default user group in an AWS account. You can create IAM groups as appropriate for the account. IAM groups cannot be nested. An IAM user can be a member of multiple groups. Permissions can be specified for the entire group. Creating IAM groups and assigning permissions to them is the recommended approach. This practice eases the administrative burden when compared to specifying permissions for individual IAM users. Permissions are defined using IAM policy documents.

IAM Policies

IAM Policies are JSON-formatted permission documents that are used to grant or deny access to IAM users. By default, an IAM user cannot do anything with any service in an AWS account, even if he has authentication credentials. Permission to take actions with a service must be explicitly provided to individual users, groups, or roles.

A *least privilege* approach should be used when assigning permissions so that there are just enough permissions to perform a job or set of tasks. Only grant the permissions necessary to perform a task. Least privilege reduces the chances of mistakes being made when actions are performed. It is easier to loosen permissions than tighten them when implementing security. IAM permissions can provide a granular level of control.

An IAM policy consists of four main elements:

- Action
- Effect
- Resource
- Conditions (optional)

IAM policy rules have an order of precedence. If an action is explicitly denied, it is denied; then if an action is explicitly allowed, it is allowed. Otherwise, all other actions are implicitly denied.

An IAM policy can also use policy conditions for extra security. For example, conditions could be set so that requests must originate from a specific range of IP addresses, or a request must use SSL or MFA.

There are two types of IAM policies:

Managed policies *Managed policies* can be customer- and AWS-managed. These policies are standalone policies that are attached to multiple users, groups, and roles. AWS-managed policies are created by AWS. Customer-managed policies are created and managed by customers. Managed policies provide several features, including reusability, central change management, versioning and rollback, and the ability to delegate permissions management to other users. Examples of the managed policies provided by AWS include administrator access, power user access, read-only access, and IAM read-only access.

Inline policies *Inline policies* are embedded in a principal entity such as a user, group, or role. The policy is an inherent part of the entity. The same policy can be used for multiple entities, but those entities do not share the policy. Instead, each entity has its own copy of the policy (that is, inline policies cannot be centrally managed).



The IAM console provides “service last accessed data” about when IAM users and roles last attempted to access AWS Cloud services. This information can help identify unnecessary permissions so that IAM policies can be refined.

IAM Roles

IAM Roles are temporary user credentials. IAM user credentials are regarded as static credentials and static AWS credentials should not be embedded in software source code or placed on Amazon EC2 instances—as there is a chance they could be compromised and used for unauthorized access to resources in an AWS account.

The principle of least privilege implies that an entity should not have excessive sets of permissions. If an IAM user undertakes a task sporadically, there is the potential for them to have a permission set that is too powerful for their daily activities.

IAM roles are used to delegate access to AWS resources. An IAM role provides temporary access and eliminates the need to use static AWS credentials. Think of IAM roles as temporary users. The IAM roles do not have a user name or password associated with them; instead, they have temporary credentials consisting of the following:

- Access key ID
- Secret access key
- Token
- Duration

IAM roles enable temporary credentials to be created and issued when needed for the following:

- Applications written with a language-specific AWS SDK
- Amazon EC2 instance access to an AWS resource
- IAM user to gain temporary credentials to undertake a more powerful action (for example, Amazon EC2 instance termination) only when needed
- Federated user accounts

Just as permission documents are used to grant or deny access to IAM users, they are used to grant/deny access to IAM roles. In the case of the IAM role, the policy is attached to the role (not to the IAM user or IAM group member assuming the IAM role). By default, AWS resources cannot interact with AWS Cloud services; the resources need to have the necessary permissions. IAM roles can be used to provide AWS resources with access to AWS Cloud services.

IAM roles can be used to provide access to resources and services within an AWS account to externally authenticated users and third parties. External users authenticate against an external identity store, and federation allows them to get temporary credentials when they assume a role.

An IAM user can switch to an IAM role to gain access to resources in their current AWS account that are not accessible with their normal permissions. An IAM user who requires cross-account access can use an IAM role to gain access to resources in another AWS account.

Best Practices for Securing Your AWS Account

AWS recommends several best practices for securing your AWS account:

1. Require MFA for root-level access.
2. Do not share root credentials with anyone other than the account holder.
3. Physically secure root account hardware MFA devices in a safe place, such as a vault.
4. Create individual IAM users.

5. Use groups to assign permissions to IAM users.
6. Enable MFA for privileged users.
7. Use IAM roles for applications that run on Amazon EC2 instances.
8. Delegate by using IAM roles instead of sharing credentials.
9. Rotate credentials regularly.
10. Remove unnecessary credentials.
11. Use policy conditions for extra security.
12. Monitor activity on your AWS account.
13. Remove root credentials.
14. Use access levels to review IAM permissions.
15. Use AWS-defined policies to assign permissions whenever possible.
16. Use IAM roles to provide cross-account access.

So far, you have learned about the shared responsibility model and how to secure your IAM account. Now let's talk about securing your AWS resources.

Securing Your AWS Cloud Services

In this section, we discuss Amazon EC2 key pairs, X.509 certificates, AWS Key Management Services (AWS KMS), and AWS CloudHSM.

Key Pairs

Amazon EC2 instances created from a public AMI use a public/private key pair instead of a password for signing in via SSH. The public key is embedded in your instance, and you use the private key to sign in securely without a password. After you create your own AMIs, you can choose other mechanisms to log in securely to your new instances. You can have a key pair generated automatically for you when you launch the instance, or you can upload your own. Save the private key in a safe place on your system, and record the location where you saved it. For Amazon CloudFront, you use key pairs to create signed URLs for private content, such as when you want to distribute restricted content that someone paid for.



Amazon CloudFront key pairs can be created only by the root account and cannot be created by IAM users.

X.509 Certificates

X.509 certificates are used to sign SOAP-based requests. X.509 certificates contain a public key and additional metadata (for example, an expiration date that AWS verifies when

you upload the certificate) and is associated with a private key. When you create a request, you create a digital signature with your private key and then include that signature in the request, along with your certificate. AWS verifies that you are the sender by decrypting the signature with the public key that is in your certificate. AWS also verifies that the certificate you sent matches the certificate that you uploaded to AWS.

AWS Key Management Service (AWS KMS) Security

AWS KMS provides a simple interface that can be used to generate and manage cryptographic keys and operate as a cryptographic service provider for protecting data. AWS KMS offers traditional key management services integrated with AWS Cloud services to provide a consistent view of customers' keys across AWS, with centralized management and auditing.

AWS KMS provides a simple web interface in the AWS Management Console, AWS CLI, and RESTful APIs to access an elastic, multi-tenant, Hardened Security Appliance (HSA).

You can establish your own HSA-based cryptographic contexts under your master keys. These keys are accessible only on the HSAs, and they can be used to perform HSA-resident cryptographic operations, including the issuance of application data keys (encrypted under your master key). You can create multiple master keys, each represented with an HS-based Customer Master Key (CMK) identified by its key ID. You can use the AWS KMS console to define access controls on who can manage and/or use master keys by creating a policy that is attached to the key. This allows you to define application-specific uses for your keys on a per-API basis.

All requests to AWS KMS must be made over the TLS protocol and terminate on an AWS KMS host. AWS KMS hosts will only allow TLS with a cipher suite that provides perfect forward secrecy. AWS KMS authenticates and authorizes customer requests using the same credential and policy mechanisms available for all other AWS APIs, including IAM. AWS KMS is designed to meet the following requirements:

Durability The durability of cryptographic keys is designed to equal that of the highest-durability services in AWS. A single cryptographic key can encrypt large volumes of customer data accumulated over a long time period. Data encrypted under a key becomes irretrievable if the key is lost.

Quorum-based access No single Amazon employee can gain access to CMKs. There is no mechanism to export plaintext CMKs. Confidentiality of your cryptographic keys is crucial.

Access control Use of keys is protected by access control policies defined and managed by you.

Low-latency and high throughput AWS KMS will provide cryptographic operations at latency and throughput levels suitable for use by other services in AWS.

Regional independence AWS provides regional independence for customer data. Key usage is isolated in an AWS Region.

Secure source of random numbers Because strong cryptography depends on truly unpredictable random number generation, AWS provides a high-quality source of random numbers.

Audit AWS records the use of cryptographic keys in AWS CloudTrail Logs. Customers can use AWS CloudTrail Logs to inspect use of their cryptographic keys, including use of keys by AWS Cloud services on the customer's behalf.

The AWS KMS system includes a set of AWS KMS operators and service host operators (collectively, “operators”) that administer “domains.” A *domain* is a regionally defined set of AWS KMS servers, HSAs, and operators. Each entity has a hardware token that contains a private and public key pair used to authenticate its actions. The HSAs have an additional private and public key pair used to establish encryption keys to protect HSA-to-HSA communications.

AWS CloudHSM Security

The AWS *CloudHSM* service provides dedicated access to a Hardware Security Module (HSM) appliance designed to provide secure cryptographic key storage and operations in an intrusion-resistant, tamper-evident device. You can generate, store, and manage the cryptographic keys used for data encryption so that they are accessible only by you.

AWS CloudHSM appliances are designed to store and process cryptographic key material securely for a wide variety of uses such as database encryption, Digital Rights Management (DRM), Public Key Infrastructure (PKI), authentication and authorization, document signing, and transaction processing. They support some of the strongest cryptographic algorithms available, including Advanced Encryption Standard (AES), RSA, Elliptic Curve Cryptography (ECC), and many others. The AWS CloudHSM service is designed to be used with Amazon EC2 and Amazon VPC, which provide the appliance with its own private IP within a private subnet. You can connect to AWS CloudHSM appliances from your Amazon EC2 servers through SSL/TLS, which uses two-way digital certificate authentication and 256-bit SSL encryption to provide a secure communication channel.

Selecting AWS CloudHSM in the same region as your Amazon EC2 instance decreases network latency, which can improve your application performance. You can configure a client on your Amazon EC2 instance that allows your applications to use the APIs provided by the HSM, including Public-Key Cryptography Standards (PKCS) #11, Microsoft Cryptographic Application Programming Interface (CAPI), and Java Cryptography Architecture/Java Cryptography Extensions (Java JCA/JCE).

AWS has administrative credentials to the appliance, but these credentials can only be used to manage the appliance, not the HSM partitions on the appliance. AWS uses these credentials to monitor and maintain the health and availability of the appliance.

AWS cannot extract your keys, nor can AWS cause the appliance to perform any cryptographic operation using your keys. The HSM appliance has both physical and logical tamper detection and response mechanisms that erase the cryptographic key material and generate event logs if tampering is detected. The HSM is designed to detect tampering if the physical barrier of the HSM appliance is breached. In addition, after three unsuccessful attempts to access an HSM partition with HSM admin credentials, the HSM appliance erases its HSM partitions.

When your AWS CloudHSM subscription ends and you have confirmed that the contents of the HSM are no longer needed, you must delete each partition and its contents as well as any logs. As part of the decommissioning process, AWS zeroizes the appliance, permanently erasing all key material.

Monitoring to Enhance Security

In this section, you are introduced to the various monitoring tools offered by AWS. This section serves as an overview of the tools. More details can be found in Chapter 9, “Monitoring and Metrics.”

AWS CloudTrail

As important as credentials and encrypted endpoints are for preventing security problems, logs are just as crucial for understanding events after a problem has occurred. To be effective as a security tool, a log must include not just a list of what happened and when, but also identify the source. To help you with your after-the-fact investigations and near-real-time intrusion detection, *AWS CloudTrail* provides a log of all requests for AWS resources in your account. For each event, you can see what service was accessed, what action was performed, and who made the request. AWS CloudTrail captures information about every API call to every AWS resource you use, including sign-in events. After you have enabled AWS CloudTrail, event logs are delivered every five minutes. You can configure AWS CloudTrail so that it aggregates log files from multiple regions into a single Amazon S3 bucket. From there, you can then upload them to your preferred log management and analysis solutions to perform security analysis and detect user behavior patterns. By default, log files are stored securely in Amazon S3, but you can also archive them to Amazon Glacier to help meet audit and compliance requirements. More information on Amazon Glacier and Amazon S3 is provided in Chapter 6, “Storage.”

In addition to AWS CloudTrail’s user activity logs, you can use the Amazon CloudWatch Logs feature to collect and monitor system, application, and custom log files from your Amazon EC2 instances and other sources in near real time. For example, you can monitor your web server’s log files for invalid user messages to detect unauthorized login attempts to your guest operating system. More information on Amazon CloudWatch and Amazon CloudWatch Logs can be found in Chapter 9.

Amazon Virtual Private Cloud (Amazon VPC) Flow Logs

AWS CloudTrail captures the API calls made as users interact with the AWS Cloud services and resources in an AWS account. You still need to have a complete view of what is happening within your Amazon VPC (for example, network traffic flowing into and out of your Amazon VPC).

Amazon VPC Flow Logs is a feature that enables you to capture information about the IP traffic going to and from network interfaces in your Amazon VPC. Flow log data is

stored using Amazon CloudWatch Logs. After you have created a flow log, you can view and retrieve its data in Amazon CloudWatch Logs.

In addition, Elastic Load Balancing and elastic network interfaces provide access logs that capture detailed information about requests or connections sent to your load balancer. Each log contains information such as the time it was received, the client's IP address, latencies, request paths, and server responses. Detailed information on Amazon VPC Flow Logs can be found in Chapter 5.

Amazon CloudWatch

Amazon CloudWatch provides a means of monitoring the use of AWS resources. There are standard metrics provided by AWS for a variety of AWS resources. Users can also create their own custom metrics using agents that they have installed to feed data to Amazon CloudWatch for monitoring.

Amazon CloudWatch Logs enable the operating system and other logs to be monitored by Amazon CloudWatch. Additionally, AWS CloudTrail logs can be monitored in real time using Amazon CloudWatch Logs. Pattern filtering is used to analyze the logs. The log events are evaluated for matches against terms, phrases, and values specified in the pattern filter. Amazon CloudWatch Alarms can be created to report out-of-bound conditions discovered in log files. Amazon CloudWatch Alarms are triggered based on thresholds that you specified in the alarm. The alarm can be configured to send notifications and perform an action.

AWS Config

AWS Config records configuration changes to an AWS account. AWS Config can be used to retrieve an inventory of AWS resources in an AWS account at a particular time. AWS Config can be used to identify new and deleted resources. AWS Config can issue notifications when resource configurations change. AWS Config use cases include the following:

- Resource discovery
- Troubleshooting
- Change management
- Audit compliance
- Security analysis

AWS Config Rules allow rules to be set up to check configuration changes recorded by AWS Config. There are prebuilt rules provided by AWS. You can also author your own rules, which will run on AWS Lambda. The rules are invoked automatically when triggered to provide continuous assessment. A dashboard can be used to visualize compliance and identify any offending changes.

AWS Config can trigger a rules evaluation when any resource that matches the rule's scope changes in configuration (as when a security group has its rules modified). A rule can also be triggered to evaluate your account's configuration periodically. For example, the AWS Config service runs evaluations for the rule at a chosen frequency (such as every 24 hours). For more information on AWS Config, refer to Chapter 9.

Amazon Inspector

Amazon Inspector is an automated security assessment service that helps improve the security and compliance of applications deployed on AWS. Amazon Inspector automatically assesses applications for vulnerabilities or deviations from best practices. After performing an assessment, Amazon Inspector produces a detailed list of security findings prioritized by level of severity.

Amazon Inspector includes a knowledge base of hundreds of rules mapped to common security best practices and vulnerability definitions. Examples of built-in rules include checking for remote root login being enabled or vulnerable software versions being installed. These rules are regularly updated by AWS security researchers. More information about AWS Inspector can be found in Chapters 4 and 9.

AWS Certificate Manager

AWS Certificate Manager is a service that lets you provision, manage, and deploy SSL/TLS certificates for use with AWS Cloud services. SSL/TLS certificates are used to secure network communications and establish the identity of websites over the Internet. With AWS Certificate Manager, you can request a certificate and deploy it on AWS resources, such as Elastic Load Balancing load balancers or Amazon CloudFront distributions. AWS Certificate Manager then handles the certificate renewals.

AWS Web Application Firewall (AWS WAF)

AWS Web Application Firewall (AWS WAF) is a web application firewall that helps protect web applications from attacks by allowing you to configure rules that allow, block, or monitor (count) web requests based on conditions that you define. These conditions include IP addresses, HTTP headers, HTTP body, Uniform Resource Identifier (URI) strings, SQL injection, and cross-site scripting.

As the underlying service receives requests for your websites, it forwards those requests to AWS WAF for inspection against your rules. Once a request meets a condition defined in your rules, AWS WAF instructs the underlying service either to block or allow the request based on the action you define.

AWS WAF is tightly integrated with Amazon CloudFront and the Application Load Balancer, services that AWS customers commonly use to deliver content for their websites and applications. When you use AWS WAF on Amazon CloudFront, your rules run in all AWS edge locations around the world close to your end users. This means that security doesn't come at the expense of performance. Blocked requests are stopped before they reach your web servers. When you use AWS WAF on Application Load Balancer, your rules run in-region and can be used to protect Internet-facing and internal load balancers.

AWS Trusted Advisor

The *AWS Trusted Advisor* customer support service not only monitors cloud performance and resiliency, but also cloud security. AWS Trusted Advisor inspects your AWS environment and makes recommendations when opportunities may exist to save money, improve

system performance, or close security gaps. It provides alerts on several of the most common security misconfigurations that can occur, including leaving certain ports open that make you vulnerable to hacking and unauthorized access, neglecting to create IAM accounts for your internal users, allowing public access to Amazon S3 buckets, not turning on user activity logging (AWS CloudTrail), or not using MFA on your root AWS account. More information on AWS Trusted Advisor checks is located in Chapter 9.

AWS Cloud Service-Specific Security

Not only is security built into every layer of the AWS infrastructure, but it's also built into each of the services available on that infrastructure. AWS Cloud services are architected to work efficiently and securely with all AWS networks and platforms. Each service provides additional security features to enable you to protect sensitive data and applications.

Compute Services

AWS provides a variety of cloud-based computing services that include a wide selection of compute instances that can scale up and down automatically to meet the needs of your application or enterprise.

Amazon Elastic Compute Cloud (Amazon EC2) Security

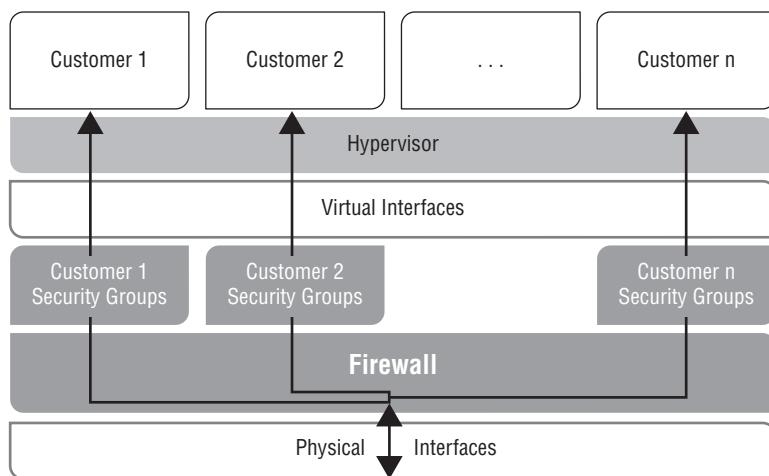
Amazon EC2 is a key component in Amazon's Infrastructure as a Service (IaaS), providing resizable computing capacity using server instances in AWS datacenters. Amazon EC2 is designed to make web-scale computing easier by enabling you to obtain and configure capacity with minimal friction. You create and launch instances, which are collections of platform hardware and software.

Multiple levels of security Security within Amazon EC2 is provided on multiple levels: the operating system of the host platform, the virtual instance operating system or guest operating system, a firewall, and signed API calls. Each of these items builds on the capabilities of the others. The goals are to prevent data contained in Amazon EC2 from being intercepted by unauthorized systems or users and to make Amazon EC2 instances themselves as secure as possible without sacrificing the flexibility in configuration that customers demand.

The Hypervisor Amazon EC2 currently uses a highly-customized version of the Xen Hypervisor, taking advantage of paravirtualization (in the case of Linux guests). Because paravirtualized guests rely on the Hypervisor to provide support for operations that normally require privileged access, the guest operating system has no elevated access to the CPU. The CPU provides four separate privilege modes, called rings: 0–3. Ring 0 is the most privileged and 3 the least. The host operating system executes in Ring 0. Rather than executing in Ring 0 as most operating systems do, the guest operating system runs in lesser-privileged Ring 1, and applications in the least-privileged Ring 3. This explicit virtualization of the physical resources leads to a clear separation between guest and Hypervisor, resulting in additional security separation between the two.

Instance isolation Different instances running on the same physical machine are isolated from each other via the Xen Hypervisor. Amazon is active in the Xen community, which provides AWS with awareness of the latest developments. In addition, the AWS firewall resides within the Hypervisor layer, between the physical network interface and the instance's virtual interface. All packets must pass through this layer; thus an instance's neighbors have no more access to that instance than any other host on the Internet and can be treated as if they are on separate physical hosts. The physical RAM is separated using similar mechanisms. Customer instances have no access to raw disk devices, but instead are presented with virtualized disks. The AWS proprietary disk virtualization layer automatically resets every block of storage used by the customer so that one customer's data is never unintentionally exposed to another customer. In addition, memory allocated to guests is scrubbed (set to zero) by the Hypervisor when it is unallocated to a guest. The memory is not returned to the pool of free memory available for new allocations until the memory scrubbing is complete. Figure 3.3 depicts instance isolation within Amazon EC2.

FIGURE 3.3 Amazon EC2 multiple layers of security



Host operating system Administrators with a business need to access the management plane are required to use MFA to gain access to purpose-built administration hosts. These administrative hosts are systems that are specifically designed, built, configured, and hardened to protect the management plane of the cloud. All such access is logged and audited. When an employee no longer has a business need to access the management plane, the privileges and access to these hosts and relevant systems can be revoked.

Guest operating system Virtual instances are completely controlled by you, the customer. You have full root access or administrative control over accounts, services, and applications. AWS does not have any access rights to your instances or the guest operating system. AWS recommends a base set of security best practices to include disabling password-only access to your guests and using some form of MFA to gain access to your instances (or at a minimum, certificate-based SSH Version 2 access). Additionally, you should employ a

privilege escalation mechanism with logging on a per-user basis. For example, if the guest operating system is Linux, after hardening your instance, you should use certificate-based SSH Version 2 to access the virtual instance, disable remote root login, use command-line logging, and use sudo for privilege escalation. You should generate your own key pairs in order to guarantee that they are unique and not shared with other customers or with AWS. AWS also supports the use of the SSH network protocol to enable you to log in securely to your UNIX/Linux Amazon EC2 instances. Authentication for SSH used with AWS is via a public/private key pair to reduce the risk of unauthorized access to your instance. You can also connect remotely to your Windows instances using Remote Desktop Protocol (RDP) by using an RDP certificate generated for your instance. You also control the updating and patching of your guest operating system, including security updates. Amazon-provided Windows- and Linux-based AMIs are updated regularly with the latest patches, so if you do not need to preserve data or customizations on your running AMI instances, you can simply relaunch new instances with the latest updated AMI. In addition, updates are provided for the Amazon Linux AMI via the Amazon Linux yum repositories.

Firewall Amazon EC2 provides a mandatory inbound firewall that is configured in a default deny-all mode; Amazon EC2 customers must explicitly open the ports needed to allow inbound traffic. The traffic may be restricted by protocol, by service port, and by source IP address (individual IP or Classless Inter-Domain Routing [CIDR] block).

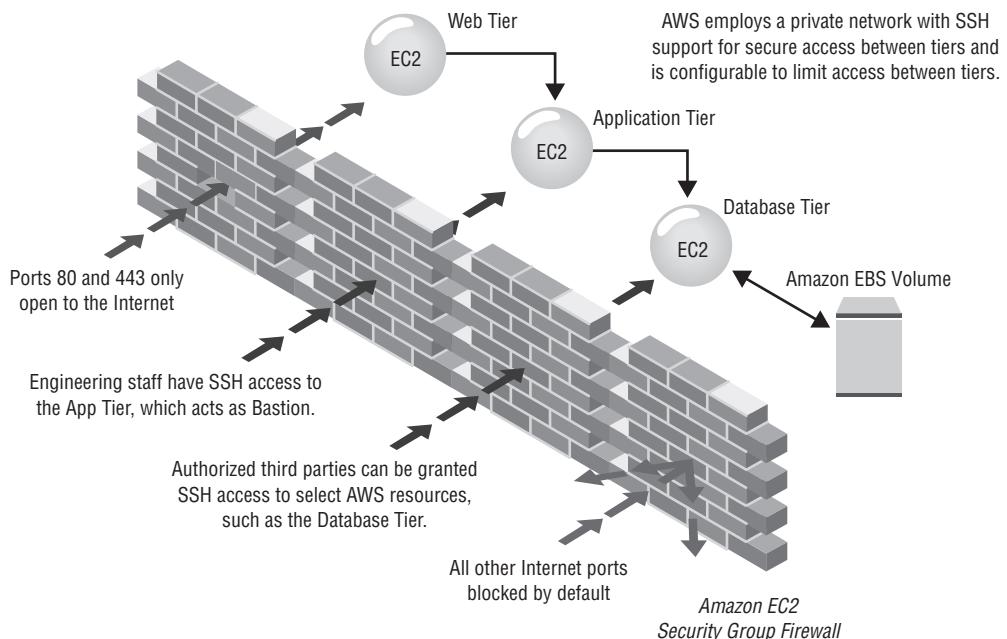
The firewall can be configured in groups, permitting different classes of instances to have different rules. Consider, for example, the case of a traditional three-tiered web application. The group for the web servers would have port 80 (HTTP) and/or port 443 (HTTPS) open to the Internet. The group for the application servers would have port 8000 (application-specific) accessible only to the web server group. The group for the database servers would have port 3306 (MySQL) open only to the application server group. All three groups would permit administrative access on port 22 (SSH), but only from the customer's corporate network. Highly secure applications can be deployed using this approach, which is depicted in Figure 3.4.

The level of security afforded by the firewall is a function of which ports you open and for what duration and purpose. Well-informed traffic management and security design are still required on a per-instance basis. AWS further encourages you to apply additional per-instance filters with host-based firewalls such as IP tables or the Windows firewall and VPNs. This can restrict both inbound and outbound traffic.



The default state is to deny all incoming traffic, and you should carefully plan what you will open when building and securing your applications.

API access API calls to launch and terminate instances, change firewall parameters, and perform other functions are all signed by your Amazon secret access key, which could be either the AWS account's secret access key or the secret access key of a user created with IAM. Without access to your secret access key, Amazon EC2 API calls cannot be made on your behalf. API calls can also be encrypted with SSL to maintain confidentiality. AWS recommends always using SSL-protected API endpoints.

FIGURE 3.4 Amazon EC2 security group firewall

Amazon Elastic Block Store (Amazon EBS) security *Amazon Elastic Block Store (Amazon EBS)* allows you to create storage volumes from 1 GB to 16 TB that can be mounted as devices by Amazon EC2 instances. Storage volumes behave like raw, unformatted block devices with user-supplied device names and a block device interface. You can create a file system on top of Amazon EBS volumes or use them in any other way you would use a block device (like a hard drive). Amazon EBS volume access is restricted to the AWS account that created the volume and to the users under the AWS account created with IAM (if the user has been granted access to the Amazon EBS operations). All other AWS accounts and users are denied the permission to view or access the volume.

Data stored in Amazon EBS volumes is redundantly stored in multiple physical locations as part of normal operation of those services, at no additional charge. However, Amazon EBS replication is stored within the same Availability Zone, not across multiple zones; therefore, it is highly recommended that you conduct regular snapshots and save them to Amazon S3 for long-term data durability. For customers who have architected complex transactional databases using Amazon EBS, it is recommended that backups to Amazon S3 be performed through the database management system so that distributed transactions and logs can be check pointed. AWS does not automatically perform backups of data that are maintained on virtual disks attached to running instances on Amazon EC2.

You can make Amazon EBS volume snapshots publicly available to other AWS accounts to use as the basis for creating duplicate volumes. Sharing Amazon EBS volume snapshots does not provide other AWS accounts with the permission to alter or delete the original snapshot,

as that right is explicitly reserved for the AWS account that created the volume. An Amazon EBS snapshot is a block-level view of an entire Amazon EBS volume. Note that data that is not visible through the file system on the volume, such as files that have been deleted, may be present in the Amazon EBS snapshot. If you want to create shared snapshots, you should do so carefully. If a volume has held sensitive data or has had files deleted from it, you should create a new Amazon EBS volume to share. The data to be contained in the shared snapshot should be copied to the new volume and the snapshot created from the new volume.

Amazon EBS volumes are presented to you as raw unformatted *block devices*, which have been wiped prior to being made available for use. Wiping occurs immediately before reuse so that you can be assured that the wipe process completed. If you have procedures requiring that all data be wiped via a specific method, you have the ability to do so on Amazon EBS. You should conduct a specialized wipe procedure prior to deleting the volume for compliance with your established requirements.

Encryption of sensitive data is generally a good security practice, and AWS provides the ability to encrypt Amazon EBS volumes and their snapshots with AES-256. The encryption occurs on the servers that host the Amazon EC2 instances, providing encryption of data as it moves between Amazon EC2 instances and Amazon EBS storage. In order to be able to do this efficiently and with low latency, the Amazon EBS encryption feature is only available on Amazon EC2's more powerful instance types.

Networking

AWS provides a range of networking services that enable you to create a logically isolated network that you define, establish a private network connection to the AWS Cloud, use a highly available and scalable Domain Name System (DNS) service, and deliver content to your end users with low latency at high data transfer speeds with a content delivery web service.

Elastic Load Balancing Security

Elastic Load Balancing is used to manage traffic on a fleet of Amazon EC2 instances, distributing traffic to instances across all Availability Zones within a region. Elastic Load Balancing has all of the advantages of an on-premises load balancer, plus several security benefits:

- Takes over the encryption and decryption work from the Amazon EC2 instances and manages it centrally on the load balancer
- Offers clients a single point of contact and can also serve as the first line of defense against attacks on the customer's network
- When used in an Amazon VPC, supports creation and management of security groups associated with Elastic Load Balancing to provide additional networking and security options
- Supports end-to-end traffic encryption using TLS (previously SSL) on those networks that use HTTPS connections. When TLS is used, the TLS server certificate used to terminate client connections can be managed centrally on the load balancer, instead of on every individual instance.

HTTPS/TLS uses a long-term secret key to generate a short-term session key to be used between the server and the browser to create the encrypted message. Elastic Load Balancing configures your load balancer with a predefined cipher set that is used for TLS negotiation when a connection is established between a client and your load balancer. The predefined cipher set provides compatibility with a broad range of clients and uses strong cryptographic algorithms. However, some customers may have requirements for allowing only specific ciphers and protocols (for example, PCI DSS, Sarbanes-Oxley Act [SOX]) from clients to ensure that standards are met. In these cases, Elastic Load Balancing provides options for selecting different configurations for TLS protocols and ciphers. You can choose to enable or disable the ciphers depending on your specific requirements.

To help ensure the use of newer and stronger cipher suites when establishing a secure connection, you can configure the load balancer to have the final say in the cipher suite selection during the client-server negotiation. When the server order preference option is selected, the load balancer will select a cipher suite based on the server's prioritization of cipher suites rather than the client's. This gives you more control over the level of security that clients use to connect to your load balancer.

For even greater communication privacy, Elastic Load Balancing allows the use of perfect forward secrecy, which uses session keys that are ephemeral and not stored anywhere. This prevents the decoding of captured data, even if the secret long-term key itself is compromised.

Elastic Load Balancing allows you to identify the originating IP address of a client connecting to your servers, whether you're using HTTPS or TCP load balancing. Typically, client connection information, such as IP address and port, is lost when requests are proxied through a load balancer. This is because the load balancer sends requests to the server on behalf of the client, making your load balancer appear as though it is the requesting client. Having the originating client IP address is useful if you need more information about visitors to your applications in order to gather connection statistics, analyze traffic logs, or manage whitelists of IP addresses.

Elastic Load Balancing access logs contain information about each HTTP and TCP request processed by your load balancer. This includes the IP address and port of the requesting client, the back-end IP address of the instance that processed the request, the size of the request and response, and the actual request line from the client (for example, GET `http://www.example.com: 80/HTTP/1.1`). All requests sent to the load balancer are logged, including requests that never make it to back-end instances (more on Elastic Load Blancing can be found in Chapter 5).

Amazon Virtual Private Cloud (Amazon VPC) Security

Normally, each Amazon EC2 instance you launch is randomly assigned a public IP address in the Amazon EC2 address space. *Amazon VPC* enables you to create an isolated portion of the AWS Cloud and launch Amazon EC2 instances that have private (RFC 1918) addresses in the range of your choice (for example, 10.0.0.0/16). You can define subnets in your Amazon VPC, grouping similar kinds of instances based on IP address range, and then set up routing and security to control the flow of traffic in and out of the instances and subnets.

Security features in Amazon VPC include security groups, *network ACLs*, routing tables, and external gateways. Each of these items is complementary to providing a secure, isolated network that can be extended through selective enabling of direct Internet access or private connectivity to another network. Amazon EC2 instances running in an Amazon VPC inherit all of the benefits described next that are related to the guest operating system and protection against packet sniffing. Note, however, that you must create security groups specifically for your Amazon VPC; any Amazon EC2 security groups that you have created will not work inside your Amazon VPC. In addition, Amazon VPC security groups have additional capabilities that Amazon EC2 security groups do not have, such as being able to change the security group after the instance is launched and being able to specify any protocol with a standard protocol number (as opposed to just TCP, User Datagram Protocol [UDP], or Internet Control Message Protocol [ICMP]).

Each Amazon VPC is a distinct, isolated network in the cloud; network traffic in each Amazon VPC is isolated from all other Amazon VPCs. At creation time, you select an IP address range for each Amazon VPC. You may create and attach an Internet gateway, virtual private gateway, or both to establish external connectivity, subject to the following controls.

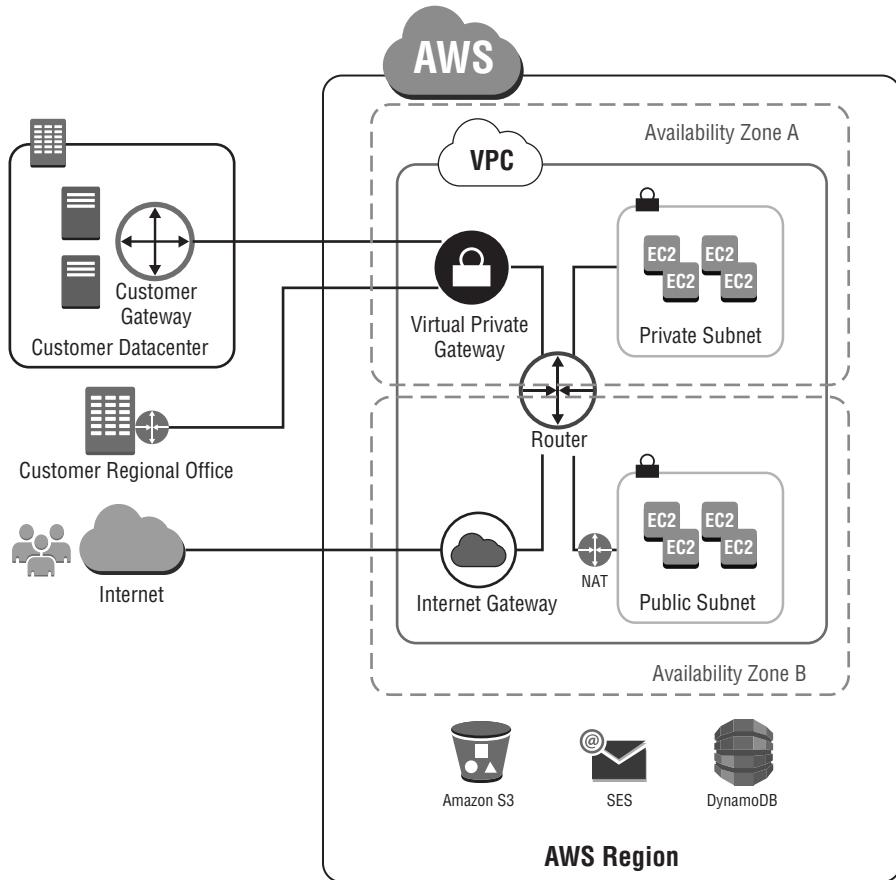
API access Calls to create and delete Amazon VPCs; change routing, security group, and network ACL parameters; and perform other functions are all signed by your Amazon secret access key, which could be either the AWS account's secret access key or the secret access key of a user created with IAM. Without access to your secret access key, Amazon VPC API calls cannot be made on your behalf. In addition, API calls can be encrypted with SSL to maintain confidentiality. AWS recommends always using SSL-protected API endpoints. IAM also enables a customer to further control what APIs a newly created user has permissions to call.

Subnets and route tables You create one or more subnets in each Amazon VPC. Each instance launched in the Amazon VPC is connected to one subnet. Traditional Layer 2 security attacks, including MAC spoofing and Address Resolution Protocol (ARP) spoofing, are blocked. Each subnet in an Amazon VPC is associated with a routing table, and all network traffic leaving the subnet is processed by the routing table to determine the destination.

Firewall (security groups) Like Amazon EC2, Amazon VPC supports a complete firewall solution, enabling filtering on both ingress and egress traffic from an instance. The default group enables inbound communication from other members of the same group and outbound communication to any destination. Traffic can be restricted by any IP protocol, by service port, and by source/destination IP address (individual IP or CIDR block). The firewall isn't controlled through the guest operating system; rather, it can be modified only through the invocation of Amazon VPC APIs. AWS supports the ability to grant granular access to different administrative functions on the instances and the firewall, therefore enabling you to implement additional security through separation of duties. The level of security afforded by the firewall is a function of which ports you open and for what duration and purpose. Well-informed traffic management and security design are still required

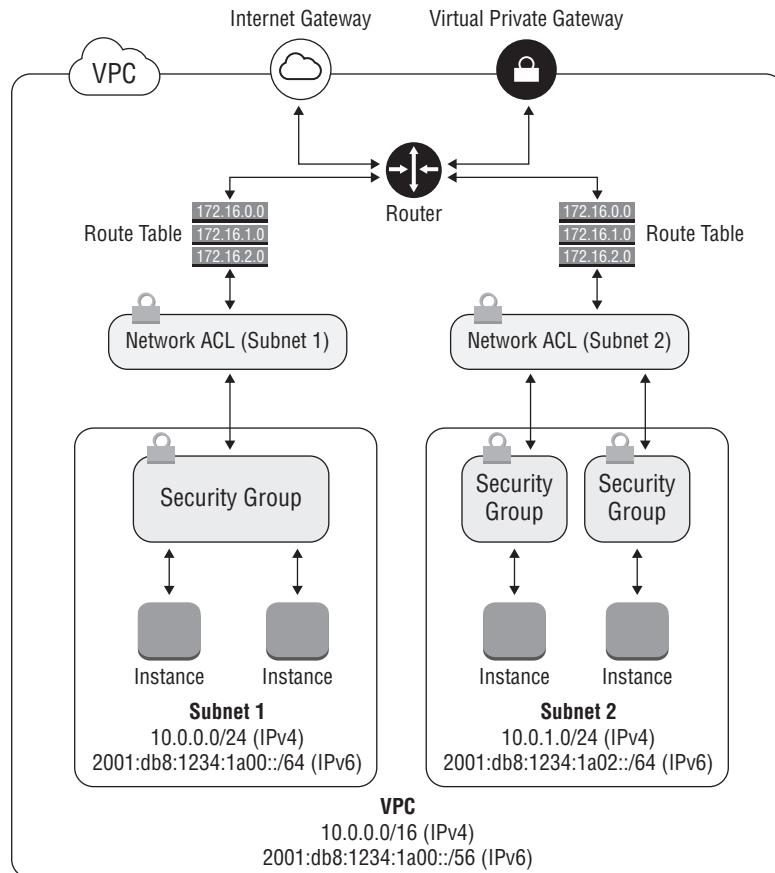
on a per-instance basis. AWS further encourages you to apply additional per-instance filters with host-based firewalls, such as IP tables or the Windows Firewall. Figure 3.5 illustrates an Amazon VPC with two types of subnets—public and private—and two network paths with two different networks—a customer datacenter and the Internet.

FIGURE 3.5 Amazon VPC network architecture



Network ACLs To add a further layer of security within Amazon VPC, you can configure network ACLs. These are stateless traffic filters that apply to all traffic inbound or outbound from a subnet within Amazon VPC. These ACLs can contain ordered rules to allow or deny traffic based on IP protocol, by service port, and source/destination IP address.

Like security groups, network ACLs are managed through Amazon VPC APIs, adding an additional layer of protection and enabling additional security through separation of duties. Figure 3.6 depicts how the security controls discussed thus far interrelate to enable flexible network topologies while providing complete control over network traffic flows.

FIGURE 3.6 Flexible network topologies

Virtual private gateways A *virtual private gateway* enables private connectivity between the Amazon VPC and another network. Network traffic within each virtual private gateway is isolated from network traffic within all other virtual private gateways. You can establish VPN connections to the virtual private gateway from gateway devices at your premises. Each connection is secured by a pre-shared key in conjunction with the IP address of the customer gateway device.

Internet gateways An *Internet gateway* may be attached to an Amazon VPC to enable direct connectivity to Amazon S3, other AWS Cloud services, and the Internet. Each instance desiring this access must either have an Elastic IP associated with it or route traffic through a Network Address Translation (NAT) instance. Additionally, network routes are configured to direct traffic to the Internet gateway. AWS provides reference NAT AMIs that you can extend to perform network logging, deep packet inspection, application layer filtering, or other security controls.

This access can only be modified through the invocation of Amazon VPC APIs. AWS supports the ability to grant granular access to different administrative functions on the instances and the Internet gateway, enabling you to implement additional security through separation of duties.

Dedicated instances Within an Amazon VPC, you can launch Amazon EC2 instances that are physically isolated at the host hardware level (that is, they will run on single-tenant hardware). An Amazon VPC can be created with “dedicated” tenancy so that all instances launched into the Amazon VPC will use this feature. Alternatively, an Amazon VPC may be created with “default” tenancy, but you can specify dedicated tenancy for particular instances launched into it. More information on networking on AWS can be found in Chapter 5.

Dedicated hosts An *Amazon EC2 Dedicated Host* is a physical server with Amazon EC2 instance capacity fully dedicated to your use. Dedicated hosts allow you to use your existing per-socket, per-core, or per-virtual machine software licenses, including Windows Server, Microsoft SQL Server, SUSE, Linux Enterprise Server, and so on. More information on dedicated hosts can be found in Chapter 4.

Amazon CloudFront Security

Amazon CloudFront gives customers an easy way to distribute content to end users with low latency and high data transfer speeds. It delivers dynamic, static, and streaming content using a global network of edge locations. Requests for customers’ objects are automatically routed to the nearest edge location, so content is delivered with the best possible performance. Amazon CloudFront is optimized to work with other AWS Cloud services like Amazon S3, Amazon EC2, Elastic Load Balancing, and Amazon Route 53. It also works seamlessly with any non-AWS origin server that stores the original, definitive versions of your files.

Amazon CloudFront requires that every request made to its control API is authenticated so that only authorized users can create, modify, or delete their own Amazon CloudFront distributions. Requests are signed with an HMAC-SHA-1 signature calculated from the request and the user’s private key. Additionally, the Amazon CloudFront control API is only accessible via SSL-enabled endpoints.

There is no guarantee of durability of data held in Amazon CloudFront edge locations. The service may sometimes remove objects from edge locations if those objects are not requested frequently. Durability is provided by Amazon S3, which works as the origin server for Amazon CloudFront by holding the original, definitive copies of objects delivered by Amazon CloudFront.

If you want control over who can download content from Amazon CloudFront, you can enable the service’s private content feature. This feature has two components. The first controls how content is delivered from the Amazon CloudFront edge location to viewers on the Internet. The second controls how the Amazon CloudFront edge locations access objects in Amazon S3. Amazon CloudFront also supports geo-restriction, which restricts access to your content based on the geographic location of your viewers.

To control access to the original copies of your objects in Amazon S3, Amazon CloudFront allows you to create one or more origin access identities and associate these with your distributions. When an origin access identity is associated with an Amazon CloudFront distribution, the distribution will use that identity to retrieve objects from Amazon S3. You can then use Amazon S3's ACL feature, which limits access to that origin access identity so that the original copy of the object is not publicly readable.

To control who can download objects from Amazon CloudFront edge locations, the service uses a signed-URL verification system. To use this system, you first create a public-private key pair and upload the public key to your account via the AWS Management Console. You then configure your Amazon CloudFront distribution to indicate which accounts you would authorize to sign requests. You can indicate up to five AWS accounts that you trust to sign requests. As you receive requests, you will create policy documents indicating the conditions under which you want Amazon CloudFront to serve your content. These policy documents can specify the name of the object that is requested, the date and time of the request, and the source IP (or CIDR range) of the client making the request. You then calculate the SHA-1 hash of your policy document and sign this using your private key. Finally, you include both the encoded policy document and the signature as query string parameters when you reference your objects. When Amazon CloudFront receives a request, it will decode the signature using your public key. Amazon CloudFront will only serve requests that have a valid policy document and matching signature.

Note that private content is an optional feature that must be enabled when you set up your Amazon CloudFront distribution. Content delivered without this feature enabled will be publicly readable.

Amazon CloudFront provides the option to transfer content over an encrypted connection (HTTPS). By default, Amazon CloudFront will accept requests over both HTTP and HTTPS protocols. You can also configure Amazon CloudFront to require HTTPS for all requests or have Amazon CloudFront redirect HTTP requests to HTTPS. You can even configure Amazon CloudFront distributions to allow HTTP for some objects but require HTTPS for other objects. More information on Amazon CloudFront can be found in Chapters 5 and 6.

Storage

AWS provides low-cost data storage with high durability and availability. AWS offers storage choices for backup, archiving, disaster recovery, and block and object storage.

Amazon Simple Storage Service (Amazon S3) Security

Amazon S3 allows you to upload and retrieve data at any time from anywhere on the web. Amazon S3 stores data as objects in buckets. An object can be any kind of file: a text file, a photo, a video, and more. When you add a file to Amazon S3, you have the option of including metadata with the file and setting permissions to control access to the file. For each bucket, you can control access to the bucket (who can create, delete, and list objects in the bucket), view access logs for the bucket and its objects, and choose the geographical region where Amazon S3 will store the bucket and its contents.

Data Access

Access to data stored in Amazon S3 is restricted by default; only bucket and object owners have access to the Amazon S3 resources that they create (note that a bucket/object owner is the AWS account owner, not the user who created the bucket/object). There are multiple ways to control access to buckets and objects.

IAM policies IAM enables organizations with many employees to create and manage multiple users under a single AWS account. IAM policies are attached to the users, enabling centralized control of permissions for users under your AWS account to access buckets or objects. With IAM policies, you can only grant users in your own AWS account permission to access your Amazon S3 resources.

ACLs Within Amazon S3, you can use ACLs to give read or write access on buckets or objects to groups of users. With ACLs, you can only grant other AWS accounts (not specific users) access to your Amazon S3 resources.

Bucket policies Bucket policies in Amazon S3 can be used to add or deny permissions across some or all of the objects in a single bucket. Policies can be attached to users, groups, or Amazon S3 buckets, enabling centralized management of permissions. With bucket policies, you can grant users in your AWS account or other AWS accounts access to your Amazon S3 resources.

Query string authentication You can use a query string to express a request entirely in a URL. In this case, you use query parameters to provide request information, including the authentication information. Because the request signature is part of the URL, this type of URL is often referred to as a pre-signed URL. You can use pre-signed URLs to embed clickable links, which can be valid for up to seven days, in HTML.

You can further restrict access to specific resources based on certain conditions. For example, you can restrict access based on request time (date condition), whether the request was sent using SSL (Boolean conditions), a requester's IP address (IP address condition), or the requester's client application (string conditions). To identify these conditions, you use policy keys.

Amazon S3 also gives developers the option to use query string authentication, which allows them to share Amazon S3 objects through URLs that are valid for a predefined period of time. Query string authentication is useful for giving HTTP for browser access to resources that would normally require authentication. The signature in the query string secures the request.

Data Transfer

For maximum security, you can securely upload/download data to Amazon S3 via the SSL-encrypted endpoints. The encrypted endpoints are accessible from both the Internet and from within Amazon EC2, so that data is transferred securely both within AWS and to and from sources outside of AWS.

Data Storage

Amazon S3 provides multiple options for protecting data at rest. If you prefer to manage your own encryption, you can use a client encryption library like the Amazon S3 Encryption Client to encrypt data before uploading to Amazon S3. Alternatively, you can use *Amazon S3 Server Side Encryption (SSE)* if you prefer to have Amazon S3 manage the encryption process for you. Data is encrypted with a key generated by AWS or with a key that you supply, depending on your requirements. With Amazon S3 SSE, you can encrypt data on upload simply by adding an additional request header when writing the object. Decryption happens automatically when data is retrieved. Note that metadata, which you can include with your object, is not encrypted.



AWS recommends that customers not place sensitive information in Amazon S3 metadata.

Amazon S3 SSE uses one of the strongest block ciphers available: AES-256. With Amazon S3 SSE, every protected object is encrypted with a unique encryption key. This object key itself is then encrypted with a regularly rotated master key. Amazon S3 SSE provides additional security by storing the encrypted data and encryption keys in different hosts. Amazon S3 SSE also makes it possible for you to enforce encryption requirements. For example, you can create and apply bucket policies that require that only encrypted data can be uploaded to your buckets.

When an object is deleted from Amazon S3, removal of the mapping from the public name to the object starts immediately and is generally processed across the distributed system in several seconds. Once the mapping is removed, there is no remote access to the deleted object. The underlying storage area is then reclaimed for use by the system.

Amazon S3 Standard is designed to provide 99.99999999 percent durability of objects over a given year. This durability level corresponds to an average annual expected loss of 0.000000001 percent of objects. For example, if you store 10,000 objects with Amazon S3, you can on average expect to incur a loss of a single object once every 10,000,000 years. In addition, Amazon S3 is designed to sustain the concurrent loss of data in two facilities.

Access Logs

An Amazon S3 bucket can be configured to log access to the bucket and objects in it. The access log contains details about each access request, including request type, the requested resource, the requestor's IP, and the time and date of the request. When logging is enabled for a bucket, log records are periodically aggregated into log files and delivered to the specified Amazon S3 bucket.

Cross-Origin Resource Sharing (CORS)

AWS customers who use Amazon S3 to host static web pages or store objects used by other web pages can load content securely by configuring an Amazon S3 bucket to explicitly enable cross-origin requests. Modern browsers use the same-origin policy to block

JavaScript or HTML5 from allowing requests to load content from another site or domain as a way to help ensure that malicious content is not loaded from a less reputable source (such as during cross-site scripting attacks). With the *Cross-Origin Resource Sharing (CORS)* policy enabled, assets such as web fonts and images stored in an Amazon S3 bucket can be safely referenced by external web pages, style sheets, and HTML5 applications. For more information on Amazon S3, refer to Chapter 6.

Amazon Glacier Security

Like Amazon S3, the *Amazon Glacier* service provides low-cost, secure, and durable storage. Where Amazon S3 is designed for rapid retrieval, Amazon Glacier is meant to be used as an archival service for data that is not accessed often and for which retrieval times of several hours are suitable.

Amazon Glacier stores files as archives in vaults. Archives can consist of any data such as a photo, video, or document, and can contain one or several files. You can store an unlimited number of archives in a single vault and can create up to 1,000 vaults per region. Each archive can contain up to 40 TB of data.

Data Transfer

For maximum security, you can securely upload/download data to Amazon Glacier via the SSL-encrypted endpoints. The encrypted endpoints are accessible from both the Internet and from within Amazon EC2, so that data is transferred securely both within AWS and to and from sources outside of AWS.

Data Retrieval

Retrieving archives from Amazon Glacier requires the initiation of a retrieval job, which is generally completed in three to five hours. You can then access the data via HTTP GET requests. The data will remain available to you for 24 hours. You can retrieve an entire archive or several files from an archive. If you want to retrieve only a subset of an archive, you can use one retrieval request to specify the range of the archive that contains the files in which you are interested, or you can initiate multiple retrieval requests, each with a range for one or more files.

You can also limit the number of vault inventory items retrieved by filtering on an archive creation date range or by setting a maximum items limit. Whichever method you choose, when you retrieve portions of your archive, you can use the supplied checksum to help ensure the integrity of the files, provided that the range that is retrieved is aligned with the tree hash of the overall archive.

Data Storage

Amazon Glacier automatically encrypts data using AES-256 and stores it durably in an immutable form. Amazon Glacier is designed to provide average annual durability of 99.99999999 percent for an archive. It stores each archive in multiple facilities and

multiple devices. Unlike traditional systems, which can require laborious data verification and manual repair, Amazon Glacier performs regular, systematic data integrity checks and is built to be self-healing.

Data Access

Only your account can access your data in Amazon Glacier. To control access to your data in Amazon Glacier, you can use IAM to specify which users in your account have rights to operations on a given vault.

AWS Snowball

AWS Snowball is a data transport solution that accelerates moving terabytes to petabytes of data into and out of AWS using storage appliances designed to be secure for physical transport.

Data Transfer

When you're using a standard AWS Snowball appliance to import data into Amazon S3, all data transferred to an AWS Snowball appliance has two layers of encryption:

1. A layer of encryption is applied in the memory of your local workstation. This layer is applied whether you're using the Amazon S3 Adapter for AWS Snowball or the AWS Snowball client. This encryption uses AES Galois/Counter Mode (GCM) 256-bit keys, and the keys are cycled for every 60 GB of data transferred.
2. SSL encryption is a second layer of encryption for all data going onto or off of a standard AWS Snowball appliance.

AWS Snowball uses SSE to protect data at rest.

Data Retrieval

To use AWS Snowball export, simply sign in to the AWS Management Console, choose AWS Snowball, and create an export job. As with an import job, you specify the AWS Region and Amazon S3 buckets that you want to use. AWS Snowball encrypts all data with 256-bit encryption.

Data Storage

AWS Snowball encrypts all data with 256-bit encryption. You manage your encryption keys by using AWS KMS. Your keys are never sent to or stored on the appliance.

Data Cleansing

When the data transfer job has been processed and verified, AWS performs a software erasure of the AWS Snowball appliance that follows the National Institute of Standards and Technology (NIST) guidelines for media sanitization.

More information on AWS Snowball is available in Chapter 6.

AWS Storage Gateway Security

The *AWS Storage Gateway* service connects your on-premises software appliance with cloud-based storage to provide seamless and secure integration between your IT environment and AWS storage infrastructure. The service enables you to upload data securely to AWS scalable, reliable, and secure Amazon S3 storage service for cost-effective backup and rapid disaster recovery.

Data Transfer

Data is asynchronously transferred from your on-premises storage hardware to AWS over SSL.

Data Storage

The data is stored encrypted in Amazon S3 using AES-256, a symmetric key encryption standard using 256-bit encryption keys. AWS Storage Gateway only uploads data that has changed, minimizing the amount of data sent over the Internet.

Database

AWS provides a number of database solutions for developers and businesses, from managed relational and NoSQL database services to in-memory caching as a service and a petabyte-scale data warehouse service.

Amazon DynamoDB Security

Amazon DynamoDB is a managed NoSQL database service that provides fast and predictable performance with seamless scalability. Amazon DynamoDB enables you to offload the administrative burdens of operating and scaling distributed databases to AWS, so you don't have to worry about hardware provisioning, setup and configuration, replication, software patching, or cluster scaling.

You can create a database table that can store and retrieve any amount of data and serve any level of request traffic. Amazon DynamoDB automatically spreads the data and traffic for the table over a sufficient number of servers to handle the request capacity you specified and the amount of data stored, while maintaining consistent, fast performance. All data items are stored on Solid State Drives (SSDs) and are automatically replicated across multiple Availability Zones in a region to provide built-in high availability and data durability.

You can set up automatic backups using a special template in AWS Data Pipeline that was created just for copying Amazon DynamoDB tables. You can choose full or incremental backups to a table in the same region or a different region. You can use the copy for disaster recovery in the event that an error in your code damages the original table or to federate Amazon DynamoDB data across regions to support a multi-region application.

To control who can use the Amazon DynamoDB resources and API, you set up permissions in IAM. In addition to controlling access at the resource level with IAM, you can also control access at the database level. You can create database-level permissions

that allow or deny access to items (rows) and attributes (columns) based on the needs of your application. These database-level permissions are called *fine-grained access controls*, and you create them using an IAM policy that specifies under what circumstances a user or application can access an Amazon DynamoDB table. The IAM policy can restrict access to individual items in a table, access to the attributes in those items, or both at the same time.

In addition to requiring database and user permissions, each request to Amazon DynamoDB must contain a valid HMAC-SHA-256 signature or the request is rejected. The AWS SDKs automatically sign your requests; however, if you want to write your own HTTP POST requests, you must provide the signature in the header of your request to Amazon DynamoDB. To calculate the signature, you must request temporary security credentials from the AWS Security Token Service (AWS STS). Use the temporary security credentials to sign your requests to Amazon DynamoDB. Amazon DynamoDB is accessible via SSL-encrypted endpoints, and the encrypted endpoints are accessible from both the Internet and from within Amazon EC2.

Amazon Relational Database Service (Amazon RDS) Security

Amazon RDS allows you to create a relational database instance (DB instance) quickly and flexibly scale the associated compute resources and storage capacity to meet application demand. Amazon RDS manages the DB instance on your behalf by performing backups, handling failover, and maintaining the database software. As of the time of this writing, Amazon RDS is available for MySQL, Oracle, Microsoft SQL Server, MariaDB, Amazon Aurora, and PostgreSQL database engines.

Amazon RDS has multiple features that enhance reliability for critical production databases, including DB security groups, permissions, SSL connections, automated backups, DB snapshots, and multiple Availability Zone (Multi-AZ) deployments. DB instances can also be deployed in an Amazon VPC for additional network isolation.

Access control When you first create a DB instance in Amazon RDS, you will create a master user account, which is used only within the context of Amazon RDS to control access to your DB instance(s). The master user account is a native database user account that allows you to log on to your DB instance with all database privileges. You can specify the master user name and password you want associated with each DB instance when you create the DB instance. Once you have created your DB instance, you can connect to the database using the master user credentials. Subsequently, you can create additional user accounts so that you can restrict who can access your DB instance.

You can control Amazon RDS DB instance access via *DB security groups*, which are similar to Amazon EC2 security groups but are not interchangeable. DB security groups act like a firewall controlling network access to your DB instance. DB security groups default to deny all access mode, and customers must specifically authorize network ingress. There are two ways of doing this:

- Authorizing a network IP range
- Authorizing an existing Amazon EC2 security group

DB security groups only allow access to the database server port (all others are blocked) and can be updated without restarting the Amazon RDS DB instance.

Using IAM, you can further control access to your Amazon RDS DB instances. IAM enables you to control what Amazon RDS operations each individual IAM user has permission to call.

Network isolation For additional network access control, you can run your DB instances in an Amazon VPC. Amazon VPC enables you to isolate your DB instances by specifying the IP range that you want to use and connect to your existing IT infrastructure through an industry-standard encrypted IPsec VPN. Running Amazon RDS in an Amazon VPC enables you to have a DB instance within a private subnet. You can also set up a virtual private gateway that extends your corporate network into your Amazon VPC and allows access to the Amazon RDS DB instance in that Amazon VPC.

For Multi-AZ deployments, defining a subnet for all Availability Zones in a region will allow Amazon RDS to create a new standby in another Availability Zone should the need arise. You can create DB subnet groups, which are collections of subnets that you may want to designate for your Amazon RDS DB instances in an Amazon VPC. Each DB subnet group should have at least one subnet for every Availability Zone in a given region. In this case, when you create a DB instance in an Amazon VPC, you select a DB subnet group. Amazon RDS then uses that DB subnet group and your preferred Availability Zone to select a subnet and an IP address within that subnet. Amazon RDS creates and associates an elastic network interface to your DB instance with that IP address.

DB instances deployed within an Amazon VPC can be accessed from the Internet or from Amazon EC2 instances outside of the Amazon VPC via VPN or bastion hosts that you can launch in your public subnet. To use a bastion host, you need to set up a public subnet with an Amazon EC2 instance that acts as an SSH bastion. This public subnet must have an Internet gateway and routing rules that allow traffic to be directed via the SSH host, which must then forward requests to the private IP address of your Amazon RDS DB instance.

DB security groups can be used to help secure DB instances within an Amazon VPC. In addition, network traffic entering and exiting each subnet can be allowed or denied via network ACLs. All network traffic entering or exiting your Amazon VPC via your IPsec VPN connection can be inspected by your on-premises security infrastructure, including network firewalls and intrusion detection systems.

Encryption You can encrypt connections between your application and your DB instance using SSL. For MySQL and SQL Server, Amazon RDS creates an SSL certificate and installs the certificate on the DB instance when the instance is provisioned. For MySQL, you launch the MySQL client using the `--ssl_ca` parameter to reference the public key in order to encrypt connections. For SQL Server, download the public key and import the certificate into your Windows operating system. Oracle RDS uses Oracle-native network encryption with a DB instance. You simply add the native network encryption option to an option group and associate that option group with the DB instance. Once an encrypted connection is established, data transferred between the DB instance and your application will be encrypted during transfer. You can also require your DB instance to accept only encrypted connections.

Amazon RDS supports Transparent Data Encryption (TDE) for SQL Server (SQL Server Enterprise Edition) and Oracle (part of the Oracle Advanced Security option available in Oracle Enterprise Edition). The TDE feature automatically encrypts data before it is written to storage and automatically decrypts data when it is read from storage. If you require your MySQL data to be encrypted while at rest in the database, your application must manage the encryption and decryption of data.

Note that SSL support in Amazon RDS is for encrypting the connection between your application and your DB instance; it should not be relied on for authenticating the DB instance itself. Although SSL offers security benefits, be aware that SSL encryption is a compute-intensive operation and will increase the latency of your database connection.

Automated backups and DB snapshots Amazon RDS provides two different methods for backing up and restoring your DB instances: automated backups and DB snapshots. Turned on by default, the automated backup feature of Amazon RDS enables point-in-time recovery for your DB instances. Amazon RDS will back up your database and transaction logs and store both for a user-specified retention period. This allows you to restore a DB instance to any second during your retention period, up to the last five minutes. Your automatic backup retention period can be configured to up to 35 days.

DB snapshots are user-initiated backups of your DB instances. These full database backups are stored by Amazon RDS until you explicitly delete them. You can copy DB snapshots of any size and move them between any of AWS public regions or copy the same snapshot to multiple regions simultaneously. You can then create a new DB instance from a DB snapshot whenever you desire.

During the backup window, storage I/O may be suspended while your data is being backed up. This I/O suspension typically lasts a few minutes. This I/O suspension is avoided with Multi-AZ deployments, because the backup is taken from the standby.

DB instance replication AWS Cloud resources are housed in highly available datacenter facilities in different regions of the world, and each region contains multiple distinct locations called *Availability Zones*. Each Availability Zone is engineered to be isolated from failures in other Availability Zones and to provide inexpensive, low-latency network connectivity to other Availability Zones in the same region.

To architect for high availability of your Oracle, PostgreSQL, or MySQL databases, you can run your Amazon RDS DB instances in several Availability Zones, an option called a Multi-AZ deployment. When you select this option, AWS automatically provisions and maintains a synchronous standby replica of your DB instances in a different Availability Zone. The primary DB instance is synchronously replicated across Availability Zones to the standby replica. In the event of DB instance or Availability Zone failure, Amazon RDS will automatically failover to the standby so that database operations can resume quickly without administrative intervention.

For customers who use MySQL and need to scale beyond the capacity constraints of a single DB instance for read-heavy database workloads, Amazon RDS provides a read replica option. Once you create a read replica, database updates on the source DB instance are replicated to the read replica using MySQL's native, asynchronous replication. You can create

multiple read replicas for a given source DB instance and distribute your application’s read traffic among them. Read replicas can be created with Multi-AZ deployments to gain read scaling benefits in addition to the enhanced database write availability and data durability provided by Multi-AZ deployments.

Automatic software patching Amazon RDS will make sure that the relational database software powering your deployment stays up to date with the latest patches. When necessary, patches are applied during a maintenance window that you can control. You can think of the Amazon RDS maintenance window as an opportunity to control when DB instance modifications (such as scaling DB instance class) and software patching occur, in the event that either are requested or required. If a maintenance event is scheduled for a given week, it will be initiated and completed at some point during the 30-minute maintenance window that you identify.

The only maintenance events that require Amazon RDS to take your DB instances offline are scale compute operations (which generally take only a few minutes from start to finish) or required software patching. Required patching is automatically scheduled only for patches that are related to security and durability. Such patching occurs infrequently (typically once every few months) and should seldom require more than a fraction of your maintenance window. If you do not specify a preferred weekly maintenance window when creating a DB instance, a 30-minute default value is assigned. If you want to modify when maintenance is performed on your behalf, you can do so by modifying a DB instance in the AWS Management Console or by using the `ModifyDBInstance` API. Each DB instance can have different preferred maintenance windows, if you so choose.

Running DB instances in a Multi-AZ deployment can further reduce the impact of a maintenance event, as Amazon RDS will conduct maintenance via the following steps:

1. Perform maintenance on standby.
2. Promote standby to primary.
3. Perform maintenance on old primary, which becomes the new standby.

When an Amazon RDS DB instance deletion API (`DeleteDBInstance`) is run, the DB instance is marked for deletion. Once the instance no longer indicates deleting status, it has been removed. At this point, the instance is no longer accessible and, unless a final snapshot copy was asked for, it cannot be restored and will not be listed by any of the tools or APIs.

Amazon Redshift Security

Amazon Redshift is a petabyte-scale SQL data warehouse service that runs on highly optimized and managed AWS compute and storage resources. The service has been architected not only to scale up or down rapidly, but also to improve query speeds significantly, even on extremely large datasets. To increase performance, Amazon Redshift uses techniques such as columnar storage, data compression, and zone maps to reduce the amount of I/O needed to perform queries. It also has a Massively Parallel Processing (MPP) architecture,

which parallelizes and distributes SQL operations to take advantage of all available resources.

Cluster access By default, clusters that you create are closed to everyone. Amazon Redshift enables you to configure firewall rules (security groups) to control network access to your data warehouse cluster. You can also run Amazon Redshift inside an Amazon VPC to isolate your data warehouse cluster in your own virtual network and connect it to your existing IT infrastructure using industry-standard encrypted IPsec VPN.

The AWS account that creates the cluster has full access to the cluster. Within your AWS account, you can use IAM to create user accounts and manage permissions for those accounts. By using IAM, you can grant different users permission to perform only the cluster operations that are necessary for their work. Like all databases, you must grant permission in Amazon Redshift at the database level in addition to granting access at the resource level. Database users are named user accounts that can connect to a database and are authenticated when they log in to Amazon Redshift. In Amazon Redshift, you grant database user permissions on a per-cluster basis instead of on a per-table basis. However, users can see data only in the table rows that were generated by their own activities; rows generated by other users are not visible to them.

The user who creates a database object is its owner. By default, only a super user or the owner of an object can query, modify, or grant permissions on the object. For users to use an object, you must grant the necessary permissions to the user or the group that contains the user. In addition, only the owner of an object can modify or delete it.

Data backups Amazon Redshift distributes your data across all compute nodes in a cluster. When you run a cluster with at least two compute nodes, data on each node will always be mirrored on disks on another node, reducing the risk of data loss. In addition, all data written to a node in your cluster is continuously backed up to Amazon S3 using snapshots. Amazon Redshift stores your snapshots for a user-defined period, which can be from 1 to 35 days. You can also take your own snapshots at any time; these snapshots leverage all existing system snapshots and are retained until you explicitly delete them.

Amazon Redshift continuously monitors the health of the cluster and automatically replicates data from failed drives and replaces nodes as necessary. All of this happens without any effort on your part, although you may see a slight performance degradation during the re-replication process.

You can use any system or user snapshot to restore your cluster using the AWS Management Console or the Amazon Redshift APIs. Your cluster is available as soon as the system metadata has been restored, and you can start running queries while user data is spooled down in the background.

Data encryption When creating a cluster, you can choose to encrypt it in order to provide additional protection for your data at rest. When you enable encryption in your cluster, Amazon Redshift stores all data in user-created tables in an encrypted format using hardware-accelerated AES-256 block encryption keys. This includes all data written to disk and any backups.

Amazon Redshift uses a four-tier, key-based architecture for encryption. These keys consist of data encryption keys, a database key, a cluster key, and a master key.

- Data encryption keys encrypt data blocks in the cluster. Each data block is assigned a randomly generated AES-256 key. These keys are encrypted by using the database key for the cluster.
- The database key encrypts data encryption keys in the cluster. The database key is a randomly generated AES-256 key. It is stored on disk in a separate network from the Amazon Redshift cluster and encrypted by a master key. Amazon Redshift passes the database key across a secure channel and keeps it in memory in the cluster.
- The cluster key encrypts the database key for the Amazon Redshift cluster. You can use either AWS or an HSM to store the cluster key. HSMs provide direct control of key generation and management and make key management separate and distinct from the application and the database.
- The master key encrypts the cluster key if it is stored in AWS. The master key encrypts the cluster-key-encrypted database key if the cluster key is stored in an HSM.

You can have Amazon Redshift rotate the encryption keys for your encrypted clusters at any time. As part of the rotation process, keys are also updated for all of the cluster's automatic and manual snapshots. Note that enabling encryption in your cluster will impact performance, even though it is hardware accelerated.

Encryption also applies to backups. When you are restoring from an encrypted snapshot, the new cluster will be encrypted as well.

To encrypt your table load data files when you upload them to Amazon S3, you can use Amazon S3 SSE. When you load the data from Amazon S3, the `COPY` command will decrypt the data as it loads the table.

Database audit logging Amazon Redshift logs all SQL operations, including connection attempts, queries, and changes to your database. You can access these logs using SQL queries against system tables or choose to have them downloaded to a secure Amazon S3 bucket. You can then use these audit logs to monitor your cluster for security and troubleshooting purposes.

Automatic software patching Amazon Redshift manages all the work of setting up, operating, and scaling your data warehouse, including provisioning capacity, monitoring the cluster, and applying patches and upgrades to the Amazon Redshift engine. Patches are applied only during specified maintenance windows.

SSL connections To protect your data in transit within the AWS Cloud, Amazon Redshift uses hardware-accelerated SSL to communicate with Amazon S3 or Amazon DynamoDB for `COPY`, `UNLOAD`, backup, and restore operations. You can encrypt the connection between your client and the cluster by specifying SSL in the parameter group associated with the cluster. To have your clients also authenticate the Amazon Redshift server, you can install the public key (.pem file) for the SSL certificate on your client and use the key to connect to your clusters.

Amazon Redshift offers the newer, stronger cipher suites that use the Elliptic Curve Diffie-Hellman Ephemeral (ECDHE) protocol. ECDHE allows SSL clients to provide perfect forward secrecy between the client and the Amazon Redshift cluster. Perfect forward secrecy uses session keys that are ephemeral and not stored anywhere, which prevents the decoding of captured data by unauthorized third parties, even if the secret long-term key itself is compromised. You do not need to configure anything in Amazon Redshift to enable ECDHE; if you connect from a SQL client tool that uses ECDHE to encrypt communication between the client and server, Amazon Redshift will use the provided cipher list to make the appropriate connection. For more information on Amazon Redshift, refer to Chapter 7, “Databases.”

Amazon ElastiCache Security

Amazon ElastiCache is a web service that makes it easy to set up, manage, and scale distributed in-memory cache environments in the cloud. The service improves the performance of web applications by allowing you to retrieve information from a fast, managed, in-memory caching system instead of relying entirely on slower disk-based databases. It can be used to improve latency and throughput significantly for many read-heavy application workloads (such as social networking, gaming, media sharing, and Q&A portals) or compute-intensive workloads (such as a recommendation engine). Caching improves application performance by storing critical pieces of data in memory for low-latency access. Cached information may include the results of I/O-intensive database queries or the results of computationally intensive calculations.

The Amazon ElastiCache service automates time-consuming management tasks for in-memory cache environments, such as patch management, failure detection, and recovery. It works in conjunction with other AWS Cloud services (such as Amazon EC2, Amazon CloudWatch, and Amazon Simple Notification Service [Amazon SNS]) to provide a secure, high-performance, and managed in-memory cache. For example, an application running in Amazon EC2 can securely access an Amazon ElastiCache cluster in the same region with very low latency.

Using the Amazon ElastiCache service, you create a *cache cluster*, which is a collection of one or more *cache nodes*, each running an instance of the Memcached service. A cache node is a fixed-size chunk of secure, network-attached RAM. Each cache node runs an instance of the Memcached service and has its own DNS name and port. Multiple types of cache nodes are supported, each with varying amounts of associated memory. A cache cluster can be set up with a specific number of cache nodes and a cache parameter group that controls the properties for each cache node. All cache nodes in a cache cluster are designed to be of the same node type and have the same parameter and security group settings.

Data access Amazon ElastiCache allows you to control access to your cache clusters using *cache security groups*. A cache security group acts like a firewall, controlling network access to your cache cluster. By default, network access is turned off to your cache clusters. If you want your applications to access your cache cluster, you must explicitly enable access from hosts in specific Amazon EC2 security groups. Once ingress rules are configured, the same rules apply to all cache clusters associated with that cache security group.

To allow network access to your cache cluster, create a cache security group and use the Authorize Cache Security Group Ingress API or AWS CLI command to authorize the desired Amazon EC2 security group (which in turn specifies the Amazon EC2 instances allowed). IP range-based access control is currently not enabled for cache clusters. All clients to a cache cluster must be within the Amazon EC2 network and authorized via cache security groups.

Amazon ElastiCache for Redis provides backup and restore functionality, where you can create a snapshot of your entire Redis cluster as it exists at a specific point in time. You can schedule automatic, recurring daily snapshots, or you can create a manual snapshot at any time. For automatic snapshots, you specify a retention period; manual snapshots are retained until you delete them. The snapshots are stored in Amazon S3 with high durability and can be used for warm starts, backups, and archiving.

Application Services

AWS offers a variety of managed services to use with your applications, including services that provide application streaming, queuing, push notification, email delivery, search, and transcoding.

Amazon Simple Queue Service (Amazon SQS) Security

Amazon Simple Queue Service (Amazon SQS) is a highly reliable, scalable message queuing service that enables asynchronous message-based communication between distributed components of an application. The components can be computers, Amazon EC2 instances, or a combination of both. With Amazon SQS, you can send any number of messages to an Amazon SQS queue at any time from any component. The messages can be retrieved from the same component or a different one, right away or at a later time (within 14 days). Messages are highly durable; each message is persistently stored in highly available, highly reliable queues. Multiple processes can read/write from/to an Amazon SQS queue at the same time without interfering with each other.

Data access Amazon SQS access is granted based on an AWS account or a user created with IAM. Once authenticated, the AWS account has full access to all user operations. An IAM user, however, only has access to the operations and queues for which they have been granted access via policy. By default, access to each individual queue is restricted to the AWS account that created it. However, you can allow other access to a queue, using either an Amazon SQS-generated policy or a policy you write.

Encryption Amazon SQS is accessible via SSL-encrypted endpoints. The encrypted endpoints are accessible from both the Internet and from within Amazon EC2. Data stored within Amazon SQS can be encrypted. Additionally, you can encrypt data before it is uploaded to Amazon SQS, provided that the application using the queue has a means to decrypt the message when it is retrieved. Encrypting messages within Amazon SQS helps protect against access to sensitive customer data by unauthorized persons, including AWS.

Amazon Simple Notification Service (Amazon SNS) Security

Amazon SNS is a web service that makes it easy to set up, operate, and send notifications from the cloud. It provides developers with a highly scalable, flexible, and cost-effective capability to publish messages from an application and immediately deliver them to subscribers or other applications. Amazon SNS provides a simple interface that can be used to create topics that customers want to notify applications (or people) about, subscribe clients to these topics, publish messages, and have these messages delivered over clients' protocol of choice (such as, HTTP/HTTPS and email).

Amazon SNS delivers notifications to clients using a push mechanism that eliminates the need to check or poll for new information and updates periodically. Amazon SNS can be leveraged to build highly reliable, event-driven workflows and messaging applications without the need for complex middleware and application management. The potential uses for Amazon SNS include monitoring applications, workflow systems, time-sensitive information updates, mobile applications, and many others.

Data access Amazon SNS provides access control mechanisms so that topics and messages are secured against unauthorized access. Topic owners can set policies for a topic that restrict who can publish or subscribe to a topic. Additionally, topic owners can encrypt transmission by specifying that the delivery mechanism must be HTTPS. Amazon SNS access is granted based on an AWS account or a user created with IAM. Once authenticated, the AWS account has full access to all user operations. An IAM user, however, only has access to the operations and topics for which they have been granted access via policy. By default, access to each individual topic is restricted to the AWS account that created it. You can allow other access to Amazon SNS using either an Amazon SNS-generated policy or a policy you write.

Analytics Services

AWS provides cloud-based analytics services to help you process and analyze any volume of data, whether your need is for managed Hadoop clusters, real-time streaming data, petabyte-scale data warehousing, or orchestration.

Amazon EMR Security

Amazon EMR is a managed web service that you can use to run Hadoop clusters that process vast amounts of data by distributing the work and data among several servers. It uses an enhanced version of the Apache Hadoop framework running on the web-scale infrastructure of Amazon EC2 and Amazon S3. You simply upload your input data and a data processing application into Amazon S3. Amazon EMR then launches the number of Amazon EC2 instances you specify. The service begins the job flow execution while pulling the input data from Amazon S3 into the launched Amazon EC2 instances. Once the job flow is finished, Amazon EMR transfers the output data to Amazon S3, where you can then retrieve it or use it as input in another job flow.

When launching job flows on your behalf, Amazon EMR sets up two Amazon EC2 security groups: one for the master nodes and another for the slaves. The master security

group has a port open for communication with the service. It also has the SSH port open to allow you to use SSH to connect into the instances using the key specified at startup. The slaves start in a separate security group, which only allows interaction with the master instance. By default, both security groups are set up to prohibit access from external sources, including Amazon EC2 instances belonging to other customers. Because these are security groups within your account, you can reconfigure them using the standard Amazon EC2 tools or dashboard. To protect customer input and output datasets, Amazon EMR transfers data to and from Amazon S3 using SSL.

Amazon EMR provides several ways to control access to the resources of your cluster. You can use IAM to create user accounts and roles and configure permissions that control which AWS features those users and roles can access. When you launch a cluster, you can associate an Amazon EC2 key pair with the cluster, which you can then use when you connect to the cluster using SSH. You can also set permissions that allow users other than the default Hadoop user to submit jobs to your cluster.

If an IAM user launches a cluster, that cluster is hidden from other IAM users on the AWS account by default. This filtering occurs on all Amazon EMR interfaces (the AWS Management Console, AWS CLI, API, and AWS SDKs) and helps prevent IAM users from accessing and inadvertently changing clusters created by other IAM users.

For an additional layer of protection, you can launch the Amazon EC2 instances of your Amazon EMR cluster into an Amazon VPC, which is like launching it into a private subnet. This allows you to control access to the entire subnet. You can also launch the cluster into an Amazon VPC and enable the cluster to access resources on your internal network using a VPN connection. You can encrypt the input data before you upload it to Amazon S3 using any common data encryption tool. If you do encrypt the data before it is uploaded, you then need to add a decryption step to the beginning of your job flow when Amazon EMR fetches the data from Amazon S3.

Amazon Kinesis Security

Amazon Kinesis is a managed service designed to handle real-time streaming of big data. It can accept virtually any amount of data, from any number of sources, scaling up and down as needed. You can use Amazon Kinesis in situations that call for large-scale, real-time data ingestion and processing, such as server logs, social media, market data feeds, and web click-stream data. Applications read and write data records to Amazon Kinesis in streams. You can create any number of Amazon Kinesis streams to capture, store, and transport data.

You can control logical access to Amazon Kinesis resources and management functions by creating users under your AWS account using IAM and controlling which Amazon Kinesis operations these users have permission to perform. To facilitate running your producer or consumer applications on an Amazon EC2 instance, you can configure that instance with an IAM role. That way, AWS credentials that reflect the permissions associated with the IAM role are made available to applications on the instance, which means that you don't have to use your long-term AWS security credentials. Roles have the added benefit of providing temporary credentials that expire within a short timeframe, which adds a measure of protection.

The Amazon Kinesis API is only accessible via an SSL-encrypted endpoint (`kinesis.us-east-1.amazonaws.com`) to help ensure secure transmission of your data to AWS. You must connect to that endpoint to access Amazon Kinesis, but you can then use the API to direct Amazon Kinesis to create a stream in any AWS Region.

Deployment and Management Services

AWS provides a variety of tools to help with the deployment and management of your applications. This includes services that allow you to create individual user accounts with credentials for access to AWS Cloud services. It also includes services for creating and updating stacks of AWS resources, deploying applications on those resources, and monitoring the health of those AWS resources. Other tools help you manage cryptographic keys using HSMs and log API activity for security and compliance purposes.

AWS Identity and Access Management (IAM) Security

As discussed previously, IAM allows you to create multiple users and manage the permissions for each of these users within your AWS account. A user is an identity (within an AWS account) with unique security credentials that can be used to access AWS Cloud services. Thus IAM eliminates the need to share passwords or keys and makes it easy to enable or disable a user's access as appropriate. IAM is integrated with AWS CloudFormation. More information on AWS CloudFormation can be found in Chapter 8, "Application Deployment and Management".

IAM enables you to minimize the use of your AWS account credentials. Once you create IAM user accounts, all interactions with AWS Cloud services and resources should occur with IAM user security credentials.

Roles As discussed earlier in this chapter, an IAM role uses temporary security credentials to allow you to delegate access to users or services that normally do not have access to your AWS resources. A *role* is a set of permissions used to access specific AWS resources, but these permissions are not tied to a specific IAM user or group. An authorized entity (for example, a mobile user or Amazon EC2 instance) assumes a role and receives temporary security credentials for authenticating to the resources defined in the role. Temporary security credentials provide enhanced security due to their short lifespan (the default expiration is 12 hours) and the fact that they cannot be reused after they expire. This can be particularly useful in providing limited, controlled access in certain situations.

Federated (Non-AWS) User Access *Federated users* are users (or applications) that do not have AWS accounts. With roles, you can give them access to your AWS resources for a limited amount of time. This is useful if you have non-AWS users who you can authenticate with an external service, such as Microsoft Active Directory, Lightweight Directory Access Protocol (LDAP), or Kerberos. The temporary AWS credentials used with the roles provide identity federation between AWS and your non-AWS users in your corporate identity and authorization system.

Security Assertion Markup Language (SAML) 2.0 If your organization supports Security Assertion Markup Language (SAML) 2.0, you can create trust between your

organization as an Identity Provider (IdP) and other organizations as service providers. In AWS, you can configure AWS as the service provider, and use SAML to provide your users with federated Single-Sign On (SSO) to the AWS Management Console or to get federated access to call AWS APIs.

Roles are also useful if you create a mobile or web-based application that accesses AWS resources. AWS resources require security credentials for programmatic requests; however, you shouldn't embed long-term security credentials in your application because they are accessible to the application's users and can be difficult to rotate. Instead, you can let users sign in to your application using Login with Amazon, Facebook, or Google, and then use their authentication information to assume a role and get temporary security credentials.

Cross-account access For organizations that use multiple AWS accounts to manage their resources, you can set up roles to provide users who have permissions in one account to access resources under another account. For organizations that have personnel who only rarely need access to resources under another account, using roles helps to ensure that credentials are provided temporarily and only as needed.

Applications running on Amazon EC2 instances that need to access AWS resources If an application runs on an Amazon EC2 instance and needs to make requests for AWS resources (such as Amazon S3 buckets, Amazon DynamoDB tables), it must have security credentials. Using roles instead of creating individual IAM accounts for each application on each instance can save significant time for customers who manage a large number of instances or an elastically scaling fleet using Auto Scaling.

The temporary credentials include a security token, an access key ID, and a secret access key. To give a user access to certain resources, you distribute the temporary security credentials to the user to whom you are granting temporary access. When the user makes calls to your resources, the user passes in the token and access key ID and signs the request with the secret access key. The token will not work with different access keys.

The use of temporary credentials provides additional protection for you because you do not have to manage or distribute long-term credentials to temporary users. In addition, the temporary credentials get automatically loaded to the target instance so you don't have to embed them somewhere unsafe like your code. Temporary credentials are automatically rotated or changed multiple times a day without any action on your part and are stored securely by default.

Mobile Services

AWS mobile services make it easier for you to build, ship, run, monitor, optimize, and scale cloud-powered applications for mobile devices. These services also help you authenticate users to your mobile application, synchronize data, and collect and analyze application usage.

Amazon Cognito Security

Amazon Cognito provides identity and sync services for mobile and web-based applications. It simplifies the task of authenticating users and storing, managing, and syncing their data across multiple devices, platforms, and applications. It provides temporary, limited-privilege credentials for both authenticated and unauthenticated users without having to manage any back-end infrastructure.

Amazon Cognito works with well-known identity providers like Google, Facebook, and Amazon to authenticate end users of your mobile and web applications. You can take advantage of the identification and authorization features provided by these services instead of having to build and maintain your own. Your application authenticates with one of these identity providers using the provider's SDK. Once the end user is authenticated with the provider, an OAuth or OpenID Connect token returned from the provider is passed by your application to Amazon Cognito, which returns a new Amazon Cognito ID for the user and a set of temporary, limited-privilege AWS credentials.

To begin using Amazon Cognito, you create an identity pool through the Amazon Cognito console. The *identity pool* is a store of user identity information that is specific to your AWS account. During the creation of the identity pool, you will be asked to create a new IAM role or pick an existing one for your end users. An IAM role is a set of permissions used to access specific AWS resources, but these permissions are not tied to a specific IAM user or group. An authorized entity (for example, mobile user, Amazon EC2 instance) assumes a role and receives temporary security credentials for authenticating to the AWS resources defined in the role. Temporary security credentials provide enhanced security due to their short lifespan (the default expiration is 12 hours) and the fact that they cannot be reused after they expire.

The role you select has an effect on which AWS Cloud services your end users will be able to access with the temporary credentials. By default, Amazon Cognito creates a new role with limited permissions; end users only have access to the Amazon Cognito Sync service and Amazon Mobile Analytics. If your application needs access to other AWS resources, such as Amazon S3 or Amazon DynamoDB, you can modify your roles directly from the IAM console.

With Amazon Cognito, there is no need to create individual AWS accounts or even IAM accounts for every one of your web/mobile application end users who will need to access your AWS resources. In conjunction with IAM roles, mobile users can securely access AWS resources and application features and even save data to the AWS Cloud without having to create an account or log in. If they choose to create an account or log in later, Amazon Cognito will merge data and identification information.

Because Amazon Cognito stores data locally and in the service, your end users can continue to interact with their data even when they are offline. Their offline data may be stale, but they can immediately retrieve anything they put into the dataset whether or not they are online. The client SDK manages a local SQLite store so that the application can work even when it is not connected. The SQLite store functions as a cache and is the target of all the read and write operations. Amazon Cognito's sync facility compares the local version of the data to the cloud version and pushes up or pulls down deltas as needed. Note that in

order to sync data across devices, your identity pool must support authenticated identities. Unauthenticated identities are tied to the device, so unless an end user authenticates, no data can be synced across multiple devices.

With Amazon Cognito, your application communicates directly with a supported public identity provider (Amazon, Facebook, or Google) to authenticate users. Amazon Cognito does not receive or store user credentials, only the OAuth or OpenID Connect token received from the identity provider. Once Amazon Cognito receives the token, it returns a new Amazon Cognito ID for the user and a set of temporary, limited-privilege AWS credentials. Each Amazon Cognito identity has access only to its own data in the sync store, and this data is encrypted when stored. In addition, all identity data is transmitted over HTTPS. The unique Amazon Cognito identifier on the device is stored in the appropriate secure location. For example, on iOS the Amazon Cognito identifier is stored in the iOS keychain, and user data is cached in a local SQLite database in the application's sandbox. If you require additional security, you can encrypt this identity data in the local cache by implementing encryption in your application.

Applications

AWS applications are managed services that enable you to provide your users with secure, centralized storage and work areas in the cloud.

Amazon WorkSpaces Security

Amazon WorkSpaces is a managed desktop service that allows you to provision cloud-based desktops quickly for your users. Simply choose a Windows 7 or Windows 10 bundle that best meets the needs of your users and the number of WorkSpaces that you would like to launch. Once the WorkSpaces are ready, users receive an email informing them where they can download the relevant client and log in to their WorkSpace. They can then access their cloud-based desktops from a variety of endpoint devices, including PCs, laptops, and mobile devices. However, your organization's data is never sent to or stored on the end-user device because Amazon WorkSpaces uses PC-over-IP (PCoIP), which provides an interactive video stream without transmitting actual data. The PCoIP protocol compresses, encrypts, and encodes the user's desktop computing experience and transmits as pixels only across any standard IP network to end-user devices.

In order to access their Amazon WorkSpaces, users must sign in using a set of unique credentials or their regular Active Directory credentials. When you integrate Amazon WorkSpaces into your corporate Active Directory, each WorkSpace joins your Active Directory domain and can be managed just like any other desktop in your organization. This means that you can use Active Directory group policies to manage your users' WorkSpaces to specify configuration options that control their desktops. If you choose not to use Active Directory or another type of on-premises directory to manage your users' WorkSpaces, you can create a private cloud directory in Amazon WorkSpaces that you can use for administration.

To provide an additional layer of security, you can also require the use of MFA upon sign-in, which is in the form of a hardware or software token. Amazon WorkSpaces supports MFA using an on-premises Remote Authentication Dial In User Service (RADIUS)

server or any security provider that supports RADIUS authentication. It currently supports the Password Authentication Protocol (PAP), Challenge-Handshake Authentication Protocol (CHAP), Microsoft CHAP (MS-CHAP) 1, and MS-CHAP2 protocols, along with RADIUS proxies.

Each WorkSpace resides on its own Amazon EC2 instance within an Amazon VPC. You can create WorkSpaces in an Amazon VPC you already own or have the Amazon WorkSpaces service create one for you automatically using the Amazon WorkSpaces Quick Start option. When you use the Quick Start option, Amazon WorkSpaces not only creates the Amazon VPC, but it also performs several other provisioning and configuration tasks for you, such as creating an Internet gateway for the Amazon VPC, setting up a directory in the Amazon VPC that is used to store user and WorkSpace information, creating a directory administrator account, creating the specified user accounts and adding them to the directory, and creating the Amazon WorkSpaces instances. The Amazon VPC can also be connected to an on-premises network using a secure VPN connection to allow access to an existing on-premises Active Directory and other intranet resources. You can add a security group that you create in your Amazon VPC to all of the WorkSpaces that belong to your Active Directory. This allows you to control network access from Amazon WorkSpaces in your Amazon VPC to other resources in your Amazon VPC and on-premises network.

Persistent storage for Amazon WorkSpaces is provided by Amazon EBS and is automatically backed up twice a day to Amazon S3. If Amazon WorkSpaces Sync is enabled on a WorkSpace, the folder a user chooses to sync will be continuously backed up and stored in Amazon S3. You can also use Amazon WorkSpaces Sync on a Mac or PC to sync documents to or from your WorkSpace so that you can always have access to your data regardless of the desktop computer you are using.

Because Amazon WorkSpaces is a managed service, AWS takes care of several security and maintenance tasks like daily backups and patching. Updates are delivered automatically to your WorkSpaces during a weekly maintenance window. You can control how patching is configured for a user's WorkSpace. By default, Windows Update is turned on, but you have the ability to customize these settings or use an alternative patch management approach if you desire. For the underlying operating system, Windows Update is enabled by default on Amazon WorkSpaces and configured to install updates on a weekly basis. You can use an alternative patching approach or configure Windows Update to perform updates at a time of your choosing. You can use IAM to control who on your team can perform administrative functions like creating or deleting WorkSpaces or setting up user directories. You can also set up a WorkSpace for directory administration, install your favorite Active Directory administration tools, and create organizational units and Group Policies in order to apply Active Directory changes more easily for all of your Amazon WorkSpaces users.

Summary

In this chapter, you learned that the first priority at AWS is the security of the cloud. Security within AWS is based on a “defense in depth” model where no one, single element is used to secure systems on AWS. Rather, AWS uses a multitude of elements—each

acting at different layers of a system—in total to secure the system. As you learned in the shared responsibility model, AWS is responsible for some layers of this model, and you are responsible for others. AWS also offers security tools and features of services for customers to use at their discretion. These concepts, tools, and features were discussed in this chapter.

Exam Essentials

Understand the shared responsibility model. AWS is responsible for securing the underlying infrastructure that supports the cloud, and you are responsible for anything you put on the cloud or connect to the cloud.

Understand regions and Availability Zones. Each region is completely independent. Each region is designed to be completely isolated from the other regions. This achieves the greatest possible fault tolerance and stability. Regions are collections of Availability Zones. Each Availability Zone is isolated, but the Availability Zones in a region are connected through low-latency links.

Understand high-availability system design within AWS. You should architect your AWS usage to take advantage of multiple regions and Availability Zones. Distributing applications across multiple Availability Zones provides the ability to remain resilient in the face of most failure modes, including natural disasters or system failures.

Understand the network security of AWS. Network devices, including firewall and other boundary devices, are in place to monitor and control communications at the external boundary of the network and at key internal boundaries within the network. These boundary devices employ rule sets, ACLs, and configurations to enforce the flow of information to specific information system services.

AWS has strategically placed a limited number of access points to the cloud to allow for a more comprehensive monitoring of inbound and outbound communications and network traffic. These customer access points are called API endpoints, and they allow HTTPS access, which lets you establish a secure communication session with your storage or compute instances within AWS.

Amazon EC2 instances cannot send spoofed network traffic. The AWS-controlled, host-based firewall infrastructure will not permit an instance to send traffic with a source IP or MAC address other than its own.

Unauthorized port scans by Amazon EC2 customers are a violation of the AWS Acceptable Use Policy. Violations of the AWS Acceptable Use Policy are taken seriously, and every reported violation is investigated.

It is not possible for an Amazon EC2 instance running in promiscuous mode to receive or sniff traffic that is intended for a different virtual instance.

Understand the use of credentials on AWS. AWS employs several credentials in order to positively identify a user or authorize an API call to the platform. Credentials include:

- Passwords
- AWS root account or IAM user account login to the AWS Management Console
- MFA
- Access keys
- Digitally signed requests to AWS APIs (using the AWS SDK, AWS CLI, or REST/Query APIs)

Understand the proper use of access keys. Because access keys can be misused if they fall into the wrong hands, AWS encourages you to save them in a safe place and not to embed them in your code. For customers with large fleets of elastically-scaling Amazon EC2 instances, the use of IAM roles can be a more secure and convenient way to manage the distribution of access keys.

Understand the value of AWS CloudTrail. AWS CloudTrail is a web service that records API calls made on your account and delivers log files to your Amazon S3 bucket. AWS CloudTrail's benefit is visibility into account activity by recording API calls made on your account.

Understand the security features of Amazon EC2. Amazon EC2 uses public-key cryptography to encrypt and decrypt login information. Public-key cryptography uses a public key to encrypt a piece of data, such as a password, and then the recipient uses the private key to decrypt the data. The public and private keys are known as a key pair.

To log in to your instance, you must create a key pair, specify the name of the key pair when you launch the instance, and provide the private key when you connect to the instance. Linux instances have no password, and you use a key pair to log in using SSH. With Windows instances, you use a key pair to obtain the administrator password and then log in using RDP.

A security group acts as a virtual firewall that controls the traffic for one or more instances. When you launch an instance, you associate one or more security groups with the instance. You add rules to each security group that allow traffic to or from its associated instances. You can modify the rules for a security group at any time; the new rules are automatically applied to all instances that are associated with the security group.

Understand AWS use of encryption of data in transit. All service endpoints support encryption of data in transit via HTTPS.

Know which services offer encryption of data at rest as a feature. The following services offer a feature to encrypt data at rest:

- Amazon S3
- Amazon EBS
- Amazon EMR

- AWS Snowball
- Amazon Glacier
- AWS Storage Gateway
- Amazon RDS
- Amazon Redshift
- Amazon WorkSpaces

Test Taking Tip

Clear your head. Studying is critical, but cramming all night long won't improve your score; in fact you are just as likely to hurt your score due to exhaustion clouding your thinking. Get a good night's rest and eat a healthy snack before your exam—it will go a long way to clearing your head.

Exercises

By now you have set up an account in AWS. If you haven't already, now would be the time to do so. It is important to note that these exercises are in your AWS account and thus are not free.

Use the Free Tier when launching resources. The AWS Free Tier applies to participating services across the following AWS Regions: US East (Northern Virginia), US West (Oregon), US West (Northern California), Canada (Central), EU (London), EU (Ireland), EU (Frankfurt), Asia Pacific (Singapore), Asia Pacific (Tokyo), Asia Pacific (Sydney), and South America (Sao Paulo). For more information, see https://aws.amazon.com/sdm/optimization/server-side-test/free-tier/free_np/.

These exercises assume that you have installed the AWS Command Line Utilities. Refer to Chapter 2, "Working with AWS Cloud Services," Exercise 2.1 (Linux) or Exercise 2.2 (Windows) if you need to install the AWS Command Line Interface (AWS CLI).

The reference for the AWS Command Line Interface can be found at: <http://docs.aws.amazon.com/cli/latest/reference/>.

These exercises will create AWS Identity and Access Management Users, Groups, Policies, and Roles. To complete the exercises, you research the "how to". By looking up how to perform a specific task, you will be on your way to mastering the task. The goal of this book isn't just to prepare you to pass the AWS Certified SysOps Administrator – Associate exam, rather this book should serve as a reference companion in your day-to-day duties as an AWS Certified SysOps Administrator.

In Exercise 3.1, you will create three IAM users. There are two ways to accomplish this task:

1. AWS Management Console (Firefox or Chrome browsers)
2. The AWS CLI command

You will use the AWS CLI for Exercises 3.1, 3.2, and 3.3.

EXERCISE 3.1**Creating AWS Identity and Access Management (IAM) Users**

Create three IAM user accounts with the user names **Alice**, **Bob**, and **Charlie** with the AWS CLI using the `aws iam create-user` command example:

```
aws iam create-user --user-name Alice
```

Take note of the information returned:

```
{
  "User": {
    "UserName": "Alice",
    "Path": "/",
    "CreateDate": "2017-04-25T00:54:38.139Z",
    "UserId": "AIDAI3QWSE7XYB6ERAQVC",
    "Arn": "arn:aws:iam::123456789012:user/Alice"
  }
}
```

EXERCISE 3.2**Create IAM Credentials.**

Now you will create credentials for the three IAM users that you created. You will use the `aws iam create-login-profile` and the `aws iam create-access-key` commands to complete this exercise.

Alice will need to access the AWS Management Console and the AWS CLI. Therefore she will need to have a password and access key generated. See the following examples for the CLI commands and related outputs.

1. Create a password for Alice.
2. Generate access keys for Alice.

```
aws iam create-login-profile --user-name Alice --password Fluffy@meow1
{
  "LoginProfile": {
    "UserName": "Alice",
    "CreateDate": "2017-04-25T01:01:53.417Z",
    "PasswordResetRequired": false
  }
}
```

EXERCISE 3.2 (continued)

```
aws iam create-access-key --user-name Alice

{
    "AccessKey": {
        "UserName": "Alice",
        "Status": "Active",
        "CreateDate": "2017-04-25T01:06:07.127Z",
        "SecretAccessKey": "bldyMBCKLFGCgizUM9q3exXnbljOhhlrBSRx32WF",
        "AccessKeyId": "AKIAIMGMTCVJV53AGT5A"
    }
}
```

3. Create a password for Bob. He will only have access to the AWS Management Console—no access keys are required.
 4. Charlie will be your Amazon EC2 and Amazon S3 administrator. He needs access to the AWS CLI, but does not need access to the AWS Management Console.
-

EXERCISE 3.3**Create IAM Groups.**

In this exercise, you will create IAM groups and add your IAM users to the groups. You will use the `aws iam create-group` and `aws iam add-user-to-group` commands to complete this exercise.

Create the following:

| Groups | Users |
|-----------|---------|
| admins | Alice |
| monitors | Bob |
| operators | Charlie |

```
aws iam create-group --group-name admins
{
    "Group": {
        "Path": "/",
        "CreateDate": "2017-04-25T01:22:20.505Z",
        "GroupName": "admins"
    }
}
```

```
        "GroupId": "AGPAIKTFN6QVR4M226BYA",
        "Arn": "arn:aws:iam::123456789012:group/admins",
        "GroupName": "admins"
    }
}

aws iam add-user-to-group --group-name admins --user-name Alice
```

EXERCISE 3.4

Working with IAM Policies

This exercise focuses on using AWS Managed Policies as the basis of the permissions that you set for the IAM groups. In this exercise, you will use the AWS Management Console.

1. Log in to the console with a user who has IAM permissions. For this exercise, the root account will work just fine, as you are creating your IAM accounts. After you have created an IAM account with admin access (for example, Alice) you can put the root account to rest.
 2. Go to the IAM console.
 3. Navigate to Groups.
 4. Select the Admin group.
 5. Go to the permissions tab, and attach the AdministratorAccess policy.
 6. Click on Show Policy to verify that you have administrator access.
 7. Perform the same steps for the monitors group, and assign ReadOnlyAccess.
 8. Perform the same steps for the operators group, and assign the AmazonEC2FullAccess policy.
-

EXERCISE 3.5

Working with IAM Roles

So far you have created IAM user accounts, IAM groups, and attached IAM policies to those groups.

Now you will create a role that will allow an Amazon EC2 instance acting as a bastion host in order to interact with the EC2 and Amazon S3 services. The bastion host will be able to create, modify, or terminate any EC2 instance and interact with S3 resources—all without having to store credentials on the bastion host.

1. Log in to the AWS Management Console.
2. Go to the IAM console.

EXERCISE 3.5 (*continued*)

3. Click on Dashboard.
 4. Take note of the IAM Users Sign-In link.
 5. Click on the link, and sign in as Alice.
 6. Go back to the IAM console.
 7. Navigate to Roles, and select the Amazon EC2 AWS Service Role.
 8. Create a new role, and attach the AmazonEC2FullAccess and AmazonS3FullAccess policy to this role.
 9. Name your IAM Role **bastionHost**. (Your role will be tested in the Chapter 4 exercises.)
-

Review Questions

- 1.** Whose responsibility is it to secure the AWS Cloud?
 - A.** Only Amazon Web Services
 - B.** Only you
 - C.** The World Wide Web Consortium (W3C)
 - D.** You and AWS share the responsibility.
- 2.** For which aspects of physical and environmental security is Amazon Web Services responsible?
 - A.** Fire detection and suppression
 - B.** Power redundancy
 - C.** Climate and temperature control in AWS datacenters
 - D.** All of the above
- 3.** True or False: The AWS network provides protection against traditional network security issues.
 - A.** True
 - B.** False
- 4.** Which AWS service provides centralized management of access and authentication of users administering the services in an AWS account?
 - A.** AWS Directory Service
 - B.** AWS Identity and Access Management Service
 - C.** Amazon Cognito
 - D.** AWS Config
- 5.** Which credentials can an IAM user have in order to access AWS services via the AWS Management Console and the AWS Command Line Interface (AWS CLI)? (Choose two.)
 - A.** Key pair
 - B.** User name and password
 - C.** Email address and password
 - D.** Access keys
- 6.** True or False: A password policy can be set in IAM that requires at least two lowercase letters and at least two non-alphanumeric characters.
 - A.** True
 - B.** False

7. The IAM access keys used to access AWS services via the AWS Command Line Interface (AWS CLI) and/or AWS Software Development Kits (SDK) consist of which two parts?
 - A. Access Key ID and password
 - B. Public Access Key and Secret Access Key
 - C. Access Key ID and Secret Access Key
 - D. User name and Public Access Key
8. Which Multi-Factor Authentication devices does the IAM service support?
 - A. Hardware devices (Gemalto)
 - B. Virtual MFA applications (for example, Google Authenticator)
 - C. Simple Message Service (SMS) (via mobile devices)
 - D. All of the above
9. Which of the following is true when using AWS Identity and Access Management groups?
 - A. IAM users are members of a default user group.
 - B. Groups can be nested.
 - C. An IAM user can be a member of multiple groups.
 - D. IAM roles can be members of a group.
10. Which of the following is *not* a best practice for securing an AWS account?
 - A. Requiring Multi-Factor Authentication for root-level access
 - B. Creating individual IAM users
 - C. Monitoring activity on your AWS account
 - D. Sharing credentials to provide cross-account access
11. Which of the following is true when using AWS Key Management Service (AWS KMS)?
 - A. All API requests to AWS KMS must be made over HTTP.
 - B. Use of keys is protected by access control policies defined and managed by you.
 - C. An individual AWS employee can access a Customer Master Key (CMK) and export the CMK in plaintext.
 - D. An AWS KMS key can be used globally in any AWS Region.
12. The AWS CloudTrail service provides which of the following?
 - A. Logs of the API requests for AWS resources within your account
 - B. Information about the IP traffic going to and from network interfaces
 - C. Monitoring of the utilization of AWS resources within your account
 - D. Information on configuration changes to AWS resources within your AWS account

- 13.** Amazon CloudWatch Logs enable Amazon CloudWatch to monitor log files. Pattern filtering can be used to analyze the logs and trigger Amazon CloudWatch alarms based on customer specified thresholds. Which types of log files can be sent to Amazon CloudWatch Logs?
- A.** Operating system logs
 - B.** AWS CloudTrail Logs
 - C.** Access Flow Logs
 - D.** All of the above
- 14.** AWS CloudTrail logs the API requests to AWS resources within your account. Which other AWS service can be used in conjunction with CloudTrail to capture information about changes made to AWS resources in your AWS account?
- A.** Auto Scaling
 - B.** AWS Config
 - C.** Amazon VPC Flow Logs
 - D.** AWS Artifact
- 15.** True or false: Amazon Inspector continuously monitors your AWS account's configuration against the Well Architected Framework's best practice recommendations for security.
- A.** True
 - B.** False
- 16.** A workload consisting of Amazon EC2 instances is placed in an Amazon VPC. What feature of VPC can be used to deny network traffic based on IP source address and port number?
- A.** Subnets
 - B.** Security groups
 - C.** Route tables
 - D.** Network Access Control Lists
- 17.** You want to pass traffic securely from your on-premises network to resources in your Amazon VPC. Which type of gateway can be used on the VPC?
- A.** Internet Gateway (IGW)
 - B.** Amazon Virtual Private Cloud endpoint
 - C.** Virtual Private Gateway
 - D.** Amazon Virtual Private Cloud peer
- 18.** To protect data at rest within Amazon DynamoDB, customers can use which of the following?
- A.** Client-side encryption
 - B.** TLS connections
 - C.** Server-side encryption provided by the Amazon DynamoDB service
 - D.** Fine-grained access controls

- 19.** When an Amazon Relational Database Service database instance is run within an Amazon Virtual Private Cloud, which Amazon VPC security features can be used to protect the database instance?
- A.** Security groups
 - B.** Network ACLs
 - C.** Private subnets
 - D.** All of the above
- 20.** Which of the following is correct?
- A.** Amazon SQS and Amazon SNS encrypt data at rest.
 - B.** Amazon SQS and Amazon SNS do not encrypt data at rest.
 - C.** Amazon SQS encrypts data at rest and Amazon SNS does not encrypt data at rest.
 - D.** Amazon SQS does not encrypt data at rest and Amazon SNS encrypts data at rest.

Chapter

4



Compute

THE AWS CERTIFIED SYSOPS ADMINISTRATOR - ASSOCIATE EXAM TOPICS COVERED IN THIS CHAPTER MAY INCLUDE, BUT ARE NOT LIMITED TO, THE FOLLOWING:

Domain 2.0 High Availability

- ✓ 2.1 Implement scalability and elasticity based on scenario

Content may include the following:

- Which AWS compute service to use
- Scalability of specific AWS compute service
- Elasticity of specific AWS compute service

- ✓ 2.2 Ensure level of fault tolerance based on business needs

Content may include the following:

- What AWS Cloud services are Availability Zone based and the implication for high availability
- What AWS Cloud services are region-based services and the implication for high availability

Domain 4.0 Deployment and Provisioning

- ✓ 4.1 Demonstrate the ability to build the environment to conform to architectural design

Content may include the following:

- Choosing the appropriate AWS Cloud service to meet the business need
- How to achieve high availability in compute services



✓ **4.2 Demonstrate the ability to provision cloud resources and manage implementation automation**

Content may include the following:

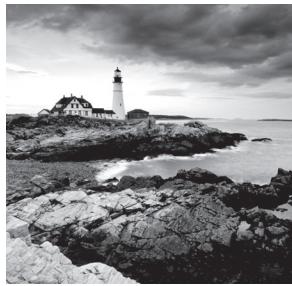
- Steps in deployment of various AWS Cloud services and key decisions regarding scaling and availability

Domain 6.0 Security

✓ **6.3 Demonstrate understanding of the shared responsibility model**

Content may include the following:

- Use of security groups to control access to compute instances
- Use of public/private keys to control access to guest operating system



Introduction to AWS Compute Services

Amazon Elastic Compute Cloud (Amazon EC2) was the third service introduced by AWS. It followed Amazon Simple Queue Service (Amazon SQS) and Amazon Simple Storage Service (Amazon S3). Amazon EC2 was announced in 2006, as a way to provide on-demand computing. Since that initial introduction, AWS has worked to expand Amazon EC2 and has incorporated a Graphical User Interface (GUI)-based management console, persistent storage in the form of both magnetic drives and Solid State Drives (SSDs), CPUs optimized for different types of compute loads, and enhanced networking capabilities.

AWS has also expanded the meaning of compute. Today, compute includes the ability to spin up any number of instances in the cloud on a pay-as-you-go basis, and it also incorporates some of the following services:

AWS Lambda This is a compute service that runs code without the need to provision or manage servers. Essentially, AWS Lambda acquires CPU cycles without acquiring the actual CPU. AWS Lambda executes code on demand and scales automatically. This scaling can range from a few requests per day to thousands per second.

AWS Elastic Beanstalk This is a service that enables you to quickly deploy and manage applications on the AWS Cloud without worrying about the infrastructure that runs those applications. AWS Elastic Beanstalk reduces management complexity without restricting choice or control. You upload your application, and AWS Elastic Beanstalk handles the details of capacity provisioning, load balancing, scaling, and application health monitoring.

Amazon EC2 Container Service (Amazon ECS) This is a highly scalable and fast container management service that makes it easy to run, stop, and manage Docker containers on a cluster of Amazon EC2 instances.

Amazon Lightsail This service is for developers who need virtual private servers. Amazon Lightsail includes everything you need to launch your project quickly—a virtual machine, SSD-based storage, data transfer, Domain Name System (DNS) management, and a static IP address—for a low, predictable price.

AWS Batch This service enables you to run batch computing workloads on the AWS Cloud. Batch computing is a common way for developers, scientists, and engineers to access large amounts of compute resources. AWS Batch removes the undifferentiated heavy lifting of configuring and managing the required infrastructure.

In this chapter, you learn about the various compute products and services described here, how to implement them, how to manage them, and how to secure them. It focuses on the following:

- Launching an Amazon EC2 instance
- The pricing model for Amazon EC2
- What is Amazon EC2 instance metadata and how to find it
- What is user data, how is it loaded into the Amazon EC2 instance, and how to find it
- How to create an Amazon Machine Image (AMI)
- Knowing when you would use a dedicated host and when you would use a dedicated instance
- The different CPU types available in Amazon EC2 and how to change instance types on running instances
- The different storage options available for Amazon EC2
- Understanding how instance size affects network speed
- Understanding how to log in remotely to an Amazon EC2 instance
- Knowing what Amazon EC2 metrics are monitored in Amazon CloudWatch
- How to configure an AWS Lambda function
- The pricing model for Amazon Lambda
- What AWS Lambda metrics Amazon CloudWatch monitors
- Understanding how to control CPU choice in AWS Lambda
- Understanding the use cases for AWS Elastic Beanstalk, Amazon Lightsail, and AWS Batch
- What is an AWS Lambda event and what are supported AWS Lambda event sources
- What services discussed in this chapter are considered highly available and what services are not
- What services AWS Elastic Beanstalk spins up by default
- What tools are available to configure the services mentioned in this chapter
- Knowing the different tools available to monitor Amazon EC2, Amazon ECS, and AWS Lambda
- Knowing the different tools available to manage security for Amazon EC2
- Understanding the shared responsibility model and how it applies to these services: Amazon EC2, Amazon ECS, AWS Lambda, and AWS Elastic Beanstalk
- Understanding what security groups do and how they protect instances
- Understanding the basic process for setting up Amazon ECS
- Understanding when you would use Amazon EC2 and when you would use Amazon Lightsail
- Knowing what services Amazon Lightsail spins up by default

Amazon Elastic Compute Cloud (Amazon EC2)

Amazon EC2 provides scalable computing capacity in the AWS Cloud. Amazon EC2 eliminates your need to invest in hardware, so that you can develop and deploy applications faster. You can use Amazon EC2 to launch as many or as few virtual servers as you need, configure security and networking, and manage storage. You launch your instances in the AWS Region and Availability Zone that you specify.

Amazon EC2 provides the following features:

- Virtual computing environments, known as *instances*
- Preconfigured templates for your instances, known as *AMIs*, that package the components you need for your server (including the operating system and additional software)
- Various configurations of CPU, memory, storage, and networking capacity for your instances, known as *instance types*
- Secure login information for your instances using *key pairs* (AWS stores the public key, and you store the private key in a secure place.)
- Storage volumes for temporary data that's deleted when you stop or terminate your instance, known as *instance store volumes*
- Persistent storage volumes for your data using Amazon Elastic Block Store (Amazon EBS), known as *Amazon EBS volumes*
- Multiple physical locations for your resources, such as instances and Amazon EBS volumes, known as *regions* and *Availability Zones*
- A firewall that enables you to specify the protocols, ports, and source IP ranges that can reach your instances using security groups
- Static IPv4 addresses for dynamic cloud computing that can be assigned to an individual instance, known as *Elastic IP addresses*
- IPv6 addresses that can also be assigned to instances
- Virtual networks that you can create that are logically isolated from the rest of the AWS Cloud and that you can optionally connect to your own network, known as *Amazon Virtual Private Clouds (Amazon VPCs)*

Implementation

In implementing an Amazon EC2 instance, you go through the following steps:

1. Decide on the AMI that you want to use.
2. Choose the instance type and size that you need.
3. Provide other needed configuration details.

This section goes through each of these steps in greater detail.

Choosing an AMI

Decide on the AMI that you want to use. An AMI provides the information required to launch an instance. An AMI includes the following:

- A template for the root volume for the instance (perhaps an operating system, an application server, and applications)
- Launch permissions that control which AWS accounts can use the AMI to launch instances
- A block device mapping that specifies the volumes to attach to the instance when it is launched

You can obtain AMIs from one of four places:

- Provided by AWS
- AWS Marketplace
- You can create and manage your own AMIs.
- A shared AMI

With AMIs provided by AWS, you have a variety of choices of either Linux or Windows operating systems. With the Windows AMI, the cost of the operating system is included in the hourly charge for the instance.

AWS Marketplace offers a wide variety of AMIs from a number of third parties. You can choose AMIs in categories such as software infrastructure, developer tools, or business software from a number of vendors, including Juniper, Bitnami, and SAP.

AWS also gives you the ability to create and manage your own AMIs. For example, you can take an existing AWS-provided Linux AMI, modify it to meet your needs, and save that AMI. You can then use that AMI as the basis for compute infrastructure. This is the route most companies take when choosing an AMI.

A shared AMI is an AMI that a developer created and made available for other developers to use. You can create your own AMIs and share them with others. Note that you use a shared AMI at your own risk—Amazon can't vouch for the integrity or security of AMIs shared by other Amazon EC2 users. Therefore, you should treat shared AMIs as you would any foreign code that you might consider deploying in your own datacenter and perform the appropriate due diligence. Only use an AMI from a trusted source.

Choosing an Instance Type

After choosing an AMI, you then need to choose an instance type. The instance type that you specify determines the hardware of the host computer used for your instance. Each instance type offers different compute, memory, and storage capabilities. Instance types are grouped in instance families based on these capabilities.

Amazon EC2 provides each instance with a consistent and predictable amount of CPU capacity, regardless of its underlying hardware. Amazon EC2 dedicates some resources of the host computer (for example, CPU, memory, and instance storage) to a particular instance. Amazon EC2 shares other resources of the host computer (such as the network and the disk subsystem) among instances.

The choice of the instance type affects the following attributes:

- CPU specialization
- Virtualization types
- Network features
- Storage features
- Instance limits

With CPU specialization, CPUs can be classified as being optimized in one of the following ways:

- General Purpose (T2, M4)
- Compute-Optimized (C3, C4)
- Memory-Optimized (X1, R3, R4)
- Storage-Optimized (I3, D2)
- Accelerated Computing (P2, G2, F1)

As AWS continues to introduce new instance families and new generations of instance families, the most up-to-date information can be found in the Amazon EC2 User Guide.

Each instance type supports one or both of the following types of virtualization: paravirtual or Hardware Virtual Machine (HVM). The virtualization type of your instance is determined by the AMI that you use to launch it.

For best performance, you may want to use an HVM AMI. In addition, HVM AMIs are required for enhanced networking. HVM virtualization uses hardware-assisted technology provided by the AWS platform. With HVM virtualization, the guest Virtual Machine (VM) runs as if it were on a native hardware platform, except that it still uses paravirtual network and storage drivers for improved performance.

Network features are also influenced by the instance type you choose. Network speed is primarily a function of the size of the instance. Larger instances provide greater network speed than smaller instances in the same family. Certain instance types offer support for enhanced networking. Enhanced networking uses Single Root I/O Virtualization (SR-IOV) to provide high-performance networking capabilities. There are also instance families that offer 10G connectivity. Support of jumbo frames (frames greater than 1,500 Maximum Transmission Unit [MTU] in size) is also dependent on the instance type chosen. More information about networking on AWS can be found in Chapter 5, “Networking.”

Storage features are also influenced by the instance type you choose. Some instance types support Amazon EBS volumes and instance store volumes, while other instance types support only Amazon EBS volumes. Some instances that support instance store volumes use SSDs to deliver very high random I/O performance. To obtain additional, dedicated capacity for Amazon EBS I/O, you can launch some instance types as Amazon EBS-optimized instances. Some instance types are Amazon EBS-optimized by default. Refer to Chapter 6, “Storage,” for a more comprehensive look at storage on AWS.

There is a limit on the total number of instances that you can launch in an AWS Region, and there are additional limits on some instance types. You can use the Amazon EC2 Service Limits page in the Amazon EC2 console to view the current limits for resources provided by Amazon EC2 and Amazon VPC on a per-region basis.

Other Configuration Details

Once you choose the AMI and the instance type, there are a number of other configuration details that you need to provide. These details include the following:

- Number of instances
- Purchasing options
- Amazon VPC to be used
- Subnet to be located in
- Public IP address
- AWS Identity and Access Management (IAM) role
- Shutdown behavior
- Monitoring
- Tenancy
- User data
- Storage type and volume
- Tagging
- Security groups
- Public/private key

You are able spin up as many instances as you want simultaneously, subject to the service limits mentioned.

AWS purchasing options allow you to request Spot Pricing. If you request Spot Pricing, you are sent to an additional screen where you can configure Availability Zone, maximum price, launch group (if applicable), and whether this is a persistent request. If it's not a persistent request, you specify what time and date the request is valid to and from.

All Amazon EC2 instances need to be spun up in an Amazon VPC. You can choose the Amazon VPC in which the instance is spun up. If you have not already created an Amazon VPC, then the instance will be spun up in the default Amazon VPC. Similarly, because Amazon EC2 is an Availability Zone-based service, you need to choose the Availability Zone in which to locate the Amazon EC2 instance. The private IP addresses assigned to the Amazon EC2 instance is determined by what Amazon VPC and what subnet you locate the instance in.

You can choose to assign a public IP address to your instance. Doing so will potentially make this Amazon EC2 instance accessible from the Internet (the other requirement is that the Amazon VPC containing the Amazon EC2 instance needs to have an Internet gateway). Public IP addresses are associated with specific interfaces, so if you have multiple interfaces on your Amazon EC2 instance, you can have multiple public IP addresses. The public IP address remains attached to the interface as long as the Amazon EC2 instance is running. If the Amazon EC2 instance is rebooted, it keeps the public IP address. If the Amazon EC2 instance is turned off, the public IP address is removed. If the Amazon EC2 instance is turned back on, a new public IP address is assigned.

Both IPv4 and IPv6 are supported in Amazon EC2.

IAM roles are discussed in more details in Chapter 3, “Security and AWS Identity and Access Management (IAM).” You can assign an IAM role to an Amazon EC2 instance as part of a configuration. This IAM role would give the Amazon EC2 instance permission to access other AWS Cloud services (such as when the Amazon EC2 instance needs to access an Amazon S3 bucket).

Shutdown behavior controls the conduct of the Amazon EC2 instance after it receives a shutdown command. Amazon EC2 instances that have instance storage can only be in a running state or be terminated—they cannot be stopped. Amazon EC2 instances that have Amazon EBS storage can be running, stopped, or terminated. The default behavior of these instances is to delete the attached storage after the instance is terminated.

Termination protection, when enabled, prevents the accidental termination of instances. After termination protection is enabled, you must disable it if you wish to terminate an instance. This gives you an extra level of protection against accidental termination.

Amazon EC2’s default mode is shared tenancy. The other choices are to run as a dedicated host or a dedicated instance. Dedicated hosts allow you to use your existing per-socket, per-core, or per-VM software licenses. Rebooting a dedicated host will reboot the instance on the same physical AWS infrastructure. Dedicated instances are Amazon EC2 instances that run in an Amazon VPC on hardware that is dedicated to a single customer. Your dedicated instances are physically isolated at the host hardware level from instances that belong to other AWS accounts.

User data is that which you can pass to the instance both upon initial boot up and, if desired, in subsequent boots. User data allows you to specify parameters for configuring your instance or attach a simple script. You can also use this data to build more generic AMIs that can be modified by configuration files supplied at launch time. For example, you can change a basic AWS Linux AMI to become a web server using user data. User data is limited to 16 KB, but it can have links to locations (like an Amazon S3 bucket) to load larger files.

Different instance types will support different storage options. The three major types of storage for Amazon EC2 are instance storage, Amazon EBS, and Amazon Elastic File System (Amazon EFS).

Instance storage is where the disks are physically attached to the host computer. Instance storage provides temporary block-level storage for instances. The data on an instance store volume persists only during the life of the associated instance; if you stop or terminate an instance, any data on instance store volumes is lost. Note that not all Amazon EC2 instance types support instance storage. Instance storage cannot be modified in terms of the amount of storage. Pricing for instance storage is included in the pricing for the Amazon EC2 instance.

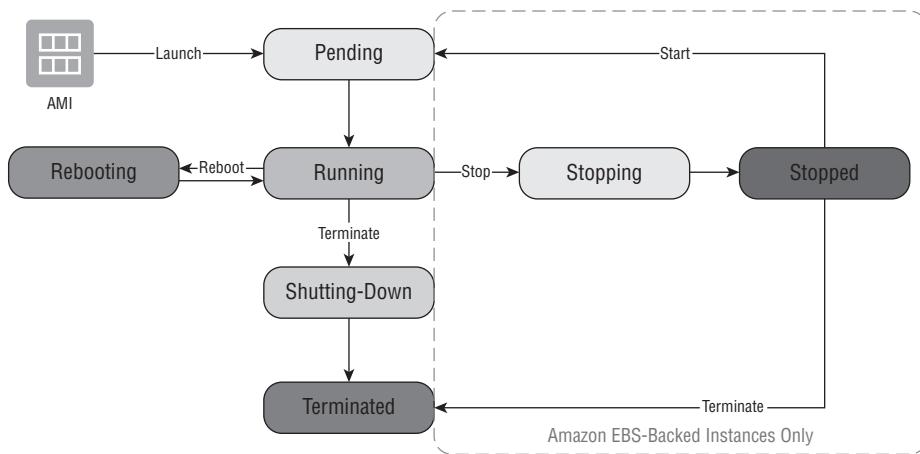
Amazon EBS provides block-level storage volumes for use with Amazon EC2 instances. Amazon EBS volumes are highly available and reliable storage volumes that can be attached to any running instance that is in the same Availability Zone. Amazon EBS volumes that are attached to an Amazon EC2 instance are exposed as storage volumes that persist independently from the life of the instance. Amazon EBS volumes can be in a variety of media (magnetic or SSD), can be in a variety of sizes, can be encrypted, and have specified I/O

speeds (with Provisioned IOPS). Because Amazon EBS exists as a separate service, it is priced independently of the Amazon EC2 instance to which it is attached.

Amazon EBS is recommended when data must be quickly accessible and requires long-term persistence. Amazon EBS volumes are particularly well-suited for use as the primary storage for file systems, databases, or for any applications that require fine granular updates and access to raw, unformatted, block-level storage. Amazon EBS is well suited to both database-style applications that rely on random reads and writes and to throughput-intensive applications that perform long, continuous reads and writes.

Another key difference between instance store-backed Amazon EC2 instances and Amazon EBS-backed Amazon EC2 instances is the ability to stop and start the Amazon EC2 instance. With Amazon EBS-backed Amazon EC2 instances, you can stop and start the Amazon EC2 instance. With instance store-backed Amazon EC2 instances, you can only delete the instance. Figure 4.1 illustrates the differences between an instance store-backed Amazon EC2 instance and an Amazon EBS-backed Amazon EC2 instance.

FIGURE 4.1 Differences between an instance store-backed Amazon EC2 instance and an Amazon EBS-backed Amazon EC2 instance



Amazon EFS provides scalable file storage for use with Amazon EC2. You can create an Amazon EFS file system and configure your instances to mount the file system. You can use an Amazon EFS file system as a common data source for workloads and applications running on multiple instances. Because Amazon EFS exists as a separate service, it is priced independently of the Amazon EC2 instance to which it is attached. It is important to note that Amazon EFS can only be used with Linux-based Amazon EC2 instances.

Tags enable you to categorize your Amazon EC2 instances in different ways (for example, by purpose, owner, or environment). This is useful when you have many resources of the same type—you can quickly identify a specific resource based on the tags you have assigned to it. Each tag consists of a *key* and a *value*, both of which you define. You can work with tags using the AWS Management Console, the AWS Command Line Interface (AWS CLI), and the Amazon EC2 Application Programming Interface (API).

Security groups are applied to the interface of an Amazon EC2 instance. Security groups allow access via port, protocol, and source. A source can be an IP address, a range of IP addresses, or another security group. You cannot set up a security group with a deny rule. Security groups control both inbound and outbound access, but the recommended practice is to use them mostly to control inbound access. By default, security groups allow all outbound access. In addition, security groups are stateful; if you send a request from your instance, the response traffic for that request is allowed to flow in regardless of inbound security group rules. You must attach a security group with either Secure Shell (SSH) or Remote Desktop Protocol (RDP) access if you intend to be able to log in to the guest operating system.

In order to protect your Amazon EC2 instance, Amazon EC2 uses public-key cryptography to encrypt and decrypt login information. To log in to your instance, you must create a key pair, specify the name of the key pair when you launch the instance, and provide the private key when you connect to the instance. Linux instances have no password, and you use a key pair to log in using SSH. With Windows instances, you use a key pair to obtain the administrator password and then log in using RDP. You can use Amazon EC2 to create your key pair, or you could use a third-party tool and then import the public key to Amazon EC2.

Management

The decision to pick an instance store-backed Amazon EC2 instance or an Amazon EBS-backed Amazon EC2 instance has implications regarding both behavior and performance. Table 4.1 summarizes some of these differences.

TABLE 4.1 Instance Store-Backed Amazon EC2 Instance vs. an Amazon EBS-Backed Amazon EC2 Instance

| Feature | Amazon EBS-Backed Amazon EC2 Instance | Instance Store-Backed Amazon EC2 Instance |
|--|---|---|
| Boot times | Boots faster | Takes longer to boot |
| Ability to reboot | Can reboot | Can reboot |
| Ability to stop | Can stop instance | Not able to stop instance, only terminate |
| Ability to resize | Able to resize by stopping instance and choosing different instance size | Not able to resize (not able to stop instance) |
| Ability to change CPU family | Able to change CPU family by stopping instance and choosing different CPU family size | Not able to change CPU family (not able to stop instance) |
| Ability to add or remove security groups from interfaces | Able to do so while the Amazon EC2 instance is stopped, running, or rebooting | Able to do so while the Amazon EC2 instance is running or rebooting |

Amazon EC2 instances that use instance storage take longer to boot than Amazon EC2 instances that use Amazon EBS. Another impact to boot times is the use of user data when initially booting an instance. Having user data slows the boot process down. Using a fully configured AMI (which you can create) speeds up the boot process. Note that a fully configured AMI, because it is hard coded, is less flexible in terms of modification (you have to create a completely new AMI), so the trade-off is between speed of booting the instance and the flexibility of configuration.

Instance metadata is data about your instance that you can use to configure or manage the running instance. Metadata can be accessed via <http://169.254.169.254/>. Amazon EC2 instances can also include dynamic data, such as an instance identity document that is generated when the instance is launched and exposed to the instance through instance metadata. It validates the attributes of the instances, such as the subscribed software, instance size, instance type, operating system, and AMI. It can be accessed via <http://169.254.169.254/latest/dynamic/instance-identity/document>.

You can also access the user data that you supplied when launching your instance. For example, you can specify parameters for configuring your instance, downloading patches, installing software, or attaching a simple script. You can also use this data to build more generic AMIs that can be modified by configuration files supplied at launch time. For example, if you run web servers for various small businesses, they can all use the same AMI and retrieve their content from the Amazon S3 bucket that you specify in the user data at launch.



Although you can only access instance metadata and user data from within the instance itself, the data is not protected by cryptographic methods. Anyone who can access the instance can view its metadata. Therefore, you should take suitable precautions to protect sensitive data (such as long-lived encryption keys). You should not store sensitive data, such as passwords, as user data.

Snapshots

You can back up the data on your Amazon EBS volumes to Amazon S3 by taking point-in-time snapshots. Snapshots are incremental backups, which means that only the blocks on the device that have changed after your most recent snapshot are saved. This minimizes the time required to create the snapshot and saves on storage costs. When you delete a snapshot, only the data unique to that snapshot is removed. Active snapshots contain all of the information needed to restore your data (from the time the snapshot was taken) to a new Amazon EBS volume.



Knowing both the performance and operating characteristics of the different types of block storage that is available with Amazon EC2 is vital as an AWS systems operator.

Monitoring the Status of Your AWS Cloud Services

AWS provides two dashboards that show the status of AWS Cloud services. The first is the *AWS Service Health Dashboard*. The dashboard displays service health across the entire AWS infrastructure, and you can personalize it to track your AWS Cloud services. With the AWS Service Health Dashboard, you can view the service health for all services across all regions (with the exception of the AWS GovCloud [US] Region and China [Beijing] Region). You are also able to subscribe to RSS feeds for chosen services across regions. In addition, you can view past system statuses for all services in all regions (again with the exception of the AWS GovCloud [US] Region and China [Beijing] Region).

The second dashboard is the *AWS Personal Health Dashboard*. The AWS Personal Health Dashboard provides information about AWS health events that can affect your account. The information is presented in two ways: a dashboard that shows recent and upcoming events organized by category and a full event log that shows all events from the past 90 days. AWS Personal Health Dashboard uses the Health API, which you can use to create your own health dashboards or feed into any current management systems you may have.

Monitoring with Amazon CloudWatch

Amazon CloudWatch monitoring is turned on by default for Amazon EC2 instances. Basic monitoring, which collects metrics every five minutes, is provided by default. You can enable detailed metrics that would collect metrics every minute. Amazon CloudWatch monitors the following metrics for Amazon EC2:

- CPU utilization
- Disk reads
- Disk writes
- Network in
- Network out
- Instance status check
- System status check
- CPU credit usage (T2 instances only)
- CPU credit balance (T2 instances only)

You can view Amazon CloudWatch metrics based on a particular instance, a particular image, membership in an Auto Scaling group, or as an aggregate. Amazon CloudWatch can be used to trigger an alarm that can stop, terminate, reboot, or recover an Amazon EC2 instance.

Using AWS CloudTrail

You can use *AWS CloudTrail* to get a history of AWS API calls and related events for your account. This includes calls made using the AWS Management Console, AWS Software Development Kits (SDKs), command-line tools, and higher-level AWS Cloud services. When

AWS CloudTrail logging is enabled, calls made to Amazon EC2 are tracked in log files, along with any other AWS Cloud service records. AWS CloudTrail determines when to create and write to a new file based on a specified time period and file size. All of the Amazon EC2 actions are logged. Every log entry contains information about who generated the request. The user identity information in the log helps you determine whether the request was made with root or IAM user credentials, with temporary security credentials for a role or federated user, or by another AWS Cloud service.



Understanding why and when you would use Amazon CloudWatch and AWS CloudTrail will not only help you in monitoring and managing Amazon EC2 instances, but it will also help you monitor and manage all of your AWS Cloud services.

Amazon EC2 Systems Manager

Amazon EC2 Systems Manager is a management service that helps you automatically collect software inventory, apply operating system patches, create system images, and configure Windows and Linux operating systems. Amazon EC2 Systems Manager can be used for both Amazon EC2 instances and for compute instances located in your own datacenters. These capabilities help you define and track system configurations, prevent drift, and maintain software compliance of your Amazon EC2 and on-premises configurations.

IP Addresses

When an Amazon EC2 instance is spun up, a private IP address is assigned to the interface of that Amazon EC2 instance. What private IP address is used for this assignment is determined by what Amazon VPC and what subnet the Amazon EC2 instance is spun up in. The private IP address will be assigned from that subnet block. That private IP address remains assigned to that interface, and it is only released once the Amazon EC2 instance is deleted. Stopping the instance will have no effect on the private IP address. Public and Elastic IP addresses behave differently. Both are also assigned to an interface and can be assigned to only one interface (though an interface can have multiple public and Elastic IP addresses assigned to it). When an Amazon EC2 instance with a public IP address assigned to it is deleted, the public IP is returned to AWS so it can be reassigned to another Amazon EC2 instance (most likely in another AWS account). In contrast, when an Amazon EC2 instance with an Elastic IP address is deleted, the Elastic IP address is not returned to AWS; it is kept by the account, available for reassignment within that particular AWS account.

Purchasing Options

There are a number of options for purchasing Amazon EC2 instances: On-Demand, Reserved, or Spot. On-Demand is straightforward—you pay for the Amazon EC2 instance as you use it. There are no up-front costs or commitments, the minimum billing increment is one hour, and you do not pay for Amazon EC2 instances that are stopped.

Reserved instances provide you with a significant discount compared to On-Demand instance pricing. Reserved instances are not physical instances; they are a billing discount applied to the use of On-Demand instances in your account. With Reserved instances, you make a commitment for a specific family of instances in a specific region or Availability Zone. You can pay for Reserved instances monthly (like On-Demand), or you can prepay the full or partial amount.

Spot instances enable you to bid on unused Amazon EC2 instances, which can lower your Amazon EC2 costs significantly. The hourly price for a Spot instance (of each instance type in each Availability Zone) is set by Amazon EC2, and it fluctuates depending on the supply of and demand for Spot instances. Your Spot instance runs whenever your bid exceeds the current market price. Spot instances are a cost-effective choice if you can be flexible about when your applications run and if your applications can be interrupted.

Estimating Costs

AWS provides two tools for estimating costs: the *AWS Simple Monthly Calculator* and the *AWS Total Cost of Ownership (TCO) Calculator*. The AWS Simple Monthly Calculator incorporates a wide array of pricing calculations across all AWS Cloud services in all AWS Regions. It also shows the breakdown of features for each service in each region. The AWS Simple Monthly Calculator has a mechanism to select a service in a particular region and add it to the total bill.

The AWS TCO Calculator allows you to estimate the cost savings when using AWS and provides a detailed set of reports that can be used in executive presentations. The AWS TCO Calculator also gives you the option to modify assumptions that best meet your business needs.

Tracking Costs

AWS provides AWS Trusted Advisor and My Billing Dashboard for managing and tracking costs. The tools can be found in the AWS Management Console. *AWS Trusted Advisor* gives you recommendations on how to potentially reduce your spending with AWS. The *My Billing Dashboard* is specifically dedicated to your charges and bills. With My Billing Dashboard, you can view and download bills, explore your bills (graph, visualize, and analyze), create budgets, and receive notifications based on those budgets. You can also create Cost Allocation Tags, which allow you to associate costs with resources that you create.

Creating Your Own AMIs

As discussed in the implementation section, you can use your own AMIs when choosing an AMI. The process of creating an AMI is different depending on whether the AMI you are creating is an instance store-backed AMI or if it is an Amazon EBS-backed AMI.

To create an instance store-backed AMI, first launch an instance from an AMI that is similar to the AMI that you would like to create. You can connect to your instance and customize it. When the instance is set up the way you want it, you can bundle it. It takes several minutes for the bundling process to complete. After the process completes, you have a bundle,

which consists of an image manifest (`image.manifest.xml`) and files (`image.part.xx`) that contain a template for the root volume. Upload the bundle to your Amazon S3 bucket and then register your AMI.

To create an Amazon EBS-backed AMI, first launch an instance from an AMI that is similar to the AMI that you would like to create. You can connect to your instance and customize it. After the instance is configured correctly, stop the instance (stopping the instance ensures data integrity of the instance; a snapshot will be taken of the instance). Next create the image. When you create an Amazon EBS-backed AMI, AWS automatically registers it for you.

Once you have created an AMI, you can control who has access to it. You can share the AMI among AWS accounts, and you can copy the AMI to other AWS Regions. Note that if the AMI is encrypted, you are not able to share it.

Placement Groups

A *placement group* is a logical grouping of instances in a single Availability Zone. Placement groups are recommended for applications that benefit from low network latency, high network throughput, or both. High Performance Computing (HPC) is an example of an application that would potentially benefit from being in a placement group. Because all instances are within a single Availability Zone, Amazon EC2 instances in the same placement group cannot, by definition, be either in different Availability Zones or in different regions. Amazon EC2 instances that are in the same placement group communicate with each other via either the private IPv4 address or the IPv6 address.

Security

Security in the Amazon EC2 environment is a responsibility shared by both the end user and AWS. This is because, within this environment, there are specific parts that AWS controls and specific parts that are controlled by the end user. Physical security and security of the Hypervisor is AWS responsibility. The end user, on the other hand, is responsible for securing the operating systems running on their instances as well as the applications running on those operating systems. You retain control of what security you choose to implement to protect your content, platform, applications, systems, and networks, no differently than you would for applications in an on-site datacenter.

Regarding Amazon EC2, you are responsible for controlling who has access to the machine (both network access and administrative access) and for keeping the guest operating system up-to-date and patched properly. If you so desire, you can apply additional layers of security (for example, running IP Tables or Windows Defender on the guest operating system) without needing authorization from AWS. AWS does not have any access rights to your instances or to the guest operating system.

Controlling Network Access

AWS provides a number of tools regarding network access: placement of the Amazon EC2 instance in an Amazon VPC (Amazon VPC security was discussed in Chapter 3) and security

groups to control access on a specific interface. With administrative access, you again can control access with placement of Amazon EC2 instances in an Amazon VPC, the use of security groups, and the use of public/private key pairs both to encrypt traffic and control access.

Controlling Administrative Access

AWS provides a number of tools to manage the security of your instances. These tools include the following:

- IAM
- AWS Trusted Advisor
- AWS Service Catalog
- Amazon Inspector

IAM allows you to create policies that control access to APIs and apply those policies to users, groups, and roles.

AWS Trusted Advisor, which comes as part of AWS Support, looks at things like open ports on security groups and level of access allowed to Amazon EC2 instances, and it makes recommendations to improve the security of your AWS infrastructure.

AWS Service Catalog allows IT administrators to create, manage, and distribute portfolios of approved products to end users who can then access the products they need in a personalized portal. AWS Service Catalog allows organizations to manage commonly deployed IT services centrally, and it helps organizations achieve consistent governance and meet compliance requirements while enabling users to deploy quickly only the approved IT services they need within the constraints the organization sets.

Amazon Inspector is a security vulnerability assessment service that helps improve the security and compliance of your AWS resources. Amazon Inspector automatically assesses resources for vulnerabilities or deviations from best practices and then produces a detailed list of security findings prioritized by level of severity.



AWS also has a number of developer documents, whitepapers, articles, and tutorials to help you in securing your environment. These are available on the Cloud Security Resources page of the AWS website.

Amazon EC2 Container Service (Amazon ECS)

Amazon ECS is a highly scalable container management service. Amazon ECS makes it easy to run, stop, and manage Docker containers on Amazon EC2 instances.

With Amazon ECS, you can launch and place containers across your cluster using API calls. You schedule the placement of containers based on your resource needs, isolation

policies, and availability requirements. You can use Amazon ECS both for cluster management and for scheduling deployments of containers onto hosts.

Amazon ECS eliminates the need for you to operate your own cluster management and configuration management systems. Amazon ECS can be used to manage and scale both batch and Extract, Transform, Load (ETL) workloads. It can also be used to build application architectures based on the microservices model.

Amazon ECS is a regionally-based service that can be used to run application containers in a highly available manner across all Availability Zones within an AWS Region.

Implementation

To implement Amazon ECS, you need to install an Amazon ECS agent on your Amazon EC2 instances. If you use Amazon ECS-optimized AMIs, that agent is already installed. Additionally, the container instance needs to have an IAM role that authenticates to your account and will need external network access to communicate with the Amazon ECS service endpoint.

Clusters are the logical grouping of container instances that you can place tasks on. Clusters are region specific and can contain multiple, different instance types (though an instance can only be a member of a single cluster).

Amazon ECS obtains a Docker image for repository. This repository can be on AWS or on other infrastructure.

Deploying a Container

To deploy a container, you need to do the following:

1. **Define a task.** This is where you assign the name, provide the image name (important for locating the correct image), and decide on the amount of memory and CPU needed.
2. **Define the service.** In this step, you decide how many instances of the task you want to run in the cluster and any Elastic Load Balancing load balancers that you want to register the instances with.
3. **Create the Amazon ECS cluster.** This is where the cluster is created and also where you specify the number of instances to run. The cluster can run across multiple Availability Zones.
4. **Create the stack.** A stack of instances is created based on the configuration information provided. You can monitor the creation of the stack in the AWS Management Console. Creation of the stack is usually completed in less than five minutes.

Management

Amazon ECS can be configured using the AWS Management Console, the AWS CLI, or the Amazon ECS CLI.

Monitoring Amazon ECS

The primary tool used for monitoring your Amazon ECS clusters is Amazon CloudWatch. Amazon CloudWatch collects Amazon ECS metric data in one-minute periods and sends them to Amazon CloudWatch. Amazon CloudWatch stores these metrics for a period of two weeks. You can monitor the CPU and memory reservation and utilization across your cluster as a whole and the CPU and memory utilization on the services in your cluster. You can use Amazon CloudWatch to trigger alarms, set notifications, and create dashboards to monitor the services.

Once it's set up, you can view Amazon CloudWatch metrics in both the Amazon ECS console and the Amazon CloudWatch console. The Amazon ECS console provides a maximum 24-hour view while the Amazon CloudWatch console provides a fine-grained and customizable view of running services.

The other tool available is AWS CloudTrail, which will log all Amazon ECS API calls.

AWS Trusted Advisor is another source for monitoring all of your AWS resources, including Amazon ECS, to improve performance, reliability, and security.

There is no additional cost for using Amazon ECS. The only charges are for the Amazon EC2 instances or AWS Lambda requests and compute time.

Security

With Amazon ECS, you need to do the following:

1. Control who can create task definitions.
2. Control who can deploy clusters.
3. Control who can access the Amazon EC2 instances.

IAM is the tool used for the first two necessities. For controlling access to Amazon EC2 instances, the tools described in the Amazon EC2 section still apply. You can use IAM roles, security groups, and (because these Amazon EC2 instances are located in an Amazon VPC) network Access Control Lists (ACLs) and route tables to control access to the Amazon EC2 instances.

AWS Elastic Beanstalk

AWS Elastic Beanstalk enables you to deploy and manage applications in the AWS Cloud without worrying about the infrastructure that runs those applications. AWS Elastic Beanstalk reduces management complexity without restricting choice or control. You simply upload your application, and AWS Elastic Beanstalk automatically handles the details of capacity provisioning, load balancing, scaling, and application health monitoring.

Languages Supported in AWS Elastic Beanstalk

AWS Elastic Beanstalk supports applications developed in the following languages:

- Java
- PHP
- .NET
- Node.js
- Python
- Ruby

Services that AWS Elastic Beanstalk Deploys

AWS Elastic Beanstalk automates the deployment of Amazon VPC, multiple compute instances (across multiple Availability Zones), security groups (both for instances and load balancers), Auto Scaling, Amazon S3 buckets, Amazon CloudWatch alarms, and domain names. Even though AWS Elastic Beanstalk automates the deployment of your environment, you can make modifications as you see fit.

Implementation

There are a number of tools that you can use to create and manage the AWS Elastic Beanstalk environment. These tools are as follows:

- AWS Management Console
- AWS Elastic Beanstalk CLI
- AWS SDK for Java
- AWS Toolkit for Eclipse
- AWS SDK for .NET
- AWS Toolkit for Visual Studio
- AWS SDK for Node.js
- AWS SDK for PHP (requires PHP 5.2 or later)
- Boto (AWS SDK for Python)
- AWS SDK for Ruby

You deploy a new application on AWS Elastic Beanstalk by uploading a source bundle. This source bundle can be a ZIP or a WAR file (you can include multiple WAR files inside of your ZIP file). This source bundle cannot be larger than 512 MB.

Management

AWS Elastic Beanstalk is by definition highly available. It creates multiple compute instances, locates those compute instances in two Availability Zones, and creates a load

balancer (a Classic Elastic Load Balancing load balancer) to spread the compute load across these multiple instances.

You can configure Auto Scaling to establish a minimum number of instances, a maximum number of instances, and what parameters would trigger either an increase or decrease in the number of instances. These triggers can either be time-based or event-based.

When you go into the AWS Elastic Beanstalk console, it displays all of the environments running in that region, sorted by application. You can further drill down by application name to see the environments, application versions, and saved configurations. There are limits on the number of applications, application versions, and environments that you can have.

As with Amazon EC2, you are responsible for patching and updating both the guest operating system and your application with AWS Elastic Beanstalk.

Making Changes in Your AWS Elastic Beanstalk Environment

It is recommended to use the AWS Elastic Beanstalk console and CLI when making changes to your AWS Elastic Beanstalk environments and configuration. Although it is possible to modify your AWS Elastic Beanstalk environment using other service consoles, CLI commands, or AWS SDKs, those changes will not be reflected in the AWS Elastic Beanstalk console and you will not be able to monitor, change, or save your environment. It will also cause issues when you attempt to terminate your environment without using the AWS Elastic Beanstalk console.

Monitoring Your AWS Elastic Beanstalk Environment

You can monitor the overall health of your environment using either the basic health reporting or enhanced health reporting and monitoring systems. Basic health reporting gives an overall status (via color key) of the environment. It does not publish any metrics to Amazon CloudWatch. Enhanced health reporting and monitoring not only provides status via color key, but it also provides a descriptor that helps indicate the severity and cause of the problem. You can also configure enhanced health reporting and monitoring to publish metrics to Amazon CloudWatch as custom metrics.

You can retrieve log files from the individual instances or have those log files uploaded to an Amazon S3 bucket for more permanent storage.

Security

When you create an environment with AWS Elastic Beanstalk, you are prompted to create two IAM roles:

Service role This is used by AWS Elastic Beanstalk to access other AWS Cloud services.

Instance profile This is applied to the instances created in your environment, and it allows them to upload logs to Amazon S3 and perform other tasks.

In addition to these two roles, you can create policies in IAM and attach them to users, groups, and roles. Just as in Amazon EC2, in AWS Elastic Beanstalk you need to create an Amazon EC2 key pair to access the guest operating system within your compute instances. Access would be via either Port 22 (for SSH) or Port 3389 (for RDP).



AWS Elastic Beanstalk allows web developers to deploy highly scalable and highly available web infrastructure without having to know or understand services like Elastic Load Balancing, Auto Scaling groups, or Availability Zones.

AWS Lambda

AWS Lambda is a serverless compute service. It runs your code in response to events. AWS Lambda automatically manages the underlying compute resources with no server configuration from you. There is no need to provision or manage servers.

The execution of code (called an AWS Lambda function) can be triggered by events internal to your AWS infrastructure or external to it.

AWS Lambda is a regionally-based service running across multiple Availability Zones. This makes it highly available and highly scalable. Because the code is stateless, it is executed almost automatically without deployment or configuration delays.

Implementation

Implementation for AWS Lambda involves the following steps:

1. Configure an AWS Lambda function.
2. Create a deployment package.
3. Upload the deployment package.
4. Create an invocation type.

The languages available to write your code are Node.js, Java, Python, and C#. The language that you choose also controls what tools are available for authoring your code. These tools can include the AWS Lambda console, Visual Studio, .NET Core, Eclipse, or your own authoring environment. You use these languages to create a deployment package.

Once a deployment package has been created, you then upload it to create an AWS Lambda function. The tools you can use to do so are the AWS Lambda console, CLI, and AWS SDKs.

The code that you have written is part of what is called an *AWS Lambda function*. The other elements of an AWS Lambda function are as follows:

- Memory
- Maximum execution time
- IAM role
- Handler name

The amount of memory that you specify controls the CPU power. The default amount of memory allocated per AWS Lambda function is 512 MB, but this can range from 128 MB to 1,536 MB in 64 MB increments.

Because you pay for the resources that are used to run your AWS Lambda function, the purpose of a timeout is to prevent your AWS Lambda function from running indefinitely. You can choose a value ranging between 1 and 300 seconds (5 minutes). When the specified timeout is reached, the AWS Lambda function is terminated.

The IAM role controls the level of access that the specific AWS Lambda function has. This is important when using AWS Lambda to access other AWS resources.

The handler refers to the method in your code where AWS Lambda begins execution. AWS Lambda passes any event information that triggered the invocation (further information on invocation and invocation types follows) as a parameter to the handler method.

When an AWS Lambda function is invoked, AWS Lambda launches a container based on your configuration specifications. Because it is launching a container, there may be some latency the first time that the AWS Lambda function is invoked. AWS Lambda will maintain the container for some time after it has been invoked to speed up response time for subsequent invocations.

How AWS Lambda Functions Are Invoked

AWS Lambda functions can be invoked either synchronously or asynchronously. If you are using AWS Cloud services to invoke AWS Lambda functions, those AWS Cloud services control whether the function is invoked synchronously or asynchronously. For example, if Amazon S3 invokes an AWS Lambda function, it does so asynchronously. In contrast, Amazon Cognito invokes an AWS Lambda function synchronously.

AWS Lambda functions are invoked by the following:

Push event model Some event sources can publish events to AWS Lambda and directly invoke your AWS Lambda function. The push model applies to Amazon S3, Amazon SNS, Amazon Cognito, Amazon Echo, and user applications, where each event triggers the AWS Lambda function.

Pull event model Some event sources publish events, but AWS Lambda must poll the event source and invoke your AWS Lambda function when events occur. This model applies when AWS Lambda is used with streaming event sources such as Amazon Kinesis and Amazon DynamoDB Streams. For example, AWS Lambda polls your Amazon Kinesis stream, or Amazon DynamoDB Stream, and it invokes your AWS Lambda function when it detects new records on the stream.

Direct invocation model This invocation type causes AWS Lambda to execute the function synchronously and returns the response immediately to the calling application. This invocation type is available for custom applications.



One of the recommendations for writing code for AWS Lambda is to make sure that the code is stateless (that is, it does not make reference to any underlying infrastructure).

Management

You can test your AWS Lambda function either in the AWS Lambda console or by using the CLI. Amazon CloudWatch provides metrics for AWS Lambda. These metrics include, but are not limited to, the following:

- Invocation count
- Invocation duration
- Number of requests
- Latency per request
- Duration of code execution

AWS CloudTrail will record the API calls used for the AWS Lambda functions. This provides another source of information for troubleshooting.

The AWS Lambda function is executed once it is invoked. If the execution fails, two more attempts are made. You can, as an option, configure AWS Lambda to forward a failed payload to a Dead Letter Queue (DLQ). This DLQ can take the form of an Amazon Simple Notification Service (Amazon SNS) topic or an Amazon Simple Queue Service (Amazon SQS) queue.

Security

The underlying compute infrastructure for AWS Lambda is the Amazon Linux AMI. Because AWS is responsible for maintaining this infrastructure (in terms of patching and upgrading), you are not able to log in to these compute instances or customize the operating system or language runtimes.

Each AWS Lambda function has a role associated with it. This role controls what AWS resources this specific AWS Lambda function has access to. As such, you need to grant the necessary level of permissions to that role.

You also manage permissions using an AWS Lambda function policy. AWS Lambda function policies allow you to have an event (for example, an event notification on an Amazon S3 bucket) to invoke an AWS Lambda function. AWS Lambda function policies are also used to invoke an AWS Lambda function when the source event is from outside the AWS account containing the AWS Lambda function.



Think of AWS Lambda as a service that lets you acquire CPU cycles without you having to acquire a CPU.

Amazon Lightsail

Amazon Lightsail is designed to give you the ability to launch and manage a Virtual Private Server (VPS) easily with AWS. When you launch an Amazon Lightsail instance, you pick the operating system, the developer stack, or application, and the size of the instance that

you wish to run. Amazon Lightsail instances come with SSD-backed storage, a publicly accessible IP address, DNS management, and data transfer capability.

Amazon Lightsail is designed to make it easy to spin up single servers. Amazon Lightsail is especially helpful for developers (both professional and hobbyist) because it enables them to focus on development instead of installing software or frameworks.

Implementation

You can access Amazon Lightsail via a console or the CLI. Access via the console can be done either through the AWS Management Console or through the separate Amazon Lightsail console.

Implementing Amazon Lightsail involves creating your instance by choosing an operating system, an application, or a developer stack.

OS Choices

The operating system choices for Amazon Lightsail are Amazon Linux or Ubuntu.

Applications and Developer Stack Choices

The following are application choices for Amazon Lightsail:

- Drupal
- Joomla
- Redmine
- GitLab
- WordPress
- Magneto
- Nginx

The following are the developer stacks choices for Amazon Lightsail:

- LAMP
- LEMP
- MEAN
- Node.js

After choosing an application or developer stack, you pick your instance plan, which determines the number of Virtual CPUs (vCPUs), amount of RAM, amount of disk space, and expected data transfer amounts. Assign a unique name (the name needs to be unique within your account), decide on the number of instances you want to spin up (the maximum is 20), and create your instance.

Your Amazon Lightsail instance is spun up in its own Amazon VPC.

Management

Once the instance is running, you can stop the instance, reboot it, or delete it. Billing stops once an instance is deleted.

Managing Amazon Lightsail involves managing the instance itself and managing the guest operating system and application or developer stack. Managing the instance can be done either via the Amazon Lightsail console or the CLI. Managing the other aspects can be done by accessing the instance via SSH connection offered in the Amazon Lightsail console or via your own SSH client. Accessing via your own SSH client involves downloading a private key from your AWS account page.

The Amazon Lightsail console provides you with access to five management features: Connect, Metrics, Networking, Snapshots, and History. Connect gives you the ability to log in to the guest operating system via the Amazon Lightsail console (versus a separate SSH client).

Available Metrics

Metrics provides you with information on the following:

- CPU utilization
- Network in
- Network out
- Status check failed instance (indicates if the specific instance is failing)
- Status check failed system (indicates if the underlying AWS hardware is failing)

All of these metrics are offered in one-hour, six-hour, one-day, one-week, and two-week increments.

Networking

Networking gives you information on your IP addresses, both the publicly accessible IP address (the address accessible from the Internet) and the private IP address (the address accessible internally). You can change the public IP to become a static IP address. Networking also lists the ports open on the firewall and provides the ability to open or close ports as needed, giving you control over who has access to your instance.

Snapshots is where you create and store snapshots. This is also where you can create new instances based on snapshots. There is no limit to the number of snapshots you can take and store. Note that you are charged for storing snapshots.

History lists when an instance was created, stopped, and rebooted; provides information on snapshots taken; and provides information on attached or detached static IP addresses.

With Amazon Lightsail, you can keep the public IP address assigned to the instance, assign a static public IP address, or create a DNS zone. Public IP addresses in Amazon Lightsail behave very similarly to public IP addresses in Amazon EC2; when you stop an Amazon Lightsail instance, the public IP address is removed; when you restart that instance, a new IP address is assigned. If you need IP address persistence, you should assign a static public IP address to the instance. A DNS zone supports A, CNAME, MX, and TXT records. You can use Amazon Route 53 to register your domain.

Costs

Costing for Amazon Lightsail is done on an hourly basis, with the minimum billing increment being one hour. While there is a monthly maximum for each different instance size, it is possible to have additional charges based on the following:

1. Storage costs of any snapshots that you take and keep
2. Charges for data transfer in excess of the amount allocated
3. Any public IP addresses that are not assigned to an instance

Changing Instance Size

Changing the instance size for Amazon Lightsail involves taking a snapshot of the existing instance and spinning up a new instance in the larger size. It is not possible to modify the number of vCPUs, RAM, or SSD size of an individual instance.

Security

Security with Amazon Lightsail follows the AWS shared responsibility model. Access to the compute instance is controlled via AWS tools (specifically IAM), although access to the guest operating system, application, or developer stack is controlled by you. Keeping the guest operating system, application, or developer stack current and patched is your responsibility.

AWS facilitates control over access to the guest operating system, application, and developer stack by only opening relevant ports for that particular operating system, application, or developer stack. Additional ports can be opened and existing ports can be modified.

AWS Batch

AWS Batch enables you to run batch computing workloads on AWS. These workloads can be at any scale. AWS Batch allows the end user to plan, schedule, and execute batch jobs while being able to control costs.

Batch jobs, by their nature, run asynchronously and have interdependencies between jobs. These interdependencies make the ability to control scheduling and sequencing very important. AWS Batch provides the tools to control the scheduling and sequencing of jobs.

Implementation

AWS Batch uses terminology that should be familiar to users of on-premises batch systems:

- Job
- Job definition

- Job queue
- Scheduler
- Compute environment

Jobs are the unit of work executed by AWS Batch. Jobs can be executed as AWS Lambda functions or as containerized applications running on Amazon EC2.

Job definitions are a similar concept to Amazon ECS task definitions—they specify how jobs are to be run. Some of the attributes specified in a job definition include vCPU requirements, memory requirements, mount points, and environmental variables, among other items.

Job queue is where jobs reside until the point that they are scheduled to a compute resource. You can have multiple job queues and set priorities using job queues. For example, you can specify having all jobs in a specific queue complete before jobs in another queue are attempted.

The scheduler evaluates the jobs in the job queue and makes decisions on when, where, and how to run those jobs. Currently, the algorithm is First In, First Out (FIFO), as long as all dependencies on other job resources have been met.

The compute environment can be of two types: managed and unmanaged. In a managed compute environment, you describe the needed requirements (such as instance types, minimum/maximum/desired vCPUs, use of Spot instances), and AWS launches and scales your resources. You specify in which Amazon VPC and subnet the compute instances are launched. In an unmanaged compute environment, you launch and manage the compute resources and need to include the Amazon ECS agent and run supported versions of Linux and Docker.

Implementing an AWS Batch Job

Steps to implementing a batch job are as follows:

1. Define a job.
2. Configure the compute environment.
3. Configure the job queue.
4. Submit a job.

Once a job is submitted it will go through the following states:

Submitted Submitted to the queue but has not yet been evaluated by the scheduler.

Pending Submitted to the queue and evaluated but not yet running due to a dependency on another job or resource.

Runnable In the queue with no outstanding dependencies and is ready to be scheduled.

Starting Scheduled to a host and container initiation operations are underway.

Running The job is running.

Succeeded The job has finished successfully.

Failed The job has failed all available attempts.

If a job fails, you can automate retry attempts up to 10 times.

Management

There are service limits to AWS Batch. You can have up to 10 compute environments, 5 job queues, and 3 compute environments per job queue. These limits can be changed via the AWS Management Console or the AWS CLI. You can also have up to 20 job dependencies and 1,000,000 jobs in a submitted state. The maximum job size is 20 KB. These three limits cannot be changed.

Monitoring

You can monitor your AWS Batch environment either via the CLI or the AWS Batch dashboard. Logs for your jobs (for example, STDERR and STDOUT) are available in the AWS Management Console and can be written to Amazon CloudWatch Logs. You can use SSH to access the Amazon EC2 instances that have been spun up to process your AWS Batch job.

Costs

There are no additional costs for AWS Batch. The only costs are for the Amazon EC2 resources or the AWS Lambda resources used. It is possible to use Spot pricing for all of your compute instances in order to minimize costs.

Security

AWS Batch uses IAM to control access to AWS resources that are used. Through IAM, you can define policies for different users in your organization to have different levels of access. Because you control the Amazon VPC in which you create the compute resources, you can implement additional layers of security with the use of private subnets, network ACLs, and security groups.

Summary

In this chapter, we discussed the following services:

- Amazon EC2
- Amazon ECS
- AWS Elastic Beanstalk
- AWS Lambda
- Amazon Lightsail
- AWS Batch

It is important to understand what each service does and does not do, even though the exam will not ask you about actual commands to launch an instance or to code an AWS Lambda function.

Exam Essentials

Know the basics about launching an Amazon EC2 instance. Launching an Amazon EC2 instance involves choosing an AMI (and knowing what are the sources for an AMI), choosing an instance type (and knowing what an instance type controls), and configuring user data.

Understand the pricing model for Amazon EC2. With Amazon EC2, there are three different pricing models: On-Demand, Reserved instances, and Spot. Understand the circumstances that would lead you to choose one pricing model over another.

Know what Amazon EC2 instance metadata is and how to find it. Instance metadata is data about your instance that you can use to configure or manage the running instance. Examples of instance metadata include AMI-id, public IP address, any roles assigned to the Amazon EC2 instance, and hostname. Metadata can be retrieved by logging in to the instance and using the following Uniform Resource Identifier (URI): <http://169.254.169.254/latest/meta-data>.

Know what user data is, how it is loaded into the Amazon EC2 instance, and how to find it. Instance user data is loaded on the initial boot of the Amazon EC2 instance, and it can be loaded in subsequent boots. User data is limited to 16 KB. User data can be used to load additional software and perform other functions. User data can be retrieved by logging into the instance and using the following URI: <http://169.254.169.254/latest/user-data>.

Know how to create an AMI. Although there are four sources for AMIs, the major source will be AMIs that you create. Creating your own AMIs gives you greater control over the actual instance. This control extends to boot times (instances spun up from AMIs boot faster than those that need to load user data), operating systems, and instance size.

Know when you would use a dedicated host and when you would use a dedicated instance. An Amazon EC2 Dedicated Host is a physical server with Amazon EC2 instance capacity fully dedicated to your use. Dedicated hosts allow you to use your existing per-socket, per-core, or per-VM software licenses. Rebooting a dedicated host will reboot the instance on the same physical AWS infrastructure. Dedicated instances are Amazon EC2 instances that run in an Amazon VPC on hardware that is dedicated to a single customer. Your dedicated instances are physically isolated at the host hardware level from instances that belong to other AWS accounts.

Understand the different CPU types available in Amazon EC2 and how to change instance types on running instances. The CPU types offered allow you to pick CPUs that are optimized for compute, storage, I/O, and graphics, among other considerations.

Know the different storage options available for Amazon EC2. Depending on the instance type chosen, you can have instance storage, Amazon EBS storage, or Amazon EBS-optimized storage.

Understand how instance size affects network speed. With the exception of certain instance types (specifically S and X instance types), network speed is a function of the size of the instance—larger instances have higher network speed. For greater network speed, some instance types support enhanced networking. Enhanced networking involves installing the correct elastic network interface and configuring the correct drivers.

Understand what an Amazon EBS-optimized instance is and when you would use it. An Amazon EBS-optimized instance uses an optimized configuration stack and provides additional, dedicated capacity for Amazon EBS I/O. This optimization provides improved performance for your Amazon EBS volumes by minimizing contention between Amazon EBS I/O and other traffic from your instance. Amazon EBS-optimized instances deliver dedicated throughput between Amazon EC2 and Amazon EBS, with options between 500 Mbps and 1,000 Mbps, depending on the instance type used.

Understand how to log in remotely to an Amazon EC2 instance. Refer to the Amazon EC2 User Guide for complete instructions.

Know what Amazon EC2 metrics are monitored in Amazon CloudWatch. Amazon CloudWatch collects metrics on CPU utilization, disk reads and writes, network in and out, status check failed instance, and status check failed system. It is important to remember that by default Amazon CloudWatch does not collect metrics beyond the Hypervisor level. Amazon CloudWatch collects log files, by default, in five-minute intervals.

Know how to configure an AWS Lambda function. AWS Lambda functions can be written in Node.js, Java, Python, and C# with a variety of tools available to create and edit code. An AWS Lambda function also has allocated memory, a maximum execution time set, an IAM role assigned (if it needs to access other AWS resources), and a handler name.

Understand the pricing model for AWS Lambda. Pricing for AWS Lambda has two components: total number of requests and duration of time your code executes. Duration is calculated from the time your code begins executing until it returns or otherwise terminates.

Know what AWS Lambda metrics Amazon CloudWatch monitors. Amazon CloudWatch will monitor AWS Lambda invocation count, invocation duration, number of requests, latency per request, and duration of code execution among other metrics.

Understand how to control CPU choice in AWS Lambda. CPU power is allocated by AWS based on the amount of memory allocated. The default amount of memory allocated is 512 MB, but that can be configured from 128 MB to 1,536 MB in 64 MB increments.

Understand the use cases for AWS Elastic Beanstalk, Amazon Lightsail, and AWS Batch. AWS Elastic Beanstalk provisions necessary infrastructure resources, such as the load balancer, Auto Scaling group, security groups, and the database (optional), and provides a unique domain name for your application infrastructure stack. AWS Elastic Beanstalk is oriented toward web developers who want to launch highly scalable, highly available web applications quickly and easily. Amazon Lightsail is most suitable for projects that require a few VPSs and for developers looking for a no-nonsense, simple management interface. AWS Batch enables developers, scientists, and engineers to run hundreds of thousands of batch computing jobs easily and efficiently on AWS.

Know what an AWS Lambda event is and what are supported AWS Lambda event sources. AWS Lambda events are events that trigger the execution of an AWS Lambda function. These events can be internal to AWS (for example, an Amazon CloudWatch notification, adding an object to an Amazon S3 bucket) or external to AWS.

Understand what services discussed in this chapter are considered highly available and what services are not. Services that are Availability Zone based (like Amazon EC2 or Amazon Lightsail) are not considered highly available. Services that are AWS Region-based

(like Amazon ECS and AWS Lambda) are considered highly available. Some services (like AWS Elastic Beanstalk) may or may not be highly available depending on how they have been configured.

Know what services AWS Elastic Beanstalk spins up by default. AWS Elastic Beanstalk spins up a default Amazon VPC (with an Internet gateway attached), subnets in multiple Availability Zones, compute instances in those Availability Zones, and Elastic Load Balancing with a public DNS name.

Know the different tools available to configure the services mentioned in this chapter. For all of the services mentioned in this chapter, with the exception of Amazon Lightsail, you can use the AWS Management Console to configure and manage them. Amazon Lightsail has its own management console that should be used for configuration, which can be accessed either directly or via the AWS Management Console. The CLI can be used to configure and manage all services mentioned in the chapter. Additionally, AWS Elastic Beanstalk and Amazon ECS have their own CLIs.

Know the different tools available to monitor Amazon EC2, Amazon ECS, and AWS Lambda. The two important tools for monitoring Amazon EC2, Amazon ECS, and AWS Lambda are Amazon CloudWatch and AWS CloudTrail. Amazon CloudWatch can receive log data from Amazon EC2, Amazon ECS, and AWS Lambda. This log data can be viewed in the AWS Management Console, the Amazon CloudWatch console, or a third-party console. Based on that log data, you can create dashboards, send notifications, and trigger alarms. AWS CloudTrail records the calls made to the AWS APIs using the AWS Management Console, the AWS Command Line Interface (AWS CLI), your own applications, and third-party software, and then publishes the resulting log files to the Amazon S3 bucket of your choice. AWS CloudTrail can also issue a notification to an Amazon SNS topic of your choice each time a file is published. Each call is logged in JSON format.

Know the different tools available to manage security for Amazon EC2. The principal tool for controlling access to AWS Cloud services is IAM. Access to interfaces on compute instances in Amazon EC2, Amazon ECS (when using Amazon EC2 instances), and AWS Elastic Beanstalk (which also uses Amazon EC2 instances) is controlled via port, protocol, and source using security groups. Interfaces on compute instances in Amazon Lightsail are protected by firewalls, which operate in a similar manner to security groups. Remote access (either via RDP or SSH) is controlled by both security groups (the proper protocol must be enabled on the interface) and the use of public/private keys.

Understand the AWS shared responsibility model and how it applies to Amazon EC2, Amazon ECS, AWS Lambda, and AWS Elastic Beanstalk. The AWS shared responsibility model means that AWS is responsible for security *of* the cloud while the AWS customer is responsible for security *in* the cloud. For Amazon EC2, Amazon ECS, AWS Elastic Beanstalk, and Amazon Lightsail, the customer has access to and control over the guest operating system running on the instance; therefore, the AWS customer is responsible for making sure that the guest operating system is properly patched and kept upgraded. For AWS Lambda, the customer does not have access to the operating system running the compute instance; therefore, AWS is responsible for making sure that the operating system is properly patched and upgraded.

Understand what security groups do and how they protect instances. Security groups control access on an interface level via port, protocol, and source. A source can be a single IP address, a range of IP addresses, or another security group. Security groups can be configured only to allow a connection, not to deny a connection.

Understand when you would use Amazon EC2 and when you would use Amazon Lightsail. Amazon Lightsail gives you a limited number of configuration options in terms of number of CPUs, amount of memory, and amount of storage. Amazon Lightsail is more applicable when you need a system that is preconfigured and preassembled and is ready to run your web applications with no system-building effort on your part. Amazon EC2, on the other hand, gives you many more options in terms of number and types of CPUs, amount of memory, and amount of storage. Amazon EC2 is more applicable when you want to (or need to) take the time to hand-select individual AWS components (for example, servers, storage, or IP addresses) and put them together on your own.

Know what services Amazon Lightsail spins up by default. With Amazon Lightsail, you choose an operating system (either AWS Linux or Ubuntu), then you can add a developer stack (such as LAMP) or an application (WordPress, for example). The plan you choose determines the number of CPUs, amount of memory, and amount of storage. A firewall is created to protect your instance with only the needed ports open. You can also add static IP addresses and DNS-hosted zones.

Test Taking Tip

You have notepaper, so make sure to use it (especially if you tend to get distracted)! You can't take the paper out of the room, but it will help you keep track of which answers you may have ruled out.

Resources to Review

Amazon EC2 page: <http://aws.amazon.com/ec2/>

Amazon EC2 FAQs: <http://aws.amazon.com/ec2/faqs/>

Amazon EC2 documentation: <http://aws.amazon.com/documentation/ec2/>

Amazon EC2 User Guide for Linux: <http://docs.aws.amazon.com/AWSEC2/latest/UserGuide/concepts.html>

Amazon EC2 User Guide for Windows: <http://docs.aws.amazon.com/AWSEC2/latest/WindowsGuide/concepts.html>

AWS Service Health Dashboard: <https://status.aws.amazon.com>

Cloud security resources: <https://aws.amazon.com/security/security-resources/>

AWS Service Health Dashboard: <http://status.aws.amazon.com/>

AWS Personal Health Dashboard: <https://phd.aws.amazon.com/>

AWS CloudTrail page: <http://aws.amazon.com/cloudtrail/>

AWS CloudTrail FAQs: <https://aws.amazon.com/cloudtrail/faqs/>

AWS CloudTrail documentation: <https://aws.amazon.com/documentation/cloudtrail/>

Amazon CloudWatch page: <https://aws.amazon.com/cloudwatch/>

Amazon CloudWatch FAQs: <https://aws.amazon.com/cloudwatch/faqs>

Amazon CloudWatch documentation: <https://aws.amazon.com/documentation/cloudwatch/>

AWS Elastic Beanstalk page: <https://aws.amazon.com/elasticbeanstalk/>

AWS Elastic Beanstalk FAQs: <https://aws.amazon.com/elasticbeanstalk/faqs>

AWS Elastic Beanstalk documentation: <https://aws.amazon.com/documentation/elastic-beanstalk>

AWS Lambda page: <https://aws.amazon.com/lambda/>

AWS Lambda FAQs: <https://aws.amazon.com/lambda/faqs>

AWS Lambda documentation: <https://aws.amazon.com/documentation/lambda/>

Amazon Lightsail page: <https://amazonlightsail.com/>

Amazon Lightsail FAQs: <https://amazonlightsail.com/docs>

Amazon Lightsail documentation: <https://aws.amazon.com/documentation/lightsail/>

AWS Batch page: <https://aws.amazon.com/batch/>

AWS Batch FAQs: <https://aws.amazon.com/batch/faqs>

AWS Batch documentation: <https://aws.amazon.com/documentation/batch/>

Amazon EC2 Container Service page: <https://aws.amazon.com/ecs/>

Amazon EC2 Container Service FAQs: <https://aws.amazon.com/ecs/faqs>

Amazon EC2 Container Service documentation: <https://aws.amazon.com/documentation/ecs/>

Exercises

By now you should have set up an account in AWS. If you haven't, now would be the time to do so. It is important to note that these exercises are in your AWS account and thus are not free.

Use the Free Tier when launching resources. The AWS Free Tier applies to participating services across the following AWS regions: US East (Northern Virginia), US West (Oregon), US West (Northern California), Canada (Central), EU (London), EU (Ireland), EU (Frankfurt), Asia Pacific (Singapore), Asia Pacific (Tokyo), Asia Pacific (Sydney), and South America (Sao Paulo). For more information, see https://aws.amazon.com/sdm/optimization/server-side-test/free-tier/free_np/.

If you have not yet installed the AWS Command Line utilities, refer to Exercise 2.1 (Linux) or Exercise 2.2 (Windows).

EXERCISE 4.1**Create a Linux Instance via the AWS Management Console.**

In this exercise, you will launch a new Linux instance, log in with SSH, and install any security updates.

1. Launch an Amazon EC2.
2. Choose the Amazon Linux AMI.
3. Choose the T.2 micro instance type.
4. Launch the instance into the default Amazon VPC.
5. Assign the instance a public IP address.
6. Add a tag to the instance of key = Name, value = Exercise 4.1 Linux.
7. Create a new security group called **CertBookL**.
8. Add a rule to the Cert Book L security group that allows SSH from anywhere (0.0.0.0/0).
9. Launch the instance.
10. When prompted for a key pair, select the Create a New Key Pair option, name it **CertBookKeyPair**, and save the private portion.
11. Use SSH to access the instance using the public IP address, username = ec2-user, and the **.CertBookKeyPair.pem** file you downloaded in Step 10.

Amazon generates a **keyname.pem** file, and you will need **keyname.ppk** if you're using the Putty client for Windows to SSH into your Linux instance. Puttygen.exe is one utility that will create a **.ppk** file from a **.pem** file.

12. From the command line, install any software updates.
13. Run **ifconfig -a**, and make note of the IP address. (It should be a private address.)
14. Access the metadata service to see the public IP address by running this command:
`--curl http://169.254.169.254/latest/meta-data/public-ipv4`.
15. Exit the SSH session and stop the instance via the AWS Management Console. You will use this instance in Exercise 4.6. If you do not plan to complete Exercise 4.6, terminate your instances.

EXERCISE 4.2**Create a Windows Instance via the AWS Management Console.**

In this exercise, you will launch a new Windows instance, log in with RDP, and read the event logs.

1. Launch an Amazon EC2 instance via the AWS Management Console.
 2. Choose the Microsoft Windows 2012 R2 Base AMI.
 3. Choose the T2.medium instance type.
 4. Launch the instance into the default Amazon VPC.
 5. Assign the instance a public IP address.
 6. Add a tag to the instance of key = Name, value = Exercise 4.2.
 7. Create a new security group called **CertBookW**.
 8. Add a rule to the security group that allows RDP from anywhere (0.0.0.0/0).
 9. Launch the instance.
 10. Use the key pair from Exercise 4.1.
 11. On the Connect Instance User Interface (UI), decrypt the administrator password, and then download the RDP file to connect to your instance.
 12. Once connected, look at the event logs, and then disconnect and stop the instance. You will use this instance in Exercise 4.6. If you do not plan to complete Exercise 4.6, terminate your instances.
-

EXERCISE 4.3**Create a Linux Instance via the AWS CLI.**

After completing Exercise 4.1, copy and paste the following information to a notepad utility:

- a. AMI ID
 - b. Keyname (should be **CertBookKeyPair**)
 - c. Security group ID
 - d. Subnet ID
 - e. Region
1. Run the following command with the values that you saved to your notepad utility:

```
aws ec2 run-instances --image-id ami-xxxxxxxx --count 1 --instance-type t2.micro  
--key-name MyKeyPair --security-group-ids sg-xxxxxxxx --subnet-id subnet-  
xxxxxxxx
```

2. Use SSH to access the instance using the public IP address, username = ec2-user, and the .pem (or .ppk) file you downloaded in the previous step.
 3. From the Linux command line, type the command below and press Enter:
`sudo yum update-security -y`
 4. Close the SSH window.
 5. Run the following command to terminate the instance:
`aws ec2-terminate-instances --instance-id i-xxxxxxxx`
-

EXERCISE 4.4

Create a Windows Instance via the AWS CLI.

1. After completing Exercise 4.1 and/or Exercise 4.3, copy and paste the following information to a notepad utility:
 - a. AMI ID
 - b. Keyname (should be **CertBookKeyPair**)
 - c. Security group ID
 - d. Subnet ID
 - e. Region
 2. Run the following command with the values that you saved to your notepad utility:
`aws ec2 run-instances --image-id ami-xxxxxxxx --count 1 --instance-type t2.micro --key-name MyKeyPair --security-group-ids sg-xxxxxxxx --subnet-id subnet-xxxxxxxx`
 3. Make note of the instance ID returned from the command or run this command:
`aws ec2 describe-instances`
 4. Use RDP to access this Windows Amazon EC2 instance using its public IP address.
 5. Run the following command to terminate the instance:
`aws ec2-terminate-instances --instance-id i-xxxxxxxx`
-

EXERCISE 4.5

Inspect the AWS Service Health Dashboards.

1. Log in to the AWS Management Console.
2. Navigate to <http://status.aws.amazon.com/>.
3. Subscribe to a service's RSS feed.
4. Open the AWS Personal Health Dashboard at <http://phd.aws.amazon.com>.

EXERCISE 4.5 (*continued*)

5. Make note of any open issues.
 6. Open the Event Log and make note of any issues and their status.
 7. Log out of the console.
-

EXERCISE 4.6**Use the Elastic IP Addresses.**

1. Log in to the AWS Management Console.
 2. Start your Linux or Windows instance from Exercises 4.1/4.2. (Notice the public IP address is different than before.)
 3. Navigate to the Amazon EC2 Console.
 4. Navigate to Elastic IPs.
 5. Allocate a new address.
 6. Associate with your running instance.
 7. Reboot your running instance.
 8. The IP address should be the same as prior to the reboot.
 9. Stop the instances.
-

EXERCISE 4.7**Work with Metadata.**

In this exercise, you will access the instance metadata from the OS.

1. Launch an instance in the Amazon EC2 console.
2. Choose the Amazon Linux AMI.
3. Choose the t2.micro instance type.
4. Launch the instance into the default Amazon VPC.
5. Assign the instance a public IP address.
6. Add a tag to the instance of Key: Name, Value: Exercise 4.7.
7. Use the Cert Book L security group.
8. Launch the instance.
9. Use the key pair from Exercise 4.1.

10. Connect to the instance via SSH using the public IP address. The user name is ec2-user and the file is CertBookKeyPair.pem or CertBookKeyPair.ppk.
 11. At the Linux command prompt, retrieve a list of available metadata by typing:
`curl http://169.254.169.254/latest/meta-data/`
 12. To see a value, add the name to the end of the URL. For example, to see the security groups, type:
`curl http://169.254.169.254/latest/meta-data/security-groups` or try
`curl http://169.254.169.254/latest/meta-data/public-ipv4`
 13. Curl the metadata service for the AMI ID (use the previous example).
 14. Close the SSH window and terminate the instance.
-

EXERCISE 4.8

Attach an AWS IAM Role to an Instance.

In this exercise, you will attach an IAM role to an instance and access.

1. Launch an instance in the Amazon EC2 console.
2. Choose the Amazon Linux AMI.
3. Choose the t2.micro instance type.
4. Launch the instance into the default Amazon VPC.
5. Select the BastionHost role that you created in Exercise 3.5.
6. Assign the instance a public IP address.
7. Add a tag to the instance of Key: Name, Value: Exercise 4.8.
8. Use the Cert Book L security group.
9. Launch the instance.
10. Use the key pair from Exercise 4.1.
11. Connect to the instance via SSH using the public IP address. The user name is ec2-user and the file is CertBookKeyPair.pem or CertBookKeyPair.ppk.
12. Run AWS configure and press Enter until you are returned back to the command prompt. AWS configure should be empty of credentials.
13. Create an Amazon S3 bucket using the AWS CLI:

```
aws s3 mb s3://mybucket --region us-west-1
```

Review Questions

1. Because of a change in your web traffic, you realize that you need a larger instance size for your web server. What is the best way to make this change?
 - A. Take a snapshot of your current instance, and use that as the basis for creating a new web server in the correct size.
 - B. Create a new Amazon Machine Image (AMI), and use that to spin up a new web server with the correct instance type.
 - C. Stop the instance, and then restart it with the correct instance type.
 - D. Assign an Elastic IP to the instance interface. Create a new instance with the right instance size. Reassign the Elastic IP from the old instance to the new instance.
2. You have configured a batch job on AWS Batch and you need it to complete. What should you do in order to make sure that the batch job completes?
 - A. Configure the pricing to be Spot pricing, but set the bid price at two times the On-Demand price for those instances.
 - B. Configure the pricing to be On-Demand.
 - C. Configure the pricing to be Spot pricing, but set the bid price for one-tenth of the On-Demand price for those instances.
 - D. There is nothing that you can do to guarantee completion of a batch job.
 - E. Configure the number of instances needed for the job as two times what is actually needed.
3. With AWS Batch, what values are you allowed to specify when configuring your instances?
 - A. Minimum number of Virtual CPUs (vCPUs), maximum number of vCPUs, and desired number of vCPUs
 - B. Minimum and maximum number of vCPUs
 - C. Desired number of vCPUs
 - D. You are not able to specify the number of vCPUs. AWS will assign the number based on current demand and availability.
4. You attempt to use Secure Shell (SSH) to access an Amazon Elastic Compute Cloud (Amazon EC2) instance but are unable to do so. What would NOT be a reason why you are unable to do so successfully?
 - A. You have an incorrect key.
 - B. The security group attached to the interface does not allow SSH.
 - C. You have an incorrect role assigned to your Amazon EC2 instance.
 - D. The Amazon EC2 instance does not have a public IP address attached to it.

5. You need to create an Amazon Elastic Compute Cloud (Amazon EC2) instance with the fastest possible boot time. What combination will give you an Amazon EC2 instance with the fastest boot time?
 - A. An instance store-backed Amazon Machine Image (AMI) with user data
 - B. An Amazon Elastic Block Store (Amazon EBS)-backed AMI with user data
 - C. An instance store-backed AMI with no user data
 - D. An Amazon EBS-backed AMI with no user data
6. Your boss has asked you to provide a report that shows your current monthly spending with AWS and an estimate of how much you will be spending this entire month. Which of these methods would be the best way to get those amounts?
 - A. Use the AWS Total Cost of Ownership (TCO) Calculator to build a model of your current infrastructure, and use that to create an estimate.
 - B. Use the My Billing Dashboard to see what your current spending is and also what your estimated total monthly spending would be.
 - C. Depending on the level of support you have with AWS, open a trouble ticket, and ask to be provided with this information.
 - D. Use the AWS Simple Monthly Calculator to build a model of your current infrastructure, and use that to create an estimate.
7. What would be a good reason NOT to use Amazon Lightsail?
 - A. You need to run a Windows instance.
 - B. You want to spin up a compute instance quickly for some basic code configuration work.
 - C. You want a high level of control over the cost of your compute instance.
 - D. You have a website that does not need to be highly available.
8. You have an application that needs to be available at all times. This application, once started, needs to run for about 35 minutes. You must also minimize the application's cost. What would be a good strategy for achieving all of these objectives?
 - A. Create an AWS Lambda function and have an Amazon Simple Notification Service (Amazon SNS) notification kick it off.
 - B. Set up a dedicated instance for this application.
 - C. Set up a dedicated host for this application.
 - D. Create an Amazon Elastic Compute Cloud (Amazon EC2) Reserved instance.
9. Which of the following would NOT be an appropriate event to trigger an AWS Lambda function?
 - A. Performing a GET operation in an Amazon Simple Storage Service (Amazon S3) bucket
 - B. Publishing a message to the Amazon Simple Notification Service (Amazon SNS) topic
 - C. An entry in an Amazon DynamoDB table
 - D. A message in an Amazon Simple Queue Service (Amazon SQS) queue

- 10.** Which of the following would be a good use case for AWS Elastic Beanstalk?
- A.** You need to make your current Amazon Relational Database Service (Amazon RDS) setup highly available.
 - B.** You want to spin up a website to serve internal customers, but you don't want to have to think about making it highly available and highly scalable.
 - C.** You need a compute instance to spin up in response to a specific network event.
 - D.** You are building a minimal website for test purposes.
- 11.** You have a relational database that is running on a Windows Amazon Elastic Compute Cloud (Amazon EC2) instance. What would be the best choice for instance type?
- A.** An instance type that uses instance storage
 - B.** An instance type that is Amazon Elastic Block Store (Amazon EBS)-optimized
 - C.** An instance type with a general-purpose CPU
 - D.** An instance type that allows you to use Amazon Elastic File System (Amazon EFS)
- 12.** You have a relational database that is running on a Windows Amazon Elastic Compute Cloud (Amazon EC2) instance. What would be the best choice for storage?
- A.** Instance store volumes
 - B.** Magnetic-based Amazon Elastic Block Store (Amazon EBS)
 - C.** Solid State Drive (SSD)-based Amazon EBS
 - D.** Amazon Elastic File System (Amazon EFS)
 - E.** Provisioned IOPS
- 13.** You have an instance store-backed Amazon Elastic Compute Cloud (Amazon EC2) instance with an interface that has a private IP address and a public IP address attached to it. You stop the instance. What happens to the IP addresses?
- A.** Both the public and private IP addresses are removed from the interface.
 - B.** The public IP address is removed, but the private IP address remains associated with the interface.
 - C.** The public IP address remains associated with the interface, but the private IP address is removed.
 - D.** Both the public IP address and the private IP address remain associated with the interface.
 - E.** None of the above. You cannot stop instance store-backed Amazon EC2 instances.
- 14.** You have an Amazon Elastic Block Store (Amazon EBS) storage-backed Amazon Elastic Compute Cloud (Amazon EC2) instance with an interface that has a private IP address and a public IP address attached to them. You stop the instance. What happens to the IP addresses?
- A.** Both the public and private IP addresses are removed from the interface.
 - B.** The public IP address is removed, but the private IP address remains associated with the interface.
 - C.** The public IP address remains associated with the interface, but the private IP address is removed.

- D. Both the public IP address and the private IP address remain associated with the interface.
 - E. None of the above. You cannot stop instance storage-backed Amazon EC2 instances.
15. You have an Amazon Elastic Compute Cloud (Amazon EC2) instance running in one AWS Region that you want to be able to run in another AWS Region. What do you need to do in order to accomplish that?
- A. You have to build the new Amazon EC2 instance from scratch because Amazon Machine Images (AMIs) are unique to an AWS Region.
 - B. You can copy an AMI from one AWS Region to another but only if they are under the same AWS account.
 - C. You cannot copy an AMI across AWS Regions because then you would have two AMIs with the same AMI ID.
 - D. You can copy AMIs across AWS Regions using the CopyImage Application Programming Interface (API).
 - E. You can copy AMIs across AWS Regions using the CopyImage API, but you need first to remove launch permissions and user defined tags.
16. You need to spin up an Apache Web Server on an Amazon Elastic Compute Cloud (Amazon EC2) instance. What is the best way to do so?
- A. Spin up an Amazon EC2 instance, use Secure Shell (SSH) to access it after it has booted up, and configure it to be an Apache Web Server.
 - B. In the metadata field, load the necessary software to spin up an Apache Web Server.
 - C. In the user data field, load the necessary software to spin up an Apache Web Server.
 - D. Go to AWS Marketplace and find an Apache Web Server Amazon Machine Image (AMI).
17. You need an Amazon Elastic Compute Cloud (Amazon EC2) instance to be able to access an object in an Amazon Simple Storage Service (Amazon S3) bucket. What is the most secure way to do this?
- A. Make sure that the Amazon EC2 instance has the necessary user permission to be able to access the Amazon S3 bucket.
 - B. Create an AWS Identity and Access Management (IAM) role. Make sure that role has the necessary level of permission to access the Amazon S3 bucket. Assign that role to the Amazon EC2 instance.
 - C. Create an IAM role. Make sure that role has the necessary level of permission to access the Amazon S3 bucket. Assign that role to the Amazon S3 bucket.
 - D. Make sure that the route table assigned to the Amazon EC2 instance has a route to the Amazon S3 bucket.
18. Why would you use AWS Elastic Beanstalk to deploy a web application that uses multiple Availability Zones?
- A. You might run out of compute instances if you ran everything in a single Availability Zone.
 - B. Using multiple Availability Zones improves the response time of your web application.
 - C. Using multiple Availability Zones makes your application highly available.
 - D. You can't run AWS Elastic Beanstalk in a single Availability Zone.

- 19.** What are some of the aspects of Amazon Elastic Compute Cloud (Amazon EC2) that are controlled by your choice of instance type?
- A.** Operating system
 - B.** User data
 - C.** Amazon Machine Image (AMI)
 - D.** CPU family availability
- 20.** You have just spun up an M4.2xlarge Amazon Elastic Compute Cloud (Amazon EC2) instance. What does the “4” stand for?
- A.** Indicates the amount of RAM associated with the instance
 - B.** Indicates the generation of M class family instance
 - C.** Indicates the baseline number of CPUs associated with the instance
 - D.** Has no meaning at all and is only an AWS naming convention

Chapter

5



Networking

THE AWS CERTIFIED SYSOPS ADMINISTRATOR - ASSOCIATE EXAM TOPICS COVERED IN THIS CHAPTER MAY INCLUDE, BUT ARE NOT LIMITED TO, THE FOLLOWING:

Domain 1.0: Monitoring and Metrics

- ✓ **1.1 Demonstrate ability to monitor availability and performance**

Content may include the following:

- Using Amazon CloudWatch to monitor Amazon Route 53, Amazon CloudFront, and Elastic Load Balancers
- Monitoring traffic in your Amazon VPC with Amazon VPC Flow Logs
- Monitoring & Alarming using Amazon CloudWatch

Domain 2.0: High Availability

- ✓ **2.2 Ensure level of fault tolerance based on business needs**

- Deploying resources in multiple subnets within multiple Availability Zones
- Using Elastic Load balancing to deliver traffic to distributed resources
- Using Amazon Route 53 to deliver fault tolerant infrastructures

Domain 3.0: Analysis

- ✓ **3.1 Optimize the environment to ensure maximum performance**

Content may include the following:

- Using Amazon CloudFront to deliver your content at improved performance for your viewers, while minimizing the operational burden and cost of scaling your infrastructure



Domain 6.0: Security

✓ 6.1 Implement and manage security policies

Content may include the following:

- Controlling access to Amazon VPC, Amazon CloudFront, Amazon Route 53, and AWS Direct Connect

Domain 7.0: Networking

✓ 7.1 Demonstrate ability to implement networking features of AWS

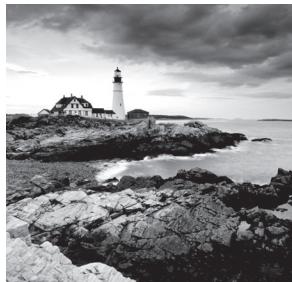
Content may include the following:

- Public and Private Subnets
- Route tables
- Elastic Load Balancing
- Elastic Network Interfaces (ENI) and Elastic IP (EIP) addresses

✓ 7.2 Demonstrate ability to implement connectivity features of AWS

Content may include the following:

- Security groups and network ACL to control traffic into and out of your Amazon VPC
- Controlling access to the Internet with Internet gateways and Network Address Translation (NAT) gateways
- Peering Amazon VPCs
- Using AWS Direct Connect and VPNs to access your Amazon VPC



Introduction to Networking on AWS

This chapter introduces you to a number of network services that AWS provides. Some of the services described in this chapter, such as Amazon Virtual Private Cloud (Amazon VPC), are fundamental to the operation of services on AWS. Others, like Amazon Route 53, offer services that, while optional, provide tight integration with AWS products and services.

The primary goal of this book is to prepare you for the AWS Certified SysOps Administrator - Associate exam; however, we want to do more for you. The authors of this book want to provide you with as much information as possible to assist you in your everyday journey as a Systems Operator.

The AWS services covered in this chapter include:

Amazon VPC With Amazon VPC, you provision a logically-isolated section of the AWS Cloud where you launch AWS resources in a virtual network that you have defined. You have complete control over your virtual networking environment.

AWS Direct Connect AWS Direct Connect allows you to establish a dedicated network connection from your premises to AWS.

Elastic Load Balancing Elastic Load Balancing automatically distributes incoming application traffic across multiple Amazon Elastic Compute Cloud (Amazon EC2) instances in an AWS Region. You can achieve fault tolerance in your applications, seamlessly providing the required amount of load balancing capacity needed to route application traffic.

Virtual Private Network (VPN) connections With VPN connections, you can connect Amazon VPC to remote networks. You can take advantage of AWS infrastructure to build a highly available, highly scalable solution, or you can build your own solution.

Amazon Route 53 Amazon Route 53 is a highly available and scalable cloud Domain Name System (DNS) web service. You can use Amazon Route 53 for domain management and for DNS service to connect to both AWS and non-AWS resources.

Amazon CloudFront Amazon CloudFront is a global Content Delivery Network (CDN) service that accelerates delivery of your websites, Application Programming Interfaces (APIs), video content, or other web assets.

Amazon Virtual Private Cloud (Amazon VPC)

Amazon Virtual Private Cloud (Amazon VPC) enables you to launch Amazon Web Services (AWS) resources into a virtual network that you've defined. This virtual network closely resembles a traditional network that you'd operate in your own datacenter, with the benefits of using the scalable infrastructure of AWS.

As you get started with Amazon VPC, you should understand the key concepts of this virtual network and how it is similar to or different from your own networks.

Amazon VPC is the networking layer for Amazon EC2. Amazon EC2 was covered in Chapter 4, “Compute.”

Amazon VPC Implementation

There are some decisions that you need to make when implementing an Amazon VPC. These decisions, because they have both design and operational implications, should be thought through thoroughly and made ahead of time. This includes decisions about the following:

- Region
- IP address type
- IP address range
- Size of Amazon VPC
- Number of subnets
- IP address range of subnets
- Size of subnets

Amazon VPC is a virtual network dedicated to your AWS account. It is logically isolated from other virtual networks in the AWS Cloud. You can launch your AWS resources, such as Amazon EC2 instances, into your VPC. You can configure your VPC; you can select its IP address range, create subnets, and configure route tables, network gateways, and security settings.

A *subnet* is a range of IP addresses in your Amazon VPC. You can launch AWS resources into a subnet that you select. Use a public subnet for resources that must be connected to the Internet, and a private subnet for resources that won't be connected to the Internet. Subnets cannot span Availability Zones, so you will need at least one subnet for each Availability Zone in which you plan to establish services. The other consideration is whether to have a public subnet and a private subnet.

Amazon VPCs are region-based services. VPCs cannot span regions. If you have (or want to have) services in two regions, you must have a minimum of two VPCs (one VPC per region). Within a region, you can have up to five VPCs per account.

Regarding IP address type, AWS supports both IPv4 and IPv6 addressing. IPv4 addressing is the default, and it is required on all Amazon VPCs. IPv6 is optional. With IPv4, the addresses are in the private address space (RFC1918), while IPv6 addresses are taken from AWS assigned IPv6 address space. It is important to remember that with IPv6, there is no concept of “private” address space, so all IPv6 addresses are reachable from the Internet. It is recommended that you choose an IPv4 address space that is currently not being used in your on-premises network, and if you’re deploying more than one VPC, those VPCs should use a different address space.

An Amazon VPC can range in size from a /16 to a /28 netmask (approximately 65,532 to 16 IP addresses). With IPv6, AWS assigns a fixed block of /56 addresses. It is recommended that the largest block be chosen (a /16 for IPv4) to maximize room for growth. Now that we have talked about VPCs, you can use the VPC wizard in the AWS Management Console or the AWS CLI to create your own VPC (See Figure 5.1).

FIGURE 5.1 AWS CLI used to create an Amazon VPC

```
[rhoodsg@chaos:~$ aws ec2 create-vpc --cidr-block 10.0.0.0/16
{
    "Vpc": {
        "VpcId": "vpc-8ac73dec",
        "InstanceTenancy": "default",
        "Tags": [],
        "Ipv6CidrBlockAssociationSet": [],
        "State": "pending",
        "DhcpOptionsId": "dopt-2ef7e74c",
        "CidrBlock": "10.0.0.0/16",
        "IsDefault": false
    }
}
rhoodsg@chaos:~$ ]
```



AWS reserves the four IP and the last IP addresses. In a 10.0.0.0/24, the following IPs are reserved:

10.0.0.0: Network address.

10.0.0.1: Reserved by AWS for the Amazon VPC router.

10.0.0.2: Reserved by AWS. The IP address of the DNS server is always the base of the Amazon VPC network range; however, the base of each subnet range is also reserved.

10.0.0.3: Reserved by AWS for future use.

10.0.0.255: Network broadcast address. AWS does not support broadcast in an Amazon VPC; therefore, we reserve this address.



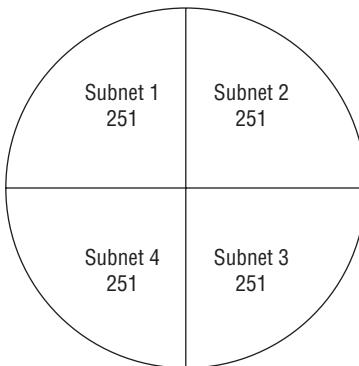
Once you choose the size of your Amazon VPC, you cannot change it. Therefore, err on the side of caution and make your VPC large. A best practice is to identify a private /16 network that is not being used and dedicate that to your VPC.

Figure 5.2 shows an Amazon VPC with CIDR /22 that includes 1,024 total IPs.

FIGURE 5.2 Amazon VPC /22 CIDR block and subnets

Example:

An Amazon VPC with CIDR /22 includes 1,024 total IPs.



Any AWS account created after December 4, 2013, has a default Amazon VPC in each AWS Region. If you do not specify a VPC to launch, your services will be launched into the default VPC.

A default Amazon VPC is set up with the following:

- A size /16 IPv4 Classless Inter-Domain Routing (CIDR) block (172.31.0.0/16)
- A default subnet in each Availability Zone (using a /20 subnet mask)
- An Internet gateway
- A main route table with a rule that sends all IPv4 traffic destined for the Internet to the Internet gateway
- A default security group that allows all traffic
- A default network ACL that allows all traffic
- A default DHCP option set

You can use a default VPC as you would use any other VPC: You can add subnets, modify the main route table, add route tables, associate additional security groups, update the rules of the default security group, and add VPN connections. You can use a default subnet as you would use any other subnet: You can add custom route tables and set network ACLs. You can also specify a default subnet when you launch an Amazon EC2 instance.

While the default subnet launches with IPv4 addresses, you can optionally associate an IPv6 CIDR block with your default VPC.



A clear understanding of route tables and network ACLs and how they operate and control access is fundamental. Know what type of traffic gets controlled by each of these and how they interact. Network connectivity troubleshooting is a key skill for this exam.

Route Tables

When a VPC is created, a route table associated with that VPC is also created. This is called the *default route table*. The default route table for VPC 10.155.0.0/16 would look like Table 5.1.

TABLE 5.1 Default Route Table for a VPC

| Destination | Target | Status | Propagated |
|---------------|--------|--------|------------|
| 10.155.0.0/16 | local | Active | No |

What this table is telling us is that any IP packet with a destination address in the 10.155.0.0/16 network (thus any packet with an address between 10.155.0.0 and 10.155.255.255) would be delivered within this VPC. What it also tells us is that any IP packet with any other IP address would be dropped because there are no instructions on how to route that packet.

Route tables can be modified to accommodate other addresses. Table 5.2 demonstrates this concept.

TABLE 5.2 Route Table for VPC with Route to the Internet

| Destination | Target | Status | Propagated |
|---------------|---------------------|--------|------------|
| 10.155.0.0/16 | local | Active | No |
| 0.0.0.0/0 | Internet-gateway-id | Active | No |

The route table now has an additional route. All network traffic bound for the 10.155.0.0/16 network would be delivered within this VPC, and all other packets would go to the Internet gateway (igw-5af67a3e). We talk about Internet gateways later in this section.

Network Access Control Lists (ACLs)

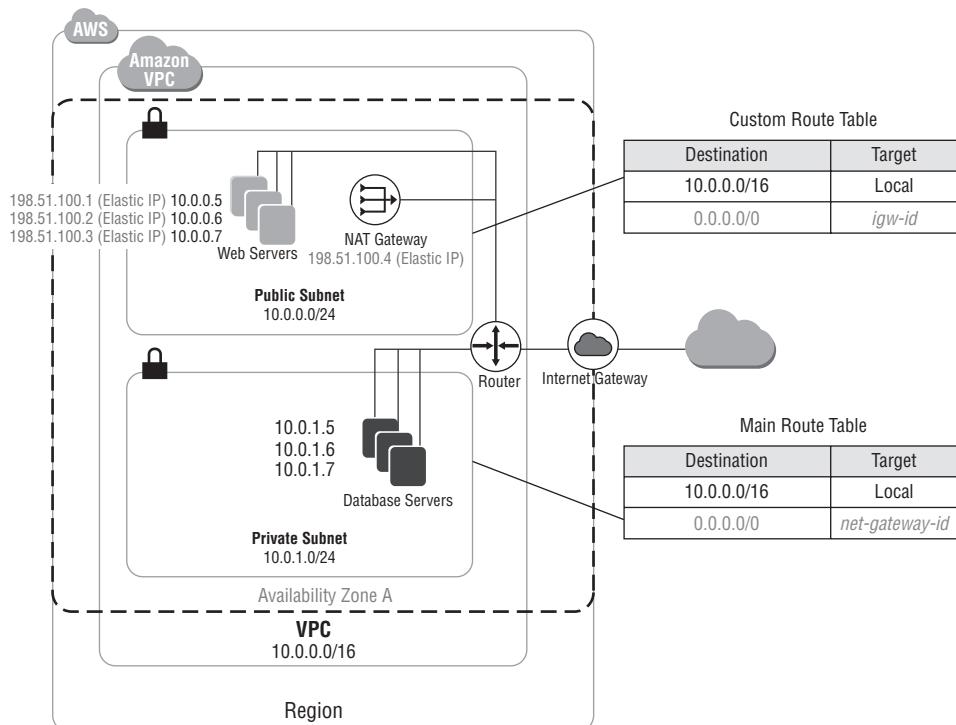
A VPC also has a default network *Access Control List (ACL)*. This default network ACL will allow all traffic. You can modify network ACLs either by allowing or denying inbound or outbound traffic based on port, protocol, and IP address. Network ACLs operate much in the same way as firewalls do in a datacenter. You create a number of rules that are evaluated in order, and when a match occurs, that rule gets invoked. Network ACLs were introduced in Chapter 3, “Security and AWS Identity and Access Management (IAM).” Figure 5.3 demonstrates a network Access Control List.

FIGURE 5.3 Default network Access Control List

| acl-4cee2929 Default | | | | | |
|---|---------------|----------------|---------------------|-----------|--------------|
| Summary | Inbound Rules | Outbound Rules | Subnet Associations | Tags | |
| Allows inbound traffic. Because network ACLs are stateless, you must create inbound and outbound rules. | | | | | |
| Edit | | | | | |
| View: All rules | | | | | |
| Rule # | Type | Protocol | Port Range | Source | Allow / Deny |
| 100 | ALL Traffic | ALL | ALL | 0.0.0.0/0 | ALLOW |
| * | ALL Traffic | ALL | ALL | 0.0.0.0/0 | DENY |

Security Groups

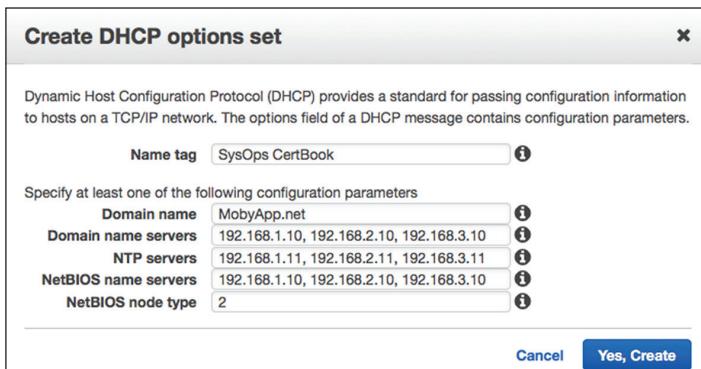
A VPC also has a default *security group*. This default security group allows all traffic. See the VPC diagram shown in Figure 5.4. It too can be modified as needed. Security groups were introduced in Chapter 3 and discussed in depth in Chapter 4.

FIGURE 5.4 VPC diagram

You control routing both within a VPC and routing into and out of a VPC by the use of route tables, network ACLs, and security groups.

One of the other things that is created with a VPC is a *Dynamic Host Configuration Protocol (DHCP) option set*. The DHCP option set controls various configuration parameters that are passed to compute instances created in the VPC. Some of the parameters are domain name, domain name server, NTP servers, NetBIOS name type, and NetBIOS node type. AWS provides a DNS server by default, but other DNS servers can be used instead. Figure 5.5 shows how to create a DHCP option set.

FIGURE 5.5 DHCP option set



When you create subnets within a VPC, those subnets inherit the route table, network ACLs, and security groups of the VPC. If you want to change the routing of packets within a subnet, you need to change the route table, network ACLs, and security groups as required.

Our recommended practice is to create two subnets per Availability Zone: one public subnet and one private subnet. The public subnet's route table includes a route to the Internet. The private subnet's route table does not include a route to the Internet. An instance in a public subnet has either a public IP or an Elastic IP address assigned to it, while an instance in a private subnet has only a private IP address assigned to it. For an instance in a private subnet to send a packet to the Internet, that packet must be routed through either a Network Address Translation (NAT) instance or a NAT gateway. For an instance in a private subnet to receive a packet from the Internet, that packet must be routed through either the Classic Load Balancer or Application Load Balancer, or through another compute instance with load balancing software installed. Table 5.3 shows a private subnet with a NAT gateway attached.

TABLE 5.3 Private Subnet Route Table with a NAT Gateway Entry

| Destination | Target | Status | Propagated |
|-------------|----------------|--------|------------|
| 10.0.0.0/16 | local | Active | No |
| 0.0.0.0/0 | nat-gateway-id | Active | No |

A VPC can be created that has no connection to resources outside of the VPC. However, most VPCs have some sort of connection to resources outside the VPC. These connections can take the form of gateways, endpoints, and peering connections.

There are three major gateways for the VPC. They are as follows:

- Internet gateway
- NAT gateway
- VPN gateway

Internet Gateway

An *Internet gateway* is a horizontally scaled, redundant, and highly available VPC component that allows communication between instances in your VPC and the Internet. It therefore imposes no availability risks or bandwidth constraints on your network traffic.

An Internet gateway serves two purposes: to provide a target in your VPC route tables for Internet-routable traffic and to perform network address translation (NAT) for instances that have been assigned public IPv4 addresses.

An Internet gateway supports IPv4 and IPv6 traffic. The Internet gateway can be created when the VPC is created, or it can be created at a later time. Figure 5.6 shows how to use the AWS CLI to create an Internet gateway.

FIGURE 5.6 AWS CLI to create Internet gateway

```
rhoodsg@chaos:~$ aws ec2 create-internet-gateway
{
    "InternetGateway": {
        "Tags": [],
        "InternetGatewayId": "igw-31308656",
        "Attachments": []
    }
}
rhoodsg@chaos:~$
```

NAT Gateway

You can use a *network address translation (NAT) gateway* to enable instances in a private subnet to connect to the Internet or other AWS services, but prevent the Internet from initiating a connection with those instances.

To create a NAT gateway, you must specify the public subnet in which the NAT gateway will reside. You must also specify an Elastic IP address to associate with the NAT gateway when you create it. After you've created a NAT gateway, you must update the route table associated with one or more of your private subnets to point Internet-bound traffic to the NAT gateway. This enables instances in your private subnets to communicate with the Internet.

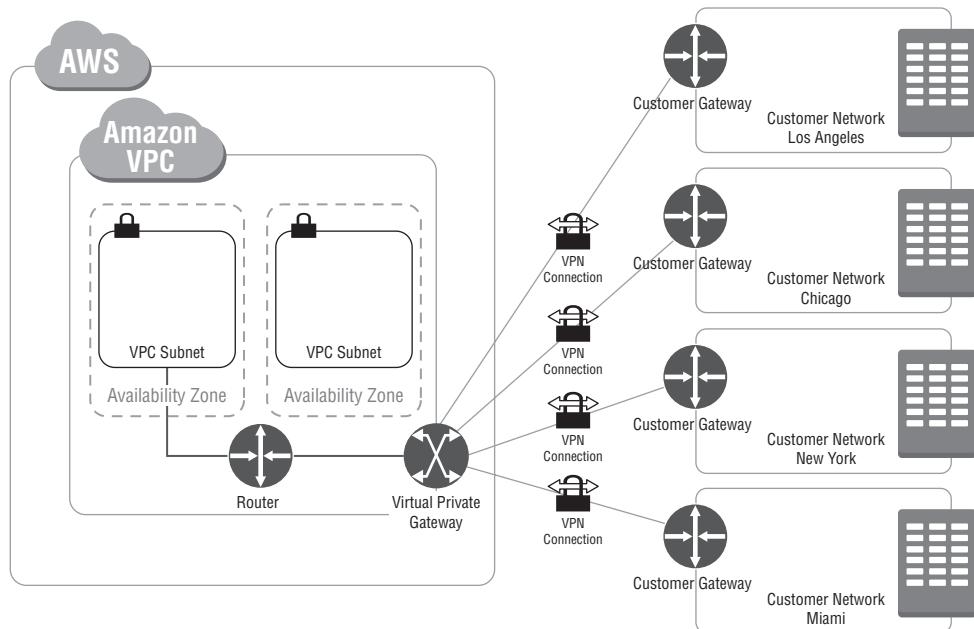
Each NAT gateway is created in a specific Availability Zone and implemented with redundancy in that zone.

VPN Gateway

A *VPN gateway* is used either to provide (1) an IPsec VPN connection through the Internet to your location or (2) an AWS Direct Connect connection to your location. In the first instance, a VPN gateway is scalable. Two connections are configured to provide availability.

When a VPN gateway is used for a VPN connection, it supports both static routing and Border Gateway Protocol (BGP). When it is used for an AWS Direct Connect connection, it supports only BGP. AWS Direct Connect is discussed in greater depth later in this chapter. See Figure 5.7 for a diagram of a VPC with VPN gateway.

FIGURE 5.7 VPC with VPN gateway



In order for an instance to communicate successfully with services outside the VPC, a public IP address needs to be associated with that instance. That IP address can be assigned to that instance when it is being created or it can be assigned by NAT. This NAT service can be provided either by an Amazon EC2 instance configured as a NAT server or by using a NAT gateway. There will be more coverage of NAT gateways later in this chapter.



The VPN connects your on-premises network to your VPC. If you need to access services outside your VPC, such as Amazon Dynamo DB, your traffic will travel the public Internet. AWS Direct Connect, discussed later in this chapter, offers a public virtual interface to handle traffic from your on-premises network to these services.

VPC Endpoint

Another way to connect to services outside the VPC, and specifically connect to Amazon Simple Storage Service (Amazon S3), is to use a VPC endpoint. A *VPC endpoint* allows you to have an Amazon S3 bucket become an entry on your route table. This allows you to create a private connection without using the VPN gateway. Table 5.4 demonstrates this concept.

TABLE 5.4 Route Table with VPC Endpoint

| Destination | Target | Status | Propagated |
|-------------|---------------|--------|------------|
| 10.0.0.0/16 | local | Active | No |
| 0.0.0.0/0 | igw-5af67a3e | Active | No |
| pl-xxxxxxxx | vpce-xxxxxxxx | Active | No |

VPC Peering

You can connect to other VPCs (both VPCs that are under your control and VPCs that are not under your control) by using VPC peering. This allows the other VPC to become another entry in your route table. Table 5.5 demonstrates this concept.

TABLE 5.5 Route Table with VPC Peering

| Destination | Target | Status | Propagated |
|---------------|--------------|--------|------------|
| 10.0.0.0/16 | local | Active | No |
| 0.0.0.0/0 | igw-5af67a3e | Active | No |
| 192.168.0.0/0 | pcx-c37bfaa | Active | No |

An Amazon *VPC peering connection* is a networking connection between two VPCs that enables you to route traffic between them using private IPv4 addresses or IPv6 addresses. Instances in either VPC can communicate with each other as if they are within the same network. You can create a VPC peering connection between your own VPCs or with a VPC in another AWS account. In both cases, the VPCs must be in the same region.

AWS uses the existing infrastructure of an Amazon VPC to create a VPC peering connection; it is neither a gateway nor a VPN connection, and it does not rely on a separate piece of physical hardware. There is no single point of failure for communication or a bandwidth bottleneck.



There are restrictions to using Amazon VPC peering: The VPCs must be in the same region, must not have overlapping IP addresses, and must trade traffic only with each other (one VPC cannot pass traffic to another VPC for it to forward on to a third VPC). Security is provided by requiring a “handshake” between the two VPCs and the use of security groups.

Elastic Network Interfaces

An *elastic network interface* (referred to as a network interface in this book) is a virtual network interface that you can attach to an instance in a VPC. Network interfaces are available only for instances running in a VPC.

A network interface can include the following attributes:

- A primary private IPv4 address
- One or more secondary private IPv4 addresses
- One Elastic IP address (IPv4) per private IPv4 address
- One public IPv4 address
- One or more IPv6 addresses
- One or more security groups
- A MAC address
- A source/destination check flag
- A description

You can create a network interface, attach it to an instance, detach it from an instance, and attach it to another instance. The attributes of a network interface follow it as it's attached to or detached from an instance and reattached to another instance. When you move a network interface from one instance to another, network traffic is redirected to the new instance.

Every instance in a VPC has a default network interface, called the *primary network interface* (*eth0*). You cannot detach a primary network interface from an instance. You can create and attach additional network interfaces. The maximum number of network interfaces that you can use varies by instance type.

When you create a network interface, it inherits the public IPv4 addressing attribute from the subnet. If you later modify the public IPv4 addressing attribute of the subnet, the network interface keeps the setting that was in effect when it was created. If you launch an instance and specify an existing network interface for *eth0*, the public IPv4 addressing attribute is determined by the network interface.

Additionally, you can associate an IPv6 CIDR block with your VPC and subnet and assign one or more IPv6 addresses from the subnet range to a network interface.

All subnets have a modifiable attribute that determines whether network interfaces created in that subnet (and therefore instances launched into that subnet) are automatically assigned an IPv6 address from the range of the subnet.



Elastic Network Interfaces (ENIs) are static, public IPv4 addresses that are associated with the account that created them. Because they are associated with the account (and not the instance), these IP addresses can be associated with any instance or network interface in that VPC. These ENIs, once assigned to an instance, will continue to remain associated with the instance even when that instance is turned off. If the instance is deleted, the ENI will remain associated with the account that created it.

Elastic IP Addresses (EIPs)

An *Elastic IP address* is a static public IPv4 address designed for dynamic cloud computing. An Elastic IP address is associated with your AWS account and allows you to mask the failure of an instance or software by remapping the address to another Amazon EC2 instance in your account.

An Elastic IP address is a public IPv4 address, which is reachable from the Internet. If your instance does not have a public IPv4 address, you can associate an Elastic IP address with your instance to enable communication with the Internet; for example, to connect to your instance from your local computer.

Currently, IPv6 is not supported on an Elastic IP address.

The following are important facts about the use of an Elastic IP address:

- To use an Elastic IP address, you first allocate one to your account and then associate it with your instance or a network interface.
- When you associate an Elastic IP address with an instance or its primary network interface, the instance's public IPv4 address (if one is assigned) is released back into Amazon's pool of public IPv4 addresses.
- You can disassociate an Elastic IP address from a resource and re-associate it with a different resource.
- A disassociated Elastic IP address remains allocated to your account until you explicitly release it.
- While your instance is running, you are not charged for one Elastic IP address associated with the instance, but you are charged for any additional Elastic IP addresses associated with the instance.
- There is a small hourly charge if an Elastic IP address is not associated with a running instance, or if it is associated with a stopped instance or an unattached network interface.
- An Elastic IP address is a regional construct.
- When you associate an Elastic IP address with an instance that previously had a public IPv4 address, the public DNS host name of the instance changes to match the Elastic IP address.

Amazon VPC Management

You can create and manage VPCs by using the AWS Management Console, the AWS Command Line Interface (AWS CLI), or AWS Software Development Kit (SDK). The two major tools used to manage and troubleshoot VPC issues are as follows:

- AWS CloudTrail
- Amazon VPC Flow Logs

With *AWS CloudTrail*, you can log various API calls and track activity that way. With Amazon VPC Flow Logs, you can capture information about IP traffic going to and from network interfaces in your VPC. AWS CloudTrail was discussed in Chapter 3. As this is the networking chapter, we discuss Amazon VPC Flow Logs in greater detail.

Amazon VPC Flow Logs

Amazon VPC Flow Logs is a feature that enables you to capture information about the IP traffic going to and from network interfaces in your VPC. Flow log data is stored using Amazon CloudWatch Logs. After you've created a flow log, you can view and retrieve its data in Amazon CloudWatch Logs.

Flow logs can help you with a number of tasks; for example, to troubleshoot why specific traffic is not reaching an instance, which in turn can help you diagnose overly restrictive security group rules. You can also use flow logs as a security tool to monitor the traffic that is reaching your instance.

You can create a flow log for a VPC, a subnet, or a network interface. If you create a flow log for a subnet or VPC, each network interface in the VPC or subnet is monitored. Flow log data is published to a log group in CloudWatch Logs, and each network interface has a unique log stream. Log streams contain flow log records, which are log events consisting of fields that describe the traffic for that network interface.



It is not possible for a virtual instance running in promiscuous mode to receive or *sniff* traffic that is intended for a different virtual instance. While customers can elect to place their interfaces into promiscuous mode, the Hypervisor will not deliver any traffic to an instance that is not addressed to it. This is where Amazon VPC Flow Logs come into play.

To create a flow log, you specify the resource for which you want to create the flow log, the type of traffic to capture (accepted traffic, rejected traffic, or all traffic), the name of a log group in CloudWatch Logs to which the flow log will be published, and the Amazon Resource Name (ARN) of an IAM role that has sufficient permission to publish the flow log to the CloudWatch Logs log group. If you specify the name of a log group that does not exist, AWS will attempt to create the log group for you. After you've created a flow log, it can take several minutes to begin collecting data and publishing to CloudWatch Logs. Flow logs do not capture real-time log streams for your network interfaces.

You can create multiple flow logs that publish data to the same log group in CloudWatch Logs. If the same network interface is present in one or more flow logs in the same log group, it has one combined log stream. If you've specified that one flow log should capture rejected traffic and the other flow log should capture accepted traffic, then the combined log stream captures all traffic.

If you launch more instances into your subnet after you've created a flow log for your subnet or VPC, then a new log stream is created for each new network interface as soon as any network traffic is recorded for that network interface.

You can create flow logs for network interfaces that are created by other AWS services; for example, Elastic Load Balancing, Amazon RDS, Amazon ElastiCache, Amazon Redshift, and Amazon WorkSpaces. However, you cannot use these services' consoles or APIs to create the flow logs. You must use the Amazon EC2 console or the Amazon EC2 API. Similarly, you cannot use the CloudWatch Logs console or API to create log streams for your network interfaces.



If you no longer require a flow log, you can delete it. Deleting a flow log disables the flow log service for the resource, and no new flow log records or log streams are created. It does not delete any existing flow log records or log streams for a network interface. To delete an existing log stream, you can use the CloudWatch Logs console. After you've deleted a flow log, it can take several minutes to stop collecting data.

AWS Direct Connect

AWS Direct Connect links your internal network to an AWS Direct Connect location over a standard 1-gigabit or 10-gigabit Ethernet fiber-optic cable. One end of the cable is connected to your router, and the other end is connected to an AWS Direct Connect router. With this connection in place, you can create virtual interfaces directly to public AWS Cloud services (for example, to Amazon S3) or to Amazon VPC, bypassing Internet service providers in your network path. An AWS Direct Connect location provides access to AWS in the region with which it is associated.

There are a number of reasons to use AWS Direct Connect instead of the Internet to access your AWS resources:

Reduce your bandwidth costs. For bandwidth-heavy workloads that you want to run on AWS, AWS Direct Connect reduces your network data out charges in two ways. First, by transferring data to and from AWS directly, you can reduce your bandwidth commitment to your Internet service provider. Second, all data transferred over your dedicated connection is charged at the reduced AWS Direct Connect data transfer rate instead of Internet data transfer rates.

Achieve consistent network performance. Network latency over the Internet can vary given that the Internet is constantly changing how data gets from point A to B. With AWS Direct Connect, you choose the data that uses the dedicated connection and how that data is routed, which can provide a more consistent network experience than Internet-based connections.

Private connectivity to AWS You can use AWS Direct Connect to establish a private virtual interface from your on-premises network directly to your Amazon VPC, providing you with a private, high-bandwidth network connection between your network and your VPC. With multiple virtual interfaces, you can even establish private connectivity to multiple VPCs while maintaining network isolation.

Elasticity and scaling AWS Direct Connect provides 1 Gbps and 10 Gbps connections, and you can easily provision multiple connections if you need more capacity. You can also use AWS Direct Connect instead of establishing a VPN connection over the Internet to your Amazon VPC, avoiding the need to use VPN hardware that frequently can't support data transfer rates above 4 Gbps.

AWS Direct Connect Implementation

You can set up an AWS Direct Connect connection in one of the following ways:

- At an AWS Direct Connect location
- Through a member of the AWS Partner Network (APN) or a network carrier
- Through a hosted connection provided by a member of the APN



There are two parts to the AWS Direct Connect service: the actual physical connection and the Virtual Interface (VIF). AWS Direct Connect uses a VPN gateway attached to the VPC you want to use.

Virtual Interfaces

To route traffic over your AWS Direct Connect connection, one or more Virtual Interfaces (VIFs) need to be created. If you have traffic that is coming from your location and bound to compute instances in your VPC, you need to create a private VIF. If the traffic is for either AWS Cloud services outside of a VPC or to the Internet, you need to create a public VIF. If you have traffic going to both, then you need to create both a private VIF and a public VIF. The components of a VIF are out of scope for the AWS Certified SysOps Exam; however, this book is meant to do more than prepare you for the exam.

Each VIF has the following components:

Virtual Local Area Network (VLAN) ID Each VIF must have its own unique VLAN ID. This has to be a value between 1 and 4094.

Peer IP address AWS Direct Connect supports both IPv6 and public and private IPv4 addresses. You can also dual stack the interface. For a public VIF, you need to specify IPv4 /30 addresses that you own. For a private VIF, private IPv4 /30 addresses are used. For IPv6, Amazon will allocate to you /125 IPv6 addresses.

BGP is the only supported routing protocol. For a private VIF, a private Autonomous System Number (ASN) is used. For a public VIF, a public ASN is used, and it is one that you must own.

Routes that you will advertise over BGP AWS Direct Connect can advertise up to 1,000 prefixes for a public VIF and 100 prefixes for a private VIF.

As previously mentioned, AWS Direct Connect uses a VPN gateway to connect to your AWS infrastructure. Therefore, you need to have a VPC configured with a VPN gateway attached.

Getting Started with AWS Direct Connect

Now we discuss the steps involved to get started with AWS Direct Connect. For the exam, understand the concepts of the public and private VIF that were discussed previously. The seven steps required to get started with Direct Connect are listed here:

1. **Sign Up for Amazon Web Services.** To use AWS Direct Connect, you need an AWS account if you don't already have one.

2. **Submit AWS Direct Connect Connection Request.** You can submit a connection request using the AWS Direct Connect console. Before you begin, ensure that you have the following information:
 - a. The port speed that you require: 1 Gbps or 10 Gbps. You cannot change the port speed after you've created the connection request.
 - b. The AWS Direct Connect location to which to connect.



If you require a port speed of less than 1 Gbps, you cannot request a connection using the console. Instead, contact an APN partner who will create a hosted connection for you. The hosted connection appears in your AWS Direct Connect console, and it must be accepted before use.

3. **Download the LOA-CFA.** AWS makes a Letter of Authorization and Connecting Facility Assignment (LOA-CFA) available to you for download, or it emails you with a request for more information after you've created the connection request. If you receive a request for more information, you must respond within seven days or the connection is deleted. The LOA-CFA is the authorization to connect to AWS, and it is required by the colocation provider or your network provider to establish the cross-network connection (cross-connect).



If the link is not enabled, the LOA-CFA is not yet available for you to download. Check your email for a request for more information. If it's still unavailable, or you haven't received an email after 72 hours, contact AWS Support.

After you've downloaded the LOA-CFA, do one of the following:

- If you're working with a network provider, send the LOA-CFA to your network provider so that they can order a cross-connect for you. You cannot order a cross-connect for yourself in the AWS Direct Connect location if you do not have equipment there. Your network provider does this for you.
- If you have equipment at the AWS Direct Connect location, contact the colocation provider to request a cross-network connection. You must be a customer of the colocation provider, and you must present them with the LOA-CFA that authorizes the connection to the AWS router, as well as the necessary information to connect to your network.



The LOA-CFA expires after 90 days. To refresh the LOA-CFA with a new issue date, you can download it again from the AWS Direct Connect console. If you do not take any action, AWS deletes the connection. Port-hour billing starts 90 days after you created the connection, or after the connection between your router and the AWS router is established, whichever comes first.

4. **(Optional) Configure Redundant Connections.** To provide for failover, we recommend that you request and configure two dedicated connections to AWS. These connections can terminate on one or two routers in your network.

There are different configuration choices available when you provision two dedicated connections: Active/Active and Active/Passive.



How you configure the connections doesn't affect redundancy, but it does affect the policies that determine how your data is routed over both connections. We recommend that you configure both connections as active.

5. **Create a Virtual Interface.** After you have placed an order for an AWS Direct Connect connection, you must create a virtual interface to begin using it. You can create a private virtual interface to connect to your VPC, or you can create a public virtual interface to connect to AWS services that aren't in a VPC. We will not discuss the specific steps needed, as it is out of the scope of this book. Visit http://docs.aws.amazon.com/directconnect/latest/UserGuide/getting_started.html.



A sub-1G connection supports only one virtual interface.

6. **Download Router Configuration.** After you have created a virtual interface for your AWS Direct Connect connection, you can download the router configuration file.
7. **Verify Your Virtual Interface.** After you have established virtual interfaces to the AWS Cloud or to Amazon VPC, you can verify your AWS Direct Connect connection.



For the exact steps and commands required to set up the AWS Direct Connect connection, refer to the AWS Direct Connect User Guide:

http://docs.aws.amazon.com/directconnect/latest/UserGuide/getting_started.html

AWS Direct Connect Management

You can use both the AWS Management Console and the AWS CLI to create and work with AWS Direct Connect. You can use tags on your AWS Direct Connect resources to categorize and manage those resources.

High Availability

AWS Direct Connect is not a highly available service by default. While it uses a VPN gateway, which is highly available, it is a single circuit, which is not. In order to achieve high availability with AWS Direct Connect, you need to have multiple AWS Direct Connect connections, and it is recommended to have those connections in different AWS Regions.



A VPN can be used as a backup connection to your AWS Direct Connect.

Route Preference

While it is beyond the scope of this book to discuss this in detail, you should be aware that when faced with a packet that has multiple routes, AWS will choose routes in this order:

1. Local routes within the VPC
2. Longest prefix match first (this would be on the route tables)
3. Static routes
4. AWS Direct Connect BGP
5. VPN static routes (defined on a VPN connection)
6. VPN BGP

Route preference is important to be aware of when using a VPN connection to back up an AWS Direct Connect connection or the inverse. VPN can be a backup solution to the AWS Direct Connect. If you're using VPN and the AWS Direct Connect becomes available, traffic should start taking advantage of the AWS Direct Connect immediately.



Costs for AWS Direct Connect include the port costs billed on an hourly basis, data transfer costs, circuit charges from the AWS Direct Connect partner, and potentially, if in a colocation facility, cross-connect charges. The data transfer costs are significantly less than similar charges for Internet data transfer.

AWS Direct Connect Security

Security is implemented in a number of ways with AWS Direct Connect. Each VIF has Layer 3 isolation from other VIFs because each operates in its own separate VLAN. In addition, it is possible to encrypt data in transit by implementing an IPsec VPN on the AWS Direct Connect connection.

Just like the other AWS services, IAM is used to control what roles, users, or groups can execute what APIs. IAM's role is in the creation and administration of AWS Direct Connect resource. Amazon CloudWatch can be used to capture API calls made by AWS Direct Connect.



AWS Direct Connect allows customers who already have an existing Wide Area Network (WAN), either Multiprotocol Label Switching (MPLS) or Virtual Private LAN Service (VPLS), to add AWS as another node in their WAN. This makes integration of AWS into the enterprise much easier.

Load Balancing

Elastic Load Balancing distributes incoming application traffic across multiple Amazon EC2 instances, in multiple Availability Zones within a single AWS Region. Elastic Load Balancing supports two types of load balancers:

- Application Load Balancers
- Classic Load Balancers

The Elastic Load Balancing load balancer serves as a single target, which increases the availability of your application. You can add and remove instances from your load balancer as your needs change without disrupting the overall flow of requests to your application. Elastic Load Balancing, working in conjunction with Auto Scaling, can scale your load balancer as traffic to your application changes over time. Refer to Chapter 10, “High Availability,” for details on how Auto Scaling works.

You can configure health checks and send requests only to the healthy instances. You can also offload the work of encryption and decryption to your load balancer. See Table 5.6 for differences between the Classic Load Balancer and the Application Load Balancer.

TABLE 5.6 Classic Load Balancer and Application Load Balancer

| Feature | Classic Load Balancer | Application Load Balancer |
|---------------------------|---------------------------------------|---------------------------|
| Protocols | HTTP, HTTPS, TCP, SSL | HTTP, HTTPS |
| Platforms | Amazon EC2-Classic, Amazon EC2-VPC | Amazon EC2-VPC |
| Sticky sessions (cookies) | ✓ | Load balancer generated |
| Idle connection timeout | ✓ | ✓ |
| Connection draining | ✓ | ✓ |
| Cross-zone load balancing | Can be enabled | Always enabled |
| Health checks | ✓ | Improved |
| Amazon CloudWatch metrics | ✓ | Improved |
| Access logs | ✓ | Improved |
| Host-based routing | | ✓ |
| Path-based routing | | ✓ |

TABLE 5.6 Classic Load Balancer and Application Load Balancer (*continued*)

| Feature | Classic Load Balancer | Application Load Balancer |
|--|-----------------------|---------------------------|
| Route to multiple ports on a single instance | ✓ | |
| HTTP/2 support | ✓ | |
| WebSocket support | | ✓ |
| Load balancer deletion protection | | ✓ |

Load Balancing Implementation

Because the Classic Load Balancer and the Application Load Balancer distribute traffic in different ways, implementation is different. Let's start with the Classic Load Balancer. Both Load Balancers will be examined in this section of the chapter.

Classic Load Balancer

With a *Classic Load Balancer*, there are a number of parameters that you need to configure. These include the following:

- Choose VPC
- Choose subnets
- Define protocols and ports
- Choose Internet facing or internal
- Determine security groups
- Configure health check
- Assign Amazon EC2 instances

If you have multiple VPCs, you determine which VPC will contain the Application Load Balancer. Load balancers cannot be shared among VPCs. From there, you are given a list of Availability Zones in which you have created subnets. In order for your application to be considered highly available, you must specify at least two Availability Zones. If you need additional Availability Zones, you need to create subnets in those Availability Zones.

High Availability

You can distribute incoming traffic across your Amazon EC2 instances in a single Availability Zone or multiple Availability Zones. The Classic Load Balancer automatically scales its request handling capacity in response to incoming application traffic.

Health Checks

The Classic Load Balancer can detect the health of Amazon EC2 instances. When it detects unhealthy Amazon EC2 instances, it no longer routes traffic to those instances and spreads the load across the remaining healthy instances.

Security Features

When using Amazon Virtual Private Cloud (Amazon VPC), you can create and manage security groups associated with Classic Load Balancers to provide additional networking and security options. You can also create a Classic Load Balancer without public IP addresses to serve as an internal (non-Internet-facing) load balancer.

SSL Offloading

Classic Load Balancers support SSL termination, including offloading SSL decryption from application instances, centralized management of SSL certificates, and encryption to backend instances with optional public key authentication.

Flexible cipher support allows you to control the ciphers and protocols the load balancer presents to clients.

Sticky Sessions

Classic Load Balancers support the ability to stick user sessions to specific EC2 instances using cookies. Traffic will be routed to the same instances as the user continues to access your application.

IPv6 Support

Classic Load Balancers support the use of Internet Protocol versions 4 and 6 (IPv4 and IPv6). IPv6 support is currently unavailable for use in VPC.

Layer 4 or Layer 7 Load Balancing

You can load balance HTTP/HTTPS applications and use layer 7-specific features, such as X-Forwarded and sticky sessions. You can also use strict layer 4 load balancing for applications that rely purely on the TCP protocol.

Operational Monitoring

Classic Load Balancer metrics, such as request count and request latency, are reported by Amazon CloudWatch.

Logging

Use the Access Logs feature to record all requests sent to your load balancer and store the logs in Amazon S3 for later analysis. The logs are useful for diagnosing application failures and analyzing web traffic.

You can use AWS CloudTrail to record classic load balancer API calls for your account and deliver log files. The API call history enables you to perform security analysis, resource change tracking, and compliance auditing.



You can assign the Classic Load Balancer to any Amazon EC2 instance that is in your VPC, including Amazon EC2 instances not in the Availability Zones that you specified. If you do so, the Classic Load Balancer will not route traffic to those Amazon EC2 instances.

Application Load Balancer

With an *Application Load Balancer*, there are a number of parameters that you need to configure. These include the following:

- Name of your load balancer
- Security groups
- The VPC
- Availability Zones to be used
- Internet-facing or internal
- IP addressing scheme to be used
- Configure one or more listeners
- Configure listener rules
- Define target group

Table 5.6 demonstrates the features of the Application Load Balancer. In order to become a proficient Systems Operator on AWS, you need to know what these features do. The features of the Application Load Balancer are defined in the following sections.

Content-Based Routing

If your application is composed of several individual services, an Application Load Balancer can route a request to a service based on the content of the request.

Host-Based Routing

You can route a client request based on the Host field of the HTTP header, allowing you to route to multiple domains from the same load balancer.

Path-Based Routing

You can route a client request based on the URL path of the HTTP header.

Containerized Application Support

You can now configure an Application Load Balancer to load balance containers across multiple ports on a single Amazon EC2 instance. Amazon EC2 Container Service (Amazon ECS)

allows you to specify a dynamic port in the ECS task definition, giving the container an unused port when it is scheduled on the EC2 instance. The ECS scheduler automatically adds the task to the ELB using this port.

HTTP/2 Support

HTTP/2 is a new version of the Hypertext Transfer Protocol (HTTP) that uses a single, multiplexed connection to allow multiple requests to be sent on the same connection. It also compresses header data before sending it out in binary format, and it supports TLS connections to clients.

WebSockets Support

WebSockets allows a server to exchange real-time messages with end users without the end users having to request (or poll) the server for an update. The WebSockets protocol provides bi-directional communication channels between a client and a server over a long-running TCP connection.

Native IPv6 Support

Application Load Balancers support native Internet Protocol version 6 (IPv6) in a VPC. This will allow clients to connect to the Application Load Balancer via IPv4 or IPv6.

Sticky Sessions

Sticky sessions are a mechanism to route requests from the same client to the same target. The Application Load Balancer supports sticky sessions using load balancer generated cookies. If you enable sticky sessions, the same target receives the request and can use the cookie to recover the session context. Stickiness is defined at a target group level.

Health Checks

An Application Load Balancer routes traffic only to healthy targets. With an Application Load Balancer, you get improved insight into the health of your applications in two ways:

1. Health check improvements that allow you to configure detailed error codes from 200-399. The health checks allow you to monitor the health of each of your services behind the load balancer.
2. New metrics that give insight into traffic for each of the services running on an Amazon EC2 instance.

High Availability

An Application Load Balancer requires you to specify more than one Availability Zone. You can distribute incoming traffic across your targets in multiple Availability Zones. An Application Load Balancer automatically scales its request-handling capacity in response to incoming application traffic.

Layer-7 Load Balancing

You can load balance HTTP/HTTPS applications and use layer 7-specific features, such as X-Forwarded-For (XFF) headers.

HTTPS Support

An Application Load Balancer supports HTTPS termination between the clients and the load balancer. Application Load Balancers also offer management of SSL certificates through AWS Identity and Access Management (IAM) and AWS Certificate Manager for predefined security policies.

Operational Monitoring

Amazon CloudWatch reports Application Load Balancer metrics, such as request counts, error counts, error types, and request latency.

Logging

You can use the Access Logs feature to record all requests sent to your load balancer and store the logs in Amazon S3 for later analysis. The logs are compressed and have a .gzip file extension. The compressed logs save both storage space and transfer bandwidth, and they are useful for diagnosing application failures and analyzing web traffic.

You can also use AWS CloudTrail to record Application Load Balancer API calls for your account and deliver log files. The API call history enables you to perform security analysis, resource change tracking, and compliance auditing.

Delete Protection

You can enable deletion protection on an Application Load Balancer to prevent it from being accidentally deleted.

Request Tracing

The Application Load Balancer injects a new custom identifier “X-Amzn-Trace-Id” HTTP header on all requests coming into the load balancer. Request tracing allows you to track a request by its unique ID as the request makes its way across various services that make up your websites and distributed applications. You can use the unique trace identifier to uncover any performance or timing issues in your application stack at the granularity of an individual request.

AWS Web Application Firewall

You can now use AWS WAF to protect your web applications on your Application Load Balancers. AWS WAF is a web application firewall that helps protect your web applications from common web exploits that could affect application availability, compromise security, or consume excessive resources.

Load Balancing Management

Elastic Load Balancing is a highly available and scalable service.

You can create, access, and manage your load balancers using any of the following interfaces:

AWS Management Console Provides a web interface that you can use to access Elastic Load Balancing.

AWS CLI Provides commands for a broad set of AWS Cloud services, including Elastic Load Balancing, and it is supported on Windows, Mac, and Linux.

AWS SDKs Provides language-specific APIs and takes care of many of the connection details, such as calculating signatures, handling request retries, and error handling.

Query API Provides low-level API actions that you call using HTTPS requests. Using the Query API is the most direct way to access Elastic Load Balancing, but it requires that your application handle low-level details such as generating the hash to sign the request and error handling.

There are three tools for managing both Classic Load Balancers and Application Load Balancers. These are Amazon CloudWatch, AWS CloudTrail, and access logs. In addition, Application Load Balancers has a feature that allows request tracing to track HTTP requests from clients to targets.

Amazon CloudWatch

Amazon CloudWatch provides a number of metrics available to track the performance of the Elastic Load Balancing load balancers. These metrics include number of health hosts, average latency, number of requests, and number of connections, among others. These metrics are updated on a 60-second interval.

AWS CloudTrail

AWS CloudTrail can capture the API calls used by the Elastic Load Balancing load balancers and deliver those logs to an Amazon S3 bucket that you specify. From there you can analyze the logs as needed. The information these logs will include are things like the time the API was invoked, what invoked the API, and the parameters of the request.

Access Logs

Access logs are an optional feature of Elastic Load Balancing. Access logs capture greater detailed information than AWS CloudTrail does, including such information as latencies, request paths, IP addresses of clients making the requests, and the instance processing the request, among other information.

Request Tracing

Request tracing is used to track HTTP requests from clients to targets or other services. When the load balancer receives a request from a client, it adds or updates the X-Amzn-Trace-Id header before sending the request to the target. Any services or applications between the load balancer and the target can also add or update this header. The contents of the header are logged in access logs.

Load Balancing Security

As mentioned earlier, security groups control the traffic allowed to and from your load balancer. You can choose the ports and protocols to allow for both inbound and outbound traffic. You apply the security group to the load balancer, so it is applied to all interfaces.

The rules associated with your load balancer security group must allow traffic in both directions. These rules need to be applied to both the listener and the health check ports. When you add a listener to a load balancer or update the health check port for a target group, you need to review your security group rules to ensure that they allow traffic on the new port in both directions.

Authentication and Access Control for Your Load Balancers

AWS uses security credentials to identify you and to grant you access to your AWS resources. You can use features of AWS Identity and Access Management (IAM) to allow other users, services, and applications to use your AWS resources fully or in a limited way, without sharing your security credentials.

By default, IAM users don't have permission to create, view, or modify AWS resources. To allow an IAM user to access resources, such as a load balancer, and perform tasks, you must create an IAM policy that grants the IAM user permission to use the specific resources and API actions they'll need and then attach the policy to the IAM user or the group to which the IAM user belongs. When you attach a policy to a user or group of users, it allows or denies the users permission to perform the specified tasks on the specified resources.

For example, you can use IAM to create users and groups under your AWS account (an IAM user can be a person, a system, or an application). Then you grant permissions to the users and groups to perform specific actions on the specified resources using an IAM policy.

Virtual Private Network (VPN)

You can connect your VPC to remote networks by using a VPN connection. There are a number ways to create this VPN connection. Your options are as follows:

- Virtual Private Gateway (VGW)
- AWS VPN CloudHub
- Software VPN

VPN Installation

To implement a Virtual Private Network, create a VGW and then attach it to a VPC. While you can have up to five VGWs per AWS Region, you can only have one VGW attached to a VPC. A VGW can be used for VPNs using IPsec and AWS Direct Connect.

A VGW can support both static routing and BGP. With static routing and with BGP, the IP ranges of the customer location and the VPC must be unique.

CloudHub

AWS VPN CloudHub also uses a VGW and, using a hub-and-spoke model, connects multiple customer gateways. AWS VPN CloudHub uses BGP with each customer location having a unique ASN.

Software VPN

Creating a software VPN involves spinning up one or more Amazon EC2 instances in one or more Availability Zones within the region and then loading VPN software onto those Amazon EC2 instances. The VPN software can be acquired directly by the customer, or it can be acquired via AWS Marketplace. AWS Marketplace offers a number of options, including OpenVPN, Cisco, Juniper, and Brocade, among others.

VPN Management

VGW is highly available and scalable. Each VGW comes with two publicly accessible IP addresses. This means that a VGW sets up two separate IPsec tunnels. You need to provision two public IP addresses on your side. These can be on a single customer gateway, two customer gateways at the same location, or two customer gateways in two different locations. You can connect multiple customer gateways to the same VGW.

AWS VPN CloudHub is highly available and scalable. With AWS VPN CloudHub, each location advertises their appropriate routes over their VPN connection. AWS VPN CloudHub receives these advertisements and re-advertises them out to the other customer gateways. This allows each site to send and receive data from the other customer sites.

Creating a software VPN gives you both the greatest level of control and the greatest level of responsibility. You spin up the instance or instances and are responsible for their placement, that they are the correct size (and can increase in size or number to meet increased demand), and that they are monitored and replaced if either is not working or working with reduced functionality.



AWS Direct Connect and VPN are the two most common methods used by large enterprises to connect their existing WAN with AWS. Most use both methods. Understanding the advantages and disadvantages of both methods and how you can use the two methods combined is very important.

Amazon Route 53

Amazon Route 53 is a highly available and scalable cloud DNS web service. DNS routes end users to Internet applications by translating names like `www.example.com` into the numeric IP addresses like `192.0.2.1` that computers use to connect to each other. Amazon Route 53 is fully compliant with IPv6.

You can use Amazon Route 53 to help you get a website or web application up and running. Amazon Route 53 enables you to perform three main functions:

Register domain names. Your website needs a name, such as `example.com`. Amazon Route 53 lets you register a name for your website or web application, known as a *domain name*.

Route Internet traffic to the resources for your domain. When a user opens a web browser and enters your domain name in the address bar, Amazon Route 53 helps the DNS connect the browser with your website or web application.

Check the health of your resources. Amazon Route 53 sends automated requests over the Internet to a resource, such as a web server, to verify that it is reachable, available, and functional. You also can choose to receive notifications when a resource becomes unavailable and choose to route Internet traffic away from unhealthy resources.

You can use any combination of these functions. For example, you can use Amazon Route 53 both to register your domain name and to route Internet traffic for the domain, or you can use Amazon Route 53 to route Internet traffic for a domain that you registered with another domain registrar. If you choose to use Amazon Route 53 for all three functions, you register your domain name, then configure Amazon Route 53 to route Internet traffic for your domain, and finally configure Amazon Route 53 to check the health of your resources. You can use Amazon Route 53 to manage both public and private hosted zones. So, you can use Amazon Route 53 to distribute traffic between multiple AWS Regions and to distribute traffic within an AWS Region.

Amazon Route 53 Implementation

In addition to registering new domains, you can transfer existing domains. When you register a domain with Amazon Route 53, a hosted zone is automatically created for that domain. This makes it easier to use Amazon Route 53 as the DNS service provider for this domain. You are, however, not obligated to use Amazon Route 53 as the DNS service provider. You may route your DNS queries to another DNS provider.

When you're using Amazon Route 53 as the DNS service provider, you need to configure the DNS service. As mentioned, when you use Amazon Route 53 as the domain registrar, a hosted zone is automatically created for you. If you are not using Amazon Route 53 as the domain registrar, then you will need to create a hosted zone.

A *hosted zone* contains information about how you want to route your traffic both for your domain and for any subdomains that you may have. Amazon Route 53 assigns a unique set of nameservers for each hosted zone that you create. You can use this set of nameservers for multiple hosted zones if you want.



A hosted zone is a collection of resource record sets for a specified domain. You create a hosted zone for a domain (such as `example.com`), and then you create resource record sets to tell the Domain Name System how you want traffic to be routed for that domain.

After you have created your hosted zones, you need to create resource record sets. This involves two parts: the record type and the routing policy. Different routing policies can be applied to different record types.

A *routing policy* determines how Amazon Route 53 responds to queries. There are five routing policies available, and each of them is explained in this chapter.

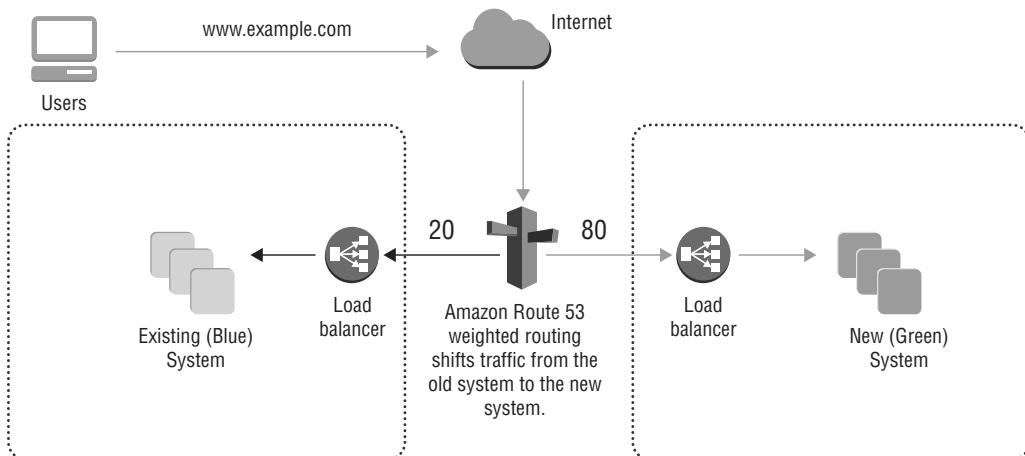
Simple Routing

Use a simple routing policy when you have a single resource that performs a given function for your domain; for example, one web server that serves content for the example.com website. In this case, Amazon Route 53 responds to DNS queries based only on the values in the resource record set; for example, the IP address in an A record.

Weighted Routing

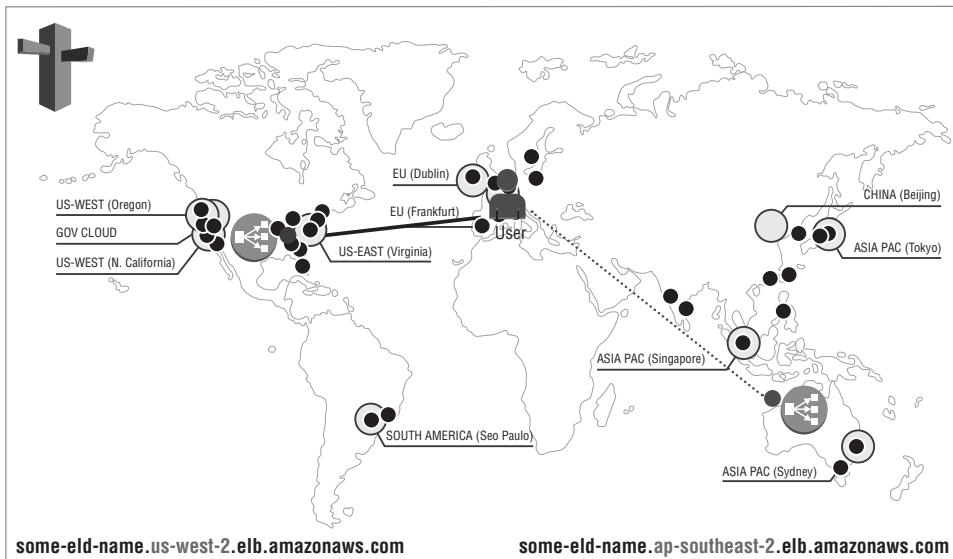
Use the weighted routing policy when you have multiple resources that perform the same function (for example, web servers that serve the same website) and you want Amazon Route 53 to route traffic to those resources in proportions that you specify (for example, one quarter to one server and three quarters to the other). Figure 5.8 demonstrates weighted routing.

FIGURE 5.8 Weighted routing



Latency-Based Routing

Use the latency routing policy when you have resources in multiple Amazon EC2 datacenters that perform the same function and you want Amazon Route 53 to respond to DNS queries with the resources that provide the best latency. For example, you might have web servers for example.com in the Amazon EC2 datacenters in Ireland and in Tokyo. When a user browses to example.com, Amazon Route 53 chooses to respond to the DNS query based on which datacenter gives your user the lowest latency. Figure 5.9 demonstrates this concept.

FIGURE 5.9 Latency-based routing

Geolocation Routing

Use the geolocation routing policy when you want Amazon Route 53 to respond to DNS queries based on the location of your users. Geolocation routing returns the resource based on the geographic location of the user. You can specify geographic locations by continent, country, or state within the United States.



Some IP addresses aren't mapped to geographic locations, so even if you create geolocation resource record sets that cover all seven continents, Amazon Route 53 will receive some DNS queries from locations that it can't identify. You can create a default resource record set that handles both queries from IP addresses that aren't mapped to any location. If you don't create a default resource record set, Amazon Route 53 returns a "no answer" response for queries from those locations.

Failover Routing

When using a failover routing policy, you designate a primary resource and a secondary resource. The secondary resource takes over in the event of a failure of the primary resource. To accomplish this, you configure a health check for the primary resource record set. If the health check fails, Amazon Route 53 routes the traffic to the secondary resource. It is recommended, but not obligatory, to configure a health check for the secondary

resource. If both record sets are unhealthy, Amazon Route 53 returns the primary resource record set. Health checks are discussed in greater detail in Chapter 10.



You can combine routing (for example, have geolocation routing backed up with failover routing) to make sure that you provide the highest level of availability possible. As you can imagine, this can get very complex (and confusing) very quickly, so good documentation is important.

DNS Record Types

Explaining the various DNS records types is out of the scope of this book. However, Table 5.7 shows the supported record types for Amazon Route 53.

TABLE 5.7 Amazon Route 53 Supported DNS Record Types

| Record Type | Description |
|-------------|--------------------------------|
| A | Address mapping records |
| AAAA | IPv6 address records |
| CNAME | Canonical name records |
| MX | Mail exchanger record |
| NAPTR | Name authority pointer record |
| NS | Name server records |
| PTR | Reverse-lookup Pointer records |
| SOA | Start of authority records |
| SPF | Sender policy framework record |
| SRV | Service record |
| TXT | Text records |

In addition to the standard DNS record types supported, Amazon Route 53 supports a record type called *Alias*. An *Alias record type*, instead of pointing to an IP address or a domain name, points to one of the following:

- An Amazon CloudFront distribution
- An AWS Elastic Beanstalk environment

- An Elastic Load Balancing Classic or Application Load Balancer
- An Amazon S3 bucket that is configured as a static website
- Another Amazon Route 53 resource record set in the same hosted zone

Health Checks

There are three types of health checks that you can configure with Amazon Route 53. They are as follows:

- The health of a specified resource, such as a web server
- The status of an Amazon CloudWatch alarm
- The status of other health checks

In this section, we explore each type. The level of detail covered may not be tested on the exam. However, as an AWS Certified Systems Operator, the material covered here is a must-know.

The health of a specified resource, such as a web server You can configure a health check that monitors an endpoint that you specify either by IP address or by domain name. At regular intervals that you specify, Amazon Route 53 submits automated requests over the Internet to your application, server, or other resource to verify that it's reachable, available, and functional. Optionally, you can configure the health check to make requests similar to those that your users make, such as requesting a web page from a specific URL.

The status of an Amazon CloudWatch alarm You can create CloudWatch alarms that monitor the status of CloudWatch metrics, such as the number of throttled read events for an Amazon DynamoDB database or the number of Elastic Load Balancing hosts that are considered healthy. After you create an alarm, you can create a health check that monitors the same data stream that CloudWatch monitors for the alarm.

To improve resiliency and availability, Amazon Route 53 doesn't wait for the CloudWatch alarm to go into the ALARM state. The status of a health check changes from healthy to unhealthy based on the data stream and on the criteria in the CloudWatch alarm. The status of a health check can change from healthy to unhealthy even before the state of the corresponding alarm has changed to ALARM in CloudWatch.

The status of other health checks You can create a health check that monitors whether Amazon Route 53 considers other health checks healthy or unhealthy. One situation where this might be useful is when you have multiple resources that perform the same function, such as multiple web servers, and your chief concern is whether some minimum number of your resources is healthy. You can create a health check for each resource without configuring notification for those health checks. Then you can create a health check that monitors the status of the other health checks and that notifies you only when the number of available web resources drops below a specified threshold.

Amazon Route 53 Management

You can access Amazon Route 53 in the following ways:

- AWS Management Console
- AWS SDKs
- Amazon Route 53 API
- AWS CLI
- AWS Tools for Windows PowerShell

The best tool for monitoring the status of your domain is the Amazon Route 53 dashboard. This dashboard will give a status of any new domain registrations, domain transfers, and any domains approaching expiration.

The tools used to monitor your DNS service with Amazon Route 53 are health checks, Amazon CloudWatch, and AWS CloudTrail. Health checks are discussed in the management section. Amazon CloudWatch monitors metrics like the number of health checks listed as healthy, the length of time an SSL handshake took, and the time it took for the health check to receive the first byte, among other metrics. AWS CloudTrail can capture all of the API requests made for Amazon Route 53. You can determine the user who invoked a particular API.

Amazon Route 53 Authentication and Access Control

To perform any operation on Amazon Route 53 resources, such as registering a domain or updating a resource record set, IAM requires you to authenticate to prove that you’re an approved AWS user. If you’re using the Amazon Route 53 console, you authenticate your identity by providing your AWS user name and a password. If you’re accessing Amazon Route 53 programmatically, your application authenticates your identity for you by using access keys or by signing requests.

After you authenticate your identity, IAM controls your access to AWS by verifying that you have permissions to perform operations and to access resources. If you are an account administrator, you can use IAM to control the access of other users to the resources that are associated with your account.

Amazon CloudFront

Amazon CloudFront is a web service that speeds up distribution of your static and dynamic web content—for example, .html, .css, .php, image, and media files—to end users. Amazon CloudFront delivers your content through a worldwide network of edge locations.

When an end user requests content that you’re serving with Amazon CloudFront, the user is routed to the edge location that provides the lowest latency, so content is delivered with the

best possible performance. If the content is already in that edge location, Amazon CloudFront delivers it immediately. If the content is not currently in that edge location, Amazon CloudFront retrieves it from an Amazon S3 bucket or an HTTP server (for example, a web server) that you have identified as the source for the definitive version of your content.

Amazon CloudFront can be used to distribute static content, dynamic content, and streaming media.

Amazon CloudFront Implementation

When implementing Amazon CloudFront, the first step is to configure your origin servers, from which Amazon CloudFront gets your files for distribution from Amazon CloudFront edge locations all over the world. An origin server stores the original, definitive version of your objects.

If you're serving content over HTTP, your origin server is either an Amazon S3 bucket or an HTTP server, such as a web server. Your HTTP server can run on an Amazon EC2 instance or on a server that you manage; these servers are also known as custom origins. If you distribute media files on demand using the Adobe Media Server Real-Time Messaging Protocol (RTMP), your origin server is always an Amazon S3 bucket.

The next step is to upload your files to your origin servers. Your files, also known as objects, typically include web pages, images, and media files, but they can be anything that can be served over HTTP or a supported version of Adobe RTMP, the protocol used by Adobe Flash Media Server.

The final step is to create an Amazon CloudFront distribution, which tells Amazon CloudFront which origin servers to get your files from when users request the files through your website or application. In addition, you can configure your origin server to add headers to the files; the headers indicate how long you want the files to stay in the cache in Amazon CloudFront edge locations.

At this point, Amazon CloudFront assigns a domain name to your new distribution and displays it in the Amazon CloudFront console or returns it in the response to a programmatic request. You can also configure your Amazon CloudFront distribution so that you can use your own domain name.

Now that we have discussed (at a very high level) the steps required to Implement an Amazon CloudFront distribution, let's talk about how that distribution is configured. The following are features that relate to Amazon CloudFront web distributions.

Cache Behaviors

A *cache behavior* is the set of rules that you configure for a given URL pattern based on file extensions, file names, or any portion of a URL path on your website (for example, *.jpg). You can configure multiple cache behaviors for your web distribution. Amazon CloudFront will match incoming viewer requests with your list of URL patterns, and if there is a match, the service will honor the cache behavior that you configure for that URL pattern. Each cache behavior can include the following Amazon CloudFront configuration values: origin server name, viewer connection protocol, minimum expiration period, query string parameters, and trusted signers for private content.

Regional Edge Caches

You can use Amazon CloudFront to deliver content at improved performance for your viewers while reducing the load on your origin resources. *Regional Edge Caches* sit in between your origin web server and the global edge locations that serve traffic directly to your viewers. As the popularity of your objects is reduced, individual edge locations may evict those objects to make room for more popular content. Regional Edge Caches have larger cache width than any individual edge location, so objects remain in cache longer at these Regional Edge Caches. This helps keep more of your content closer to your viewers, reducing the need for CloudFront to go back to your origin web server and improves overall performance for viewers. For instance, all our edge locations in Europe now go to the Regional Edge Cache in Frankfurt to fetch an object before going back to your origin web server. Regional Edge Cache locations are currently utilized only for requests that need to go back to a custom origin; requests to Amazon S3 origins skip Regional Edge Cache locations.

You do not need to make any changes to your Amazon CloudFront distributions; this feature is enabled by default for all CloudFront distributions. There is no additional cost for using this feature.

Origin Servers

You can configure one or more origin servers for your Amazon CloudFront web distribution. Origin servers can be an AWS resource, such as Amazon S3, Amazon EC2, Elastic Load Balancing, or a custom origin server outside of AWS. Amazon CloudFront will request content from each origin server by matching the URLs requested by the viewer with rules that you configure for your distribution. This feature allows you the flexibility to use each AWS resource for what it's designed for—Amazon S3 for storage, Amazon EC2 for compute, and so forth—with the need to create multiple distributions and manage multiple domain names on your website. You can also continue to use origin servers that you already have set up without the need to move data or re-deploy your application code. Furthermore, Amazon CloudFront allows the directory path as the origin name; that is, when you specify the origin for a CloudFront distribution, you can specify a directory path in addition to a domain name. This makes it easier for you to deliver different types of content via CloudFront without changing your origin infrastructure.

Private Content

You can use Amazon CloudFront's private content feature to control who is able to access your content. This optional feature lets you use Amazon CloudFront to deliver valuable content that you prefer not to make publicly available by requiring your users to use a signed URL or have a signed HTTP cookie when requesting your content.

Device Detection

Amazon CloudFront edge locations can look at the value of the User Agent header to detect the device type of all the incoming requests. Amazon CloudFront can determine whether the end user request came from a desktop, tablet, smart TV, or mobile device and pass that

information in the form of new HTTP headers to your origin server—Amazon EC2, Elastic Load Balancing, or your custom origin server. Your origin server can use the device type information to generate different versions of the content based on the new headers. Amazon CloudFront will also cache the different versions of the content at that edge location.

Geo Targeting

Amazon CloudFront can also detect the country from where the end users are accessing your content. Amazon CloudFront can then pass the information about the country in a new HTTP header to your custom origin server. Your origin server can generate different versions of the content for users in different countries and cache these different versions at the edge location to serve subsequent users visiting your website from the same country.

Cross-Origin Resource Sharing

Amazon CloudFront may be configured to forward the origin header value so that your origin server (Amazon S3 or a custom origin) can support cross-origin access via *Cross-Origin Resource Sharing (CORS)*. CORS defines a way for client web applications that are loaded in one domain to interact with resources in a different domain.

Viewer Connection Protocol

Content can be delivered to viewers using either the HTTP or HTTPS protocol. By default, your web distribution will accept requests on either protocol. However, if you want all of your content or certain URLs delivered only over an HTTPS connection, you can configure your distribution to only accept requests that come over HTTPS for that content. You can configure this feature separately for each URL pattern in your web distribution as part of the cache behavior for that URL pattern.

Protocol Detection

You can configure Amazon CloudFront to include the protocol (HTTP vs. HTTPS) of your end user's request as part of the cache key to identify an object uniquely in cache. This allows you to customize your content based on the protocol that your end users are using to access your content.

Custom SSL

Custom SSL certificate support lets you deliver content over HTTPS using your own domain name and your own SSL certificate. This gives visitors to your website the security benefits of Amazon CloudFront over an SSL connection that uses your own domain name in addition to lower latency and higher reliability. You can also configure CloudFront to use HTTPS connections for origin fetches so that your data is encrypted end-to-end from your origin to your end users. Configuring custom SSL certificate support is easy; you don't need to learn any proprietary code or hire any consultants to configure it for you.

You can provision SSL/TLS certificates and associate them with Amazon CloudFront distributions within minutes. Simply provision a certificate using the new AWS Certificate

Manager (ACM) and deploy it to your CloudFront distribution with a couple of clicks. Then let ACM manage certificate renewals for you. ACM allows you to provision, deploy, and manage the certificate with no additional charges.



Amazon CloudFront still supports using certificates that you obtained from a third-party certificate authority and uploaded to the IAM certificate store.

Geo Restriction

Geo Restriction or *Geoblocking* lets you choose the countries in which you want to restrict access to your content. By configuring either a whitelist or a blacklist of countries, you can control delivery of your content through Amazon CloudFront only to countries where you have the license to distribute. To enable this feature, you can either use the Amazon CloudFront API or the Amazon CloudFront Management Console. When a viewer from a restricted country submits a request to download your content, Amazon CloudFront responds with an HTTP status code 403 (Forbidden). You can also configure Custom Error Pages to customize the response that Amazon CloudFront sends to your viewers.

TTL Settings: Min, Max, and Default TTL

Amazon CloudFront lets you configure a minimum time-to-live (Min TTL), a maximum TTL (Max TTL), and a default TTL to specify how long CloudFront caches your objects in edge locations. Amazon CloudFront uses the expiration period that your origin sets on your files (through Cache-Control headers) to determine whether CloudFront needs to check the origin for an updated version of the file. If you expect that your files will change frequently, you can configure your origin to set a short expiration period on your files. Amazon CloudFront accepts expiration periods as short as 0 seconds (in which case CloudFront will revalidate each viewer request with the origin). Amazon CloudFront also honors special Cache-Control directives such as private, no-store, and so on. These are often useful when delivering dynamic content that you don't want CloudFront to cache.

If you have not set a Cache-Control header on your files, Amazon CloudFront uses the value of Default TTL to determine how long the file should be cached at the edge before Amazon CloudFront checks the origin for an updated version of the file. If you don't want to rely on the Cache-Control headers set by your origin, you can now easily override the Cache-Control headers by setting the same value for Max TTL, Min TTL, and Default TTL. By setting both a Min TTL and a Max TTL, you can override origin misconfigurations that might cause objects to be cached for longer or shorter periods than you intend. Min TTL, Max TTL, and Default TTL values can be configured uniquely for each of the cache behaviors you define. This allows you to maximize the cache duration for different types of content on your site by setting a lower bound, upper bound, or a default value on the length of time each file can remain in cache.

Query String Parameters

Query string parameters are often used to return customized content generated by a script running on the origin server. By default, Amazon CloudFront does not forward query string parameters (for example, "?x=1&y=2") to the origin. In addition, the query string portion of the URL is ignored when identifying a unique object in the cache. However, you can optionally configure query strings to be forwarded to the origin servers and be included in the unique identity of the cached object. This feature can be enabled separately for each unique cache behavior that you configure. Query string parameters can thus help you customize your web pages for each viewer while still taking advantage of the performance and scale benefits offered by caching content at Amazon CloudFront edge locations.

GZIP

You can configure Amazon CloudFront to apply GZIP compression automatically when browsers and other clients request a compressed object with text and other compressible file formats. This means that if you are already using Amazon S3, CloudFront can transparently compress this type of content. For origins outside S3, doing compression at the edge means that you don't need to use resources at your origin to do compression. The resulting smaller size of compressed objects makes downloads faster and reduces your CloudFront data transfer charges. To use the feature, simply specify within your cache behavior settings that you would like CloudFront to compress objects automatically and ensure that your client adds `Accept-Encoding: gzip` in the request header (most modern web browsers do this by default).

HTTP Cookie Support

Amazon CloudFront supports delivery of dynamic content that is customized or personalized using HTTP cookies. To use this feature, you specify whether you want Amazon CloudFront to forward some or all of your cookies to your custom origin server. You may also specify wildcard characters in the cookie name to forward multiple cookies matching a string format. Amazon CloudFront then considers the forwarded cookie values when identifying a unique object in its cache. This way, your end users get both the benefit of content that is personalized just for them with a cookie and the performance benefits of Amazon CloudFront.

Forward Headers to Origin

You can use Amazon CloudFront to forward all (or a whitelist of) request headers to your origin server. These headers contain information, such as the device used by your visitors or the country from which they accessed your content. You can configure CloudFront to cache your content based on the values in the headers, so that you can deliver customized content to your viewers. For example, if you are hosting multiple websites on the same web server, you can configure Amazon CloudFront to forward the Host header to your origin. When your origin returns different versions of the same object based on the values in the Host header, Amazon CloudFront will cache the objects separately based on those values.

Add or Modify Request Headers Forwarded from Amazon CloudFront to Origin

You can configure Amazon CloudFront to add custom headers or override the value of existing request headers when CloudFront forwards requests to your origin. You can use these headers to help validate that requests made to your origin were sent from CloudFront (shared secret) and configure your origin only to allow requests that contain the custom header values that you specify. This feature also helps with setting up Cross-Origin Request Sharing (CORS) for your CloudFront distribution: You can configure CloudFront always to add custom headers to your origin to accommodate viewers that don't automatically include those headers in requests. It also allows you to disable varying on the origin header, which improves your cache hit ratio, yet forward the appropriate headers so that your origin can respond with the CORS header.

Enforce HTTPS-Only Connection Between Amazon CloudFront and Your Origin Web Server

You can configure Amazon CloudFront to connect to your origin server using HTTPS regardless of whether the viewer made the request by using HTTP or HTTPS.

Support for TLSv1.1 and TLSv1.2 Between Amazon CloudFront and Your Origin Web Server

Amazon CloudFront supports the TLSv1.1 and TLSv1.2 protocols for HTTPS connections between CloudFront and your custom origin web server (along with SSLv3 and TLSv1.0). You can choose the protocols that you want CloudFront to use when communicating with your origin so that you can, for example, choose not to allow CloudFront to communicate with your origin by using SSLv3, which is less secure than TLS.

Default Root Object

You can specify a default file (for example, `index.html`) that will be served for requests made for the root of your distribution without an object name specified, for instance, requests made to `http://abc123.cloudfront.net/` alone, without a file name.

Object Versioning and Cache Invalidation

You have two options to update your files cached at the Amazon CloudFront edge locations. You can use *object versioning* to manage changes to your content. To implement object versioning, you create a unique file name in your origin server for each version of your file and use the file name corresponding to the correct version in your web pages or applications. With this technique, Amazon CloudFront caches the version of the object that you want without needing to wait for an object to expire before you can serve a newer version.

You can also remove copies of an object from all Amazon CloudFront edge locations at any time by calling the *invalidation API*. This feature removes the object from every Amazon CloudFront edge location regardless of the expiration period you set for that object.

on your origin server. If you need to remove multiple objects at once, you may send a list of invalidation paths (up to 3,000) in an XML document. Additionally, you can request up to 15 invalidation paths with a wildcard character. The invalidation feature is designed to be used in unexpected circumstances; for example, to correct an encoding error on a video you uploaded or an unanticipated update to your website's CSS file. However, if you know beforehand that your files will change frequently, it is recommended that you use object versioning to manage updates to your files. This technique gives you more control over when your changes take effect, and it also lets you avoid potential charges for invalidating objects.

Access Logs

You can choose to receive more information about the traffic delivered or streamed by your Amazon CloudFront distribution by enabling access logs. *Access logs* are activity records that show you detailed information about each request made for your content. CloudFront access files are automatically delivered multiple times per hour and the logs in those files will typically be available within an hour of your viewers requesting that object.

Amazon CloudFront Usage Charts

Amazon *CloudFront Usage Charts* let you track trends in data transfer and requests (both HTTP and HTTPS) for each of your active CloudFront web distributions. These charts show your usage from each CloudFront region at daily or hourly granularity, going back up to 60 days. They also include totals, average, and peak usage during the time interval selected.

Amazon CloudFront Monitoring and Alarming Using Amazon CloudWatch

You can monitor, alarm, and receive notifications on the operational performance of your Amazon CloudFront distributions using Amazon CloudWatch, giving you more visibility into the overall health of your web application. CloudFront automatically publishes six operational metrics, each at 1-minute granularity, into Amazon CloudWatch. You can then use CloudWatch to set alarms on any abnormal patterns in your CloudFront traffic. These metrics appear in CloudWatch within a few minutes of the viewer's request for each of your Amazon CloudFront web distributions.

Zone Apex Support

You can use Amazon CloudFront to deliver content from the root domain, or "zone apex" of your website. For example, you can configure both <http://www.example.com> and <http://example.com> to point at the same CloudFront distribution, without the performance penalty or availability risk of managing a redirect service.

Using Amazon CloudFront with AWS WAF to Protect Your Web Applications

AWS WAF is a web application firewall that helps detect and block malicious web requests targeted at your web applications. AWS WAF allows you to create rules based on IP addresses, HTTP headers, and custom URIs. Using these rules, AWS WAF can block, allow, or monitor (count) web requests for your web application.

HTTP Streaming of On-Demand Media

Amazon CloudFront can be used to deliver your on-demand adaptive bit-rate media content at scale to a global audience. Whether you want to stream your content using Microsoft Smooth Streaming format to Microsoft Silverlight players or stream to iOS devices using HTTP Live Streaming (HLS) format, you can do so using Amazon CloudFront without the need to set up and manage any third-party media servers. Furthermore, there are no additional charges for using this capability beyond Amazon CloudFront's standard data transfer and request fees. Simply encode your media files for the format you want to use and upload it to the origin they plan to use.

On-demand Smooth Streaming You can specify in the cache behavior of an Amazon CloudFront web distribution to support Microsoft Smooth Streaming format for that origin.

On-demand HLS Streaming Streaming on-demand content using the HLS format is supported in Amazon CloudFront without having to do any additional configurations. You store your content in your origin (for example, Amazon S3). Amazon CloudFront delivers this content at a global scale to a player (such as the iOS player) requesting the HLS segments for playback.

RTMP Distributions for On-Demand Media Delivery

Amazon CloudFront lets you create *RTMP distributions*, which deliver content to end users in real time—the end users watch the bytes as they are delivered. Amazon CloudFront uses Adobe's Flash Media Server 3.5 to power its RTMP distributions. RTMP distributions use the Real-Time Messaging Protocol (RTMP) and several of its variants, instead of the HTTP or HTTPS protocols used by other Amazon CloudFront distributions.

Content Delivery

Now that you have configured Amazon CloudFront to deliver your content, the following will happen when users request your objects:

1. A user accesses your website or application and requests one or more objects.
2. DNS routes the request to the Amazon CloudFront edge location that can best serve the user's request, typically the nearest Amazon CloudFront edge location in terms of latency, and routes the request to that edge location.
3. In the edge location, Amazon CloudFront checks its cache for the requested files. If the files are in the cache, Amazon CloudFront returns them to the user. If the files are not in the cache, CloudFront does the following:
 - a. Amazon CloudFront compares the request with the specifications in your distribution and forwards the request for the files to the applicable origin server for the corresponding file type.
 - b. The origin servers send the files back to the Amazon CloudFront edge location.
 - c. As soon as the first byte arrives from the origin, Amazon CloudFront begins to forward the files to the user.
 - d. Amazon CloudFront also adds the files to the cache in the edge location for the next time someone requests those files.

As mentioned earlier, you can configure headers to indicate how long you want the files to stay in the cache in Amazon CloudFront edge locations. By default, each object stays in an edge location for 24 hours before it expires. The minimum expiration time is 0 seconds; there isn't a maximum expiration time limit.

The Amazon CloudFront console includes a variety of reports:

Amazon CloudFront cache statistics reports These reports use the Amazon CloudFront console to display a graphical representation of statistics related to CloudFront edge locations. Data for these statistics are drawn from the same source as CloudFront access logs. You can display charts for a specified date range in the last 60 days, with data points every hour or every day.

Amazon CloudFront popular objects reports These reports are available via the Amazon CloudFront console. You can display a list of the 50 most popular objects for a distribution during a specified date range in the previous 60 days.

Amazon CloudFront top referrers reports These reports provide a list of the 25 domains of the websites that originated the most HTTP and HTTPS requests for objects that CloudFront is distributing for a specified distribution. These top referrers can be search engines, other websites that link directly to your objects, or your own website.

Amazon CloudFront usage reports These are reports which are more detailed than the billing report but less detailed than CloudFront access logs. The usage report provides aggregate usage data by hour, day, or month, and it lists operations by region and usage type.

Amazon CloudFront viewers reports These reports show devices, browsers, and operating systems' versions used to access your content, and also from what locations they are accessing your content.

Amazon CloudFront Management

You can configure Amazon CloudFront using the AWS Management Console, the Amazon CloudFront console, the AWS CLI, or various SDKs available for Amazon CloudFront.

Amazon CloudFront metrics are also accessible in the Amazon CloudWatch Console.

Amazon CloudFront Security

You can control user access to your private content in two ways:

1. Restrict access to objects in the Amazon CloudFront edge cache using either signed URLs or signed cookies.
2. Restrict access to objects in your Amazon S3 bucket so that users can access it through Amazon CloudFront, thus direct access to the S3 bucket. See Amazon S3 bucket policies in Chapter 6, “Storage Systems,” for more details.



Understanding how Amazon CloudFront works and how it can deliver content to your end users is important. Knowing how to manage content both at the origin and at the edge is crucial.

Summary

We covered a lot of material in this chapter. While the exam questions may not go into as great a depth in terms of detail, understanding this material will assist you in providing the best answers to the questions on the exam.

In this chapter, we discussed the following:

- Amazon VPC as a logically-isolated section of the AWS Cloud
- AWS Direct Connect and how it allows you to establish a dedicated network connection from your premises to AWS
- The two types of Elastic Load Balancers and their features (Application and Classic Load Balancers)
- How Elastic Load Balancers automatically distribute incoming application traffic across multiple Amazon Elastic Compute Cloud (Amazon EC2) instances within an AWS Region
- Virtual Private Network (VPN) connections and how you can connect Amazon VPC to remote networks to make your AWS infrastructure highly available
- Internet gateways, which are used to connect your public subnets to the Internet
- NAT gateways, which are used to provide connectivity to the Internet from your private instances
- Elastic Network interfaces, which are used to multi-hone an Amazon EC2 instance and can be re-assigned to another Amazon EC2 instance
- Elastic IP addresses (EIP), which are public IPv4 addresses that can be assigned and reassigned to Amazon EC2 instances. IPv6 is not (currently) supported with EIP.
- You learned that Amazon Route 53 is a highly available and scalable cloud Domain Name System (DNS) web service. We discussed the various routing types: Simple, Weighted Round Robin (WRR), Latency Based Routing (LBR), geolocation, and Failover routing.
- You learned that Amazon CloudFront is a global Content Delivery Network (CDN) service that accelerates delivery of your websites, Application Programming Interfaces (APIs), video content, or other web assets.

Resources to Review

AWS YouTube channel: <https://www.youtube.com/user/AmazonWebServices>

AWS What's new: <https://aws.amazon.com/new>

Jeff Barr's blog: <https://aws.amazon.com/blogs/aws/>

Amazon VPC documentation: <https://aws.amazon.com/documentation/vpc>

Amazon VPC peering guide:

<http://docs.aws.amazon.com/AmazonVPC/latest/PeeringGuide/Welcome.html>

VPN options:

<http://docs.aws.amazon.com/AmazonVPC/latest/UserGuide/vpn-connections.html>

NAT gateway fundamentals on AWS:

<http://docs.aws.amazon.com/AmazonVPC/latest/UserGuide/vpc-nat-gateway.html>

Amazon CloudFront documentation:

<https://aws.amazon.com/documentation/cloudfront/>

Amazon Route 53 documentation: <https://aws.amazon.com/documentation/route53>

Elastic Load Balancing documentation:

<https://aws.amazon.com/documentation/elastic-load-balancing/>

AWS Direct Connect and VPN deep dive:

<https://www.youtube.com/watch?v=Qep11X1r1QA>

Amazon CloudFront best practices: <https://www.youtube.com/watch?v=fgbJJ412qRE>

Exam Essentials

Understand what a VPC is. Know how to set up a VPC, and what are the minimum and maximum size of both a VPC and subnets.

Understand the purpose and use of route tables, network ACLs, and security groups. Know how to use each for controlling access and providing security.

Know what are the default values for route tables, network ACLs, and security groups. Know where those default values come from, how to modify them, and why you would modify them.

Understand the difference between a private and public subnet. Public subnets allow traffic to the Internet; private subnets do not. Know how to use Amazon EC2 instances in private subnets to have access the Internet.

Understand the role and function of the various ways to connect the VPC with outside resources. This includes Internet gateway, VPN gateway, Amazon S3 endpoint, VPC peering, NAT instances, and NAT gateways. Understand how to configure these services.

Understand what is an Elastic IP (EIP). Elastic supports public IPv4 addresses (as of this publication). Understand the difference between EIP and an ENI.

Understand what is an Elastic Network Interface (ENI). Elastic Network Interfaces can be assigned and reassigned to an Amazon EC2 instance. Understand why this is important.

Know what services operate within a VPC and what services operate outside a VPC. Amazon EC2 lives within a VPC. Services such as Amazon S3 live outside the VPC. Know the various ways to access these services.

Know what AWS Direct Connect is. Understand why it is used and the basic steps for setting it up. (Remember the seven steps listed in the AWS Direct Connect section of this chapter.)

Understand the concept of VIFs. Understand what a VIF is and the difference between a public and private VIF. Why would you use one versus the other? Understanding these concepts will be very helpful on the exam.

Understand the options for Elastic Load Balancing (Classic Load Balancer vs. Application Load Balancer). Know how each type of load balancer operates, why you would choose one over the other, and how to configure each.

Understand how health checks work in each type of load balancer. Classic Load Balancers and Application Load Balancers have different health check options. Know what they are!

Understand how listeners work. Understand rules, priorities, and conditions and how they interact.

Know how Amazon CloudWatch, AWS CloudTrail, and access logs work. Know what type of information each one provides.

Understand the role of security groups with load balancers. Be able to configure a security group and know how rules are applied.

Understand the various options for establishing an IPsec VPN tunnel from an Amazon VPC to a customer location. Know the operational and security implications of these options.

Know how Amazon Route 53 works as a DNS provider. Understand how it can be used for both public and private hosted zones.

Know what the different routing options are for Amazon Route 53. Understand how to configure the various routing options and how they work.

Know what an Amazon Route 53 routing policy is. Understand how it is applied in Amazon Route 53.

Understand what record types Amazon Route 53 supports and how they work. Know both standard and non-standard record sets.

Know the tools for managing and monitoring Amazon Route 53. Understand how Amazon CloudWatch and AWS CloudTrail work with Amazon Route 53. Go deep in understanding how all of the services in this chapter are monitored.

Know the purpose of Amazon CloudFront and how it works. Know what a distribution is and what an origin is. Know what types of files Amazon CloudFront can handle.

Know the steps to implement Amazon CloudFront. Remember there are three steps to do this.

Know the various methods for securing content in Amazon CloudFront. Know how to secure your content at the edge and at the origin.

Test Taking Tip

For this exam, focus on what services do by themselves and what they do in partnership with other services. For example, Amazon CloudWatch does not send notifications; Amazon SNS does.

Exercises

By now you should have set up an account in AWS. If you haven't, now would be the time to do so. It is important to note that these exercises are in your AWS account and thus are not free.

Use the Free Tier when launching resources. The AWS Free Tier applies to participating services across the following AWS Regions: US East (Northern Virginia), US West (Oregon), US West (Northern California), Canada (Central), EU (London), EU (Ireland), EU (Frankfurt), Asia Pacific (Singapore), Asia Pacific (Tokyo), Asia Pacific (Sydney), and South America (Sao Paulo). For more information, see https://aws.amazon.com/sdm/optimization/server-side-test/free-tier/free_np/.

Remember to delete and terminate resources to minimize usage charges.

EXERCISE 5.1**Create an Elastic IP (EIP).**

1. Sign in to the AWS Management Console.
2. Type **EC2** in the AWS Services Search box, and select EC2.
3. Navigate to the Elastic IP console.
4. Allocate a new address.
5. Make note of the public IPv4 address.

You have just created an Elastic IP address. Hold on to it, as you will use it in the next exercise.

EXERCISE 5.2**Create an Amazon VPC.**

1. Sign in to the AWS Management Console.
2. Type **VPC** in the AWS Services Search box, and select VPC.
3. Use the VPC wizard, and select VPC with public and private subnets.
4. Use a CIDR block equal to 10.0.0.0/16 and a VPC name of VPC-1.

5. Assign a public CIDR block equal to 10.0.0.0/24 and no preference for the Availability Zone.
6. Assign a private CIDR block equal to 10.0.1.0/24 and no preference for the Availability Zone.
7. Assign a NAT gateway, and use the Elastic IP from the previous exercise.
8. Assign an Amazon S3 endpoint, and leave the default DNS and tenancy options.

You have just created a VPC with two subnets (one public and one private), an Amazon S3 endpoint, and a NAT gateway.

EXERCISE 5.3

Tag Your Amazon VPC and Subnets.

1. Sign in to the AWS Management Console.
2. Type **VPC** in the AWS Services Search box, and select VPC.
3. Navigate to your VPCs.
4. Select your VPC. (*Hint:* CIDR = 10.0.0.0/16.)
5. Navigate to the Tags console, and add VPC1 to the value of the name key.
6. Go to the Subnets Console, and add the following key values:

Public and Private to the name keys (*Hint:* 10.0.0.0/24 = public).

You have just tagged your VPC and subnets.

EXERCISE 5.4

Create an Elastic Network Interface (ENI).

1. Sign in to the AWS Management Console.
2. Type **EC2** in the AWS Services Search box, and select EC2.
3. Select Network Interfaces. Create it in the public subnet that you made in Exercise 5.2 and the default security group of your VPC.
4. Create an Elastic IP. (See Exercise 5.1.)
5. Go to the Amazon EC2 console, and navigate to the Network Interfaces Console.
6. Create a network interface, and select your public subnet with the default subnet group.
7. Tag the network interface. (*Hint:* Go to the Network Interfaces Console.)

You have just created an Elastic Network Interface.

EXERCISE 5.5**Associate the ENI.**

1. Sign in to the AWS Management Console.
2. Create an Amazon EC2 instance, and launch it into the public subnet, security group from Exercise 4.1, with a tag of key = name, value = Exercise 5.5. Use **CertBookKeyPair**. See Chapter 4, and follow Exercise 4.1 for complete details.
3. After the Amazon EC2 instance is launched, navigate to the Network Interfaces Console, and select the ENI that you created in Step 5.4, Step 4.
4. Attach your ENI to your newly created Amazon EC2 instance.

You have just attached your ENI to an Amazon EC2 instance.

EXERCISE 5.6**Test Your ENI.**

1. Sign in to the AWS Management Console.
2. Navigate to the Amazon EC2 console.
3. Find your Amazon EC2 instance (the one you launched in Exercise 5.5). Notice that you have two IP addresses.
4. Create an EIP (see Exercise 5.1 for details), and associate it with the network interface. (The tag that you created will make finding your network interface much easier.)
5. Go back to the Amazon EC2 Console, and take note of the IP addresses.
6. SSH in to the public IP address of the instance.

You have just tested connectivity to your Amazon EC2 instance via the Elastic Network Interface (eth1).

EXERCISE 5.7**Delete VPC.**

1. Sign in to the AWS Management Console.
2. Navigate to the VPC Section of Services.
3. Choose the VPC you wish to delete via the radio button on the left.
4. Under the “Actions” drop-down box choose Delete VPC.
5. Confirm the deletion.

You have just deleted a VPC.

Review Questions

1. You have implemented a Classic Load Balancer. You now need to collect some information, specifically what is the IP address of the client making the request on the Classic Load Balancer. How would you collect this information?
 - A. Enable CloudWatch and monitor the HostConnect metric.
 - B. Use AWS CloudTrail and monitor the eventSource API.
 - C. You would not be able to collect this information in a Classic Load Balancer—it is only available with Application Load Balancers.
 - D. Use access logs.
2. You just updated the port of the health check for an Application Load Balancer. You go into the CloudWatch console, but you are not seeing the health check. What could possibly be the reason?
 - A. The CloudWatch console does not display information on a number of healthy hosts.
 - B. You find this information in the Amazon EC2 Management console.
 - C. You should review your security group's rules to make sure that traffic is allowed to that port.
 - D. You need to restart your Application Load Balancer after you make this change.
3. You need to establish a highly available connection between your Amazon VPC and your datacenter. What is the best way to accomplish this?
 - A. Create an AWS Direct Connect connection between your datacenter and your AWS VPC.
 - B. Spin up multiple Amazon EC2 instances across two Availability Zones. Load VPN software onto the Amazon EC2 instances. Set internal routing such that if one Amazon EC2 instance fails the other takes over.
 - C. Set up a Virtual Private Gateway with a route out to your datacenter.
 - D. Set up a Virtual Private Gateway. Make sure that you have two customer gateways configured.
4. You are using Amazon Route 53 as your DNS provider. You have a web application that is running in your datacenter located in Las Vegas, NV and in the AWS Frankfort, Germany Region. What steps would you take to minimize the load times for this web application?
 - A. Implement a geolocation routing policy, where all requests from users in the United States are routed to your Las Vegas location, and everything in Europe is routed to your AWS service in the Frankfort region.
 - B. Set up your web application in an AWS Region in the United States because you are not able to route traffic to non-AWS locations.
 - C. Set up a simple routing policy that routes all European traffic to Frankfort and all United States traffic to Las Vegas.
 - D. Set up a weighted routing policy, and split that traffic evenly between Frankfort and Las Vegas.
 - E. Set up latency-based routing on your service.

5. You are trying to SSH in to an Amazon EC2 instance in your Amazon VPC but are unable to do so. What should you be checking?
 - A. Make sure that you have a Virtual Private Gateway attached to your VPC, that the VPC route table has an entry that routes packets to the Internet, and that the Network ACL has an inbound rule that allows traffic on port 80.
 - B. Make sure that you have an Internet gateway attached to your VPC, that the VPC route table has an entry that routes packets to the Internet, and that the Network ACL has an inbound rule that allows SSH.
 - C. Make sure that you have an Internet gateway attached to your Amazon VPC, that the VPC route table has an entry that routes packets to the Internet, that the Network ACL has an inbound and an outbound rule that allows SSH, and that the Amazon EC2 instance has a security group rule that allows inbound SSH.
 - D. Make sure that you have an Internet gateway attached to your Amazon VPC, that the VPC route table has an entry that routes packets to the Internet, that the Network ACL has an inbound and an outbound rule that allows SSH, and that the Amazon EC2 instance has a security group rule that allows inbound SSH. Make sure that the EC2 instance has a Public or Elastic IP address associated with it.
6. Why would you place an Amazon EC2 instance in a private subnet?
 - A. To decrease the latency in reaching the instance
 - B. Because you have more available IP addresses in a private subnet than you do in a public subnet
 - C. As a way of providing an additional layer of security
 - D. Because with some Amazon EC2 instances, you are obligated to place them in a private subnet
7. You need to order an AWS Direct Connect circuit. What do you need on your side to implement AWS Direct Connect successfully?
 - A. A router that supports BGP, with single mode fiber, and with a 1 or 10 Gig Ethernet port. You also need both a public and a private Autonomous System Number.
 - B. A router that supports OSPF and has a 10 Gig Ethernet port. You also need a private Autonomous System Number.
 - C. A switch that supports single mode fiber with a 1 or 10 Gig Ethernet port
 - D. A router that supports BGP with single mode fiber and with a 1 or 10 Gig Ethernet port. You also need both a public and private Autonomous System Number. Finally, you need the ability to issue a LOA/CFA to AWS.
 - E. A router that supports static routing with single mode fiber and that has a 1 or 10 Gig Ethernet port
8. What would NOT be a reason to get AWS Direct Connect?
 - A. Increased latency
 - B. Decreased data transfer out costs
 - C. Connecting VPCs in different regions
 - D. Connectivity between your WAN and AWS

9. What is a private VIF?
- A. The physical connection between AWS and the customer location
 - B. The logical interface between the customer location and those AWS resources located inside the VPC
 - C. The logical interface between the customer location and those AWS services located outside the VPC
 - D. The logical connection between two VPCs when you establish VPC peering
10. The network ACL shown here has been applied as an inbound rule to a subnet. What statement is correct?

| Rule # | Type | Protocol | Port Range | Source | Allow / Deny |
|--------|-------------|----------|------------|-----------|--------------|
| 50 | SMTP (25) | TCP (6) | 25 | 0.0.0.0/0 | DENY |
| 100 | ALL Traffic | ALL | ALL | 0.0.0.0/0 | ALLOW |
| * | ALL Traffic | ALL | ALL | 0.0.0.0/0 | DENY |

- A. All traffic is blocked in both directions.
 - B. All traffic is allowed, except for SMTP, which is blocked inbound and outbound.
 - C. All traffic is allowed inbound, except for any response to an SMTP packet.
 - D. All traffic is allowed inbound, except SMTP, which is blocked within the subnet.
 - E. All traffic is allowed inbound, except SMTP, which is blocked from entering the subnet.
11. What statement is true about Internet gateways?
- A. For high availability, you should have one Internet gateway per Availability Zone.
 - B. Internet gateways come with public IP addresses already assigned.
 - C. You cannot have a VPC with both an Internet gateway and a Virtual Private Network gateway.
 - D. An Internet gateway is needed if you want to connect to AWS services outside of the VPC.
12. You have noticed that your web servers have come under a phishing attack. You have identified the IP address that is the source of this attack. What should you do to mitigate this attack?
- A. Configure a route table that directs packets from this IP address to a fictitious Amazon EC2 instance.
 - B. Configure the Network ACLs to block traffic from this IP address.
 - C. Configure the security group for your web servers to deny any protocols from this IP address.
 - D. Contact the AWS Help Desk, and ask them to put a block on the offending subnet.

- 13.** You have established three VPCs all within the same region but in different accounts. What is the easiest way to establish connectivity between all three VPCs?
- A.** Designate one VPC as the master, and establish VPN peering between the master VPC and each of the other VPCs.
 - B.** Establish an AWS Direct Connect connection among all of the VPCs.
 - C.** Establish a CloudHub with all three VPCs as participants.
 - D.** Establish VPC peering between each pair of VPCs.
 - E.** Install a Virtual Private Gateway (VPG) in each VPC, and establish an IPsec tunnel between each VPC using the AWS infrastructure.
- 14.** What is the difference between an Internet-facing load balancer and an internal-facing load balancer? (Choose two.)
- A.** There is no difference between the two.
 - B.** Internet-facing load balancers are larger than internal load balancers.
 - C.** By default, Internet-facing load balancers get their DNS names from DHCP servers, while internal load balancers do not.
 - D.** The DNS name of an Internet-facing load balancer is publicly resolvable to the public IP addresses of the nodes.
 - E.** The DNS name of an internal load balancer is publicly resolvable to the private IP addresses of the nodes.
- 15.** What does a default VPC come with?
- A.** A /20 address space
 - B.** Both an Internet gateway and a Virtual Private Network gateway
 - C.** A route table that sends all IPv4 traffic destined for the Internet to the Internet gateway
 - D.** A NAT instance
- 16.** You need to monitor all traffic from the Internet to Amazon EC2 instances in a VPC. What AWS tool do you have at your disposal?
- A.** Amazon VPC Flow Logs
 - B.** Amazon CloudWatch
 - C.** AWS CloudTrail
 - D.** AWS Network Management Console
- 17.** Which statement is correct regarding Amazon CloudFront?
- A.** Amazon CloudFront will forward a file to the user as soon as it gets the first bytes.
 - B.** Amazon CloudFront will wait until the entire file downloads in order to perform error checking before it forwards the file to the user.
 - C.** Amazon CloudFront always delivers the most current version of the file to the user.
 - D.** Amazon CloudFront is only located in AWS Regions.

18. What are the best ways to control access to your content in the Amazon CloudFront edge locations? (Choose two.)
- A. Use Origin Access Identity (OAI).
 - B. Signed URLs
 - C. Signed cookies
 - D. Policies that restrict access by IP address

19. You have created a VPC with an Internet Gateway attached. The VPC is 10.155.0.0/16. You have created a subnet in that VPC. The Subnet is 10.155.1.0/23. What is the Route Table for the subnet?

| A. | Destination | Target | Status | Propagated |
|----|---------------|--------|--------|------------|
| | 10.0.0.0/16 | local | Active | No |
| | 10.155.0.0/23 | local | Active | No |

| B. | Destination | Target | Status | Propagated |
|----|-------------|--------|--------|------------|
| | 10.0.0.0/16 | local | Active | No |

| C. | Destination | Target | Status | Propagated |
|----|---------------|--------|--------|------------|
| | 10.155.0.0/16 | local | Active | No |

- D. There is no Route Table. When you create a subnet, you have to define a Route Table.
20. When you configure Amazon CloudFront, an origin refers to which of the following?
- A. The AWS server that is holding your static content.
 - B. Either an HTTP server or an Amazon S3 bucket.
 - C. For static content only, it is an Amazon S3 bucket.
 - D. For static content, it is either an HTTP server or an Amazon S3 bucket. For media files on demand it is an S3 bucket.

Chapter

6



Storage Systems

THE AWS CERTIFIED SYSOPS ADMINISTRATOR - ASSOCIATE EXAM TOPICS COVERED IN THIS CHAPTER MAY INCLUDE, BUT ARE NOT LIMITED TO, THE FOLLOWING:

Domain 4.0: Deployment and Provisioning

- ✓ 4.1 Demonstrate ability to build the environment to conform to the architected design

Content may include the following:

- Optimizing storage for performance

- ✓ 4.2 Demonstrate ability to provision cloud resources and manage implementation automation

Content may include the following:

- Deploying storage solutions on the cloud
- Understanding the different storage options available on the AWS platform
- Optimizing storage for performance

Domain 5.0: Data Management

- ✓ 5.1 Demonstrate ability to implement backups for different services

Content may include the following:

- Working with snapshots
- Using AWS Storage Gateway

- ✓ 5.2 Demonstrate ability to enforce compliance requirements

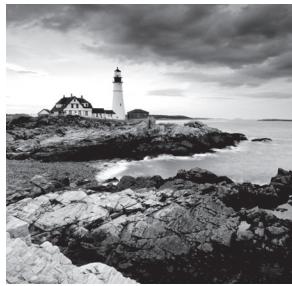


Domain 6.0: Security

✓ 6.2 Ensure data integrity and access controls when using the AWS platform

Content may include the following:

- How to secure data at rest
- Using Amazon Simple Storage Service (Amazon S3) lifecycle policies
- Using Amazon Glacier vaults



Understanding Different Storage Options

One of the advantages of running your application on AWS is the wide variety of storage options that are available to you. AWS provides solutions to allow your team to optimize for the business needs of specific data. Variances in read or write capacity, simultaneous access options, storage cost, durability, availability, and retrieval speeds all become factors in choosing the right service. For any given enterprise application, you may find yourself managing many, if not all, of these different options as you optimize for different use cases.

Block Storage vs. Object Storage

The first factor in choosing a storage solution concerns *block storage vs. object storage*. Although this is primarily an architecture discussion, as systems operators you need to be aware of their primary differentiators in order to optimize usage on your end.

The key difference between block and object storage systems is the fundamental unit of storage. For block storage, each file (object) being saved to the drive is broken down into “blocks” of a specific size. If you were using 512 byte blocks and had a 10 KB file, you would create 20 blocks worth of drive space. Each block is saved and tracked independently of the others.

For object storage, each file is saved as a single object regardless of size. A 10 KB file is saved as a single 10 KB object, and a 30 MB file is saved as a single 30 MB object.

The best way to visualize this behavior is to compare what happens on a file update.



Real World Scenario

Updating a File (Block Storage vs. Object Storage)

In this scenario, we begin with a chapter for a book. After all of the graphics are embedded in the document, and the file size happens to be 30 MB. After it is saved to storage, the editor does one final review and she finds a capitalization error on the last page of the chapter. She finishes the edit and overwrites the file with the correction (the update is the key to this scenario).

If she was using block storage (assuming the 512-byte block), a 30 MB file is broken down into 60,000 blocks. Her edit of a single capitalization error only affected one out of the 60,000, so her save action only updates the one 512-byte block (or the delta—the difference between the documents) and her save is complete. This is a great method for objects with regular updates because the save only has to account for the delta and the rest of the object is untouched.

If she was using object storage, there is no concept of just updating the delta. Each save creates a unique, discrete object. An update will simply discard the previous object and upload all 30 MB as a brand new object.

On the surface, the object storage approach seems like a waste of network bandwidth and time required for loading a complete document just to update a single character. As you look closer at the specific use cases for block vs. object storage, however, you will understand why they behave so differently and why one would be preferable to the other in certain circumstances.

Block Storage Basics

To make it easier to understand, *block storage* is mountable drive storage. AWS provides unformatted drive space, then based on your operating system configuration, you choose the format, block size, and other factors. In most cases, these formatting decisions are pre-made by the Amazon Machine Image (AMI) selected, but AWS does not prevent you from making your own selections.

Block storage volumes are provisioned and attached to Amazon Elastic Compute Cloud (Amazon EC2) instances. In the case of Amazon Elastic Block Store (Amazon EBS), the volume's lifecycle is independent of the instance itself.

Content management in block storage is controlled entirely by the operating system of the instance to which the volume is attached. AWS has no visibility inside the individual blocks; it only has visibility into the volume properties (such as volume encryption, volume size, IOPS).

Object Storage Basics

Unlike block storage, which functions as provisioned, mountable volumes controlled by the operating systems, *object storage* is 100 percent Application Programming Interface (API) driven. Applications with the proper credentials and authorization make calls to the object storage services for reads, writes, updates, deletes, and other functions.

Because all actions in and out of AWS object storage are API controlled, AWS can provide you with much more granular control over your content and more visibility of actual usage.

With the notable exception of AWS Snowball, all AWS object storage systems are regional in scope. Content placed in Amazon Simple Storage Service (Amazon S3) or Amazon Glacier is automatically replicated across multiple facilities in multiple Availability Zones, providing data durability that is not possible with single datacenter storage solutions. A later section in this chapter provides more information on durability and availability in object storage.

Because AWS has more visibility into your interactions with content in object storage, AWS does not require any pre-provisioning for object storage (again with the notable exception of AWS Snowball). You are billed only for the exact volume in storage for the hours that it is in storage.

Retrieval Times (Hot vs. Cold Storage)

When choosing the right storage medium, for both block storage and object storage, the second criteria to consider is *retrieval times*. In many cases, this consideration is framed in terms of hot and cold storage options.

The classic example is Amazon S3 vs. Amazon Glacier (which is examined in more detail later in this chapter). Amazon S3 is a highly available system where content can be delivered to many requests simultaneously for high and immediate parallel throughput. Amazon Glacier is designed for cold storage of content that is not needed regularly, and when it is needed for auditing or analysis, operations can wait three to five hours for the content to be made available.

Block storage does not have such a dramatic differentiation, but there are still media selections that can have an effect on the volume and velocity of requests in and out of the provisioned drives. The easy example here that you might see on the certification exam is the difference between a 1 TB magnetic volume and a 1 TB Solid State Drive (SSD) general purpose (gp2) volume in Amazon EBS. The first has a hard cap of no more than 100 IOPS while the gp2 volume has provisioned 3,000 IOPS as a baseline. If your data needed high throughput without bottlenecks, you would want to stay in the gp2 line of volumes.



As of the time of this writing, the largest Amazon EBS volume size is 16 TB (gp2, io1, sc2, and st1 types). The previous magnetic generation was 1 TB (standard type).

Cost Efficiency

When choosing and managing the proper storage for your content, the third major criteria is *cost*. This book does not list actual prices because they can be different across the various regions and because AWS is constantly lowering prices wherever they can engineer new efficiencies. Be aware of the current pricing sheet for your AWS Region of choice.

The driving principles of AWS pricing are that customers only pay for what they provision, and customers with efficient operations only provision what they are actually using.

With block storage, this is specifically an operational consideration. Block storage volumes are provisioned for a specific size, and you are billed based on provisioned size each hour. AWS pricing charts show block storage as a monthly rate to avoid publishing a long list of zeros after a decimal place, even though the hourly rate is the actual billing that is applied to your volume.

Cost-efficient methods will focus on not overprovisioning volumes based on projected storage needs; rather, they will provision based on actual amounts and then grow the storage as needed. Provisioning a 3 TB drive based on estimates of needing that much in the next few years, even though current data sizes are only around 100 MB, would be a classic example of a cost scenario that systems operators can and should address.

With object storage, there is no concept of pre-provisioning (once again with the exception of AWS Snowball). AWS knows exactly how much data you have in the object storage systems, and you are billed for exactly that amount. You never have to tell AWS in advance how much you plan on storing, and your billing stops once you delete the content.

There are significant cost efficiencies that can be leveraged when examining the various types of object storage. Based on durability needs, retrieval times, and frequency, you will be able to improve the cost efficiency of your application dramatically. These different options are discussed in the Amazon S3 section of this chapter.

Block Storage on AWS

From a systems operation viewpoint, block storage must be provisioned, inventoried, attached, secured, monitored, duplicated, shared, resized, restored, detached, and decommissioned.

In some cases, the methods that you will use are common throughout all types of block storage. Let's examine the various types and go through AWS operations.

Amazon Elastic Block Store (Amazon EBS)

Amazon EBS is the foundational block storage service from AWS. It is built as network-attached storage that is variable in size from 1 GB to 16 TB. As mentioned earlier, volumes are provisioned by the operator and then attached to instances. Figure 6.1 shows how the command-line interface is used to create an Amazon EBS volume.

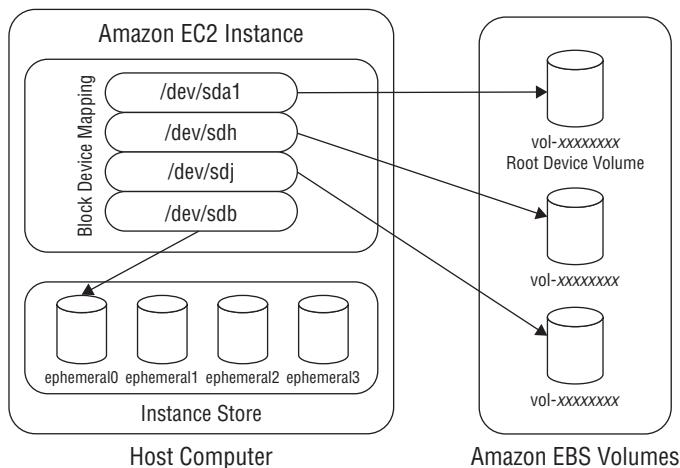
FIGURE 6.1 AWS CLI to create an Amazon EBS volume

```
rhoodsg@chaos:~$ aws ec2 create-volume --size 80 --region us-east-1 --availability-zone us-east-1a --volume-type gp2
{
  "AvailabilityZone": "us-east-1a",
  "Encrypted": false,
  "VolumeType": "gp2",
  "VolumeId": "vol-0e40b03560ceb54fb",
  "State": "creating",
  "Iops": 240,
  "SnapshotId": "",
  "CreateTime": "2017-07-04T01:18:47.283Z",
  "Size": 80
}
rhoodsg@chaos:~$
```

Amazon EBS volume usage is traditionally divided into boot volumes and data volumes. While boot and data can be provisioned on a single Amazon EBS volume, resizing a single volume requires a complete shutdown of the instance. For stateless servers that effectively ignore any data storage, however, there is no reason to have a separate data volume.

Once created, a volume's lifecycle is fundamentally independent from the Amazon EC2 instances. However, boot volumes are often destroyed when instances are terminated. Figure 6.2 shows an Amazon EC2 instance with Amazon EBS volumes attached.

The fact that Amazon EBS is network-attached is the distinguishing attribute of the service. Because Amazon EBS is a mounted volume, minimizing latency is critical to AWS operations. To ensure the lowest latency possible, volumes cannot be mounted across Availability Zones.

FIGURE 6.2 Amazon EC2 instance with attached Amazon EBS volumes

For example, an account has an Amazon EC2 instance running in us-west-2a and two unattached volumes: One is running in us-west-2a, and the other is running in us-west-2b. Only the volume in -2a is allowed to be mounted. If the operator is in the Amazon EBS Console, only the -2a volume will be selectable. If the operator is using the API, the attempt to attach the volume will be rejected. Figure 6.3 demonstrates the AWS CLI with a rejected volume attachment.

FIGURE 6.3 AWS CLI with a rejected volume attachment

```
rhoodsg@chaos:~$ aws ec2 attach-volume --volume-id vol-1234567890abcdef0 --instance-id i-01474ef662b89480 --device /dev/sdf
An error occurred (InvalidParameterValue) when calling the AttachVolume operation: Value (vol-1234567890abcdef0) for parameter volume is invalid. Expected: 'vol-...'.
rhoodsg@chaos:~$
```

Amazon EBS Volume Types

There are five types of Amazon EBS volumes that can be provisioned as of this writing. Table 6.1 details the various Amazon EBS volume types.

TABLE 6.1 The Various Amazon EBS Volume Types

| Volume Type | API Reference | Sizes | Max IOPS* | Max Throughput | Price (us-east-1) |
|----------------------|---------------|------------|-----------|----------------|--|
| General Purpose SSD | gp2 | 1 GB–16 TB | 10,000 | 160 MB/s | \$.10/GB-month |
| Provisioned IOPS SSD | io1 | 4 GB–16 TB | 20,000 | 320 MB/s | \$.125/GB-month \$.065/provisioned IOPS |

TABLE 6.1 The Various Amazon EBS Volume Types (*continued*)

| Volume Type | API Reference | Sizes | Max IOPS* | Max Throughput | Price (us-east-1) |
|--------------------------|---------------|--------------|-----------|----------------|-------------------------------------|
| Throughput Optimized HDD | st1 | 500 GB–16 TB | 500 | 500 MB/s | \$.045/GB-month |
| Cold HDD | sc1 | 500 GB–16 TB | 250 | 250 MB/s | \$.025/GB-month |
| Magnetic (Previous Gen) | standard | 1 GB–1 TB | 40-200 | 40-90 MB/s | \$.05/GB-month \$.05/million I/O |

* io1/gp2/standard based on 16 K I/O size, st1/sc1 based on 1 MB I/O packet size.

The most visible difference between types is whether they are SSD or magnetic Hard Disk Drives (HDD). Do not assume that SSD is always the best performance choice for your Amazon EC2 instances. If you will be using Amazon EBS, you must decide which Amazon EBS volume type is the best for your business value.

For the certification exam, it is important to align the volume type's characteristics to the values and desired outcome presented in the question. If you are looking for the lowest-cost option, provisioned IOPS volumes are almost always the wrong choice. If maximum throughput is the need, you should avoid choosing default magnetic drives.

Pricing

Let's start with the last column: Price. Although actual costs vary region to region, the prices in Table 6.1 are good representations of comparative costs.

A common statement that we hear about Amazon EBS is that customers choose gp2 for speed and magnetic for price. With the introductions of st1 and sc1 volumes, there are now magnetic drives that become the most performant options for some workloads while still reducing costs.

Determining the cheapest Amazon EBS volume is more than just choosing the lowest GB-month rate—there is also the minimum volume size to consider. The Cold HDD (sc1) is the lowest cost at \$.025 GB-month, but there is a minimum volume size of 500 GB. For example, if you only need a 50 GB volume for all of your needs, assuming cost is your only concern, going with the legacy default magnetic drive at 50 GB will be the least expensive. The flex point for choosing an Amazon EBS volume type is 250 GB. Once you require that much space, changing from default to sc1 will start saving money while providing significantly improved performance. Don't let the label "cold" keep you from using this volume type in production systems. Compared to the legacy default volumes, sc1 is a significant improvement on all performance metrics.

Provisioned IOPS is often the most expensive option. As a rule, these drives should not be used arbitrarily in operations unless you are specifically ensuring the availability of more than 3,000 IOPS. Provisioned IOPS volumes not only have the highest GB-month storage cost, but they have the additional \$.065 per provisioned IOPS per month. A 10,000 IOPS volume will include a monthly cost of \$650 above and beyond the monthly cost for the storage itself.



In some of the AWS documentation, you may see RAM and storage displayed as GiB or GB. For example, despite the different numbers, 128 GB and 119.2 GiB are the same amount—no space is “lost” or “missing.” The size is simply being expressed in a different unit. The use of GB or gigabyte can refer to 1,000,000,000 bytes in some contexts and to 1,073,741,824 bytes in others, sometimes in reference to the same device. This difference has caused some confusion among consumers. This book uses the GB convention instead of GiB for consistency.

Maximum IOPS

IOPS is the measurement of how many read/write actions can happen against the volume per second. The ratings for the volumes are based on a standard packet size, although you can choose the actual packet size that your application will use based on your needs. Because of the number of variables in play, it is important to understand the fundamentals of calculating potential IOPS.

The ratings on the volumes are based on 16 KB packets for SSD volumes and 1 MB packets for HDD volumes. The reason for the difference is to optimize each media type based on their specialty. In this case, SSD volumes handle many small packets with greater efficiency, while HDD is best suited for fewer large packets.

You can choose the actual packet size your application uses, but there are some limitations:

- Packet sizes for SSD are capped at 256 KB.
- HDD volumes can go as large as 1 MB.



The math for possible IOPS is based on packet size in use compared to rated IOPS for the volume. For example, a 100 GB gp2 volume has a rating of 300 IOPS at 16 KB packets. In practice, operations could expect the volume to sustain 300 random (non-sequential) read/write operations per second as long as each packet was <16 KB in size. But if the packets were 48 KB in size for the same 100 GB volume, the actual IOPS that would be recorded would be 100 IOPS.

Maximum Throughput

Throughput is the product of packet size and IOPS. Even though st1 and sc1 volumes have relatively small maximum IOPS (hundreds compared to thousands for gp2 or io1), the total throughput for st1 and sc1 far outstrips the SSD options because of the significantly larger packet options.

For large numbers of small random writes, gp2 (or io1) will provide the most throughput due to the higher IOPS. Using the smallest packet size of 16 KB, an SSD with 10,000 IOPS would sustain 160 MB/s throughput (which is the maximum throughput for gp2). Using the same 16 KB packet with an st1 volume, even at max IOPS with only 500 IOPS available, the volume would reach capacity at a paltry 8 MB/s—only 5 percent of the throughput of the SSD options.

By changing the packet size to the maximum allowed, however, the st1 volume running 1 MB packets at the same 500 IOPS is now able to read/write at 500 MB/s or over 300 percent of the maximum throughput of gp2.

Provisioning Amazon EBS

When provisioning Amazon EBS volumes, the systems operator declares the following:

- Volume type
- Size
- Availability Zone
- IOPS (only if io1)
- Snapshot ID (if this volume is a copy of a saved snapshot)
- Encryption options (including AWS Key Management Service [AWS KMS] key ID)

Once a drive is provisioned, the values are not editable. If you want to change a volume from gp2 to io1, you must take a snapshot of the original gp2 volume and create a new io1 volume based on that snapshot ID (decommissioning the original gp2 drive as it is no longer needed).

The same process applies to any of the other properties not on the list that you would want to change (such as volume encryption or volume size). Changing these properties requires taking a snapshot, creating a new volume, and then deleting the old one.

Calculating IOPS and Throughput

Amazon EBS gp2 volumes have a set amount of IOPS based on volume size: 3 IOPS/GB. For example, a 100 GB volume has a 300 IOPS baseline, a 1 TB volume has 3,000 IOPS, and so on up to a maximum of 10,000 IOPS with a 3.33 TB volume. Any volume larger than that will be throttled at the 10,000 IOPS maximum per volume.

Note that the maximum output of an Amazon EC2 instance is not tied to the IOPS of individual volumes. For example, let's say that your application is using a c4.2xlarge Amazon EC2 instance and you need approximately 15 TB of storage for your database. You could provision a single 15 TB gp2 Amazon EBS volume with 10,000 IOPS maximum, or you could provision three 5 TB gp2 volumes with 10,000 IOPS each and stripe them together at Raid0. This would present a logical volume of 15 TB and give you an aggregate IOPS of 30,000, assuming that you were distributing read/writes randomly throughout the array.



There is a minimum of 100 IOPS provisioned on all gp2 volumes no matter how small they are. Even though a 10 GB volume would only have 30 IOPS under the original formula, it will have a 100 IOPS throttle. st1 volumes have a similar sliding scale on throughput: 40 MB/s per TB. A 2 TB volume has a baseline of 80 MB/s, a 4 TB volume has a baseline of 160 MB/s, and so on up to a maximum of a 12.5 TB volume with a baseline of 500 MB/s.

Bursting IOPS and Throughput

Both gp2 and st1 volumes have bursting models for drive communication. In the case of gp2, the standard formula for 3 IOPS/GB provisioned is the baseline until you get to 10,000 IOPS (3.33 TB volume). For volumes under 1 TB (< 3,000 IOPS), AWS provides the options to occasionally burst up to 3,000 IOPS for a short period of time. This means that even a 50 GB volume, which has a baseline performance of 150 IOPS, could operate on demand at 3,000 IOPS until the bursting window closes.

The ability to burst is based on I/O credits accrued by the volume at a rate of three credits per GB provisioned per second. Once bursting is triggered, the credits are consumed at three credits per IOPS. The maximum amount of credits a volume can keep is approximately 54 million, which is enough for max bursting (3,000 IOPS) for 30 minutes. Once a volume is out of credits, AWS will throttle communication back down to the baseline amount.

st1 has a similar bursting model, but based on MB/s, not IOPS. In the case of st1 volumes, the cap is 500 MB/s, which becomes the bursting cap with credits accrued in a similar manner. Throughput credits accrue at 40 MB/s per TB provisioned. A volume then spends credits to burst to 500 MB/s until credits are depleted. Note that the 500 MB/s cap only applies to volumes that are equal to or larger than 2 TB; volumes that are smaller have reduced cap sizes based on the formula of 250 MB/s per provisioned TB. Thus a 500 GB volume (smallest size allowed in st1) would only be able to burst to 125 MB/s before being capped, which is still significantly more throughput than its baseline of 20 MB/s.

Provisioned IOPS

Provisioned IOPS is the primary feature of io1 volumes. Although there are no minimums on how many IOPS can be provisioned, as a general rule it is considered to be a costing anti-pattern to use io1 with less provisioned IOPS than a similarly sized volume for gp2 would have for baseline IOPS. Although there may be edge cases where an enterprise might choose that scenario, for the certification exam, consider any of those cases to be “zebras” instead of “horses.”

There is another good practice minimum to consider when provisioning IOPS when latency to the drive matters. To get the best I/O latency to an io1 volume, it is recommended to have at least two IOPS provisioned for each GB; with any less, you might encounter latency issues. For example, a 3,000 GB volume should be provisioned with at least 6,000 IOPS for best performance.

There are maximums on how many IOPS can be provisioned: a ratio of 50 IOPS per GB provisioned up to a max of 20,000 IOPS per volume. Thus a 100 GB volume could only be provisioned for 5,000 IOPS, but anything over 400 GB can be provisioned for the maximum amount.

Amazon EC2 Capacity

You should pay attention to the IOPS/throughput capacity of the Amazon EC2 instances that will use the volume. Technically, a systems operator could provision five volumes with all at 20,000 IOPS to give a 100,000 IOPS possible aggregate capacity and attach them to a single instance. However, if that instance was an m4.large, there are only 3,600 maximum IOPS that can come out of that instance. It’s the equivalent of buying a race car and deciding to drive it in a residential area—there is no way to use the full potential of the one because of the limits of the other.

In fact, there are no instances (as of this printing) that could use that kind of potential. The maximum IOPS in an Amazon EC2 instance caps at 65,000. Any application that needs more than 65,000 IOPS will either need to shard the data transactions onto multiple Amazon EC2 instances or consider instance store, which is discussed later in this chapter.

Throughput also has Amazon EC2 limitations. Some instances can go as high as 12 GB/s for maximum throughput, but only if the instance has been launched with the Amazon EBS Optimization property turned on. (Refer to Chapter 4, “Compute,” for more information on Amazon EBS optimization on Amazon EC2.)

Mounting Amazon EBS Volumes

Amazon EBS volumes operate independently from Amazon EC2 instances. Although they can be created at launch time by the AMI and can be terminated simultaneously as well, it is important to remember from an operations perspective that Amazon EBS and Amazon EC2 are separate services. A key benefit is that Amazon EC2 enables the resizing of instances (as mentioned in Chapter 4). Amazon EBS volumes persist when an instance is stopped, so the instance type can be changed.

When Amazon EC2 instance types are changed, the underlying hardware is always swapped out. Amazon EBS persistence makes this changing of infrastructure seamless to the operator.

To mount an Amazon EBS volume, use the following commands:

```
aws ec2 attach-volume --volume-id vol-1234567890abcdef0  
--instance-id i-abc1234 --device /dev/sdf
```

Amazon EBS volumes can only attach to one Amazon EC2 instance at a time. Although there is no limit to how many instances a volume may have been attached to, there is no concurrent attachment option. For use cases such as shared volumes where multiple instances must communicate with a single volume, the best architecture is to use Amazon Elastic File System (Amazon EFS), which is discussed later in this chapter.

Amazon EBS Snapshots

Amazon EBS volumes are redundant arrays and can withstand the loss of individual drives without loss of data. As previously mentioned, however, Amazon EBS volumes live in one—and only one—Availability Zone. If data durability is important, systems operations teams should include volume snapshots as part of their regular operational configuration.

Snapshots create permanent images of the volume and store them at the regional level (across multiple Availability Zones) in Amazon S3. It is important to realize that the snapshots are not stored as Amazon S3 objects and cannot be viewed using the Amazon S3 Console or APIs. They are unique Amazon EBS images and can only be accessed through the Amazon EC2 service.

Because they are not true Amazon S3 objects, snapshots only store the sectors that have changed (the delta) from the previous snapshot, allowing the total volume of data stored (and thus billed) to be dramatically reduced.



The caveat for only storing the delta is that if more than 10 GB of sectors have been modified since the last snapshot, the system will save a new 100 percent snapshot rather than try to parse out such a large change.

Snapshots for data volumes do not require the volume to be stopped (or quiesced), but choosing to hold writes to the volume will provide a much more accurate dataset. Boot volumes must be stopped before they can be snapped—another key reason to keep boot and data volumes separate.

Snapshots are locked to the region where they were created. If your operational policy requires keeping copies of data in different regions, you must initiate a copy of that image to the target region. The new snapshot will have a new ID. This fact is important when creating automated deployment methods because new volumes cannot be created from image IDs that are not in the region where the new volume is being created. Figure 6.4 demonstrates the AWS CLI command used to create snapshots.

FIGURE 6.4 AWS CLI command to create snapshots

```
rhoodsg@chaos:~$ aws ec2 create-snapshot --volume-id vol-0cb5b2dee8805e563 --description "This is my root volume snapshot."
{
    "Description": "This is my root volume snapshot.",
    "Encrypted": false,
    "VolumeId": "vol-0cb5b2dee8805e563",
    "State": "pending",
    "VolumeSize": 80,
    "Progress": "",
    "StartTime": "2017-07-04T01:50:26.000Z",
    "SnapshotId": "snap-0cb14f04261cd0417",
    "OwnerId": "123456789abc"
}
rhoodsg@chaos:~$
```

Sharing Snapshots

Your team can share snapshots to other accounts both in and out of your organization by managing the permissions on those images. After a volume is shared, anyone granted access to the volume will be able to make a complete copy of the volume (there is no partial sharing).

Public sharing can only happen on unencrypted volumes. If your volume is encrypted, you can share the volume to specific accounts as long as you provide the custom Customer Master Key (CMK) used to encrypt the volume. Volumes encrypted with your account's default CMK can never be shared.

Creating an AMI from a Snapshot

A snapshot of a root volume has most of the pieces needed to create an AMI. What is left is to put the additional metadata pieces in the system and register the snapshot as an AMI. Using the `register-image` command in the AWS CLI, you can identify which snapshot you want to use, along with the new name for the AMI.

Deregistration will cause the image to no longer be available as an AMI. Figure 6.5 demonstrates the AWS CLI used to describe your snapshots.

FIGURE 6.5 AWS CLI command to describe snapshots

```
rhoodsg@chaos:~$ aws ec2 describe-snapshots
{
    "Snapshots": [
        {
            "OwnerAlias": "amazon",
            "Description": "Business/Industry Summary (Windows)",
            "Encrypted": false,
            "VolumeId": "vol-e1ac4888",
            "State": "completed",
            "VolumeSize": 15,
            "Progress": "100%",
            "StartTime": "2008-11-19T12:15:17.000Z",
            "SnapshotId": "snap-8af818e3",
            "OwnerId": "123456789abc"
        }
    ]
}
```

Securing Amazon EBS

All Amazon EBS volumes are automatically locked to the account that created them and the Availability Zone where they were created. Unless the volume owner makes a snapshot and shares the snapshot with another account, there is no way for an instance in another account to have direct access to any data on the original Amazon EBS volume.

Customers who want additional security on their volumes can encrypt the data at either the client level or the volume level. Client-level encryption is done in the operating system of the instance to which the volume is currently attached. Because this is controlled by the operating system of the Amazon EC2 instance, AWS has no access or visibility into what keys or algorithms are being used. From an operational perspective, this means that customers using client-side encryption must be responsible for maintaining availability and durability for all keys in use.

AWS offers volume-level encryption as well. In the case of volume-level encryption, all data is encrypted at rest and in transit between the instance and the volumes. With Amazon EBS volume-level encryption, each volume gets a unique encryption key only used by that volume or other volumes created from snapshots of that volume. AWS uses the industry-standard Advanced Encryption Standard (AES)-256 algorithm to encrypt data and the volume keys. For greater security, customers can choose to use AWS KMS for the volume key management, which can be combined with AWS Identity and Access Management (IAM) permissions and AWS CloudTrail logging to provide greater protection and visibility.

Monitoring Amazon EBS

Basic volume monitoring (included with all volumes) happens in five-minute intervals. The metrics cover the fundamental pieces of disk management: number of bytes (read/write), I/O per second (read/write), idle time, queue length, and so on.

For volumes with provisioned IOPS, an additional metric of *throughput percentage* is also available. Throughput percentage is a critical metric because it shows a percentage of actual throughput used compared to provisioned. For cost analysis, this number must be closely regulated to ensure that you are not overpaying for IOPS.

For gp2, st1, and sc1 volumes, *burst balance* is also available. This feature provides visibility into burst bucket credits that are available to the volume.

Decommissioning Amazon EBS

Amazon EBS volumes are designed for persistence—as such, they can survive the termination of associated Amazon EC2 instances or even be directly detached from instances (as long as they are not the root volume). Detaching a volume does not change the billing for the volume, and it continues to count toward the storage limit for the account that owns the volume. A detached volume can be reattached to any other Amazon EC2 instance in the same Availability Zone of the same account.



Volumes that are no longer needed should be terminated using the delete-volume API. If there are snapshots of the volume, deleting the Amazon EBS volume does not delete the snapshots. If the data is sensitive and should be deleted, then the snapshots should be deleted as part of the operational process.

Instance Store

For many Amazon EC2 instance types, AWS provides an additional direct-attach block storage option through *instance store* (also referred to as *ephemeral storage*). From an operational perspective, instance store has some very specific advantages and some considerations.

Instance store is ideal for temporary storage of information that changes frequently (for example buffers, caches, scratch data, and other temporary content) or for data that is replicated across a fleet of instances (such as a load-balanced pool of web servers).

Instance Store vs. Amazon EBS

Amazon EBS is designed for persistence, whereas instance store is designed for comparatively inexpensive, disposable, usually high-speed communication. Instance store lives on the same hardware as the Amazon EC2 instance itself and is included in the hourly cost of the instance. This also means that there is no external network overhead that must be factored into communication potential.

The primary consideration is also a potential benefit: The fact that the storage is on the same hardware as the instance means that the storage does not have any persistence beyond the life of the Amazon EC2 instance. If the instance is stopped and started, all data on the instance store volume is lost.

Root volumes can run on Amazon EBS or instance store, but if the root volume is on instance store, there is no “stop” option, as a stop will terminate the instance.

Provisioning Instance Store

Unlike Amazon EBS, instance store is not flexible when considering volume size, media type, or throughput. Each instance type has a single specific storage option, if instance storage is available for that instance—some instances only support Amazon EBS for volume types and do not have instance storage options. Check the Amazon EC2 documentation for current instance storage options for each instance type.

Instance store-backed volumes are only available on creation of the Amazon EC2 instance. If an instance is launched without the volumes, they cannot be added later.

Instance Store Security

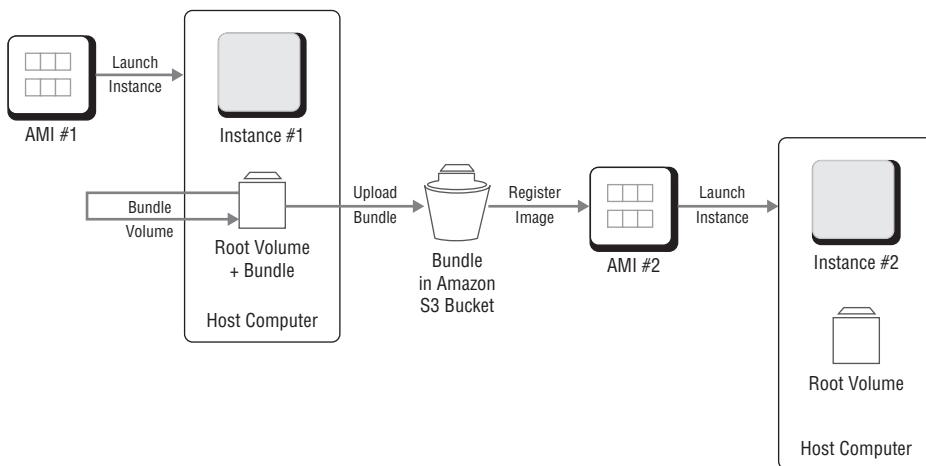
Security concerns are simplified to some degree when using instance store. The data never leaves the hardware, so encryption in transit is not a consideration. The instance cannot be detached and reattached to another instance. All encryption must be done at the operating system level.

These considerations mean that all security controls are managed at the operating system and above, hence it’s your responsibility to secure. Refer to the shared responsibility model described in Chapter 3, “Security and AWS Identity and Access Management (IAM),” for more information.

AMIs on Instance Store

To create an instance store-backed AMI, start from an instance that you have launched from an existing instance store-backed Linux AMI. Customize the instance to suit your needs, then bundle the volume and register a new AMI. Figure 6.6 summarizes the process of creating an AMI from an instance store-backed instance.

FIGURE 6.6 Process to create an AMI from an instance store-backed instance



The bundling step is different compared to Amazon EBS, with which you can create an AMI directly from an instance or a snapshot.

Once an instance store AMI is created, all root volumes will be built on the instance store when using that AMI.

The bundling process is a multi-step function that we are not going to cover in this book, as it falls outside the scope of the exam. Nevertheless, you should look up the online documentation and practice making an instance store-backed AMI to understand the process.

Amazon Elastic File System (Amazon EFS)

Amazon EFS is not a true unformatted block service but rather a fully managed file storage service. Multiple Linux Amazon EC2 instances connect to Amazon EFS through standard operating system I/O APIs. Amazon EFS can also connect to on-premises Linux servers through AWS Direct Connect.

Amazon EFS is designed to be automatically flexible in sizing, growing from gigabytes of shared storage up to petabytes in a single logical volume.

Amazon EFS vs. Amazon EBS

When comparing the two options, it is important to look closely at the primary use cases. Amazon EFS is designed to provide regional, durable, multi-user file systems. Amazon EBS is designed for single Amazon EC2 volumes.

Because of its design, Amazon EBS is significantly cheaper when looking at per-GB cost of storage. If there is no need for multiple instances to attach to the storage, Amazon EBS will likely be a good place to start your operation.

Once you have multiple users, however, Amazon EFS will often be a much less expensive option. Cost savings are more apparent once you consider the regional redundancy and durability that is rolled into Amazon EFS, bypassing the need to build your own highly available, regionally redundant file system. This becomes even more important when you consider the maintenance overhead that you avoid when using Amazon EFS over a DIY solution.

Provisioning Amazon EFS

After creating Amazon EFS using the Create API, you create endpoints in your Amazon Virtual Private Cloud (Amazon VPC) called “mount targets.” When mounting from an Amazon EC2 instance, you provide the Amazon EFS Domain Name System (DNS) name in your mount command. The DNS resolves to the mount target’s IP address. If your instance has access to the mount target in the VPC, you will then be able to access the file system.

This will be the same mount target that peered Amazon VPC instances or on-premises instances will use. They are granted access to the host VPC through the networking methods, and that will complete their ability to connect to the file system as well.

Instances then connect to the system with an NFSv4.1 client. See Figure 6.7 for the AWS CLI command to create an Amazon EFS.

FIGURE 6.7 Amazon EFS create command

```
rhoodsg@chaos:~$ aws efs create-file-system --creation-token CertBook --region us-east-1
{
    "SizeInBytes": {
        "Value": 0
    },
    "CreationToken": "CertBook",
    "CreationTime": 1499132308.0,
    "PerformanceMode": "generalPurpose",
    "FilesystemId": "fs-6d41ce24",
    "NumberOfWorkers": 0,
    "LifecycleState": "creating",
    "OwnerId": "123456789abc"
}
rhoodsg@chaos:~$
```

Securing Amazon EFS

The mount target you created when you provisioned the file system can use Amazon VPC security groups (the same ones that you use to secure access to Amazon EC2). All of the same rules of ingress and egress control can then be managed in detail with that toolset.

The mount targets are also specifically assigned to subnets (which by extension means that you should include mount points in a subnet in each Availability Zone in which you intend to run connected instances). This means that network Access Control List (ACL) control would also apply at the subnet level, if needed, for additional controls over an instance connecting from outside the subnet where the mount targets exist.

Encryption is not yet available natively on Amazon EFS as of this writing.

File access inside the system is controlled with standard UNIX-style read/write/execute permissions based on the user and group ID asserted by the mounting NFSv4.1 client.

Object Storage on AWS

The decision to use block storage vs. object storage is often an architecting question, as discussed in the introduction to this chapter. Understanding their core differences and maintaining content in object stores falls under the systems operator's scope of responsibility.

In AWS, object storage systems are fully managed, abstracted services. Interactions are all API driven, which means that IAM privileges are critical to security, because these services live outside of the subnets of your Amazon VPC.

Amazon Simple Storage Service (Amazon S3)

Amazon S3 is regionally provisioned object storage service. Content saved into Amazon S3 is automatically replicated into at least three facilities. The multiple copies provide the extreme levels of durability, while the regional distribution allows for massive parallel throughput. AWS datacenters are geographically separated, providing protection against natural disasters.

There is no data minimum or data limit to total Amazon S3 content. You are billed for exactly the amount in storage for the time it is in storage, plus the charges for API calls in and out of the system and data volume costs for data transfer out of the region.

Content stored in Amazon S3 is organized in buckets. Buckets are globally unique named objects; they are unique across all AWS accounts. If Customer A has a bucket called “projectatestresults,” Customer B would be rejected in an attempt to create the exact same bucket. You cannot create a bucket in another bucket.

The reason behind the uniqueness is based on the property of Amazon S3 storage in that all content in Amazon S3 is automatically web-enabled. When content is saved into Amazon S3, there is a unique URL created for the object, which includes the bucket name as the domain, hence the need for global uniqueness.



Amazon S3 defines a bucket name as a series of one or more labels, separated by periods, which adhere to the following rules:

- The bucket name can be between 3 and 63 characters long and can contain only lowercase characters, numbers, periods, and dashes.
- Each label in the bucket name must start with a lowercase letter or number.
- The bucket name cannot contain underscores, end with a dash, have consecutive periods, or use dashes adjacent to periods.
- The bucket name cannot be formatted as an IP address (198.51.100.24).



Amazon S3 supports both virtual-hosted-style and path-style URLs to access a bucket.

- In a virtual-hosted-style URL, the bucket name is part of the domain name in the URL. For example:
 - `http://bucket.s3.amazonaws.com`
 - `http://bucket.s3-aws-region.amazonaws.com`

In a virtual hosted-style URL, you can use either of these endpoints. If you make a request to the `http://bucket.s3.amazonaws.com` endpoint, the DNS has sufficient information to route your request directly to the region where your bucket resides.

- In a path-style URL, the bucket name is not part of the domain (unless you use a region-specific endpoint). For example:
 - US East (N. Virginia) region endpoint, `http://s3.amazonaws.com/bucket`
 - Region-specific endpoint, `http://s3-aws-region.amazonaws.com/bucket`.

In a path-style URL, the endpoint you use must match the region in which the bucket resides. For example, if your bucket is in the South America (São Paulo) region, you must use the `http://s3-sa-east-1.amazonaws.com/bucket` endpoint. If your bucket is in the US East (N. Virginia) region, you must use the `http://s3.amazonaws.com/bucket` endpoint.

The fact that content is web-enabled does not mean that it is exposed to the public. Default properties of Amazon S3 keep content private for the account owner until explicit commands to share are invoked. (This is covered further later in this chapter.)

Durability vs. Availability

One of the big engineered benefits of Amazon S3 is that it is designed to provide eleven 9s of durability for content. In other words, 99.99999999 percent durability of objects over a given year. This durability level corresponds to an average annual expected loss of 0.00000001 percent of objects.

To run the numbers in a more specific example, if you store 10,000 objects in Amazon S3, you can, on average, expect to incur a loss of a single object once every 10,000,000 years.

As a systems operator, do not confuse the durability numbers with availability numbers. Amazon S3 Standard is designed to deliver up to four 9's (99.99 percent) of availability. AWS even provides a Service Level Agreement (SLA) with penalties if availability drops below 99.9 percent on any given month of service (which translates to approximately 44 minutes of time in a month).

To understand the difference between durability and availability, think in terms of the question each is trying to answer. Availability addresses the question, “Can I reach my

content?” While AWS is very proud of our track record on availability, there are many factors that control access and are taken into account in the engineering profile. Durability addresses the question, “Does my content still exist?” From an enterprise survivability perspective, this is a much more critical question. Most businesses could survive the unfortunate loss of service for a few minutes a year; few businesses would be well positioned if their data was lost.

Durability is the primary reason that AWS recommends that all Amazon EBS volumes that contain mission-critical data be saved as snapshots into Amazon S3. Availability can always be restored as long as the data survives.

Accessing Amazon S3

Amazon S3 is accessed using APIs. Amazon S3 objects can also be accessed directly through HTTP calls with the right permissions.

Permissions to content (or buckets) are granted through IAM, or they can be granted directly to the operator through object-level permissions. Either approach is sufficient to grant access.

IAM policy documents and Amazon S3 bucket policies appear almost identical. The key difference between them is that the bucket policies include an account number. The account number is used to explicitly authorize users (or roles) from that account.

Amazon S3 – Reduced Redundancy Storage (Amazon S3-RSS)

Amazon S3 is designed to deliver eleven 9s of durability, four 9s of availability, and low-cost throughput, which solves most architectural storage needs. In some cases, however, the business need does not require durability.

For example, you have an application that manages high-definition imagery and uses a front end with a thumbnail browser. The high-definition images are best stored in Amazon S3 because they are mission-critical assets, but the thumbnails are all derivatives of the original image. In this scenario, there is very little damage to the business if a thumbnail is lost, because it can be detected and regenerated quickly from the original. Because durability is not as critical for the thumbnails, it doesn’t make sense to pay for the massive redundancies and durability of regular Amazon S3 to store them.

This is an example of the business scenarios behind *Amazon S3 – Reduced Redundancy Storage (Amazon S3-RRS)*. Unlike standard Amazon S3, which replicates your data across at least three geographically separate facilities, Amazon S3-RRS only uses two facilities. There is no difference in the availability (99.99 percent) or the SLA between Amazon S3 and Amazon S3-RRS, but there is significant difference in the results of the durability engineering. Amazon S3 is designed for eleven 9s, whereas Amazon S3-RRS is designed for four 9s of durability.

This reduction in engineered durability means that Amazon S3-RRS can provide somewhat cheaper storage without reducing availability.

From a certification perspective, you must look at a question and determine the business value behind the content being stored. If the question refers to storing content that must be highly available and cost effective, do not immediately assume that the correct approach is

to use Amazon S3-RRS. Unless the question includes some mention about the data being reproducible or the original content is stored separately, the loss of durability might be a problem.

If the question asks for the least-expensive solution as the primary goal and does not mention anything about needing durable content however, you can assume that Amazon S3-RRS would be a better choice than standard Amazon S3.

If the question specifically mentions that the data being stored can be re-created on demand, then you should absolutely look at Amazon S3-RRS as the primary candidate for your answer.

Amazon S3 Infrequent Access (Amazon S3-IA)

Amazon S3 Infrequent Access (Amazon S3-IA) solves a completely different business need. In the scenarios suited for Amazon S3-RRS, high availability and high parallel throughput were still needed, but durability was not a primary value. For Amazon S3-IA, the objects still need the massive durability of Amazon S3 and the high availability, but the throughput is not expected at all.

The example scenario in this case would be from the same imaging application mentioned previously that has an archive section for images that are more than six months old. Although they might occasionally be called up for a review, the reality is that they won't be looked at more than once or twice a year, if ever.

For those business cases, AWS offers Amazon S3-IA. It is still Amazon S3 in terms of durability and availability, but the difference is that the way the storage is engineered for accessibility is much less expensive in general, but the cost per individual requests are notably higher. The reduced cost for availability means that Amazon S3-IA is the most cost-efficient option, as long as the objects are only retrieved infrequently.

The breakpoint for cost effectiveness in Amazon S3-IA is around two retrievals per month. If you expect more retrievals than that on average, it is cheaper to stay with standard Amazon S3.

Exam questions that specifically call out objects that are only accessed once a month or less than 20 times a year should most likely be addressed with Amazon S3-IA. As always, look for other clues that would invalidate the savings presented by Amazon S3-IA.

For example, an additional attribute of Amazon S3-IA is minimum file size (128 KB). Standard Amazon S3 has no file size minimum—a 4 KB file in Amazon S3 is only billed for the 4 KB space. The same 4 KB file stored in Amazon S3-IA would be billed at 128 KB, so even if Amazon S3-IA was normally half the cost of regular Amazon S3, in this edge case, Amazon S3-IA would be 16 times the cost of storing the same 4 KB file in Amazon S3. If the exam question specifically mentioned tiny files (less than 64 KB in size), that would be a signal not to choose Amazon S3-IA as the cost-effective answer.

The other edge case, where Amazon S3-IA is not the best answer, is when files are only stored for a few days. Amazon S3-IA has a minimum 30-day storage time. Content stored for one week is still billed for the entire month. Amazon S3-IA would normally be less expensive, but the shortened shelf life makes standard Amazon S3 a more cost-effective choice.

Versioning

The default behavior of Amazon S3 is to have a single version of an object. When the object is updated, all copies of that object are updated as well. When the object is deleted, all copies are deleted. For operational needs that require back versions to be retained, Amazon S3 *versioning* can be enabled.

After versioning is enabled, updates to an object do not replace the original object; rather, the new version is stored and marked as the current version of the object and the previous version is retained but no longer retrieved when the object is HTTP requested.

The previous versions are still accessible through the APIs. Operators can select those versions and mark them as the current version (restore) or delete/expire them when they are no longer needed.

Deletes in versioned buckets get divided into two operational functions. “Normal” deletes actually add another version to the stack of versions referenced by the object. In this case, it is just a marker indicating that the file does not exist. If there is an HTTP call to the object, it will return a 404 error. Deleting this 0 KB version will restore the previous version (effectively providing a “recycle bin” function).

Because normal deletes in this case only serve to add a 404 header, to actually purge the object from Amazon S3, you must perform “delete version.” This can be done version by version or wildcarded to delete the entire stack.

It is important to account for cost whenever versioning is enabled. Each object version is billed at the full Amazon S3 storage rates. Operations teams that do not have a lifecycle plan for versioning can find storage costs escalating.

As such, versioning only makes sense on the exam when the questions are concerned with object survivability in the event of intentional or accidental deletes or overwrites. If those are not concerns, then for price considerations, versioning is likely not the best choice.



After you enable versioning on a bucket, it can be in either the enabled or suspended state; you cannot disable versioning on a bucket. If you suspend versioning, Amazon S3 suspends the creation of object versions for all operations but preserves any existing object versions.

Amazon S3 Security Options

Amazon S3 is API driven and is a completely abstracted service. As such, there are no security groups or Amazon VPC subnets that control access to the Amazon S3 endpoints (although individual instances can certainly block themselves from accessing Amazon S3).

Assigning the correct IAM policy document to individual users, groups, and roles is the first method for granting access to Amazon S3. You can assign permissions to individual objects or to buckets.

The advantage to using group policies is that individual buckets don't have to be named. If new buckets are created in an account, anyone with wildcarded Amazon S3 permissions will be able to access the new bucket without edits to the policy document.

Bucket policies have the advantage of being able to grant access to other accounts without needing to create a role for them in the account that owns the bucket.

Cross-Region Replication

There are additional security concerns when dealing with Amazon S3: termination protection and tampering protection. Both of those can be addressed by enabling cross-region replication.

At first glance, *cross-region replication* is a tool designed to provide extreme global durability. If you have a business that for regulatory reasons needs data replicated with hundreds or thousands of miles of distance between copies, this feature will make those copies automatically.

To enable cross-region replication, you must first turn on versioning in both the source and destination buckets. After versioning is turned on, you can enable the replication by identifying the source and destination. Note that this is not bucket synchronization—deleted versions in the source bucket are not deleted from the target bucket.

How does this protect your content? Even if someone accidentally (or intentionally) deletes all content from the source bucket, operations can go to the target bucket and find the content in its entirety.

In addition, cross-region replication can be done from a source bucket in one account to a target bucket in a completely separate account. This multi-account strategy allows a security team to ensure that no single individual has permissions to both buckets where the data is copied. Even full administrator privileges on one account would not automatically give them any access to modify or delete content in the other.

Multi-Factor Authentication (MFA) Delete

The other unique element of security that Amazon S3 and Amazon Glacier offers is the ability to attach a physical or virtual Multi-Factor Authentication (MFA) device against an object or bucket.

Once this MFA device is attached, any delete statement will go through two checks. First, the properly authorized operator must still have the right IAM or object permissions to delete the object. Second, the operator must also present the current one-time password generated by the MFA device.

This feature will prevent the deletion of content by a compromised account or malicious agent. It will prevent the deletion of content by an automation method that would unintentionally have caught the protected object.

It also solves these problems without any billing expense. *MFA Delete* is provided at no additional charge, whereas enabling cross-region replication will incur the cost of data transfer between regions and the storage costs of the duplicate datasets.



Cross-region replication and MFA Delete provide even stronger protection when used together. If you have to protect data at all costs, these methods are complementary.

Amazon Glacier

If we look at the use cases of standard Amazon S3, Amazon S3-IA, or Amazon S3-RRS, there may have been variable demands on durability requirements or how often an object would be retrieved. In all three primary use cases, however, high availability was always the need—being able to get to the content immediately and on demand was critical.

But what about a situation where availability is not needed, where as long as the data is protected, intact, and eventually retrievable, then the business need is solved? This is the primary scenario for Amazon Glacier.

Unlike the various flavors of Amazon S3, which are all enabled for immediate consumption online, there is no online availability for content stored in *Amazon Glacier*. The content is in “cold storage.” Durability, however, is designed to be the same between Amazon S3 and Amazon Glacier: eleven 9s.

Retrieving content from Amazon Glacier is done through the normal API request; you can request content for immediate retrieval at additional expense. Once content is retrieved, a copy is placed into Amazon S3 where it becomes a normal Amazon S3 object (with associated availability and cost structure). Retrieving an object does not delete the object from Amazon Glacier.



Amazon Glacier has traditionally been tied to a three- to five-hour retrieval time period. At the end of 2016, AWS released the immediate retrieval option.



Regarding the certification exam, Amazon Glacier presents a great example of “horses, not zebras.” If a question involves Amazon Glacier, unless the question specifically discusses the immediate retrieval option (which is notably more expensive), assume the standard three- to five-hour retrieval time in your decisions.

You store your data in Amazon Glacier as archives. You may upload a single file as an archive, but your costs will be lower if you aggregate your data. TAR and ZIP are common formats that customers use to aggregate multiple files into a single file before uploading to Amazon Glacier. The total volume of data and number of archives that you can store are virtually unlimited. Individual Amazon Glacier archives can range in size from 1 byte to 40 TB. The largest archive that can be uploaded in a single upload request is 4 GB. For items larger than 100 MB, customers should consider using the multipart upload capability. Archives stored in Amazon Glacier are immutable (that is, archives can be uploaded and deleted but cannot be edited or overwritten).



An archive is any object, such as a photo, video, or document, that you store in a vault. The archive is a base unit of storage in Amazon Glacier. You can have 1,000 vaults per account per region.

Security

From a security perspective, Amazon Glacier operates in a similar manner to Amazon S3. Users can be granted IAM permissions to individual archives or vaults (which are root-level storage containers similar to Amazon S3 buckets). Object-level permission is also possible, which can allow for multi-account workflows to the vaults or archives. MFA Delete offers additional protection against accidental or intentional deletions.

In addition, Amazon Glacier offers *Vault Lock* for customers with specific compliance needs. For example, the United States Securities and Exchange Commission (SEC) rule 17a-4 states that all “electronic records must be preserved exclusively in a non-rewriteable and non-erasable format.”

Legal clarifications to that rule require that there be no possibility of content deletion. An operations team can store content in Amazon Glacier, make copies of that archive in vaults owned by different accounts and different regions, remove all IAM permissions, attach deny policies against the objects and the vaults, and attach different MFA devices against each copy protecting against deletion, finally locking those devices in a safe owned by an escrow company. However, without Amazon Glacier Vault Lock, they still would not be compliant with SEC rule 17a-4. That is because with the right root-level permissions on all of the accounts and the cooperation of the escrow company holding the MFA devices, the content could be deleted or altered.

Although such a complex action would be difficult to do unnoticed, it is still technically possible. Vault Lock adds an additional, immutable element to the storage engine.

Enabling Vault Lock requires two separate API requests. The first request establishes the parameters of the Vault Lock in terms of duration. The second API (which must be executed within 24 hours of the first) confirms the Vault Lock option.

Once Vault Lock is confirmed, any content stored in the vault enters a locked state until that duration period passes. For example, if a duration of one year is put on a vault, content placed in the vault on January 1, 2020 would not be capable of being deleted until January 1, 2021. It does not mean that it will be deleted on that day, but under no conditions can it be deleted or altered during the year 2020.

What if the company changes its mind? The company can change its mind all it wants; however, until the duration expires for any content under Vault Lock, there is no way for the company to delete the content. What if the company’s account is delinquent? In those cases, under the AWS Terms and Conditions, AWS will continue to honor the Vault Lock conditions and not delete the content, although access to the content may be denied until the account is made current.

The only way to “disable” a Vault Lock is to delete the vault after all archives in the vault have outlived their duration protection. Given the irrevocable nature of the Vault Lock feature, it is considered a best practice not to use Vault Lock unless a company is under specific compliance language that provides no other interpretation.

A final security note on Vault Lock is that the option does not provide read-lock. Users with permissions may still download copies of the content, so sensitive material will still need proper governance models to protect against unauthorized downloads.

Lifecycle Policies

An important aspect of the relationship between Amazon S3 and Amazon Glacier is the ability to set *lifecycle policies* that automatically manage the transitions of content and their expiry.

For example, a company might have content that needs accessibility for the first 90 days followed by occasional usage for the next two years, at which point the records must be retained but would only be accessed in the event of an audit. Finally, the content must be deleted after a total of seven years for compliance. To accommodate this, systems operators can create a lifecycle policy on the original bucket:

1. On object creation plus 90 days, change attribute of content to Amazon S3-IA.
2. The next part of the lifecycle policy statement reads: At object creation plus two years, copy the content to an Amazon Glacier vault, and delete the content from the Amazon S3-IA bucket.
3. The final policy is on the Amazon Glacier vault where creation date plus five years deletes the object (the five years in Amazon Glacier, plus the two years in Amazon S3, results in a seven-year total object lifecycle).



There are two types of lifecycle actions: Transition actions, where you define when objects transition to another storage class, and expiration actions, where you specify when the objects expire.

When creating a lifecycle policy, it is the responsibility of the systems operator to ensure that only the correct content is affected by the scope of the policy. To clarify, if you have a deletion policy on an Amazon Glacier vault, you should only place content in that vault if that content has the same lifecycle rules. Failure to do so will result in content being purged unexpectedly.



Amazon Glacier offers a 10 GB retrieval free tier. The AWS Free Tier applies to participating services across the following AWS Regions: US East (Northern Virginia), US West (Oregon), US West (Northern California), Canada (Central), EU (London), EU (Ireland), EU (Frankfurt), Asia Pacific (Singapore), Asia Pacific (Tokyo), Asia Pacific (Sydney), and South America (Sao Paulo). You can retrieve 10 GB of your Amazon Glacier data per month for free. The free tier allowance can be used at any time during the month and applies to standard retrievals.

Systems Operator Scenario: The Newspaper

You have just been put in charge of operations for the city *Tribune*'s Content Management System (CMS). The system has already been architected, and now you need to make sure that it is operating properly on the AWS Cloud.

For the needs of this chapter, let's look at how the storage systems would be designed based on the various needs of the CMS.

Storage Needs

The *Tribune*'s CMS covers a number of processes of the production system: The reporter process, along with storage of all notes; the photo editing process, including storage of outtakes; the web publishing system for all stories, photos, and videos; and the archive system for research internally and externally.

Solution Breakdown

Here is a breakdown of the *Tribune*'s CMS processes.

Reporter

The reporter uses a word processor for all of her work. This implies a heavy read/write workload best served by Amazon EBS at first glance. However, given the fact that editors are involved and story creation is a collaborative effort, a better operational solution would be to use Amazon EFS as a mounted volume for the word processor application.

Using Amazon EFS would provide immediate shared storage for all content in the creation and editing phases of the workflow without any additional operational overhead.

After the content is fully edited and ready for publication, you could serve the content on the website from the Amazon EFS (or Amazon EBS) volume. A more scalable solution would be to load the document into Amazon S3 and have the web page call the story directly from Amazon S3, which is a more cost-efficient and scalable publishing option.

The final concern for reporters is their story notes, which are not intended for the public but must be retained for verification in the event of legal challenges to the story. Amazon Glacier becomes the ideal solution for these documents because retrieval needs are rarely immediate and storage costs are the lowest.

Photo Editor

The photographer will often shoot hundreds of photos or hours of video for a single shot or clip that will end up in the publication. The outtakes can often be mined years later for important details that seemed insignificant during the shoot. After photo selection is complete, storing the outtakes in Amazon Glacier again provides the most cost-effective solution for a deep archive.

For media that is marked for production, you can store the raw files in Amazon S3 and serve them directly from their bucket to internal production applications or to the public websites.

If the publication needs pre-rendered thumbnails or mobile-sized versions of the media, they can be stored in Amazon S3-RRS as a way of saving costs, so that those versions can be re-created on demand from the originals if they are lost.

Production and Archive

By storing the content (text and media) in Amazon S3 (or Amazon S3-RRS), the CMS is already positioned with the most cost-effective, scalable option to deliver the content.

At some point, the content will be determined to be out of date. In the case of a news website, it may take months before consumers stop searching for or clicking on old stories, but eventually all content will reach a point of diminishing results. When the content reaches that point, automated processes can transition the content to Amazon S3-IA. Why not Amazon Glacier? Because in the case of a news archive, even if consumers don't care about the old story for years, once someone is doing research and wants to open that file (or video), odds are they don't want to wait five hours for the content to be available. By using Amazon S3-IA, the archive is still cost effective but immediately available and searchable.

Additional Storage Solutions

Amazon S3, Amazon EBS, Amazon EFS, and Amazon Glacier represent the core of the AWS storage solutions. When studying to be successful on the exam, you must be very comfortable with these services.

AWS does provide a number of other services that may be considered part of the storage solutions discussion. You should be familiar with them, but they will not likely be a significant portion of the AWS Certified SysOps Administrator – Associate exam.

Amazon CloudFront

Global content distribution can be delivered using *Amazon CloudFront*. This Content Delivery Network (CDN) can help accelerate the delivery of your content by caching copies close to consumers.

Content is stored in origin locations that can be on AWS in an Amazon S3 bucket or served from an Amazon EC2 instance. Amazon CloudFront can even cache content stored on-premises.

Amazon CloudFront is a good choice for distribution of frequently accessed static content that benefits from edge delivery, like popular website images, videos, media files, or software downloads. For on-demand media files, you can also choose to stream your content using Real-Time Messaging Protocol (RTMP) delivery. Amazon CloudFront also supports delivery of live media over HTTP.

Lastly, Amazon CloudFront has a geo-restriction feature that lets you specify a list of countries in which your users can access your content. Alternatively, you can specify the countries in which your users cannot access your content. In both cases, Amazon CloudFront responds to a request from a viewer in a restricted country with an HTTP status code 403 (Forbidden).

AWS Storage Gateway

AWS Storage Gateway allows existing on-premises storage solutions to be extended into AWS. By installing a virtual appliance in the local datacenter, storage can be duplicated or extended into the AWS Region.

For the exam, you will likely see at least a couple of questions that involve AWS Storage Gateway. The important things to understand are the different options available when configuring the appliance.

AWS Storage Gateway Options

The first option is called the *file interface*. This enables you to use Amazon S3 with your on-premises workflows by accessing the Amazon S3 bucket through a Network File System (NFS) mount point. This allows customers to migrate their files into Amazon S3 through object-based workloads while maintaining their on-premises investments.

The second option is the *volume interface*. In this case, you are presented with an iSCSI block storage endpoint. Data is accessed on the local volumes, which is then stored on AWS in the form of Amazon EBS snapshots.

There are two configuration modes using the volume interface. The *cached mode* is designed to extend your local storage. All content is saved in AWS, and only the frequently accessed data is cached locally. This is an excellent operational solution for on-premises storage solutions that are exceeding their local capacity limits. The other mode is the *stored mode*, which is a one-to-one backup of all local content on AWS. This is a primary solution when durable and inexpensive off-site backups are needed. This becomes an excellent start to a hybrid disaster recovery plan.

The third option for AWS Storage Gateway is the *tape interface*. In this configuration, you connect to your existing backup application using an iSCSI Virtual Tape Library (VTL). These virtual tapes are then asynchronously stored in Amazon S3 or Amazon Glacier if less expensive, longer-term storage is needed.

AWS Snowball

For massive data transfers, ground transport can be faster than the Internet. Being able simply to ship large hard drives will get the data into AWS faster (and often cheaper) than public Internet speeds can accommodate. For example, suppose you had 100 TB of data to transfer to AWS. Even over a 100 Mbps data transfer line, it would take 120 days to complete the data transfer. Physically shipping the data is a far faster option.

For those data transfer cases, AWS provides *AWS Snowball*, a physically hardened, secure appliance that ships directly to your on-premises location. Each device weighs around 50 pounds and stores 80 TB (as of this writing). Using multiple AWS Snowball appliances in parallel can provide an easy to implement a migration solution for datasets that are multiple petabytes in size.

AWS Snowball is secured through tamper-resistant seals and a built-in Trusted Platform Module (TPM) that uses a dedicated processor designed to detect any unauthorized modifications to the hardware, firmware, or software.

Data is automatically encrypted when using AWS Snowball with encryption keys managed through AWS KMS. The actual encryption keys are never stored on the appliance.

AWS Snowball with AWS Greengrass

As AWS Snowball has increased in popularity, AWS has added additional functionality, including the ability to access APIs run on the device by leveraging *AWS Greengrass* technology. Those APIs allow the AWS Snowball appliance to act as if it was an Amazon S3 endpoint itself, even when disconnected from the Internet.

AWS Snowmobile

Some companies have dataset transfer needs that range in the exabyte scale. For those unique transfers, AWS can deploy 45-foot-long shipping containers called *AWS Snowmobiles*. Pulled by tractor trailers, each container can transfer up to 100 PB of data.

For security, AWS Snowmobile uses dedicated security personnel, GPS tracking, alarm monitoring, 24/7 video surveillance, AWS KMS encryption, and an optional escort security vehicle while in transit. For more information on AWS KMS, refer to Chapter 3.

Like AWS Snowball, AWS Snowmobile can deliver multiple containers to a single location, providing exabytes of capacity where needed.

Summary

There is no one-size-fits-all solution when it comes to storage. As you prepare for the exam, you need to dive deep into the core storage solutions: the block storage options of Amazon EBS and Amazon EFS and the object storage solutions of Amazon S3 and Amazon Glacier. Make sure that you are comfortable with their use cases and when to deploy each option.

Resources to Review

Amazon EBS documentation:

<http://docs.aws.amazon.com/AWSEC2/latest/UserGuide/AmazonEBS.html>

Amazon EFS main page: <https://aws.amazon.com/efs/>

Amazon EFS documentation: <https://aws.amazon.com/documentation/efs/>

Amazon S3 main page: <http://aws.amazon.com/s3/>

Amazon S3 documentation: <http://aws.amazon.com/documentation/s3>

“IAM Policies and Bucket Policies and ACLs! Oh, My! (Controlling Access to Amazon S3 Resources)” blog post: <https://aws.amazon.com/blogs/security/iam-policies-and-bucket-policies-and-acls-oh-my-controlling-access-to-s3-resources/>

Configuring Static Website Hosting section of the Amazon S3 user guide: <http://docs.aws.amazon.com/AmazonS3/latest/user-guide/static-website-hosting.html>

Amazon Glacier main page: <https://aws.amazon.com/glacier/>

Amazon Glacier documentation: <https://aws.amazon.com/documentation/glacier/>

Amazon CloudFront main page: <https://aws.amazon.com/cloudfront/>

Amazon CloudFront documentation:
<https://aws.amazon.com/documentation/cloudfront/>

AWS Storage Gateway main page: <https://aws.amazon.com/storagegateway/>

AWS Storage Gateway documentation:
<https://aws.amazon.com/documentation/storage-gateway/>

“File Interface to Storage Gateway” blog post: <https://aws.amazon.com/blogs/aws/category/aws-storage-gateway/>

AWS Snowball main page: <https://aws.amazon.com/snowball/>

AWS Snowball documentation: <https://aws.amazon.com/documentation/snowball/>

What’s new at AWS: <https://aws.amazon.com/new>

AWS Simple Monthly Calculator: <http://calculator.s3.amazonaws.com/index.html>

Exam Essentials

Understand block storage vs. object storage. The difference between block storage and object storage is the fundamental unit of storage. With block storage, each file being saved to the drive is broken down into “blocks” of a specific size. With object storage, each file is saved as a single object regardless of size.

Understand when to use Amazon S3 and when to use Amazon EBS or Amazon EFS. This is an architectural decision based on the content type and the rate of change. Amazon S3 can hold any type of data; however, Amazon S3 would not be a good choice for a database or any rapidly changing data types. Remember the case for eventual consistency.

Understand the lifecycle of Amazon EBS volumes. Amazon EBS volumes can be provisioned at the launch of an Amazon EC2 instance or created and attached at a later time. Amazon EBS volumes can persist through the lifecycle of an Amazon EC2 instance, provided the Deleted on Termination flag is set to false. The AWS best practice for an unused Amazon EBS volume is to snapshot it and delete it. The Amazon EBS volume can be created from the snapshot if needed. Remember that with Amazon EBS, you pay for what you provision, as opposed to paying for what you use with Amazon S3.

Understand different types of Amazon EBS volumes. Understand bursting. Amazon EBS volumes are tied to the Availability Zone.

Understand how Amazon EBS snapshots work. Snapshots are stored in Amazon S3; however, you do not access them via the Amazon S3 Console. They are accessed via the Amazon EC2 console under Snapshots.

Understand instance store storage. Depending on the Amazon EC2 instance family, you have access to the local storage. This storage is called the instance store. Snapshots cannot be taken of the instance store. Instance store storage is ephemeral and doesn't survive a restart or termination of the Amazon EC2 instance. For more information, refer to Chapter 4.

Have a detailed understanding of how Amazon S3 lifecycle policies work. Lifecycle configuration enables you to specify the lifecycle management of objects in a bucket. The configuration is a set of one or more rules, where each rule defines an action for Amazon S3 to apply to a group of objects. Know the two types: Transition actions and expiration actions.

Understand Amazon S3 versioning. When Amazon S3 versioning is turned on, it cannot be turned off, only paused. Understand that when versioning is turned on, items deleted are assigned a delete marker and are unable to be accessed. The deleted objects are still in Amazon S3 and you still pay for them.

Understand how to interface with Amazon Glacier. When objects are moved from Amazon S3 to Amazon Glacier, they can only be accessed from the Amazon S3 APIs (for example, in the case of an object that has been moved to Amazon Glacier as the result of a lifecycle policy).

Understand Amazon Glacier vaults. Amazon Glacier stores objects as archives and stores the archives in vaults. You can have (as of this writing) 1,000 vaults per account per region. Vaults are immutable; when retrieving an object, it is not removed from Amazon Glacier but is copied to Amazon S3.

Know how MFA Delete works with Amazon S3. You can optionally add another layer of security by configuring a bucket to enable MFA Delete, which requires additional authentication for changing the versioning state of your bucket and for permanently deleting an object version. MFA Delete requires two forms of authentication together: your security credentials and the combination of a valid serial number, a space, and the six-digit code displayed on an approved authentication device. MFA Delete thus provides added security in the event, for example, that your security credentials are compromised.

Know how to control access to Amazon S3 resources. IAM policies specify what actions are allowed or denied on what AWS resources. Amazon S3 bucket policies, on the other hand, are attached only to Amazon S3 buckets. Amazon S3 bucket policies specify what actions are allowed or denied for which principals on the bucket to which the bucket policy is attached. Amazon S3 bucket policies are a type of ACL.

Understand the features of Amazon CloudFront. Amazon CloudFront is a CDN that can help accelerate the delivery of content by caching copies close to consumers. Content is stored in origin locations, which can be on AWS in an Amazon S3 bucket or served from an Amazon EC2 instance. Amazon CloudFront can even cache content stored on-premises.

Understand media that can be delivered from Amazon CloudFront. This media includes static website CSS style sheets, HTML pages, images, videos, media files, or software downloads. For on-demand media files, you can also choose to stream your content using RTMP delivery. Amazon CloudFront also supports delivery of live media over HTTP.

Understand Amazon CloudFront's geo-restriction feature. This feature lets you specify a list of countries in which your users can access your content. Alternatively, you can specify the countries in which your users cannot access your content.

Understand AWS Storage Gateway's interfaces and modes of operation. The first option is called the file interface. This enables you to use Amazon S3 with your on-premises workflows by accessing the Amazon S3 bucket through an NFS mount point. The second option is the volume interface. In this case, you are presented with an iSCSI block storage endpoint. The third option for AWS Storage Gateway is the tape interface. In this configuration, you connect to your existing backup application using an iSCSI VTL. The file volume interface operates in one of two modes: cached mode and stored mode. Understand the differences.

Test Taking Tip

Multiple choice questions that ask you to choose two or three true answers require that ALL of your answers be correct. There is no partial credit for getting a fraction correct. Pay extra attention to those questions when doing your review.

Exercises

By now you should have set up an account in AWS. If you haven't, now would be the time to do so. It is important to note that these exercises are in your AWS account and thus are not free.

Use the Free Tier when launching resources. The AWS Free Tier applies to participating services across the following AWS Regions: US East (Northern Virginia), US West (Oregon), US West (Northern California), Canada (Central), EU (London), EU (Ireland), EU (Frankfurt), Asia Pacific (Singapore), Asia Pacific (Tokyo), Asia Pacific (Sydney), and South America (Sao Paulo). For more information, see https://aws.amazon.com/sdm/optimization/server-side-test/free-tier/free_np/.

If you have not yet installed the AWS Command Line utilities, refer to Chapter 2, "Working with AWS Cloud Services," Exercise 2.1 (Linux) or Exercise 2.2 (Windows).

The reference for the AWS CLI can be found at <http://docs.aws.amazon.com/cli/latest/reference/>.

Getting to know storage systems can be simple when using your own account. Follow the steps in previous chapters to log in to your account. As always, remember that although AWS is a very cost-effective solution, you will be responsible for all charges incurred. If you don't want to keep the items permanently, remember to delete them after your practice session.

Remember, the goal of this book isn't just to prepare you to pass the AWS Certified SysOps Administrator – Associate exam. It should also serve as a reference companion in your day-to-day duties as an AWS Certified SysOps Administrator.

EXERCISE 6.1**Create an Encrypted Amazon EBS Volume.**

In this exercise, you will create an Amazon EBS volume with either native or AWS KMS encryption.

1. While signed in to the AWS Management Console, open the Amazon EC2 service dashboard.
2. Under Elastic Block Store, select Volumes.
3. Select Create Volume.
4. Choose the options that are appropriate for your volume, including size and Availability Zone.
5. For master key, select either the default native encryption or an available AWS KMS master key. If one is not available, you may create one in the IAM dashboard under encryption keys.
6. Review the configuration options, and create the volume.

You have now created an encrypted Amazon EBS volume.

EXERCISE 6.2**Monitor Amazon EBS Using Amazon CloudWatch.**

In this exercise, you will explore the monitoring options available to your volume.

1. While signed in to the AWS Management Console, open the Amazon CloudWatch service dashboard.
2. Select Metrics. Under All Metrics, choose EBS and Per-Volume Metrics.
3. Active EBS volumes will now be shown by ID number along with all metrics that are available.

You have now explored the metrics dashboard for Amazon CloudWatch.

EXERCISE 6.3**Create and Attach an Amazon EFS Volume.**

In this exercise, you will create a new Amazon EFS volume and attach it to a running instance.

1. While signed in to the AWS Management Console, open the Amazon EC2 service dashboard. If you don't have a running Linux instance, create, and launch one.
2. Open the Amazon EFS service dashboard. Select Create File System.

3. Select the Amazon VPC where your Linux instance is running.
4. Accept the default mount targets, and take note of the security group ID assigned to the targets.
5. Choose any settings, and then create the file system.
6. Assign the same default security group used by the file system to your Linux instance.
7. Log in to the console of the Linux instance, and install the NFS client on the Amazon EC2 instance. For Amazon Linux:
`sudo yum install -y nfs-utils`
8. Create a new directory on your Amazon EC2 instance, such as efs:
`sudo mkdir efs`
9. Mount the file systems using the DNS name:
`sudo mount -t nfs4 -o nfsvers=4.1,rsize=1048576,wsize=1048576,hard,timeo=600,retrans=2 fs-12341234.efs.region-1.amazonaws.com:/ efs`

You have now mounted the Amazon EFS volume to the instance.

EXERCISE 6.4

Create and Use an Amazon S3 Bucket.

In this exercise, you will create an Amazon S3 bucket and then load and publish content.

1. While signed in to the AWS Management Console, open the Amazon S3 service dashboard.
2. Select Create Bucket.
3. Provide a globally unique name for the bucket, and choose the appropriate region. Leave all other settings at their defaults for now. Create the bucket.
4. Select the bucket, and choose Upload.
5. Click Add Files, and navigate to an image that you want to share publicly. Click Next.
6. Because you want this image to be publicly accessible, under Manage Public Permissions, give everyone read access to the object. Review, and upload the image.
7. Click on the object name to go to the properties screen. Select, and open the URL for the image in a new browser window. You should now see your image.

You have now created a bucket and loaded content that is web-enabled and publicly accessible.

EXERCISE 6.5**Enable Amazon S3 Versioning.**

In this exercise, you will activate Amazon S3 versioning, which protects objects from overwrites and accidental deletions.

1. While signed in to the AWS Management Console, open the Amazon S3 service dashboard.
2. Select the bucket that you created in Exercise 6.4, and open the properties tab for the bucket.
3. Click on the versioning box, enable versioning, and save.

The bucket now has versioning turned on until it is suspended.

EXERCISE 6.6**Enable Cross-Region Replication.**

In this exercise, you will enable cross-region replication of the contents of the original bucket to a new bucket in a different region.

1. While signed in to the AWS Management Console, open the Amazon S3 service dashboard.
2. Create a new bucket in a different region from the bucket that you created in Exercise 6.4. Enable versioning on the new bucket.
3. Select the original bucket as the source and the new bucket as the target.
4. Choose the storage class for the target.
5. For IAM, select Create New Role, and save.
6. Load a new image in the origin bucket. It should appear in the target.

You have now enabled cross-region replication, which can be used for compliance and to protect content from extreme disaster.

EXERCISE 6.7**Create an Amazon Glacier Vault.**

In this exercise, you will create an Amazon Glacier vault. Be careful if you choose to load anything to the vault as you will be responsible for the storage costs for a minimum of 90 days.

1. While signed in to the AWS Management Console, open the Amazon Glacier service dashboard.
2. Create a new vault.

3. Choose the region and provide a globally unique vault name.
4. Set notification rules as needed.
5. Save changes to create the vault.

The new vault has been created.

EXERCISE 6.8

Enable Lifecycle Rules.

In this exercise, you will link content created in the Amazon S3 bucket with a lifecycle that will move the content to Amazon Glacier and eventually purge the object.

1. While signed in to the AWS Management Console, open the Amazon S3 service dashboard.
2. Select the bucket that you created in Exercise 6.4, and go to the Management tab.
3. Under Lifecycle, choose Add Lifecycle Rule.
4. Name the rule **Move to Glacier**.
5. For transition, select Current Version. Add a transition rule to move to Amazon Glacier after 30 days.
6. For Expiration, select current and previous versions and Expire After 31 Days.
7. For incomplete multipart uploads, expire them after seven days.

Your account will now automatically move content from Amazon S3 to Amazon Glacier after 30 days and delete the content from Amazon S3 one day later.

Review Questions

1. You are running a website that keeps historical photos from previous elections. Users need to be able to search for and display images from the last 100 years. On average, most images are requested only once or twice a year. What would be the most cost-effective, highly available method of storing and serving the images?
 - A. Save images on Amazon Elastic Block Store (Amazon EBS) and use a web server running on Amazon Elastic Compute Cloud (Amazon EC2).
 - B. Save images on Amazon Elastic File System (Amazon EFS) and use a web server running on Amazon EC2.
 - C. Save images on Amazon Simple Storage Service (Amazon S3) and serve direct from Amazon S3.
 - D. Save images on Amazon Simple Storage Service – Infrequent Access (Amazon S3-IA) and serve direct from Amazon S3.
 - E. Save images on Amazon Glacier and serve direct from Amazon Glacier.
2. You are running a legacy Sybase database on an Amazon Elastic Compute Cloud (Amazon EC2) instance running Windows. You average 1,200 transactions per second against the database, but at peak levels have as many as 2,500 transactions per second. The current size of the database is 1.8 TB. What is the best data volume Amazon Elastic Block Store (Amazon EBS) configuration for cost without sacrificing performance?
 - A. One Amazon EBS Magnetic Volume provisioned at 2 TB with 2,500 provisioned IOPS
 - B. Two Amazon EBS Magnetic Volumes, each provisioned at 1 TB and Raid0 connected
 - C. One Amazon EBS gp2 volume provisioned at 2 TB
 - D. One Amazon EBS Solid State Drive (SSD) provisioned at 2 TB with 2,500 provisioned IOPS
 - E. Two Amazon EBS gp2 volumes, each provisioned at 1 TB and Raid0 connected
3. Your application's Amazon Elastic Compute Cloud (Amazon EC2) instances need a single shared volume to edit common files. What is the best volume to attach to the instances?
 - A. One Amazon Elastic Block Store (Amazon EBS) volume with IOPS
 - B. One Amazon EBS ma1 volume
 - C. One Amazon Elastic File System (Amazon EFS) volume
 - D. One Amazon Simple Storage Service (Amazon S3) volume
 - E. One Amazon EBS gp2 volume
4. Your company is planning to keep a media archive of photos that are rarely accessed (no more than 10 times a year on average). Business needs expect that the media be available for publishing on request with a response time of less than 800 ms. What is the most cost-efficient storage method for the media?
 - A. Amazon Elastic Block Store (Amazon EBS) with provisioned IOPS
 - B. Amazon EBS gp2

- C. Amazon Simple Storage Service – Reduced Redundancy Storage (Amazon S3-RSS)
 - D. Amazon S3 – Infrequent Access (Amazon S3-IA)
 - E. Amazon Glacier
5. Which of the following would be good use cases for Amazon Simple Storage Service (Amazon S3)? (Choose three.)
- A. Compiled application installers
 - B. Video clips storage from motion-activated cameras
 - C. .dat files from active databases
 - D. Scratch disk for video transcoders
 - E. Data warehouse repositories
 - F. Amazon Elastic Compute Cloud (Amazon EC2) session state offloading
6. Your company has a compliance requirement to record all writes to an Amazon Simple Storage Service (Amazon S3) bucket and any time that content was read publicly. What are the two steps needed to achieve compliance? (Choose two.)
- A. Activate AWS Identity and Access Management (IAM) logging.
 - B. Activate AWS CloudTrail logging.
 - C. Activate Amazon CloudWatch logging.
 - D. Activate Server Access logging.
 - E. Activate ClearCut logging.
7. Your company must retain legacy compliance data for five years in “an immutable form” in the unlikely event of a tax audit. What is the most cost-effective method that will achieve compliance?
- A. Amazon Glacier with Vault Lock activated
 - B. Amazon Glacier with AWS Identity and Access Management (IAM) permissions to edit and delete objects denied
 - C. Amazon Simple Storage Service (Amazon S3) with cross-region replication activated
 - D. Amazon S3 Infrequent Access (Amazon S3-IA) with Bucket Lock activated
 - E. AWS Storage Gateway with tape interface
8. You want to use AWS Storage Gateway to increase the amount of storage available to your on-premises application block storage systems. What is the correct configuration?
- A. AWS Storage Gateway file interface
 - B. AWS Storage Gateway volume interface with cached mode
 - C. AWS Storage Gateway volume interface with stored mode
 - D. AWS Storage Gateway tape interface

9. What is a good use case for Amazon Elastic Compute Cloud (Amazon EC2) instance store?
 - A. Compiled application installers
 - B. Video clips storage from motion-activated cameras
 - C. .dat files from active databases
 - D. Scratch disk for video transcoders
 - E. Data warehouse repositories
 - F. Amazon EC2 session state offloading
10. What step must you do as part of provisioning and mounting Amazon Elastic File System (Amazon EFS) on your Amazon Elastic Compute Cloud (Amazon EC2) instance?
 - A. Select the authorized AWS Key Management Service (AWS KMS) master key.
 - B. Create mount targets in the appropriate Amazon Virtual Private Cloud (Amazon VPC) subnets.
 - C. Activate versioning on the associated Amazon Simple Storage Service (Amazon S3) buckets.
 - D. Create the Amazon EFS role in AWS Identity and Access Management (IAM).
 - E. Connect to the iSCSI Amazon EFS endpoint.
11. In what ways does Amazon Simple Storage Service (Amazon S3) object storage differ from block and file storage? (Choose two.)
 - A. Amazon S3 stores data in fixed blocks.
 - B. Objects can be any size.
 - C. Objects are stored in buckets.
 - D. Objects contain both data and metadata.
12. Which of the following are features of Amazon Elastic Block Store (Amazon EBS)? (Choose two.)
 - A. Data stored on Amazon EBS is automatically replicated within an Availability Zone.
 - B. Amazon EBS data is automatically backed up to tape.
 - C. Amazon EBS volumes can be encrypted transparently to workloads on the attached instance.
 - D. Data on an Amazon EBS volume is lost when the attached instance is stopped.
13. You need to take a snapshot of an Amazon Elastic Block Store (Amazon EBS) volume. How long will the volume be unavailable?
 - A. It depends on the provisioned size of the volume.
 - B. The volume will be available immediately.
 - C. It depends on the amount of data stored on the volume.
 - D. It depends on whether the attached instance is an Amazon EBS-optimized instance.

- 14.** You are restoring an Amazon Elastic Block Store (Amazon EBS) volume from a snapshot. How long will it be before the data is available?
- A.** It depends on the provisioned size of the volume.
 - B.** The data will be available immediately.
 - C.** It depends on the amount of data stored on the volume.
 - D.** It depends on whether the attached instance is an Amazon EBS-optimized instance.
- 15.** You store critical data in Amazon Simple Storage Service (Amazon S3); the data must be protected against inadvertent or intentional deletion. How can this data be protected? (Choose two.)
- A.** Use cross-region replication to copy data to another bucket automatically.
 - B.** Set a vault lock.
 - C.** Enable versioning on the bucket.
 - D.** Use a lifecycle policy to migrate data to Amazon Glacier.
 - E.** Enable MFA Delete on the bucket.
- 16.** Amazon Glacier is well-suited to data that is which of the following? (Choose two.)
- A.** Is infrequently or rarely accessed
 - B.** Must be immediately available when needed
 - C.** Is available after a three- to five-hour restore period
 - D.** Is frequently erased within 30 days
- 17.** Amazon EFS supports all of the Windows operating systems.
- A.** True
 - B.** False
- 18.** When using Amazon Glacier, the size of an individual archive can be virtually unlimited.
- A.** True
 - B.** False
- 19.** You can take periodic snapshots of instance storage.
- A.** True
 - B.** False
- 20.** How do you resize an instance store-backed volume?
- A.** Stop the Amazon Elastic Cloud Compute (Amazon EC2) instance and use the resize volume command.
 - B.** Take a snapshot of the instance volume, resize the instance store volume, and use the snapshot to restore data to the new resized instance-store volume.
 - C.** You cannot resize an instance store volume.
 - D.** Attach another instance volume, copy the data to the new volume, and delete the old volume.

Chapter

7



Databases

THE AWS CERTIFIED SYSOPS ADMINISTRATOR - ASSOCIATE EXAM TOPICS COVERED IN THIS CHAPTER MAY INCLUDE, BUT ARE NOT LIMITED TO, THE FOLLOWING:

Domain 1.0: Monitoring and Metrics

- ✓ 1.1 Demonstrate ability to monitor availability and performance
- ✓ 1.2 Demonstrate ability to monitor and manage billing and cost optimization processes

Domain 2.0: High Availability

- ✓ 2.1 Implement scalability and elasticity based on scenario
- ✓ 2.2 Ensure level of fault tolerance based on business needs

Domain 4.0: Deployment and Provisioning

- ✓ 4.1 Demonstrate the ability to build the environment to conform with the architected design
- ✓ 4.2 Demonstrate the ability to provision cloud resources and manage implementation automation

Domain 5.0: Data Management

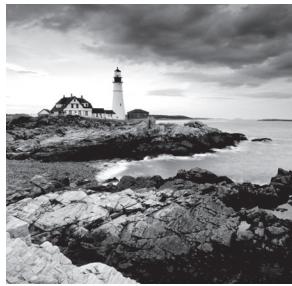
- ✓ 5.1 Demonstrate ability to create backups for different services
- ✓ 5.3 Manage backup and disaster recovery processes

Domain 6.0: Security

- ✓ 6.1 Implement and manage security policies
- ✓ 6.2 Ensure data integrity and access controls when using the AWS platform
- ✓ 6.3 Demonstrate understanding of the shared responsibility model

Domain 7.0: Networking

- ✓ 7.2 Demonstrate ability to implement connectivity features of AWS



Introduction to AWS Databases

This chapter is an overview of relational and non-relational database systems that can be deployed on AWS. It will explore the following topics:

- Amazon Relational Database Service (Amazon RDS)
- Amazon DynamoDB
- Amazon ElastiCache
- Amazon Redshift
- Monitoring and maintaining databases
- Database security

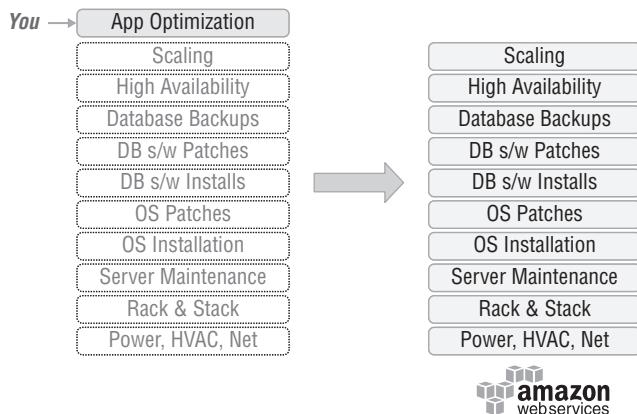
AWS provides fully managed relational and NoSQL database services, a fully managed in-memory database caching service, and a fully managed petabyte-scale data warehouse service. If you require a highly-customized database solution, you can deploy it using Amazon Elastic Compute Cloud (Amazon EC2) and Amazon Elastic Block Store (Amazon EBS).

When deploying a database solution on-premises, or using an Amazon EC2 instance, all aspects of the database must be managed by the customer. In the case of using an Amazon EC2 instance, the instance needs to be provisioned with an operating system, the operating system has to be patched/maintained, the Database Management System (DBMS) has to be installed/patched/maintained, the DBMS has to be architected to be highly available and to scale appropriately, and the data has to be managed.

AWS can manage the provisioning and maintenance of a DBMS and allow customers to focus their time and energy on managing and maintaining their data.

AWS offers a wide range of fully managed database services. These services include Amazon RDS, Amazon DynamoDB, Amazon Redshift, and Amazon ElastiCache.

As part of AWS database offering, there is also the AWS Database Migration Service, which makes it easy and inexpensive to migrate databases into the AWS Cloud. Figure 7.1 illustrates how database administration changes when managed by AWS.

FIGURE 7.1 The benefit of a managed service

Amazon RDS is designed to make it easy to set up, operate, and scale a relational database in the cloud. It provides cost-efficient and resizable capacity while managing time-consuming database administration tasks, freeing customers to focus on applications and business requirements. Amazon RDS has six database engines to choose from: Amazon Aurora, PostgreSQL, MySQL, MariaDB, Oracle, and Microsoft SQL Server.

SQL vs. NoSQL

In general, database solutions are often referred to as being one of two types: SQL or NoSQL. These descriptions are not entirely accurate, however. The correct descriptors are relational and non-relational. SQL, the Structured Query Language, can actually be used to run reports from both relational and non-relational databases but it is associated with relational databases. Non-relational databases can use SQL, but they also can use other languages for processing. Because of this, NoSQL/non-relational systems are sometimes referred to as “Not Only SQL” databases.

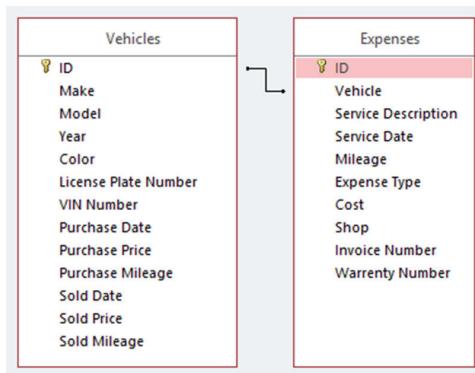
Amazon RDS makes it easy to set up, operate, and scale a relational database in the cloud. It provides cost-efficient, resizable capacity for industry-standard relational databases and manages common database administration tasks.

Amazon DynamoDB is a fully managed NoSQL database service that provides fast and predictable performance with seamless scalability. It enables customers to offload the administrative burdens of operating and scaling distributed databases to AWS so that they don't have to worry about hardware provisioning, setup and configuration, replication, software patching, or cluster scaling. Amazon DynamoDB automatically partitions and repartitions data and provisions additional server capacity as table sizes grow or provisioned throughput increases. In addition, Amazon DynamoDB synchronously replicates data across three facilities in an AWS Region, providing high availability and data durability automatically.

Relational Databases Overview

A *Relational Database Management System (RDBMS)* stores data in one or more tables formatted into rows and columns, with each row getting a unique key. Database tables usually contain a specific type of information, such as an address or a person's name. Because each row in a table has a unique key, it can be related to a unique key (or set of keys) in another table. Figure 7.2 shows two tables, Vehicles and Expenses, and the shared key of ID in both.

FIGURE 7.2 Relational database tables with a shared key



Relational databases are incredibly complex. A relatively simple query could have multiple execution paths which an optimizer evaluates at runtime. When taken as a whole, these key relationships are called *schemas*. In a relational database, the schema must be clearly defined before any information can be added to it. A well-designed schema minimizes data redundancy and ensures data integrity. After a schema has been designed and implemented, changes are both difficult and expensive.

SQL was initially developed at IBM in the 1970s, and it is the most widely used database language in the industry. Because of this widespread usage, relational databases are often referred to as SQL databases. SQL, however, is a language and not a database engine.

Relational Database Design

SQL database design centers on a technique called *normalization*. The goal of normalization is to reduce or eliminate duplicate data in a database to reduce errors in stored data. Each table in a relational database ideally holds data of one type or thing (for example, addresses). The trade-off is less flexibility when application changes impact more than one database table.

ACID

Relational databases use a set of rules called Atomicity, Consistency, Isolation, and Durability (ACID) to guide database design. A sequence of database operations that perform a single

logical operation is called a *transaction*. The four ACID components ensure that a database transaction is completed in a timely manner.

Atomicity Database transactions can be broken down into smaller parts. An atom is the smallest part of an element that can exist and still be recognized. *Atomicity* refers to the integrity of an entire database transaction (as opposed to a single part of that transaction). Atomic data, for example, might be a person's name.

Consistency For any database to operate as intended, it must follow appropriate validation rules. *Consistency* means that only data that follows these rules is permitted to be written to the database. After a successful transaction, new data will be added to the database and the state of the database will be consistent with existing rules. If a transaction occurs and the result is data that does not follow the rules, it will be "rolled back." The rollback is to a previous state that complies with the rules.

Here is a simplistic example:

If a database entry is an email address, then a valid address with an at sign (@) is required. Without this character, the rule is broken and the data should not be written.

Isolation Every database transaction acts on the data separately. *Isolation* refers to the ability of a database to process multiple transactions simultaneously without interference. If two transactions attempt to perform an operation on the same dataset, the first must be completed before the second begins.

If a database application is accessing inventory, two transaction cannot deplete the total at the same time. The first transaction should be completed before the second one starts.

Durability "Everything fails, all the time." This oft-quoted statement from Amazon's Chief Technology Officer, Werner Vogels, is very relevant to database discussions. The idea is that if a system is designed to tolerate failure, things will not really fail. At the very least, as systems operators, the goal is to make failures invisible to customers. In this context, *durability* means that data is saved only when a transaction has been completed, and data is stored in a way that tolerates power loss, system crash, or other error.

In a distributed system, one way to achieve ACID is to use a two-phase commit. This ensures that all involved databases must commit to transaction completion. If one or more cannot commit, the transaction is rolled back.

Non-Relational Database Overview

In contrast to relational databases, non-relational databases were created to support the requirements of cloud-based applications and were built to overcome the limitations of relational databases in terms of scale, performance, data modeling, and data distribution. Just as relational databases are referred to as SQL databases, non-relational databases are called *NoSQL databases*. As SQL is a language, however, it is possible to use it to access non-relational databases.

NoSQL databases use a variety of data models, including document, graph, key/value, and columnar. NoSQL databases are widely recognized for ease of development, scalable

performance, high availability, and resilience. Key/value pairs are the main feature of NoSQL databases. Keys are names or unique ID numbers, and values range from simple data to documents to columns of data to structured lists (arrays) of key/value data. Each row in a NoSQL table includes the key and its value.

The design of NoSQL depends on the type of database, called *stores*.

Document stores These types of stores pair each key identifier with a document that can be a document, key/value pair, or key/value array.

Graph stores These stores are designed to hold data best represented by graphs and interconnected data with an unknown number of relations between the data points.

Key/value stores These types of stores are the simplest type, with data stored with a name, the key, and its associated data; that is, the value.

Wide column stores These stores are optimized for queries across large datasets.



Amazon DynamoDB is a NoSQL database that uses documents and key/value pairs to store data.

There are other ways to describe the range of NoSQL databases available, but these are the simplest and most comprehensive categories. Within each type of NoSQL database, the available functionality differs and can impact database design. As an example, MongoDB evolved from MySQL. Because of this, MongoDB retains many of features native to relational databases.

NoSQL databases often trade some ACID properties of traditional relational database management systems for a more flexible data model that scales horizontally and can be modified dynamically. These characteristics make NoSQL databases an excellent choice in situations where traditional RDBMSs encounter architectural challenges overcoming some combination of performance bottlenecks, scalability, operational complexity, and increasing administration and support costs.

NoSQL databases were designed to complement existing relational database systems, not replace them. NoSQL is designed for distributed data stores for very large-scale data needs.

In a NoSQL database, there is no fixed schema and no table joins. An RDBMS “scales up” by adding faster CPUs and increasing memory. NoSQL databases can take advantage of “scaling out.” Scaling out refers to adding more identical nodes to spread the load. This is what makes a NoSQL database an inexpensive solution for large datasets.

Amazon RDS Features and Benefits

Amazon RDS includes the following features and benefits:

- CPU, memory, storage, and IOPS can be scaled independently.
- AWS manages backups, software patching, automatic failure detection, and recovery.
- Automated backups can be performed as needed, or manual backups can be triggered. Backups can be used to restore a database; the Amazon RDS restore process works reliably and efficiently.

- Amazon RDS provides high availability with a primary instance that can fail over to a secondary instance when a problem occurs.
- Amazon RDS provides elasticity and scalability by enabling Amazon Aurora, MySQL, MariaDB, or PostgreSQL Read Replicas to increase read scaling.
- Amazon RDS supports MySQL, MariaDB, PostgreSQL, Oracle, Microsoft SQL Server, and the MySQL-compatible Amazon Aurora database engines.
- AWS Identity and Access Management (IAM) can help to control which AWS resources have access to Amazon RDS databases.
- Databases can be further protected by putting them in an Amazon Virtual Private Cloud (Amazon VPC) using Secure Sockets Layer (SSL) for data in transit and encryption for data at rest.

Amazon RDS Overview

The Amazon RDS Database Engines are the Following:

- Amazon Aurora
- PostgreSQL
- MySQL
- MariaDB
- Oracle
- Microsoft SQL Server

Consider Using a Relational Database When the Following is Required:

- Transaction support
- ACID compliance
- Table joins
- SQL

If these items are not required, consider using a NoSQL database solution, such as Amazon DynamoDB.

Amazon RDS DB Instances

A *DB instance* is a logically-isolated database environment running in the cloud, and it is the basic building block of Amazon RDS. A DB instance can contain multiple user-created databases, and these databases are accessed using the same client tools and applications that you use to access a standalone DB instance. DB instances are created and modified using the AWS command-line tools, Amazon RDS Application Programming Interfaces (APIs), or the AWS Management Console.



Amazon RDS only supports access to its databases using SQL client applications. Amazon RDS does *not* allow direct host access.

There is a soft limit of 40 Amazon RDS DB instances per account. Of these 40 instances, up to 10 can be Oracle or Microsoft SQL Server DB instances under the License Included model. Customers who use the Bring Your Own License (BYOL) model can use all 40 DB instances for Oracle or Microsoft SQL Server.



If more DB instances are required, the soft limit can be raised by contacting AWS Support.

There are three DB instance types used by Amazon RDS. They are the *M* (for Multipurpose) type, the *R* (for RAM/Memory optimization) type, and the *T* (for Tiny/Burstable) type.

Amazon Aurora

Amazon Aurora is a fully managed, MySQL-compatible, relational database engine that combines the speed and reliability of high-end commercial databases with the simplicity and cost effectiveness of open source databases. It delivers up to five times the performance of MySQL without requiring changes to most of your existing applications.

Amazon RDS performs routine database tasks, such as provisioning, patching, backup, recovery, failure detection, and repair. Amazon RDS also provides push-button migration tools to convert existing Amazon RDS for MySQL applications to Amazon Aurora.

Amazon Aurora is a drop-in replacement for MySQL. The code, tools, and applications you use today with your existing MySQL databases can be used with Amazon Aurora.

When creating an Amazon Aurora instance, a DB cluster is created. This DB cluster consists of one or more DB instances and a volume that manages the data for those instances. An Amazon Aurora cluster volume is a virtual database storage volume that spans multiple Availability Zones, with each Availability Zone containing a copy of the cluster data.

Two types of instances make up an Aurora DB cluster: a primary instance and an Aurora Replica.

Primary Instance

A *primary instance* supports read-write workloads and performs all of the data modifications to the cluster volume. Each Amazon Aurora DB cluster has one primary instance.

Aurora Replica

An *Aurora Replica* supports only read operations. Each DB cluster can have up to 15 Aurora Replicas in addition to the primary instance, which supports both read and write workloads. Multiple Aurora Replicas distribute the read workload, and by locating Aurora Replicas in separate Availability Zones, you can also increase database availability.

Amazon Aurora Volumes

Amazon Aurora data is stored in the cluster volume. This is a single, virtual volume that uses Solid State Drives (SSDs). A cluster volume consists of copies of the data across multiple Availability Zones in a single region. Because the data is automatically replicated across

Availability Zones, data is highly durable. This replication also ensures that databases are more available during failover events because data copies already exist in the other Availability Zones and continue to serve data requests to the instances in your DB cluster.

Amazon Aurora cluster volumes automatically grow as the amount of data in a database increases. An Amazon Aurora cluster volume can grow to a maximum size of 64 TB. Table size is limited to the size of the cluster volume. This means that the maximum size for a table in an Amazon Aurora DB cluster is 64 TB.

Even though an Amazon Aurora cluster volume can grow to up to 64 TB, customers are only charged for the space that is used in an Amazon Aurora cluster volume.

Amazon Aurora Replication

Aurora Replicas are independent endpoints in an Amazon Aurora DB cluster. They provide read-only access to the data in the DB cluster volume and enable customers to scale the read workload for data over multiple replicated instances. This improves the performance of data reads and increases the availability of the data in your Amazon Aurora DB cluster.

Aurora Replicas are also failover targets. They can be quickly promoted if the primary instance in an Amazon Aurora DB cluster fails.

Amazon Aurora Reliability

Amazon Aurora is designed to be reliable, durable, and fault tolerant. An Amazon Aurora DB cluster can be architected to improve availability by adding Aurora Replicas and placing them in different Availability Zones.

Amazon Aurora also includes several automatic features to enhance its reliability.

Storage auto-repair Because Amazon Aurora maintains multiple copies of data in three Availability Zones, the chance of losing data as a result of a disk failure is greatly minimized. Amazon Aurora automatically detects failures in the disk volumes that make up the cluster volume. When a segment of a disk volume fails, Amazon Aurora immediately repairs the segment. When Amazon Aurora repairs the disk segment, it uses the data in the other volumes that make up the cluster volume to ensure that the data in the repaired segment is current. As a result, Amazon Aurora avoids data loss and reduces the need to perform a point-in-time restore to recover from a disk failure.

“Survivable” cache warming Amazon Aurora “warms” the buffer pool cache when a database starts after it has been shut down or restarted after a failure. This means that Amazon Aurora preloads the buffer pool with the pages for known common queries that are stored in an in-memory page cache. This provides a performance gain by bypassing the need for the buffer pool to “warm up” from normal database use.

The Amazon Aurora page cache is managed in a separate process from the database, which allows the page cache to “survive” independently of the database. In the unlikely event of a database failure, the page cache remains in memory, which ensures that the buffer pool is warmed with the most current state when the database restarts.

Crash recovery Amazon Aurora is designed to recover from a crash nearly instantaneously and continue to serve application data. Amazon Aurora performs crash recovery asynchronously on parallel threads. After a crash, a database is open and available immediately.

Amazon Aurora Security

Security for Amazon Aurora is managed at three levels: IAM, Amazon VPC security groups, and database authentication.

AWS Identity and Access Management (IAM) Use IAM to control who can perform Amazon RDS management actions on Amazon Aurora DB clusters and DB instances. When connecting to AWS using IAM credentials, the IAM account must have IAM policies that grant the permissions required to perform Amazon RDS management operations.

Amazon VPC security groups Amazon Aurora DB clusters must be created in an Amazon VPC. Amazon VPC security groups control which services and Amazon EC2 instances can connect to the endpoint and port of an Amazon Aurora DB instance.

Database authentication To authenticate login and permissions for an Amazon Aurora DB cluster, one or both of the following approaches can be used.

MySQL commands MySQL commands such as CREATE USER, RENAME USER, GRANT, REVOKE, and SET PASSWORD manage access the same way in Amazon Aurora as they do in other MySQL databases. It is also possible to modify database schema tables directly.

IAM database authentication With IAM database authentication, you authenticate to a DB cluster by using an IAM user or IAM role and an authentication token. An authentication token is a unique value that is generated using the Signature Version 4 signing process. By using IAM database authentication, it is possible to use the same credentials to control access to both your AWS resources and your databases.

Securing Amazon Aurora Data with SSL

Amazon Aurora DB clusters support SSL connections from applications using the same process and public key as Amazon RDS MySQL DB instances.

Amazon RDS creates an SSL certificate and installs the certificate on the DB instance when Amazon RDS provisions the instance. These certificates are signed by a certificate authority. The SSL certificate includes the DB instance endpoint as the Common Name for the SSL certificate to guard against spoofing attacks.

As a result, it is only possible to use the DB cluster endpoint to connect to a DB cluster using SSL if the client supports Subject Alternative Names. Otherwise, the endpoint of the primary instance must be used.



AWS recommends using the MariaDB Connector/J client because it supports Subject Alternative Names with SSL.



Using the Amazon RDS APIs or the AWS Management Console, it is possible to scale up or down the compute and memory resources powering an Amazon Aurora DB instance deployment using *Push-Button Compute Scaling*. The maximum compute available is 32 vCPUs and 244 GB of memory. Compute scaling operations typically complete in a few minutes.

Amazon Aurora PostgreSQL Edition

In addition to being MySQL compatible, Amazon Aurora is also fully compatible with the open source version of PostgreSQL version 9.6.1.

Amazon RDS Storage

Amazon RDS provides three storage types: magnetic, General Purpose (SSD), and Provisioned IOPS. They differ in performance characteristics and price, which allows users to tailor storage performance and cost to the needs of their respective workloads.

When using the Provisioned IOPS and General Purpose (SSD) storage types, note the following:

- MySQL, MariaDB, PostgreSQL, and Oracle Amazon RDS DB instances can be created with up to 6 TB of storage.
- Microsoft SQL Server Amazon RDS DB instances can be provisioned with up to 4 TB of storage.

General Purpose (SSD) General Purpose (SSD) volumes offer cost-effective storage that is ideal for a broad range of workloads. These volumes deliver single-digit millisecond latencies and the ability to burst to 3,000 IOPS for extended periods of time. This storage type is excellent for small- to medium-sized databases.



General Purpose (SSD) is also called gp2.

Provisioned IOPS Provisioned IOPS storage is designed to meet the needs of I/O-intensive workloads, such as databases, which are sensitive to storage performance and consistency in random access I/O throughput. When using Provisioned IOPS, specify the amount of storage to be allocated and then specify the amount of dedicated IOPS desired.



For backwards compatibility, Amazon RDS supports magnetic storage. The recommendation from AWS is to use General Purpose (SSD) or Provisioned IOPS for new storage needs.

Several factors can affect the performance of Amazon RDS volumes, such as instance configuration, I/O characteristics, and workload demand.

Amazon RDS provides several metrics to determine how the storage of a DB instance is performing. These metrics can be viewed in the Amazon RDS console by selecting the DB instance and clicking Show Monitoring. Amazon CloudWatch can also be used to monitor these metrics.

IOPS The number of I/O operations completed per second. This metric is reported as the average IOPS for a given time interval. Amazon RDS reports read and write IOPS separately in one-minute intervals. Total IOPS is the sum of the read and write IOPS. Typical values for IOPS range from zero to tens of thousands per second.

Latency The elapsed time between the submission of an I/O request and its completion. This metric is reported as the average latency for a given time interval. Amazon RDS reports read and write latency separately in one-minute intervals in units of seconds. Typical values for latency are reported as milliseconds. For example, Amazon RDS reports 2 milliseconds as 0.002 seconds.

Throughput The number of bytes per second transferred to or from disk. This metric is reported as the average throughput for a given time interval. Amazon RDS reports read and write throughput separately in one-minute intervals using units of megabytes per second (MB/s). Typical values for throughput range from zero to the I/O channel's maximum bandwidth.

Queue depth The number of I/O requests in the queue waiting to be serviced. These are I/O requests that have been submitted by the application but have not been sent to the device because the device is busy servicing other I/O requests. This metric is reported as the average queue depth for a given time interval. Amazon RDS reports queue depth in one-minute intervals. Typical values for queue depth range from zero to several hundred.



Time spent waiting in the queue is a component of latency and service time. This is not available as a metric. The longer the queue, the worse the latency.

Additional Factors That Impact Storage Performance

All of the following system-related activities consume I/O capacity and may reduce database instance performance while running:

- DB snapshot creation
- Nightly backups
- Multi-AZ peer creation
- Read Replica creation
- Scaling storage

Adding Storage and Changing the Storage Type

For existing MySQL, MariaDB, PostgreSQL, and Oracle DB instances, you might observe some I/O capacity improvement if you scale up your storage.



Because of the extensibility limitations of striped storage attached to a Windows Server environment, Amazon RDS does not currently support increasing storage for Microsoft SQL Server.

It is possible to modify a DB instance to use additional storage and/or convert to a different storage type. Adding storage or converting to a different storage type can take time

and reduces the performance of a DB instance. Plan on scheduling this type of activity in off hours.

Although a DB instance is available for reads and writes when adding storage, it is possible to experience degraded performance until the process is complete. Adding storage could take several hours.

The duration of the process depends on several factors, such as:

- Database load
- Storage size
- Storage type
- Amount of IOPS provisioned

The typical amount of time needed to scale storage will be under 24 hours. In some cases, however, it can take up to several days. During the scaling process, the DB instance will be available for use but might experience performance degradation.



While storage is being added, nightly backups are suspended and no other Amazon RDS operations can take place, including modify, reboot, delete, create Read Replica, and create DB snapshot.

Storage conversions between magnetic storage and General Purpose (SSD) storage can potentially deplete the initial 5.4 million I/O credits ($3,000 \text{ IOPS} \times 30 \text{ minutes}$) allocated for General Purpose (SSD) storage. When performing these storage conversions, the first 82 GB of data will be converted at approximately 3,000 IOPS, while the remaining data will be converted at the base performance rate of 100 IOPS per GB of allocated General Purpose (SSD) storage. This can result in longer conversion times.



It is possible to provision more General Purpose (SSD) storage to increase the base I/O performance rate and improve the conversion time. Once storage has been allocated, it cannot be removed.

I/O Credits and Burst Performance

General Purpose (SSD) storage performance is governed by volume size, which dictates the base performance level of the volume and how quickly it accumulates I/O credits. Larger volumes have higher base performance levels and accumulate I/O credits faster. I/O credits represent the available bandwidth that a General Purpose (SSD) storage can use to burst large amounts of I/O when more than the base level of performance is needed. The more credits a storage volume has for I/O, the more time it can burst beyond its base performance level and the better its performance when more throughput is needed.

When using General Purpose (SSD) storage, a DB instance receives an initial I/O balance of 5.4 million I/O credits, which is enough to sustain a burst performance of 3,000 IOPS for 30 minutes. This initial credit balance is designed to provide a fast initial boot cycle for

boot volumes and to provide a good bootstrapping experience for other applications. Volumes earn I/O credits at the baseline performance rate of 3 IOPS per GB of volume size. For example, a 100 GB General Purpose (SSD) volume has a baseline performance of 300 IOPS.

When storage requires more than the base performance I/O level, it uses I/O credits in the credit balance to burst to the required performance level up to a maximum of 3,000 IOPS. Storage larger than 1,000 GB has a base performance that is equal to or greater than the maximum burst performance. Because of this, its I/O credit balance never depletes and it can burst indefinitely.

When storage uses fewer I/O credits than it earns in a second, unused I/O credits are added to the I/O credit balance. The maximum I/O credit balance for a DB instance using General Purpose (SSD) storage is equal to the initial credit balance (5.4 million I/O credits).

If a storage volume uses all of its I/O credit balance, its maximum performance will remain at the base performance level until I/O demand drops below the base level and unused credits are added to the I/O credit balance. The more storage, the greater the base performance is and the faster it replenishes the credit balance.

Table 7.1 lists several storage sizes and the associated base performance of the storage, the burst duration at the 3,000 IOPS maximum, and the time in seconds that the storage takes to refill an empty credit balance.

TABLE 7.1 Storage and Bursting Rates

| Storage Size (GB) | Base Performance (IOPS) | Maximum Burst Duration @ 3,000 IOPS (Seconds) | Seconds to Fill Empty Credit Balance |
|-------------------|-------------------------|---|--------------------------------------|
| 1 | 100 | 1,862 | 54,000 |
| 100 | 300 | 2,000 | 18,000 |
| 250 | 750 | 2,400 | 7,200 |
| 500 | 1,500 | 3,600 | 3,600 |
| 750 | 2,250 | 7,200 | 2,400 |
| 1,000 | 3,000 | Infinite | N/A |

The burst duration of your storage depends on the size of the storage, the burst IOPS required, and the credit balance when the burst begins.

Burst is calculated using this formula:

$$\text{Burst duration} = \frac{(\text{Credit balance})}{(\text{Burst IOPS}) - 3(\text{Storage size in GB})}$$

If storage performance is frequently limited to the base level due to an empty I/O credit balance, consider allocating more General Purpose (SSD) storage with a higher base performance level. Alternatively, switch to Provisioned IOPS storage for workloads that require sustained IOPS performance.

For workloads with steady-state I/O requirements, provisioning less than 100 GB of General Purpose (SSD) storage may result in higher latencies if you exhaust your I/O burst credit balance.

Amazon RDS Supported Storage Engines

Although MySQL supports multiple storage engines with varying capabilities, not all of them are optimized for recovery and data durability. Amazon RDS fully supports the InnoDB storage engine for MySQL DB instances. Amazon RDS features such as point-in-time restore and snapshot restore require a recoverable storage engine and are only supported using the InnoDB storage engine.



MySQL 5.6 or later is required for the InnoDB Memcached interface.

Multi-AZ Deployments

Amazon RDS *Multi-AZ deployments* provide enhanced availability and durability for DB instances, and it makes them suitable for production database workloads. When provisioning a Multi-AZ DB instance, Amazon RDS automatically creates a primary DB instance and synchronously replicates the data to a standby instance in a different Availability Zone. Each Availability Zone runs on its own physically distinct, independent infrastructure and is engineered to be highly reliable.

In case of an infrastructure failure, Amazon RDS performs an automatic failover to the standby so that database operations resume as soon as the failover is complete. Because the endpoint for a DB instance remains the same after a failover, applications resume operation without the need for manual intervention.



When using Amazon Aurora and there is an infrastructure failure, Amazon RDS can also fail over to a Read Replica.

Enhanced Durability

Multi-AZ deployments for the MySQL, MariaDB, Oracle, and PostgreSQL engines use synchronous physical replication to keep data on the standby up to date with the primary. Multi-AZ deployments for the SQL Server engine use synchronous logical replication to achieve the same result, employing SQL Server-native mirroring technology. Both approaches safeguard data in the event of a DB instance failure or loss of an Availability Zone.

If a storage volume on your primary instance fails in a Multi-AZ deployment, Amazon RDS automatically initiates a failover to the up-to-date standby. In the case of a Single-AZ database failure, a user-initiated, point-in-time-restore operation is required. This operation can take considerable time to complete, and it will include any updates since the latest restorable time.

Amazon Aurora employs a highly durable, SSD-backed virtualized storage layer purpose-built for database workloads. Amazon Aurora automatically replicates a DB volume six ways across three Availability Zones. Amazon Aurora storage is fault tolerant and transparently handles the loss of up to two copies of data without affecting database write availability and up to three copies without affecting read availability. Amazon Aurora storage has been designed to be self-healing. Data blocks and disks are continuously scanned for errors and replaced automatically.

Increased Availability

If an Availability Zone failure or DB instance failure occurs, the impact on availability is limited to the time automatic failover takes to complete. For Amazon Aurora, this is typically under one minute. For other DB engines, the failover time is between one to two minutes.

The availability benefits of Multi-AZ deployments also extend to planned maintenance and backups. In the case of system upgrades like operating system patching or DB instance scaling, they are applied to the standby prior to the automatic failover. As a result, the availability impact is limited to the time for automatic failover to complete.

In a Single-AZ deployment, I/O activity is suspended on the DB instance during backups. During backups on Multi-AZ deployments using MySQL, MariaDB, Oracle, and PostgreSQL engines, I/O does not need to be suspended because the backup is taken from the standby.



It is possible to experience higher than usual latencies during backups in Multi-AZ DB deployments.

On instance failure in Amazon Aurora deployments, Amazon RDS uses Multi-AZ technology to automate failover to one of up to 15 Aurora Replicas created in any of up to three Availability Zones. If no Aurora Replicas have been provisioned, Amazon RDS will automatically attempt to create a new Amazon Aurora DB instance.

Failover

DB instance failover is fully automatic and requires no administrative intervention. Amazon RDS monitors the health of your primary and standby DB instances and initiates a failover automatically in response to a variety of failure conditions.

Failover Conditions

Amazon RDS detects and automatically recovers from the most common failure scenarios for Multi-AZ deployments so that database operations are resumed as quickly as possible without administrative intervention.

Amazon RDS automatically performs a failover in the event of any of the following:

- Loss of availability in primary Availability Zone
- Loss of network connectivity to primary instance
- Compute unit failure on primary instance
- Storage failure on primary instance



Amazon RDS Multi-AZ deployments do not fail over automatically in response to database operations such as long running queries, deadlocks, or database corruption errors.

Create Multi-AZ Amazon RDS Deployments

Using the AWS Management Console, it is possible to create new Multi-AZ deployments or modify existing Single-AZ instances to change them to be Multi-AZ deployments.

To create a new Multi-AZ deployment using the AWS Management Console, click the Yes option for the Multi-AZ Deployment parameter when launching a DB instance.

To convert an existing Single-AZ DB Instance to a Multi-AZ deployment, use the Modify option corresponding to your DB instance in the AWS Management Console.

Other Replication Options

Amazon RDS for MySQL and PostgreSQL combines built-in replication with Read Replicas to scale beyond capacity constraints with read-heavy database workloads. Use Multi-AZ deployments and Read Replicas together to enjoy the complementary benefits of each.

Specify a given Multi-AZ deployment as the source DB instance for a Read Replica. This provides both data durability and availability benefits of Multi-AZ deployments and the read scaling benefits of Read Replicas.



For Multi-AZ deployments, there is also the option to create a Read Replica in an Availability Zone other than the primary and the standby ones for even more redundancy.

High Availability

Amazon RDS provides high availability and failover support for DB instances by using multiple Availability Zones.

Depending on the database engine, Amazon RDS uses one of several different technologies to provide failover support in Multi-AZ deployments:

- Oracle, PostgreSQL, MySQL, and MariaDB DB instances use physical layer replication to keep data consistent on the standby instance.
- Microsoft SQL Server DB instances use SQL Server Mirroring.
- Amazon Aurora instances store copies of the data in a DB cluster across multiple Availability Zones in a single region, regardless of whether the instances in the DB cluster span multiple Availability Zones.



When using the BYOL model, licenses are required for both the primary DB instance and the standby replica.

While AWS is engineered with low-latency network connectivity between Availability Zones, latency might increase in Multi-AZ deployments or when a DB instance fails over to a standby instance. For production workloads, the recommendation is to use Provisioned IOPS and DB instance classes optimized for Provisioned IOPS for fast and consistent performance.



The Amazon RDS high availability feature is not a scaling solution for read-only databases. It is not possible to use a standby replica to serve read traffic. For read-only traffic, Read Replicas should be used.

Read Replicas

Amazon RDS uses functionality built into MySQL, MariaDB, and PostgreSQL (version 9.3.5 and later) database engines to create a special type of DB instance called a *Read Replica* from a source DB instance. Updates made to the source DB instance are asynchronously copied to the Read Replica.



To maintain consistency, Multi-AZ Amazon RDS implementations use synchronous replication. Due to the potential for data loss, asynchronous replication is only used for Read Replicas.

The load on the source DB instance can be reduced by sending read queries from your applications to the Read Replica. Using Read Replicas, it is possible to elastically scale out beyond the capacity constraints of a single DB instance for read-heavy database workloads.

When a Read Replica is created in a single region, Amazon RDS takes a snapshot of the source instance and creates a read-only instance from it. Amazon RDS then uses asynchronous replication to update the Read Replica whenever there is a change to the source DB instance. The Read Replica operates as a DB instance that allows only read-only connections. Applications connect to a Read Replica in the same way that they do to any DB

instance, with a connection string. Amazon RDS replicates all databases in the source DB instance.

When a Read Replica is created, a new Domain Name System (DNS) endpoint is created. To use the Read Replica, applications need to be updated/modified to use this connection string.

If the source DB instance is in a different region than the Read Replica, Amazon RDS sets up a secure communications channel between them.

For a single-AZ DB instance, the DB instance pauses briefly to stop the I/O before a snapshot is taken. The instance then resumes, and the Read Replica is created from the snapshot.

For a Multi-AZ DB instance, a snapshot is made from the standby without pausing the primary. The Read Replica is created from the snapshot.

Important things to know about Read Replicas:

- Read Replicas cannot be used behind an Elastic Load Balancing load balancer.
- Read Replicas can be promoted to standalone/primary databases. This breaks replication and will create a new connection string.
- MySQL and PostgreSQL can have up to five Read Replicas.
- Amazon Aurora can have up to 15 Aurora Replicas as well as 5 MySQL replicas.
- Multi-region Read Replicas are supported using MySQL, Amazon Aurora, and PostgreSQL.
- MySQL Read Replicas can be created from existing Read Replicas. This impacts latency.
- Database snapshots cannot be taken from Read Replicas.
- Read Replicas can be created from the AWS Management Console or by using the API call `CreateDBInstanceReadReplica`.

Failover Process for Amazon RDS

If an Amazon RDS DB instance has been configured for Multi-AZ, in the event of a planned or unplanned outage of a DB instance, Amazon RDS will automatically switch to a standby replica in another Availability Zone.

The time it takes for the failover to complete depends on the database activity and other conditions at the time the primary DB instance became unavailable. Failover times are typically 60-120 seconds. Large transactions or a lengthy recovery process can increase failover time. When the failover is complete, it can take additional time for the Amazon RDS console user interface to reflect the new Availability Zone.

The failover mechanism automatically changes the DNS record of the DB instance's connection string to point to the standby DB instance. As a result, existing connections will need to be reestablished.



For customers that have Java Virtual Machine (JVM) environments, the Java DNS caching mechanism might require a reconfiguration of your environment in order to implement failover policies. Details can be found inside the AWS Software Development Kit (SDK) for Java.

Amazon RDS handles failovers automatically and database operations resume as quickly as possible without administrative intervention.

A Primary DB Instance will Switch Over Automatically to the Standby Replica if Any of the Following Conditions Occur:

- An Availability Zone outage
- The primary DB instance fails.
- The DB instance's server type is changed.
- The operating system of the DB instance is being patched.
- A manual failover of the DB instance was initiated using *Reboot with Failover*.

There are Several Ways to Determine if a Multi-AZ DB Instance Has Failed Over:

- Receiving DB event subscriptions, which can be set up to send notifications via email or Short Message Service (SMS) when a failover has been initiated
- Viewing DB events via the Amazon RDS console or APIs
- Viewing the current state of a Multi-AZ deployment via the Amazon RDS console and APIs

Availability Zone failover can be forced by rebooting the primary DB instance from the Amazon RDS console or using the API call `RebootDBInstance`.

Amazon RDS Backup and Restore

As part of disaster recovery planning, Amazon RDS creates and saves automated backups of DB instances. Amazon RDS creates a storage volume snapshot of your entire DB instance. This results in a backup of the entire instance and not just individual databases.

Amazon RDS creates automated backups of your DB instance during the backup window of a DB instance. Amazon RDS saves the automated backups of a DB instance according to the backup retention period defined when it was created. If necessary, a database can be recovered to any point in time during the backup retention period.

A DB instance must be in the ACTIVE state for automated backups to occur. If a database is in another state, such as `STORAGE_FULL`, automated backups do not occur.

It is possible to create snapshots of a DB instance manually. There is a soft limit of 100 manually created DB instance snapshots.



When deleting an Amazon RDS DB instance, the snapshots created as part of the automatic backups are deleted. Manually created snapshots are *not* automatically deleted, however. These snapshots persist until they are deleted manually.

Backup Storage

Amazon RDS backup storage for each region is composed of the automated backups and manual DB snapshots for that region. Backup storage is equivalent to the sum of the database storage for all instances in that region. Moving a DB snapshot to another region increases the backup storage in the destination region.

All automated backups are deleted when a DB instance is deleted. After a DB instance is deleted, the automated backups cannot be recovered.

MySQL requires InnoDB for automated backups.

When performing an Amazon RDS restore, the engine type (SQL Standard to Enterprise, for example) can be changed if there is enough storage space.

Amazon RDS Snapshot Copy

Amazon RDS offers the ability to copy snapshots from one region to another. The copy can be initiated from the AWS Management Console, the AWS Command Line Interface, or using the Amazon RDS APIs. You can copy snapshots of any size and from any of the database engines that are supported by RDS. Copies can be moved between any of the public AWS Regions, and you can copy the same snapshot to multiple regions simultaneously by initiating more than one transfer.

As is the case with the other copy operations, the copy is done on an incremental basis. Only the data that has changed since the last snapshot of a given database instance will be copied. When you delete a snapshot, deletion is limited to the data that will not affect other snapshots.

There is no charge for the copy operation itself; you pay only for the data transferred out of the source region and for the data storage in the destination region. You are not charged if the copy fails, but you are charged if you cancel a snapshot that is underway at the time.

The Democratization of Disaster Recovery

This feature has been described as “democratizing data recovery.” More than this, it is the democratization of disaster recovery. Imagine all of the headaches and expenses incurred while setting up the network, storage, processing, and security infrastructure needed to do this in an on-premises datacenter. You would have to acquire space, add racks, set up network links and encryption, create the backups, and arrange to copy them from location to location. Without the cloud, you would have to invest thousands of dollars in infrastructure and the same amount, if not more, in DBA and system administrator time.

All of these pain points and associated costs go away when copying backups from region to region using Amazon RDS.

The Backup Window

Automated backups occur daily during the preferred backup window. If the backup requires more time than allotted to the backup window, the backup continues after the window ends until it finishes.

Backup windows cannot overlap with the weekly maintenance window for the DB instance.

During the automatic backup window, storage I/O might be suspended briefly while the backup process initializes. Typically, this is less than a few seconds. Elevated latencies might occur for a few minutes during backups for Multi-AZ deployments.

For MariaDB, MySQL, Oracle, and PostgreSQL, I/O activity is not suspended on your primary during backup for Multi-AZ deployments because the backup is taken from the standby. For SQL Server, I/O activity is suspended briefly during backup for Multi-AZ deployments.

If a preferred backup window is not specified when creating a DB instance, Amazon RDS assigns a default 30-minute backup window selected at random from an 8-hour block of time per region.

The Backup Retention Period

The backup retention period is set when a DB instance is created. If the backup retention period is not set, a default backup retention period is set.

- If the DB instance is created using the Amazon RDS API or the AWS CLI, the default retention period is one day.
- If the DB instance is created using the console, the default is seven days.
- For Amazon Aurora DB clusters, the default backup retention period is one day regardless of how the DB cluster is created.

After a DB instance has been created, the backup retention period can be modified. The backup retention period can be set to be between 1 and 35 days.

If the backup retention period is set to 0, automatic backups will be disabled. Manual snapshot limits (100 per region) do not apply to automated backups.

Disabling and Enabling Automated Backups

It is possible to disable automated backups; however, this practice is highly discouraged.



Disabling automated backups disables point-in-time recovery and *deletes all existing automated backups for the instance*. If automatic backups are disabled and then re-enabled, users can only restore starting at the time they were re-enabled.

A use case for disabling automated backups is when you're loading large amounts of data.

You can disable automated backups for a DB instance by setting the backup retention parameter to 0. To enable automated backups, set the backup retention period to a positive, non-zero value.

Amazon RDS and MySQL Security

Security for Amazon RDS MySQL DB instances is managed at three levels: IAM, security groups, and database authentication.

AWS Identity and Access Management (IAM)

IAM controls who can perform Amazon RDS management actions on DB instances. When connecting to AWS using IAM credentials, the IAM account must have policies granting the permissions required to perform Amazon RDS management operations.

Security Groups

When creating a DB instance, either an Amazon VPC security group or a DB security group controls which devices and Amazon EC2 instances can open connections to the endpoint and port of the DB instance.

Database Authentication

To authenticate login and permissions for a MySQL DB instance, you can take either of the following approaches or a combination of them.

You can take the same approach as with a standalone instance of MySQL. Commands such as CREATE USER, RENAME USER, GRANT, REVOKE, and SET PASSWORD work just as they do in on-premises databases, as does directly modifying database schema tables. For information, see the MySQL User Account Management section in the MySQL documentation <https://dev.mysql.com/doc/>.

You can also use IAM database authentication. With IAM database authentication, you authenticate to your DB instance by using an IAM user or IAM role and an authentication token. An authentication token is a unique value that is generated using the Signature Version 4 signing process. By using IAM database authentication, you can use the same credentials to control access to your AWS resources and your databases.

Amazon RDS Security Groups

Security groups control the access traffic in and out of a DB instance. Three types of security groups are used with Amazon RDS.

DB Security Groups

- A DB security group controls access to Amazon EC2-Classic DB instances that are *outside* of an Amazon VPC.
- DB security group rules apply to inbound traffic only. Outbound traffic is not currently permitted for DB instances.

Amazon VPC Security Groups

- An Amazon VPC security group controls access to DB instances and Amazon EC2 instances inside an Amazon VPC. The source can be a range of addresses or another Amazon VPC security group.
- By specifying an Amazon VPC security group as the source, incoming traffic from all instances that use the source Amazon VPC security group is allowed.
- Amazon VPC security groups can have rules that govern both inbound and outbound traffic, though the outbound traffic rules do not apply to DB instances.
- To create Amazon VPC security groups, you must use the Amazon EC2 API or the Security Group option on the Amazon VPC console.

Amazon EC2 Security Groups

- An Amazon EC2 security group controls access to and from an Amazon EC2 instance.
- Rules can be modified at any time, and new rules are automatically applied.

By default, network access is turned off to DB instances. To gain access to an Amazon RDS DB instance, create a security group that allows access from an IP address range, port, or Amazon EC2 security group. After the ingress rules are configured, the same rules apply to all DB instances that are associated with that security group.



Security groups can only contain up to 20 rules. This is a soft limit. It can be raised by contacting AWS Support. Raising security group limits may impact performance.

Encrypting Amazon RDS Resources

Amazon RDS instances and snapshots can be encrypted at rest by enabling the encryption option. Data encrypted at rest includes the underlying storage for a DB instance, its automated backups, Read Replicas, and snapshots.

Amazon RDS encrypted instances use the industry-standard Advanced Encryption Standard (AES) 256 encryption algorithm to encrypt data. After data is encrypted, Amazon RDS handles authentication of access and decryption of data transparently with a minimal impact on performance. Database clients do not need to be modified to use encryption.

The *AWS Key Management Service (AWS KMS)* is used to manage the keys for encrypting and decrypting Amazon RDS resources. AWS KMS combines secure, highly available hardware and software to provide a key management system scaled for the cloud.

AWS KMS can be used both to create encryption keys and to define policies that control how these keys can be used. AWS KMS supports AWS CloudTrail, so that key usage can be audited.

After an Amazon RDS DB instance has been encrypted, all logs, backups, and snapshots are encrypted. Read Replicas of an Amazon RDS encrypted instance are encrypted using the same key as the primary DB instance.

Limitations of Amazon RDS Encryption:

- Encryption for an Amazon RDS DB instance must be enabled when it is being created.
- DB instances that are encrypted cannot be modified to disable encryption.
- If a DB instance has been encrypted, the Read Replica must also be encrypted.
- Encrypted Read Replicas must be encrypted with the same key as the source DB instance.
- It is not possible to restore an unencrypted backup or snapshot to an encrypted DB instance.
- It is not possible to restore an encrypted MySQL DB snapshot to an Amazon Aurora DB cluster.
- To copy an encrypted snapshot from one region to another, the AWS KMS key identifier of the destination region must be specified. AWS KMS encryption keys are specific to the region in which they are created.

Option Groups

Some DB engines offer additional features that make it easier to manage data and databases as well as provide additional security. Amazon RDS uses *option groups* to enable and configure these features.

Options can have settings that specify how an individual option is configured. When a DB instance is associated with an option group, the specified options and their related settings are enabled for that DB instance.

Currently, Amazon RDS Supports Options for the Following Database Engines:

- MariaDB
- Microsoft SQL Server
- MySQL
- Oracle

Scaling Relational Databases

As a managed service, Amazon RDS takes care of the scaling of relational databases so that they can keep up with demand. Scaling methods are typically categorized as either horizontal or vertical scaling, with some combination of both occasionally referred to as *scaling diagonally*. *Vertical scaling*, also referred to as *scaling up*, is adding additional resources to an existing instance to improve performance. *Horizontal scaling*, referred to as *scaling out*, is the addition of identical resources to manage the workload.

Scaling Up

For large databases, especially ones used for web applications, a primary concern for users is scalability. As more and more applications are created in environments that have massive workloads, scalability requirements can change very quickly and grow very large. Relational databases scale well, but usually only on a single server.

To handle a higher load in your database, you can vertically scale up your primary DB instance with a push of a button. There are currently more than 18 instance sizes to choose from when resizing an Amazon RDS MySQL, PostgreSQL, MariaDB, Oracle, or Microsoft SQL Server instance. For Amazon Aurora, there are five memory-optimized instance sizes available.

It is possible to scale the compute resources and storage capacity allocated to your DB instance with the AWS Management Console, the Amazon RDS API, or the AWS CLI. Memory and CPU resources are modified by changing your DB instance class, and storage available is changed when storage allocation is modified.

When a DB instance class or allocated storage is modified, the requested changes will be applied during the specified maintenance window. It is possible to set the apply-immediately flag to force the changes. Any pending system changes will be applied as well.

The Following are Points to Consider When Scaling up an Amazon RDS Instance:

- Before scaling, make sure to have the correct licensing in place for commercial engines: Microsoft SQL Server and Oracle. For commercial engines, licenses are often tied to the CPU sockets or cores.
- Determine when the scaling change should be applied. Should it be immediately or during the maintenance window?
- Storage and instance type are decoupled. When scaling the DB instance type up or down, the storage size is not affected by the change. Likewise, a DB instance can be modified to increase the allocated storage space/type without changing the instance class.

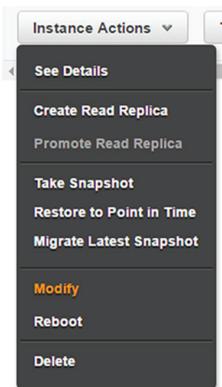
There is minimal downtime when scaling up on a Multi-AZ environment because the standby database gets upgraded first, then a failover will occur to the newly sized database. A Single-AZ instance will be unavailable during the scaling operation.

Vertical Scaling

To handle a higher load in a database, vertically scale up a database with a click of a mouse.

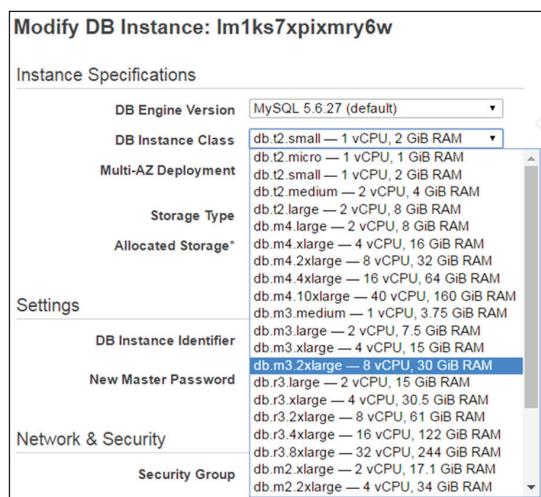
To change the instance type, choose Modify from the Instance Actions menu on the Amazon RDS console. Refer to Figure 7.3 for more detail.

FIGURE 7.3 The Instance Actions menu



Then choose the new DB instance class. Figure 7.4 displays the DB Instance Class drop-down list.

FIGURE 7.4 DB Instance Class drop-down list



Finally, determine if the change is to be applied immediately. If it is, select the **Apply Immediately** check box at the bottom of the Modify page. Otherwise, the change will be scheduled to occur during the preferred maintenance window that was defined when the Amazon RDS DB instance was created.



NOTE There are currently more than 18 instance sizes to choose from when resizing an Amazon RDS MySQL, PostgreSQL, MariaDB, Oracle, or Microsoft SQL Server instance.

For Amazon Aurora, there are five memory-optimized instance sizes from which to choose.

Scaling Out

When the capacity of a single server is reached, additional servers can be added to create a distributed database. The complexity of relational databases causes problems with their potential to scale. In general, relational databases are aware of their schema but unaware of the nature of the stored data. That is, a relational database knows how to format address information, but it cannot distinguish what addresses are to be entered.

Horizontal Scaling

It is possible to improve the performance of a read-heavy database by using Read Replicas to scale a database horizontally.

Read Replicas are read-only copies synchronized with a master database. It is possible to place a Read Replica in a different AWS Region to be closer to users and improve performance.

Read Replicas can increase the availability of a database because they can be promoted to a primary DB instance in the event of a disaster.



WARNING Read Replicas are not a replacement for the high availability and automatic failover capabilities that Multi-AZ configurations provide.

Amazon RDS Read Replicas support transparent load balancing of queries and connections. Each replica has a unique DNS endpoint so that an application can implement load balancing by connecting to the replica endpoint.

Currently, Elastic Load Balancing load balancers do not support the routing of traffic to Amazon RDS instances. Therefore, consider other options such as HAProxy, an open source, software-based load balancer. Configure HAProxy to listen on one port for read queries and another port for write queries.

Another option is to use a layer 7 SQL-aware load balancer, such as MaxScale, ProxySQL, and MySQL Proxy, to forward queries to databases using complex rules. This type of load balancer has a sophisticated capability of understanding how to perform read/write splits properly.



Amazon RDS MySQL, PostgreSQL, and MariaDB can have up to five Read Replicas. Amazon Aurora can have up to 15 Aurora Replicas plus 5 MySQL Read Replicas.

Sharding

To scale out a relational database and distribute the load, it is necessary to create “horizontal partitions” of data called shards. *Shards* are separate instances of a database that contain a subset of the data for the entire database. Each shard acts as the single/sole source of the subset it contains. When creating shards, intelligence has to be added to the application to know which shard to access.

Relational databases are schema-aware. The built-in intelligence controls how the data is stored and its format; however, relational databases are not designed to be aware of what data is appropriate.

For example, if a database with customer data is split into a pair of shards with half of the customers in each shard, intelligence has to be added to the application to make it aware of the correct shard. Additionally, it must *always* use the same shard for the same customer.



A shard architecture is sometimes referred to as a “shared nothing” architecture, because no data is shared between shards. Each shard works independently of other shards and is self-sufficient.

When scaling a database to hundreds or thousands of servers, the complexities become overwhelming. The characteristics that make relational databases appealing are the same ones that reduce their viability as platforms for large distributed systems.

Tagging Amazon RDS Resources

It is possible to use tags to add metadata to Amazon RDS resources. These tags can be used with IAM policies to manage access to Amazon RDS resources and control what actions can be applied to them. In addition, these tags can be used to track costs by running reports on similarly tagged resources.

All Amazon RDS Resources can be Tagged:

- DB instances
- DB clusters
- Read Replicas
- DB snapshots
- DB cluster snapshots
- Reserved DB instances

- Event subscriptions
- DB option groups
- DB parameter groups
- DB cluster parameter groups
- DB security groups
- DB subnet groups

An *Amazon RDS tag* is a customer-defined name/value pair associated with an Amazon RDS resource. Customers can use tags to assign arbitrary information to an Amazon RDS resource. AWS recommends the use of a consistent set of tags for tracking, reporting, and billing.

Each Amazon RDS resource has a *tag set*, which contains all of the tags that are assigned to that Amazon RDS resource. A tag set can contain as many as 10 tags, or it can be empty. If a tag is added to an Amazon RDS resource that has the same key as an existing tag, the new value overwrites the old value.

The AWS Management Console, AWS CLI, or Amazon RDS API can be used to add, list, and delete tags on Amazon RDS resources. When using the AWS CLI or API, you must provide the Amazon Resource Name (ARN) for the Amazon RDS resource you want to work with.



It is possible to add, list, or remove tags using the AWS CLI.

- To add one or more tags to an Amazon RDS resource, use the AWS CLI command: `add-tags-to-resource`
- To list tags on an Amazon RDS resource, use the AWS CLI command: `list-tags-for-resource`
- To remove one or more tags from an Amazon RDS resource, use the AWS CLI command: `remove-tags-from-resource`

For example, to add a key named `location` with a value of `prod` to a DB instance named `my-rds` owned by customer `867530955512` in `US-East-1`, use the following command:

```
rds-add-tag-to-resource arn:aws:rds:us-east-1:867530955512:db:my-rds -tk location -tv prod
```



Tags are cached for authorization purposes. Because of this, additions and updates to tags on Amazon RDS resources may take several minutes before they are available.

Monitoring Amazon RDS

Monitoring is an important part of maintaining the reliability, availability, and performance of Amazon RDS. With Amazon RDS, it is possible to monitor network throughput; I/O for read, write, and/or metadata operations; client connections; and burst-credit balances for DB instances.

Create a baseline of metrics to know how a database should perform. When the performance of a database falls outside of this established baseline, you update the database infrastructure. This could mean changing the instance class of the database or adding Read Replicas.

In general, acceptable values for performance metrics depend on having an existing/established baseline for database and application activity. Investigate any activity that differs from this baseline.

High CPU or RAM consumption High values for CPU or RAM consumption might be appropriate, provided that they are within an expected range.

Disk space consumption Investigate disk space consumption if there is consistently less than 15 percent of available free space.

Network traffic Investigate network traffic if throughput is consistently lower than expected.

Database connections If an increase of usage causes performance issues, consider limiting the number of available database connections. The best number of user connections for a DB instance will depend on the instance class and the complexity of the operations performed.

IOPS metrics The expected values for IOPS metrics depend on disk specification and server configuration. Establish a baseline to know what utilization is typical. For best IOPS performance, make sure that the typical working set will fit into memory to minimize read and write operations.

Monitoring Tools

Amazon RDS provides Amazon CloudWatch metrics for database instances at no additional charge. The AWS Management Console can be used to view key operational metrics, including compute/memory/storage capacity utilization, I/O activity, and instance connections.

Amazon RDS also provides Enhanced Monitoring, providing access to more than 50 CPU, memory, file system, and disk I/O metrics.

Automated Monitoring Tools

Monitoring is an important part of maintaining the reliability, availability, and performance of Amazon RDS and related AWS solutions. Monitoring information should be

collected from all parts of a production environment in order to debug multi-point failures more easily.

AWS recommends that customers create a monitoring plan that includes answers to the following questions:

- What are the monitoring goals?
- What resources will be monitored?
- How often will these resources be monitored?
- What monitoring tools are available?
- Who will perform the monitoring tasks?
- Who should be notified when something goes wrong?

After these questions have been asked and answered, the next step is to establish a baseline for normal Amazon RDS performance in an environment. To do this, measure performance at various times and under different load conditions.

While monitoring Amazon RDS, consider storing historical monitoring data. This information will provide a baseline to compare with current performance data, identify normal performance patterns and performance anomalies, and assist in the development of methods to address issues.

In general, acceptable values for performance metrics depend on an established baseline.

AWS provides various tools to monitor Amazon RDS. Some of these tools can be configured to monitor an environment automatically. Others require manual intervention. In general, AWS recommends that monitoring tasks be automated as much as possible.

The following automated tools can be used to monitor Amazon RDS and report when something is wrong.

Amazon CloudWatch Alarms This tool monitors a single metric over a user-specified period of time and can be configured to perform one or more actions. There are three factors evaluated while monitoring: A baseline value of the metric, a threshold value above or below an established baseline, and a number of time periods the threshold has been exceeded.

Amazon CloudWatch Alarms trigger actions when the state of an alarm changes *and* that change is maintained over a specified number of time periods. A single event or breach of a threshold does not trigger an alarm.

Amazon CloudWatch Logs This feature monitors, stores, and accesses log files from AWS CloudTrail or other sources, including on-premises software. Amazon CloudWatch Logs is covered in more detail in Chapter 9, “Monitoring and Metrics.”

Amazon RDS Enhanced Monitoring This feature provides metrics in real time for the operating system running a DB instance or DB cluster.

Amazon CloudWatch Events This feature matches events and routes them to one or more target functions or streams to make changes, capture state information, and take corrective action.

AWS CloudTrail Log monitoring This feature shares log files between accounts, monitors AWS CloudTrail log files in real time by sending them to Amazon CloudWatch Logs, writes log processing applications in Java, and validates that log files have not changed after delivery by AWS CloudTrail.

Amazon RDS events Amazon RDS uses Amazon Simple Notification Service (Amazon SNS) to provide notifications when an Amazon RDS event occurs. These notifications can be in any notification format supported by Amazon SNS for an AWS Region. Example event notifications include changes made to a DB instance, DB cluster, DB snapshot, DB cluster snapshot, DB parameter group, or DB security group.

Database log files View, download, or watch database log files using the Amazon RDS console or Amazon RDS APIs. It is also possible to query some database log files that are loaded into database tables.

Amazon CloudWatch Amazon RDS Metrics

Amazon RDS sends metrics to Amazon CloudWatch for each active database instance every minute. Detailed monitoring is enabled by default.

BinLogDiskUsage: Amount of disk space occupied by binary logs on the primary database. Applies to MySQL Read Replicas and is measured in bytes

CPUUtilization: Percentage of CPU utilization

DatabaseConnections: Number of database connections in use

DiskQueueDepth: The number of outstanding read and write requests waiting to access the disk

FreeStorageSpace: Amount of available storage space

FreeableMemory: Amount of available RAM

NetworkReceiveThroughput: Inbound network traffic on the DB instance that includes both customer database traffic and Amazon RDS traffic used for monitoring and replication

NetworkTransmitThroughput: Outbound network traffic on the DB instance that includes both customer database traffic and Amazon RDS traffic used for monitoring and replication

ReadIOPS: Average number of disk I/O operations per second

ReadLatency: Average amount of time taken per disk I/O operation

ReadThroughput: Average number of bytes read from disk per second

ReplicaLag: Amount of time a Read Replica DB instance lags behind the source DB instance. Applies to MySQL, MariaDB, and PostgreSQL Read Replicas

SwapUsage: Amount of swap space used on the DB instance

WriteIOPS: Average number of disk I/O operations per second

WriteLatency: Average amount of time taken per disk I/O operation

WriteThroughput: Average number of bytes written to disk per second



There are a number of commonly used Amazon RDS metrics that can appear on the exam. Pay close attention to these Amazon CloudWatch metrics: ReplicaLag, DiskQueueDepth, FreeStorageSpace, ReadIOPS, WriteIOPS, ReadLatency, and WriteLatency.

Manual Monitoring Tools

Another important part of monitoring Amazon RDS involves manually monitoring those items that the Amazon CloudWatch Alarms don't cover. The Amazon RDS, Amazon CloudWatch, AWS Trusted Advisor, and other AWS console dashboards provide an at-a-glance view of the state of an AWS environment.

From the Amazon RDS Console:

- The number of connections to a DB instance
- The amount of read and write operations to a DB instance
- The amount of storage that a DB instance is currently utilizing
- The amount of memory and CPU being utilized for a DB instance
- The amount of network traffic to and from a DB instance

From the AWS Trusted Advisor Dashboard:

- Amazon RDS idle DB instances
- Amazon RDS security group access risk
- Amazon RDS backups
- Amazon RDS Multi-AZ
- Amazon Aurora DB instance accessibility

From the Amazon CloudWatch Console:

- Current alarms and status
- Graphs of alarms and resources
- Service health status
- Create customized dashboards to monitor the services you care about.
- Graph metric data to troubleshoot issues and discover trends.
- Search and browse all of your AWS resource metrics.
- Create and edit alarms to be notified of problems.

Amazon RDS Metrics and Dimensions

Amazon RDS sends metrics to Amazon CloudWatch for each active database instance every minute.



In the free tier, basic monitoring for services such as Amazon EC2 happens at five-minute intervals. For Amazon RDS, detailed monitoring is enabled by default. This provides additional metrics as well as a faster interval. Detailed monitoring metrics are collected in one-minute intervals. The AWS Free Tier applies to participating services across the following AWS Regions: US East (Northern Virginia), US West (Oregon), US West (Northern California), Canada (Central), EU (London), EU (Ireland), EU (Frankfurt), Asia Pacific (Singapore), Asia Pacific (Tokyo), Asia Pacific (Sydney), and South America (Sao Paulo).

Amazon RDS Pricing

When using Amazon RDS, customers pay only for what is used and there are no minimum or setup fees.

Bills are generated using the criteria discussed in the following sections.

Instance Class

Pricing is based on the class (for example, micro, small, large, and xlarge) of the DB instance.

Running Time

Running time is calculated by the instance hour, which is equivalent to a single instance running for an hour. For example, both a single instance running for two hours and two instances running for one hour consume two instance hours.

If a DB instance runs for only part of an hour, the charge is for a full instance hour.

Storage

The storage capacity that is provisioned for a DB instance is billed per GB per month. If provisioned storage capacity is scaled within the month, the bill will be prorated.

I/O Requests per Month

This shows the total number of storage I/O requests that were made in a billing cycle.

Backup Storage

Backup storage is that which is associated with automated database backups and any active database snapshots that have been taken. Increasing the backup retention period or taking additional database snapshots increases the backup storage consumed by a database.

Amazon RDS provides backup storage up to 100 percent of provisioned database storage at no additional charge.

In general, most databases require less raw storage for a backup than for the primary dataset. If customers have only one backup set, it should not incur costs. If the backups are larger in size than the DB instance, the additional storage will be charged.



Backup storage is available at no additional charge for active DB instances.

Data Transfer

This shows the Internet data transfer in and out of your DB instance.



In addition to regular Amazon RDS pricing, reserved DB instances are also available for purchase. Reserved DB instances let customers make a one-time, upfront payment for a DB instance and reserve the DB instance for a one-year or three-year term at significantly lower rates.

Non-Relational Databases

Whereas relational databases use tables with rows and columns that have well-defined relationships, non-relational databases use a variety of data models, including document, graph, key/value, and columnar. Non-relational databases typically do not enforce a schema. Instead, a partition key is used to retrieve values, column sets, JSON, XML, or other information containing related-item attributes.



Non-relational databases are often referred to as NoSQL databases. Even so, it is possible to use SQL or SQL-like commands with some non-relational databases. Because of this, some people consider the term "NoSQL" to mean "Not only SQL."

NoSQL databases are widely recognized for ease of development, scalable performance, high availability, and resilience. These features come with a price, however. NoSQL databases often trade some ACID properties of traditional relational database management systems for a more flexible data model that efficiently scales horizontally. This makes NoSQL databases an excellent choice in situations where traditional RDBMSs encounter architectural challenges to overcome some combination of performance bottlenecks, scalability, operational complexity, and increasing costs of administration and support.

Many NoSQL database systems compromise consistency in favor of availability, partition tolerance, and speed. Instead, they offer a concept of “eventual consistency,” in which database changes are propagated to all nodes “eventually.” Typically, this propagation occurs within milliseconds. Queries for data might not return updated data immediately or might result in reading data that is not accurate—a problem known as *stale reads*.

Amazon DynamoDB

Amazon DynamoDB is a fast and flexible NoSQL database service for applications that need consistent, single-digit millisecond latency at any scale. It is a fully-managed NoSQL database supporting both document and key/value store models, and it is ideal for mobile, web, gaming, ad tech, and Internet of Things (IoT) applications.

Amazon DynamoDB Core Components

In Amazon DynamoDB, tables, items, and attributes are the core components with which you work. A table is a collection of items, and each item is a collection of attributes. Amazon DynamoDB uses primary keys to identify each item in a table uniquely and secondary indexes to provide flexibility with queries.

Tables, Items, and Attributes

The basic components of Amazon DynamoDB are tables, items, and attributes. A table is a collection of items and each item is a collection of attributes. DynamoDB uses primary keys to identify each item in a table uniquely and secondary indexes to provide querying flexibility.

Tables

Similar to other database systems, Amazon DynamoDB stores data in tables. A *table* is a collection of data. When creating a table in Amazon DynamoDB, it must have a table name, a primary key, and the required read and write throughput values.

The Naming Rules for Amazon DynamoDB Tables are as Follows:

- All names must be encoded using UTF-8 and be case sensitive.
- Table names and index names must be between 3 and 255 characters long and can contain only the following characters:
 - a-z
 - A-Z
 - 0-9
 - _ (underscore)
 - - (dash)
 - . (dot)
- Attribute names must be between 1 and 255 characters long.



Amazon DynamoDB has three default settings:

- No secondary indexes
- Provisioned capacity set to five reads and five writes
- Basic alarms with 80 percent upper threshold using Amazon SNS topic “dynamodb”

Items

Each table contains multiple items. An *item* is a group of attributes uniquely identifiable among all of the other items. Items in Amazon DynamoDB are similar in many ways to rows, records, or tuples in relational database systems.

Attributes

Each item is composed of one or more attributes. An *attribute* is a fundamental data element—something that does not need to be broken down any further. Attributes in Amazon DynamoDB are similar to fields or columns in relational database systems.

Streams

An Amazon *DynamoDB Stream* is an ordered flow of information about changes to items in an Amazon DynamoDB table. When you enable a stream on a table, DynamoDB captures information about every modification to data items in the table.

Whenever an application creates, updates, or deletes items in the table, Amazon DynamoDB Stream writes a stream record with the primary key attribute(s) of the items that were modified. A *stream record* contains information about a data modification to a single item in a DynamoDB table. You can configure the stream so that the stream records capture additional information, such as the "before" and "after" images of modified items. DynamoDB streams can be used to copy data from one table to another within the same region.

Data Types

Amazon DynamoDB supports different data types for attributes within a table.

Scalar

A *scalar type* can represent exactly one value. The scalar types described below are Number, String, Binary, Boolean, and Null.

NUMBER

Numbers can be positive, negative, or zero. Numbers can have up to 38 digits of precision. Exceeding this limit will result in an exception.

In Amazon DynamoDB, numbers are represented as variable length. Leading and trailing zeros are trimmed.

All numbers are sent to Amazon DynamoDB as strings to maximize compatibility across languages and libraries. Amazon DynamoDB still treats them as number type attributes for mathematical operations.



If number precision is important, pass numbers to Amazon DynamoDB using strings that converted from number type.

The number data type can be used to represent a date or a timestamp. One way to do this is by using epoch time—the number of seconds since 00:00:00 UTC on January 1, 1970.

STRING

Strings are Unicode with UTF-8 binary encoding. The length of a string must be greater than zero and is constrained by the maximum Amazon DynamoDB item size limit of 400 KB.

If a primary key attribute is a string type, the following additional constraints apply:

- For a simple primary key, the maximum length of the first attribute value (the partition key) is 2,048 bytes.
- For a composite primary key, the maximum length of the second attribute value (the sort key) is 1,024 bytes.
- Amazon DynamoDB collates and compares strings using the bytes of the underlying UTF-8 string encoding. For example, “a” (0x61) is greater than “A” (0x41), and “ξ” (0xC2BF) is greater than “z” (0x7A).

The string data type can be used to represent a date or a timestamp. One way to do this is by using ISO 8601 strings, as shown here:

2017-09-28

2017-07-19T12:34:56Z

20161221T080000Z

BINARY

Binary type attributes can store any binary data, such as compressed text, encrypted data, or images. Whenever Amazon DynamoDB compares binary values, it treats each byte of the binary data as unsigned.

The length of a binary attribute must be greater than zero and is constrained by the maximum Amazon DynamoDB item size limit of 400 KB.

If a primary key attribute is a binary type, the following constraints apply:

- For a simple primary key, the maximum length of the first attribute value (the partition key) is 2,048 bytes.
- For a composite primary key, the maximum length of the second attribute value (the sort key) is 1,024 bytes.

Applications must encode binary values in base64-encoded format before sending them to Amazon DynamoDB. Upon receipt, Amazon DynamoDB decodes the data into an unsigned byte array and uses it as the length of the binary attribute.

BOOLEAN

A Boolean type attribute can store one of two values: true or false.

NULL

Null represents an attribute with an unknown or undefined state.

Document

There are two document types, list and map, that can be nested within each other to represent complex data structures up to 32 levels deep.

There is no limit on the number of values in a list or a map as long as the item containing the values fits within the Amazon DynamoDB item size limit of 400 KB.

An attribute value cannot be an empty string or an empty set; however, empty lists and maps are allowed.

The List Type

A *list type attribute* can store an ordered collection of values. Lists are enclosed in square brackets: [...].

A list is similar to a JSON array. There are no restrictions on the data types that can be stored in a list element, and the elements in a list element can be different types.

Here is an example of a list with strings and numbers:

```
MyFavoriteThings: ["Raindrops on Roses", "Whiskers on Kittens", 1986, 4]
```



It is possible to work with individual elements within lists, even if those elements are deeply nested.

The Map Type

A *map type attribute* can store an unordered collection of name/value pairs. Maps are enclosed in curly braces: { ... }.

A map is similar to a JSON object. There are no restrictions on the data types that can be stored in a map element, and elements in a map do not have to be the same type.

Maps are ideal for storing JSON documents in Amazon DynamoDB. The following example shows a map that contains a string, a number, and a nested list that contains another map.

```
{  
    Location: "Labyrinth",  
    MagicStaff: 1,  
    MagicRings: [  
        "The One Ring",  
        {  
            ElevenKings: { Quantity : 3},  
            DwarfLords: { Quantity : 7},  
            MortalMen: { Quantity : 9}  
        }  
    ]  
}
```



Amazon DynamoDB lets you work with individual elements within maps, even if those elements are deeply nested.

Set

Amazon DynamoDB supports types that represent sets of number, string, or binary values. All of the elements within a set must be of the same type.

There is no limit on the number of values in a set, as long as the item containing the values fits within the Amazon DynamoDB 400 KB item size limit.

Each value within a set must be unique. The order of the values within a set are not preserved. Applications must not rely on the order of elements within the set.



Amazon DynamoDB does not support empty sets.

Provisioned Throughput

When creating a table, specify how much provisioned throughput capacity is required to reserve for reads and writes. Amazon DynamoDB will reserve the necessary resources to meet throughput needs while ensuring consistent, low-latency performance. It is possible to change provisioned throughput settings, increasing or decreasing capacity as needed.

Capacity Units

In Amazon DynamoDB, provisioned throughput requirements are calculated in terms of capacity units.

Read Capacity Units

One read capacity unit represents one strongly consistent read per second or two eventually consistent reads per second for items up to 4 KB in size. For items larger than 4 KB, Amazon DynamoDB will consume additional read capacity units. The total number of read capacity units required depends on the item size and if the required read type is eventually consistent or strongly consistent.

If items are smaller than 4 KB in size, each read capacity unit will yield one strongly consistent read per second or two eventually consistent reads per second.

Consider a database that needs to read 80 items per second from a table with items 3 KB in size and that also needs to have strongly consistent reads. To determine the provisioned read capacity unit requirement, start by dividing the item size by 4 KB, and then round up to the nearest whole number.

$$3 \text{ KB} / 4 \text{ KB} = 0.75 \rightarrow 1$$

Then multiply this number by the number of reads required per second. In this example, that is 80.

$$1 \text{ Read Capacity Unit per item} \times 80 \text{ reads per second} = 80 \text{ Read Capacity Units}$$

If eventually consistent reads are acceptable, only 40 read capacity units would be required.

Consider a database that needs to perform 100 strongly consistent reads per second with items 6 KB in size. First, determine the number of read capacity units required per item and round up to the nearest whole number.

$$6 \text{ KB} / 4 \text{ KB} = 1.5 \rightarrow 2$$

Because Amazon DynamoDB reserves read capacity in 4 KB blocks, 8 KB are provisioned to contain 6 KB of data. The result is two read capacity units per item. Next, multiply this by the number of strongly consistent reads per second:

$$2 \text{ Read Capacity Units per item} \times 100 \text{ reads per second} = 200$$

Set the provisioned read capacity units to 200.

For eventually consistent reads, provision 100 read capacity units.



To read a single item, use the `GetItem` operation. To read multiple items, use the `BatchGetItem` operation to retrieve up to 100 items from a table.

Use the `Query` and `Scan` operations to retrieve multiple consecutive items from a table or an index in a single request. With these operations, Amazon DynamoDB uses the cumulative size of the processed items to calculate provisioned throughput.

For example, if a `Query` operation retrieves 100 items that are 1 KB each, the read capacity calculation is *not* $(100 \times 4 \text{ KB}) = 100$ read capacity units, as if those items had been retrieved individually. Instead, the total would be only 25 read capacity units, as shown here:

$$(100 * 1024 \text{ bytes} = 100 \text{ KB}) / 4 \text{ KB} = 25 \text{ Read Capacity Units}$$

Write Capacity Units

One write capacity unit represents one write per second for items up to 1 KB in size. For items larger than 1 KB, Amazon DynamoDB will consume additional write capacity units. The total number of write capacity units required depends on the item size.

If items are smaller than 1 KB in size, each write capacity unit will yield one write per second. To write 100 items per second to a table with items 512 bytes in size, each write would require one provisioned write capacity unit.

To determine this, divide the item size of the operation by 1 KB and then round up to the nearest whole number:

$$512 \text{ bytes} / 1 \text{ KB} = 0.5 \rightarrow 1$$

Set the provisioned write throughput to 100 write capacity units:

$$1 \text{ Write Capacity Unit per item} \times 100 \text{ writes per second} = 100 \text{ Write Capacity Units}$$

If items are larger than 1 KB in size, round up the item size. This will increase the size of the write capacity unit to the next 1 KB boundary.

Consider a database that needs to perform 10 writes per second with items 1.5 KB in size. First, determine the number of write capacity units required per item, rounding up to the nearest whole number:

$$1.5 \text{ KB} / 1 \text{ KB} = 1.5 \rightarrow 2$$

The result is two write capacity units per item. Next, multiply that value by the number of writes per second:

$$2 \text{ Write Capacity Units per item} \times 10 \text{ writes per second} = 20 \text{ Write Capacity Units}$$

Set the table's provisioned write capacity units to 20.

To work with a single item, use the `PutItem`, `UpdateItem`, or `DeleteItem` operation as appropriate. Use `BatchWriteItem` to put or delete up to 25 items in a single operation.

Reserved Capacity

AWS allows customers to purchase reserved capacity for Amazon DynamoDB. With reserved capacity, customers pay a one-time upfront fee and commit to a minimum usage level over a period of time. By reserving read and write capacity units ahead of time, it is possible to realize significant cost savings compared to on-demand provisioned throughput settings.

Secondary Indexes

For tables with secondary indexes, Amazon DynamoDB will consume additional capacity units. To add a single 1 KB item to a table with an indexed attribute, two write capacity units are required: one for writing to the table and another for writing to the index.

Read and Write Consistency

When an application writes data to an Amazon DynamoDB table and receives an HTTP 200 response (OK), all copies of the data are updated. The data will eventually be consistent across all storage locations. This is usually within one second or less.

Amazon DynamoDB supports eventually consistent and strongly consistent reads.

Eventually Consistent Reads

When data is read from an Amazon DynamoDB table, the response might not reflect the results of a recently completed write operation. The response might include some stale data.

Strongly Consistent Reads

When a strongly consistent read is requested, Amazon DynamoDB returns a response with the most up-to-date data that reflects the updates from all prior write operations that were successful.



Unless specified otherwise, DynamoDB uses eventually consistent reads as the default.

If an application's read or write requests exceed the provisioned throughput for a table, Amazon DynamoDB might throttle that request. When this happens, the request fails with an HTTP 400 code (Bad Request), accompanied by a `ProvisionedThroughputExceededException`.

The AWS SDKs have built-in support for retrying throttled requests. Even so, consider using exponential backoff logic in error handling code.

The AWS Management Console can be used to monitor provisioned and actual throughput to determine whether requests are being throttled. If requests are throttled, change the provisioned capacity settings.

Metadata Tags and Amazon DynamoDB

Amazon DynamoDB resources can be labeled with metadata using tags. Tags allow for the cataloging of resources in an environment.

Among the Many Uses of Tags, They can Help with the Following:

- Resource identification
- Billing and cost allocation reports
- Ownership
- Business purpose identification
- Operating environment identification

Tagging Restrictions

Each tag consists of a key and a value, both of which are customer-defined.

The following restrictions apply:

- Key names can only be used once per Amazon DynamoDB table. When adding a key that already exists, the existing tag will be updated to the new value.
- Tag keys and values are case sensitive.
- Maximum key length is 128 Unicode characters.
- Maximum value length is 256 Unicode characters.
- Allowed characters are letters, whitespace, and numbers, plus the following special characters: + - = . _ : /
- The maximum number of tags per resource is 50.
- AWS-assigned tag names and values are automatically assigned the “aws:” prefix. AWS-assigned tag names do not count toward the tag limit of 50. This prefix is reserved by AWS and cannot be used by customers.
- In cost allocation reports, user-assigned tag names have the prefix: “user:”.
- Resources cannot be tagged at creation. This is a separate action that can be performed only after the resource has been generated.
- Tags cannot be backdated.

Efficient tagging can provide cost insights by allowing you to create reports across services that carry specific tags.

Amazon VPC Endpoints for Amazon DynamoDB

It is possible to access Amazon DynamoDB from an Amazon VPC using *VPC endpoints*. Using VPC endpoints, you can configure network traffic to be limited to the AWS Cloud and avoid using the public Internet.

Benefits of using VPC endpoints:

- With VPC endpoints for Amazon DynamoDB, there is no need for an Internet or NAT. This keeps VPCs isolated from the public Internet.
- VPC endpoints offer simplified network configuration and remove the need to create and maintain firewalls to keep a VPC secure from network attacks.
- IAM policies can be used to restrict Amazon DynamoDB access through VPC endpoints.
- There is no additional cost for using a VPC endpoint.

A VPC endpoint is an entry in a VPC routing table that directs traffic to a specific AWS resource. (As of this writing, the available resources that use VPC endpoints are Amazon S3 and Amazon DynamoDB.) With a VPC endpoint configured to access Amazon DynamoDB, it will permit direct access to Amazon DynamoDB in the same region without traversing the public Internet.

All addresses used in IAM policies will be private IPs. Since there is no need for a NAT instance/gateway, a VPC endpoint provides reliable, high-throughput access to Amazon DynamoDB.

Amazon Redshift

Amazon Redshift is a fast, easy-to-use, fully managed data warehousing solution. It automates infrastructure provisioning and administrative tasks, such as backups, replication, and patching. It integrates seamlessly with third-party BI and Extract, Transform, Load (ETL) tools to generate reports in minutes. There is no limit to the amount of data that can be loaded and analyzed.

Amazon Redshift also includes Amazon Redshift Spectrum. This service runs SQL queries directly against exabytes of unstructured data in Amazon S3. No loading or transformation is required, and it uses open-data formats. Amazon Redshift Spectrum automatically scales query compute capacity based on the data being retrieved, so queries against Amazon S3 run quickly regardless of dataset size.

Data warehouses store data efficiently to minimize I/O and have fast query results. In addition to this, warehouses can serve concurrent users numbering in the hundreds to thousands.

A data warehouse functions as a central repository of information coming from one or more data sources. Data flows in from transactional systems and other relational databases and can include structured, semi-structured, and unstructured data. This data is processed, transformed, and ingested at regular intervals. Data scientists, business analysts, and decision-makers access the processed data in the data warehouse through Business Intelligence (BI) tools, SQL clients, and spreadsheets.

Cluster Management

An *Amazon Redshift cluster* is a set of nodes that consists of a leader node and one or more compute nodes. The type and number of compute nodes needed depends on the size of the data, the number of queries executed, and the required query execution performance.

Leader Nodes

A *leader node* receives queries from client applications, parses the queries, and develops execution plans. These plans are an ordered set of steps to process these queries. The leader node coordinates the parallel execution of these plans with the compute nodes, aggregates the intermediate results from these nodes, and returns the results to the client applications.

Compute Nodes

Compute nodes execute the steps specified in the execution plans and transmit data among themselves to serve these queries. The intermediate results are sent back to the leader node for aggregation before being sent to the client applications.

High Availability Considerations

As of this writing, Amazon Redshift only supports Single-AZ deployments. To run in multiple Availability Zones, create clusters in separate Availability Zones and load data from the same Amazon S3 source.

With Amazon Redshift Spectrum, spin up multiple clusters across Availability Zones and access data in Amazon S3 without having to load it into a cluster.

It is also possible to restore a data warehouse cluster to a different Availability Zone from your data warehouse cluster snapshots.

Cluster Access and Security

There are several features related to cluster access and security in Amazon Redshift. These features help control access to a cluster, define connectivity rules, and encrypt data and connections. These features are in addition to ones related to database access and security in Amazon Redshift.

AWS Accounts and IAM Credentials

By default, an Amazon Redshift cluster is only accessible to the AWS account that creates the cluster. The cluster is locked down so that no one else has access. Within an AWS account, use the IAM service to create user accounts and manage permissions for those accounts to control cluster operations.

Security Groups

By default, clusters are closed to everyone. IAM credentials only control access to the Amazon Redshift API-related resources. These resources include the Amazon Redshift

console, AWS CLI, API, and SDK. To enable access to the cluster from SQL client tools via JDBC or ODBC, use security groups.

- To use the Amazon EC2-Classic platform to access an Amazon Redshift cluster, Amazon Redshift security groups are required.
- To use the Amazon EC2-VPC platform to access an Amazon Redshift cluster, Amazon VPC security groups are required.

In both cases, rules are added to the security group to grant explicit inbound access to a specific range of Classless Inter-Domain Routing (CIDR)/IP addresses or to an Amazon EC2 security group if the SQL client runs on an Amazon EC2 instance.



In addition to the inbound access rules, create database users for credentials to authenticate to the database within the cluster.

Encryption

When provisioning a cluster, there is an option to encrypt the cluster for additional security. When encryption is enabled, Amazon Redshift stores all data in user-created tables in an encrypted format. Use either AWS KMS or a Hardware Security Module (HSM) to manage Amazon Redshift encryption keys.

Encryption is an immutable property of the cluster. The only way to switch from an encrypted cluster to a non-encrypted cluster is to unload the data and reload it into a new cluster.

Encryption applies to the cluster and any backups. When restoring a cluster from an encrypted snapshot, the new cluster is encrypted as well.

SSL Connections

Use SSL encryption to encrypt the connection between a SQL client and your cluster.

Databases

Amazon Redshift creates one database when a cluster has been provisioned. This is the database used to load data and run queries. Additional databases can be created, as needed, by running a SQL command.

When provisioning a cluster, specify a master user with access to all the databases within the cluster. This master user is a super user who is the one with initial access to the database. This super user can create other super user accounts as well as ones with minimal privileges.

Amazon Redshift uses parameter groups to define the behavior of all databases in a cluster. Parameter groups can define, but are not limited to, things like date presentation style and floating-point precision. If a parameter group is not specified during provisioning, Amazon Redshift will use a default parameter group.

Monitoring Clusters

There are several features related to monitoring in Amazon Redshift. It is possible to use database audit logging to generate activity logs, configure events and notification subscriptions to track information of interest, and use the metrics in Amazon Redshift and Amazon CloudWatch to learn about the health and performance of clusters and databases.

Database Audit Logging

Use the database audit logging feature to track information about authentication attempts, connections, disconnections, changes to database user definitions, and queries run in the database. This information is useful for security and troubleshooting purposes in Amazon Redshift.

Logs are stored in Amazon S3 buckets.

Events and Notifications

Amazon Redshift tracks events and retains information about them for a period of several weeks. For each event, Amazon Redshift reports information such as the date the event occurred, a description, the event source (a cluster, a parameter group, or a snapshot), and the source ID.

It is possible to create Amazon Redshift event notification subscriptions that specify a set of event filters. When an event occurs that matches the filter criteria, Amazon Redshift uses Amazon SNS to actively inform subscribers that the event has occurred.

Performance

Amazon Redshift provides performance metrics and data to track the health and performance of clusters and databases. Amazon Redshift uses Amazon CloudWatch metrics to monitor the physical aspects of the cluster, such as CPU utilization, latency, and throughput.

Amazon Redshift also provides query and load performance data to help monitor database activity in a cluster.

Billing

Customers pay only for what resources are used. There are no minimum or setup fees. Bills are based on the following factors described.

Compute Node Hours

Compute node hours are the total number of hours run across all compute nodes for the billing period. Bills are generated for one unit per node per hour. A three-node data warehouse cluster running persistently for an entire month would incur 2,160 instance hours. There is no charge for leader node hours; only compute nodes incur charges.

Backup Storage

Backup storage is the storage associated with automated and manual snapshots for a data warehouse. Increasing the backup retention period or taking additional snapshots increases the backup storage consumed by a data warehouse.

There is no additional charge for backup storage up to 100 percent of the provisioned storage for an active data warehouse cluster. Backup storage beyond the provisioned storage size of a data warehouse and backups stored after a cluster is terminated are billed at standard Amazon S3 rates.

Data Transfer

There is no Amazon Redshift specific data transfer charge for data transferred to or from Amazon Redshift outside of Amazon VPC. Data transfer to or from Amazon Redshift in Amazon VPC accrues standard AWS data transfer charges.

Data Scanned

With Amazon Redshift Spectrum, charges are generated based on the amount of Amazon S3 data scanned to execute a query. There are no charges for Amazon Redshift Spectrum when it is idle.

If data is stored in a columnar format, charges will go down because Amazon Redshift Spectrum only scans the columns needed by the query, not entire rows. Similarly, if data is compressed using one of Amazon Redshift Spectrum's supported formats, costs will also go down.

Charges are incurred based on standard Amazon S3 rates for data storage and Amazon Redshift instance rates for the cluster used.

Amazon Redshift vs. Amazon RDS

Both Amazon Redshift and Amazon RDS run traditional relational databases in the cloud while offloading database administration. Customers use Amazon RDS databases for both OLTP and reporting and analysis. Amazon Redshift harnesses the scale and resources of multiple nodes and uses a variety of optimizations to provide major improvements over traditional databases for analytic and reporting workloads against very large datasets.

Amazon Redshift provides an excellent scale-out option as data and query complexity grows or to prevent reporting and analytic processing from interfering with the performance of an OLTP workload.

Amazon ElastiCache

Amazon ElastiCache is a web service that makes it easy to set up, manage, and scale a distributed in-memory cache environment in the cloud. It provides a high-performance, scalable, and cost-effective caching solution, while removing the complexity associated with deploying and managing a distributed cache environment.

With Amazon ElastiCache, it is possible to deploy a cache environment quickly without having to provision hardware or install software. Choose from Memcached or Redis protocol-compliant cache engine software, and let Amazon ElastiCache perform software upgrades and patch management.

For enhanced security, Amazon ElastiCache can be run in the Amazon VPC environment, which provides complete control over network access to clusters.

Inside the AWS Management Console, resources such as nodes, clusters, or Read Replicas can be easily added to an Amazon ElastiCache environment to meet business needs and application requirements.

Existing applications that use Memcached or Redis can use Amazon ElastiCache with almost no modification. Applications need to know the hostnames and port numbers of the Amazon ElastiCache nodes.

Redis

Redis is a fast, open source, in-memory data store and cache. Both single-node and up to 15-shard clusters are available. These options enable scalability of up to 3.55 TB of in-memory data.

Amazon ElastiCache for Redis is fully managed, scalable, and secure. It has been designed around the needs of high-performance web, mobile, gaming, ad tech, and IoT applications.

Memcached

Memcached is a widely-adopted memory object caching system. Amazon ElastiCache is protocol compliant with Memcached, so popular tools used with existing Memcached environments will work seamlessly with Amazon ElastiCache.

The Amazon ElastiCache Auto Discovery feature for Memcached lets applications identify all of the nodes in a cache cluster and connect to them automatically. Applications are effectively insulated from changes to node membership in a cluster.

High Availability and Fault Tolerance

Amazon ElastiCache automatically detects and replaces failed nodes, reducing the overhead associated with self-managed infrastructures and provides a resilient system that mitigates the risk of overloaded databases that can slow website and application load times.

Reliability Enhancements for Critical Deployments Include:

- Automatic detection and recovery from cache node failures
- Multi-AZ with automatic failover of a failed primary cluster to a Read Replica in Redis clusters that support replication
- Flexible Availability Zone placement of nodes and clusters
- Integration with other AWS Cloud services such as Amazon EC2, Amazon CloudWatch, AWS CloudTrail, and Amazon SNS to provide a secure, high-performance, managed in-memory caching solution

Summary

In this chapter, you learned about the differences between relational databases and non-relational databases and how AWS provides them as a managed service.

The Amazon Relational Database Service consists of six database engines:

- MySQL
- MariaDB
- PostgreSQL
- Microsoft SQL Server
- Oracle
- Amazon Aurora

Other AWS database services covered were as follows:

Amazon Dynamo DB The NoSQL/non-relational database offering from AWS

Amazon Redshift The fully managed data warehouse service from AWS

Amazon ElastiCache A web service that makes it easy to set up, manage, and scale a distributed in-memory data store or cache in the cloud

Additionally, this chapter covered topics such as fault tolerance, high availability, security, monitoring, performance, and disaster recovery.

A great deal of material was covered in this chapter. Although the exam questions will not dive deep into all of the content presented in this chapter, understanding as much content as possible will help you determine the best answer for the questions in the exam.

Resources to Review

The following resources can help you learn more about databases on AWS:

AWS Database Blog <https://aws.amazon.com/blogs/database/>

Amazon RDS User Guide <http://docs.aws.amazon.com/AmazonRDS/latest/UserGuide/Welcome.html>

Monitoring Amazon RDS http://docs.aws.amazon.com/AmazonRDS/latest/UserGuide/CHAP_Monitoring.html

Best Practices for Amazon RDS http://docs.aws.amazon.com/AmazonRDS/latest/UserGuide/CHAP_BestPractices.html

Security in Amazon RDS <http://docs.aws.amazon.com/AmazonRDS/latest/UserGuide/UsingWithRDS.html>

Getting Started with Amazon DynamoDB <http://docs.aws.amazon.com/amazondynamodb/latest/gettingstartedguide/Welcome.html>

Authentication and Access Control for Amazon DynamoDB <http://docs.aws.amazon.com/amazondynamodb/latest/developerguide/authentication-and-access-control.html>

Best Practices for Amazon DynamoDB <http://docs.aws.amazon.com/amazondynamodb/latest/developerguide/BestPractices.html>

Amazon ElastiCache User Guide <http://docs.aws.amazon.com/AmazonElastiCache/latest/UserGuide/WhatIs.html>

AWS Database Blog: Scaling Your Amazon RDS Instance Vertically and Horizontally
<https://aws.amazon.com/blogs/database/scaling-your-amazon-rds-instance-vertically-and-horizontally/>

Exam Essentials

Understand what a DB instance is. A DB instance is the basic building block of an isolated database environment in the cloud. Each DB instance runs a DB engine. AWS currently supports MySQL, MariaDB, PostgreSQL, Oracle, Microsoft SQL Server, and Amazon Aurora DB engines. DB instances can be accessed using command-line tools, the Amazon RDS APIs, and the AWS Management Console.

Know what types of storage are available for DB instances. Storage comes in three types: magnetic, General Purpose (SSD), and Provisioned IOPS.

Understand the difference between horizontal and vertical scaling. Vertical scaling adds compute power. Horizontal scaling adds computing resources. Relational and non-relational databases scale differently.

Understand high availability and fault tolerance. High availability is the ability to recover after failure. Fault tolerance is about preventing the downtime from happening. To enable both, use Multi-AZ DB instances and Read Replicas. With Multi-AZ deployments, maintenance is performed on the standby instance before the primary instance. With Multi-AZ, Read Replicas are created from the standby instance to minimize the impact on the primary instance.

Know how security groups control access. A security group controls the access to a DB instance by allowing access from specific IP address ranges or Amazon EC2 instances.

Know what DB parameter groups are and what they do. A DB parameter group contains engine configuration values that can be applied to one or more DB instances of the same instance type.

Understand how DB option groups are different from DB parameter groups. Some DB engines offer tools that simplify managing the databases and making the best use of data. Amazon RDS makes such tools available through option groups (such as Oracle Application Express [APEX], Microsoft SQL Server Transparent Data Encryption, and MySQL Memcached support).

Test Taking Tip

While you prepare for the exam, test yourself often. Research shows that pre-testing as a type of study helps to improve learning.

“... testing proved itself to be superior to additional study, in a broad variety of academic topics, and the same is likely true of things like music and dance, practicing from memory. Now we’re beginning to understand that some kinds of tests improve later learning—even if we do poorly on them.”

Carey, Benedict. *How We Learn: The Surprising Truth About When, Where, and Why It Happens* (p. 101). Random House Publishing Group. Kindle Edition.

Exercises

By now you have set up an account in AWS. If you haven’t, now would be the time to do so. It is important to note that these exercises are in your AWS account and thus are not free.

Use the Free Tier when launching resources. The AWS Free Tier applies to participating services across the following AWS Regions: US East (Northern Virginia), US West (Oregon), US West (Northern California), Canada (Central), EU (London), EU (Ireland), EU (Frankfurt), Asia Pacific (Singapore), Asia Pacific (Tokyo), Asia Pacific (Sydney), and South America (Sao Paulo). For more information, see https://aws.amazon.com/sdm/optimization/server-side-test/free-tier/free_np/.

If you have not yet installed the AWS Command Line utilities, refer to Chapter 2, “Working with AWS Cloud Services,” Exercise 2.1 (Linux) or Exercise 2.2 (Windows). The reference for the AWS Command Line Interface can be found here: <http://docs.aws.amazon.com/cli/latest/reference/>

These exercises will walk you through some scenarios and will have you research the “how to” in order to complete the exercise. By looking up how to perform a specific task, you will be on your way to mastering the task. The goal of this book isn’t just to prepare to pass the AWS Certified SysOps Administrator - Associate exam; this book should serve as a reference companion in your day-to-day duties as an AWS Certified SysOps Administrator.

EXERCISE 7.1

Create a New Option Group Using the Console.

1. Sign in to the AWS Management Console, and open the Amazon RDS console.
2. In the navigation pane, choose Option Groups.
3. Choose Create Group.
4. In the Create Option Group dialog box, do the following:
 - a. For Name, type a name for the option group that is unique within your AWS account. The name can contain only letters, digits, and hyphens.

- b. For Description, type a brief description of the option group. The description is used for display purposes.
 - c. For Engine, choose the DB engine that you want.
 - d. For Major Engine Version, choose the major version of the DB engine that you want.
5. To continue, choose Yes, Create. To cancel the operation instead, choose Cancel.
-

EXERCISE 7.2

Create an Amazon DynamoDB Table from the AWS CLI.

This exercise assumes that the AWS CLI has been installed. For readability, long commands in this section are broken into separate lines. The backslash character lets you copy and paste (or type) multiple lines into a Linux terminal. For Windows PowerShell, the backtick (`) does the same thing.

This exercise will create an Amazon DynamoDB table called MusicCollection that has the attributes Artist and Title.

Type the following command into the terminal window with the AWS CLI installed:

For Linux, use the following:

```
aws dynamodb create-table \
--table-name MusicCollection \
--attribute-definitions \
  AttributeName=Artist,AttributeType=S AttributeName=Title,AttributeType=S \
--key-schema AttributeName=Artist,KeyType=HASH \
  AttributeName=Title,KeyType=RANGE \
--provisioned-throughput ReadCapacityUnits=1,WriteCapacityUnits=1
```

For Microsoft Windows, use the following:

```
aws dynamodb create-table ` \
--table-name MusicCollection ` \
--attribute-definitions ` \
  AttributeName=Artist,AttributeType=S AttributeName=Title,AttributeType=S ` \
--key-schema AttributeName=Artist,KeyType=HASH ` \
  AttributeName=Title,KeyType=RANGE ` \
--provisioned-throughput ReadCapacityUnits=1,WriteCapacityUnits=1
```

EXERCISE 7.3**Add Items to the Amazon DynamoDB Table MusicCollection Using the AWS CLI.**

Now that the table has been created, add data to it using the AWS CLI.

For Linux, use the following:

```
aws dynamodb put-item \
--table-name MusicCollection \
--item '{
  "Artist": {"S": "Mystical Father"}, 
  "Title": {"S": "Song for Elijah"}, 
  "AlbumTitle": {"S": "Lovely Laura Likes Llamas"} }' \
--return-consumed-capacity TOTAL

aws dynamodb put-item \
--table-name MusicCollection \
--item '{
  "Artist": {"S": "Steps in Springtime"}, 
  "Title": {"S": "Allemande Left but will Return"}, 
  "AlbumTitle": {"S": "Square Dances for Round Rooms"} }' \
--return-consumed-capacity TOTAL
```

For Microsoft Windows, use the following:

```
aws dynamodb put-item ` 
--table-name MusicCollection ` 
--item '{
  \"Artist\": {\"S\": \"Mystical Father\"}, 
  \"Title\": {\"S\": \"Song for Elijah\"}, 
  \"AlbumTitle\": {\"S\": \"Lovely Laura Likes Llamas\"} }' ` 
--return-consumed-capacity TOTAL

aws dynamodb put-item ` 
--table-name MusicCollection ` 
--item '{
  \"Artist\": {\"S\": \"Steps in Springtime\"}, 
  \"Title\": {\"S\": \"Allemande Left but will Return\"}, 
  \"AlbumTitle\": {\"S\": \"Square Dances for Round Rooms\"} }' ` 
--return-consumed-capacity TOTAL
```



Microsoft Windows eats double quotes ("") for breakfast. To hide them so that Amazon DynamoDB (or other services) can use them, use the backslash (\) as an escape character.

In the Amazon DynamoDB console, the table with items will look like the following graphic.

| Artist | Title | AlbumTitle |
|---------------------|--------------------------------|-------------------------------|
| Steps in Springtime | Allemande Left but will Return | Square Dances for Round Rooms |
| Mystical Father | Song for Elijah | Lovely Laura Likes Llamas |

EXERCISE 7.4

Create a MySQL Amazon RDS DB Instance.

The basic building block of Amazon RDS is the DB instance. This is the environment in which you will run your MySQL databases.

In this exercise, create a DB instance running the MySQL database engine called **west2-mysql-instance1**, with a db.m1.small DB instance class, 5 GB of storage, and automated backups enabled with a retention period of one day.

Create a MySQL DB Instance

1. Sign in to the AWS Management Console, and open the Amazon RDS console.
2. In the top-right corner of the Amazon RDS console, choose the region in which to create the DB instance.
3. In the navigation pane, choose Instances.
4. Choose Launch DB Instance. The Launch DB Instance wizard opens on the Select Engine page.
5. On the Select Engine page, choose the MySQL icon, and then choose Select for the MySQL DB engine.

EXERCISE 7.4 (continued)

6. On the Specify DB Details page, specify the DB instance information as shown here:

DB Details

| For this Parameter | Do This |
|--------------------------------------|--|
| License Model | Choose the default, general-public license to use the general license agreement for MySQL. MySQL has only one license model. |
| DB Engine Version | Choose the default version of MySQL. Note that Amazon RDS supports multiple versions of MySQL in some regions. |
| DB Instance Class | Choose db.m1.small for a configuration that equates to 1.7 GB memory, 1 ECU (1 virtual core with 1 ECU), 64-bit platform, and moderate I/O capacity. |
| Multi-AZ Deployment | Choose Yes to have a standby replica of the DB instance created in another Availability Zone for failover support. AWS recommends Multi-AZ for production workloads to maintain high availability. |
| Allocated Storage | Type 5 to allocate 5 GB of storage for the database. In some cases, allocating a higher amount of storage for your DB instance than the size of your database can improve I/O performance. |
| Storage Type | Choose the Magnetic storage type. |
| DB Instance Identifier | Type a name for the DB instance that is unique to the account in the region chosen. Feel free to add some intelligence to the name, such as including the region and DB engine chosen; for example, west2-mysql-instance1 . |
| Master Username | Type a name using alphanumeric characters to use as the master user name to log on to the DB instance. This will be the user name used to log on to the database on the DB instance for the first time. |
| Master Password and Confirm Password | Type a password that contains from 8 to 41 printable ASCII characters (excluding /, ", and @) for the master user password. This will be the password used for the user name to log on to the database. Type the password again in the Confirm Password box. |

7. When the settings have been configured, choose Next.
8. On the Configure Advanced Settings page, provide additional information that Amazon RDS needs to launch the MySQL DB instance. The following table shows settings for the Amazon DB instance in this exercise:

Configure Advanced Settings

| For This Parameter | Do This |
|----------------------------|---|
| VPC | Choose the name of the Amazon VPC that will host your MySQL DB instance. |
| Availability Zone | On the previous page, choosing Yes for the Multi-AZ Deployment parameter removes this option. Otherwise, select an AZ in which to deploy the database. |
| DB Security Groups | Choose the security group to use with this DB instance. |
| Database Name | Type a name for the default database that is between 1 to 64 alphanumeric characters long. Without a name, Amazon RDS will not automatically create a database on the newly provisioned DB instance. |
| Database Port | Leave the default value of 3306. |
| DB Parameter Group | Leave the default value. |
| Option Group | Choose the default value. This option group is used with the MySQL version chosen on the previous page. |
| Copy Tags To Snapshots | Choose this option to have any DB instance tags copied to a DB snapshot when creating a snapshot. |
| Enable Encryption | Choose Yes to enable encryption at rest for this DB instance. |
| Backup Retention Period | Set this value to 1. |
| Backup Window | Use the default of No Preference. |
| Enable Enhanced Monitoring | Use the default of No. |
| Auto Minor Version Upgrade | Choose Yes to enable the DB instance to receive minor DB engine version upgrades automatically when they become available. |
| Maintenance Window | Choose No Preference. |

EXERCISE 7.4 (*continued*)

9. Specify your DB instance information and then choose Launch DB Instance.

On the Amazon RDS console, the new DB instance appears in the list of DB instances. The DB instance will have a status of creating until the DB instance is created and ready for use. When the state changes to available, connect to the database created on the DB instance.

Depending on the DB instance class and store allocated, it could take several minutes for the new DB instance to become available.

Review Questions

1. How are charges calculated for data transferred between Amazon Relational Database Service (Amazon RDS) and Amazon Elastic Compute Cloud (Amazon EC2) instances in the same Availability Zone?
 - A. GB out from Amazon RDS
 - B. GB in from Amazon RDS
 - C. IOPS from Amazon EC2
 - D. There is no charge.
2. On an Amazon Elastic Compute Cloud (Amazon EC2) instance using an Amazon Elastic Block Store (Amazon EBS) volume for persistence, there is a large inconsistency in the response times of the database queries. While investigating the issue, you find that there is a large amount of wait time on the database's disk volume. How can you improve the performance of the database's storage while maintaining the current persistence of the data? (Choose two.)
 - A. Move to a Solid-State Drive (SSD)-backed instance.
 - B. Use an Amazon EC2 instance with an Instance Store volume.
 - C. Use an Amazon EC2 instance type that is Amazon EBS-Optimized.
 - D. Use a Provisioned IOPs Amazon EBS volume.
3. Amazon Relational Database Service (Amazon RDS) automated backups and DB snapshots are currently supported for which storage engine?
 - A. JDBC
 - B. ODBC
 - C. InnoDB
 - D. MyISAM
4. In Amazon CloudWatch, which metric monitors the amount of free space on a DB instance?
 - A. FreeStorageTotal
 - B. FreeStorageVolume
 - C. FreeStorageSpace
 - D. FreeRDSStorage
5. In an Amazon DynamoDB table, what happens if the application performs more reads or writes than the provisioned capacity?
 - A. Requests above the provisioned capacity will be performed, but it will return HTTP 400 error codes.
 - B. Requests above the provisioned capacity will be throttled, and it will return HTTP 400 error codes.
 - C. Requests above the provisioned capacity will be performed, but it will return HTTP 500 error codes.
 - D. Requests above the provisioned capacity will be throttled, and it will return HTTP 500 error codes.

6. Which metric should be monitored carefully in order to know when a Read Replica should be recreated if it becomes out of sync due to replication errors?
 - A. ReadLag
 - B. ReadReplica
 - C. ReplicaLag
 - D. ReplicaThreshold
7. Can connections between my application and a DB instance be encrypted using Secure Sockets Layer (SSL)?
 - A. Yes
 - B. No
 - C. Yes, but only inside an Amazon Virtual Private Cloud (Amazon VPC)
 - D. Yes, but only inside certain regions
8. After inadvertently deleting a database table, which Amazon Relational Database Service (Amazon RDS) feature will allow you to restore a database reliably to within five minutes of the deletion?
 - A. Amazon RDS Automated Backup
 - B. Multi-AZ Amazon RDS
 - C. Amazon RDS Read Replicas
 - D. Amazon RDS Snapshots
9. What is the range, in days, to which the backup retention period can be set?
 - A. From 1 day to 7 days
 - B. From 1 day to 28 days
 - C. From 1 day to 35 days
 - D. From 1 day to 90 days
10. Regarding Amazon DynamoDB, which of the following statements is correct?
 - A. Each item should have at least two value sets: a primary key and an attribute.
 - B. An item can have multiple attributes.
 - C. Primary keys must store a single value.
 - D. Individual attributes can have one or more sub-attributes.
11. The maximum Amazon DynamoDB item size limit is which of the following?
 - A. 256 KB
 - B. 400 KB
 - C. 512 KB
 - D. 1,024 KB

- 12.** Will a standby Amazon Relational Database Service (Amazon RDS) instance be in the same Availability Zone as the primary?
- A.** Yes
 - B.** No
 - C.** Only for Oracle and Microsoft SQL Server-based Amazon RDS instances
 - D.** Only if configured at launch
- 13.** For both Single and Multi-AZ deployments, defining a subnet for all Availability Zones in a region allows Amazon RDS to create a new standby in another Availability Zone should the need arise. Is creating a Read Replica of another Read Replica supported inside Amazon Relational Database Service (Amazon RDS)?
- A.** No
 - B.** Only in certain regions
 - C.** Only with MySQL-based Amazon RDS instances
 - D.** Only for Oracle Amazon RDS DB instances
- 14.** When running a DB instance as a Multi-AZ deployment, can the standby DB instance be used for read or write operations?
- A.** Yes
 - B.** No
 - C.** Only with Microsoft SQL Server-based Amazon Relational Database Service (Amazon RDS) DB instances
 - D.** Only for Oracle-based Amazon RDS DB instances
- 15.** If there are multiple Read Replicas for a primary DB instance and one of them is promoted, what happens to the rest of the Read Replicas?
- A.** The remaining Read Replicas will still replicate from the original primary DB instance.
 - B.** The other Read Replicas will be deleted.
 - C.** The remaining Read Replicas suspend operations.
 - D.** Read Replicas cannot be promoted.
- 16.** When using a Multi-AZ deployment, in the event of a planned or unplanned outage of the primary DB instance, Amazon RDS automatically switches to the standby replica. Which DNS record is updated by the automatic failover mechanism and points to the standby DB instance?
- A.** The A Record
 - B.** CNAME
 - C.** MX
 - D.** SOA

17. When automatic failover occurs, Amazon Relational Database Service (Amazon RDS) will create a DB instance event. Using the AWS CLI, which of the following will return information about the events related to the DB instance?
- A. ReturnEventFailure
 - B. DescribeFailureEvent
 - C. DescribeEvents
 - D. ReturnEvents
18. Is it possible to force a failover for a MySQL Multi-AZ DB instance deployment?
- A. Yes
 - B. No
 - C. Only in certain regions
 - D. Only in Amazon Virtual Private Cloud (Amazon VPC)
19. How does the Amazon Relational Database Service (Amazon RDS) Multi-AZ model work?
- A. A second, standby database is deployed and maintained in a different Availability Zone from the primary database, using synchronous replication.
 - B. A second, standby database is deployed and maintained in a different region from the primary database, using synchronous replication.
 - C. A second, standby database is deployed and maintained in a different Availability Zone from the primary database, using asynchronous replication.
 - D. A second, standby database is deployed and maintained in a different region from the primary database, using asynchronous replication.
20. What does Amazon ElastiCache provide?
- A. An Amazon Elastic Compute Cloud (Amazon EC2) instance with a large amount of memory and CPU
 - B. A managed in-memory cache service
 - C. An Amazon EC2 instance with Redis and Memcached preinstalled
 - D. A highly available and fast indexing service for searching
21. When developing a highly available web application using stateless web servers, which services are suitable for storing session-state data? (Choose three.)
- A. Amazon Simple Queue Service (Amazon SQS)
 - B. Amazon Relational Database Service (Amazon RDS)
 - C. Amazon CloudWatch
 - D. Amazon ElastiCache
 - E. Amazon DynamoDB
 - F. Amazon CloudFront

- 22.** Which statement best describes Amazon ElastiCache?
- A.** It reduces the latency of a DB instance by splitting the workload across multiple Availability Zones.
 - B.** It provides a simple web interface to create and store multiple datasets, query your data easily, and return the results.
 - C.** It is a managed service from AWS that makes it easy to set up, operate, and scale a relational database in the cloud.
 - D.** It offloads the read traffic from a database in order to reduce latency caused by read-heavy workload.
- 23.** Which two AWS Cloud services provide out-of-the-box, user-configurable, automatic backup-as-a-service and backup rotation options? (Choose two.)
- A.** AWS CloudFormation
 - B.** Amazon Simple Storage Service (Amazon S3)
 - C.** Amazon Relational Database Service (Amazon RDS)
 - D.** Amazon Elastic Block Store (Amazon EBS)
 - E.** Amazon DynamoDB
 - F.** Amazon Redshift



Chapter 8

Application Deployment and Management

**THE AWS CERTIFIED SYSOPS
ADMINISTRATOR - ASSOCIATE EXAM
TOPICS COVERED IN THIS CHAPTER MAY
INCLUDE, BUT ARE NOT LIMITED TO, THE
FOLLOWING:**

Domain 2.0 High Availability

- ✓ 2.1 Implement scalability and elasticity based on scenarios

Content may include the following:

- Which AWS compute service to use for deploying scalable and elastic environments
- Including scalability of specific AWS compute service in the deployment and management of applications
- Methods for implementing upgrades while maintaining high availability

- ✓ 2.2 Ensure level of fault tolerance based on business needs

Content may include the following:

- What AWS Cloud deployment services can be used for deploying fault-tolerant applications

Domain 4.0 Deployment and Provisioning

- ✓ 4.1 Demonstrate the ability to build the environment to conform to architectural design

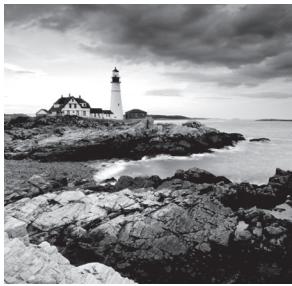
Content may include the following:

- Choosing the appropriate AWS Cloud service to meet requirements for deploying applications

- ✓ 4.2 Demonstrate the ability to provision cloud resources and manage implementation automation

Content may include the following:

- Automating the deployment and provisioning of AWS Cloud services



Introduction to Application Deployment and Management

As a candidate for the AWS Certified SysOps Administrator – Associate certification, you will need to be familiar with application deployment strategies and services used for the deployment and management of applications. AWS offers many capabilities for provisioning your infrastructure and deploying your applications. The deployment model varies from customer to customer depending on the capabilities required to support operations. Understanding these capabilities and techniques will help you pick the best strategy and toolset for deploying the infrastructure that can handle your workload.

An experienced systems operator is aware of the “one size doesn’t fit all” philosophy. For enterprise computing or to create the next big social media or gaming company, AWS provides multiple customization options to serve a broad range of use cases. The AWS platform is designed to address scalability, performance, security, and ease of deployment, as well as to provide tools to help migrate applications and an ecosystem of developers and architects that are deeply involved in the growth of its products and services.

This chapter details different deployment strategies and services. It reviews common features available on these deployment services, articulates strategies for updating application stacks, and presents examples of common usage patterns for various workloads.

Deployment Strategies

AWS offers several key features that are unique to each deployment service that will be discussed later in this chapter. There are some characteristics that are common to these services, however. This section discusses various common features and capabilities that a systems operator will need to understand to choose deployment strategies. Each feature can influence service adoption in its own way.

Provisioning Infrastructure

You can work with building-block services individually, such as provisioning an Amazon Elastic Compute Cloud (Amazon EC2) instance, Amazon Elastic Block Store (Amazon EBS)

volume, Amazon Simple Storage Service (Amazon S3) bucket, Amazon Virtual Private Cloud (Amazon VPC) environment, and so on. Alternately, you can use automation provided by deployment services to provision infrastructure components. The main advantage of using automated capabilities is the rich feature set that they offer for deploying and configuring your application and all the resources it requires. For example, you can use an AWS CloudFormation template to treat your infrastructure as code. The template describes all of the resources that will be provisioned and how they should be configured.

Deploying Applications

The AWS deployment services can also make it easier to deploy your application on the underlying infrastructure. You can create an application, specify the source repository to your desired deployment service, and let the deployment service handle the complexity of provisioning the AWS resources needed to run your application. Despite providing similar functionality in terms of deployment, each service has its own unique method for deploying and managing your application.

Configuration Management

Why does a systems operator need to consider configuration management? You may need to deploy resources quickly for a variety of reasons—upgrading your deployments, replacing failed resources, automatically scaling your infrastructure, etc. It is important for a systems operator to consider how the application deployment should be configured to respond to scaling events automatically.

In addition to deploying your application, the deployment services can customize and manage the application configuration. The underlying task could be replacing custom configuration files in your custom web application or updating packages that are required by your application. You can customize the software on your Amazon EC2 instance as well as the infrastructure resources in your stack configuration.

Systems operators need to track configurations and any changes made to the environments. When you need to implement configuration changes, the strategy you use will allow you to target the appropriate resources. Configuration management also enables you to have an automated and repeatable process for deployments.

Tagging

Another advantage of using deployment services is the automation of tag usage. A *tag* consists of a user-defined key and value. For example, you can define tags such as application, project, cost centers, department, purpose, and stack so that you can easily identify a resource. When you use tags during your deployment steps, the tools automatically propagate the tags to underlying resources such as Amazon EC2 instances, Auto Scaling groups, or Amazon Relational Database Service (Amazon RDS) instances.

Appropriate use of tagging can provide a better way to manage your budgets with cost allocation reports. Cost allocation reports aggregate costs based on tags. This way, you can determine how much you are spending for each application or a particular project.

Custom Variables

When you develop an application, you want to customize configuration values, such as database connection strings, security credentials, and other information, which you don't want to hardcode into your application. Defining variables can help loosely couple your application configuration and give you the flexibility to scale different tiers of your application independently. Embedding variables outside of your application code also helps improve portability of your application. Additionally, you can differentiate environments into development, test, and production based on customized variables. The deployment services facilitate customizing variables so that once they are set, the variables become available to your application environments. For example, an AWS CloudFormation template could contain a parameter that's used for your web-tier Amazon EC2 instance to connect to an Amazon RDS instance. This parameter is inserted into the user data script so that the application installed on the Amazon EC2 instance can connect to the database.

Baking Amazon Machine Images (AMIs)

An *Amazon Machine Image (AMI)* provides the information required to launch an instance. It contains the configuration information for instances, including the block device mapping for volumes and what snapshot will be used to create the volume. A *snapshot* is an image of the volume. The root volume would consist of the base operating system and anything else within the volume (such as additional applications) that you've installed.

In order to launch an Amazon EC2 instance, you need to choose which AMI you will use for your application. A common practice is to install an application on an instance at the first boot. This process is called *bootstrapping an instance*.



The bootstrapping process can be slower if you have a complex application or multiple applications to install. Managing a fleet of applications with several build tools and dependencies can be a challenging task during rollouts. Furthermore, your deployment service should be designed to perform faster rollouts to take advantage of Auto Scaling.

Baking an image is the process of creating your own AMI. Instead of using a bootstrap script to deploy your application, which could take an extended amount of time, this custom AMI could contain a portion of your application artifacts within it. However, during deployment of your instance, you can also use user data to customize application installations further. AMIs are regionally scoped—to use an image in another region, you will need to copy the image to all regions where it will be used.

The key factor to keep in mind is how long it takes for the instance to launch. If scripted installation of each instance takes an extended amount of time, this could impact your ability to scale quickly. Alternatively, copying the block of a volume where your application is already installed could be faster. The disadvantage is that if your application is changed, you'll need either to bake a new image, which can also be automated, or use a configuration management tool to apply the changes.

For example, let's say that you are managing an environment consisting of web, application, and database tiers. You can have logical grouping of your base AMIs that can take 80 percent of application binaries loaded on these AMI sets. You can choose to install the remaining applications during the bootstrapping process and alter the installation based on configuration sets grouped by instance tags, Auto Scaling groups, or other instance artifacts. You can set a tag on your resources to track for which tier of your environment they are used. When deploying an update, the process can query for the instance tag, validate whether it's the most current version of the application, and then proceed with the installation. When it's time to update the AMI, you can simply swap your existing AMI with the most recent version in the underlying deployment service and update the tag.

You can script the process of baking an AMI. In addition, there are multiple third-party tools for baking AMIs. Some well-known ones are Packer by HashiCorp and Aminator by Netflix. You can also choose third-party tools for your configuration management, such as Chef, Puppet, Salt, and Ansible.

Logging

Logging is an important element of your application deployment cycle. Logging can provide important debugging information or provide key characteristics of your application behavior. The deployment services make it simpler to access these logs through a combination of the AWS Management Console, AWS Command Line Interface (AWS CLI), and Application Programming Interface (API) methods so that you don't have to log in to Amazon EC2 instances to view them.

In addition to built-in features, the deployment services provide seamless integration with Amazon CloudWatch Logs to expand your ability to monitor the system, application, and custom log files. You can use Amazon CloudWatch Logs to monitor logs from Amazon EC2 instances in real time, monitor AWS CloudTrail events, or archive log data in Amazon S3 for future analysis.

Instance Profiles

Applications that run on an Amazon EC2 instance must include AWS credentials in their API requests. You could have your developers store AWS credentials directly within the Amazon EC2 instance and allow applications in that instance to use those credentials. But developers would then have to manage the credentials and ensure that they securely pass the credentials to each instance and update each Amazon EC2 instance when it's time to rotate the credentials. That's a lot of additional work. There is also the potential that the credentials could be compromised, copied from the Amazon EC2 instance, and used elsewhere.

Instead, you can and should use an AWS Identity and Access Management (IAM) role to manage temporary credentials for applications that run on an Amazon EC2 instance. When you use a role, you don't have to distribute long-term credentials to an Amazon EC2 instance. Instead, the role supplies temporary permissions that applications can use when they make calls to other AWS resources. When you launch an Amazon EC2 instance, you specify an IAM role to associate with the instance. Applications that run on the instance can then use the role-supplied temporary credentials to sign API requests.

Instance profiles are a great way of embedding necessary IAM roles that are required to carry out an operation to access an AWS resource. An instance profile is a container for an IAM role that you can use to pass role information to an Amazon EC2 instance when the instance starts. An instance profile can contain only one IAM role, although a role can be included in multiple instance profiles. These IAM roles can be used to make API requests securely from your instances to AWS Cloud services without requiring you to manage security credentials. The deployment services integrate seamlessly with instance profiles to simplify credentials management and relieve you from hardcoding API keys in your application configuration.

For example, if your application needs to access an Amazon S3 bucket with read-only permission, you can create an instance profile and assign read-only Amazon S3 access in the associated IAM role. The deployment service will take the complexity of passing these roles to Amazon EC2 instances so that your application can securely access AWS resources with the privileges that you define.

Scalability Capabilities

Scaling your application fleet automatically to handle periods of increased demand not only provides a better experience for your end users, but it also keeps the cost low. As demand decreases, resources can automatically be scaled in. Therefore, you're only paying for the resources needed based on the load.

For example, you can configure Auto Scaling to add or remove Amazon EC2 instances dynamically based on metrics triggers that you set within Amazon CloudWatch (such as CPU, memory, disk I/O, and network I/O). This type of Auto Scaling configuration is integrated seamlessly into AWS Elastic Beanstalk and AWS CloudFormation. Similarly, AWS OpsWorks and Amazon EC2 Container Services (Amazon ECS) have capabilities to manage scaling automatically based on time or load. Amazon ECS has Service Auto Scaling that uses a combination of the Amazon ECS, Amazon CloudWatch, and Application Auto Scaling APIs to scale application containers automatically.

Monitoring Resources

Monitoring gives you visibility into the resources you launch in the cloud. Whether you want to monitor the resource utilization of your overall stack or get an overview of your application health, the deployment services are integrated with monitoring capabilities to provide this info within your dashboards. You can navigate to the Amazon CloudWatch console to get a system-wide view into all of your resources and operational health. Alarms can be created for metrics that you want to monitor. When the threshold is surpassed, the alarm is triggered and can send an alert message or take an action to mitigate an issue. For example, you can set an alarm that sends an email alert when an Amazon EC2 instance fails on status checks or trigger a scaling event when the CPU utilization meets a certain threshold.

Each deployment service provides the progress of your deployment. You can track the resources that are being created or removed via the AWS Management Console, AWS CLI, or APIs.

Continuous Deployment

This section introduces various deployment methods, operations principles, and strategies a systems operator can use to automate integration, testing, and deployment.

Depending on your choice of deployment service, the strategy for updating your application code could vary a fair amount. AWS deployment services bring agility and improve the speed of your application deployment cycle, but using a proper tool and the right strategy is key for building a robust environment.

The following section looks at how the deployment service can help while performing application updates. Like any deployment lifecycle, the methods you use have trade-offs and considerations, so the method you implement will need to meet the specific requirements of a given deployment.

Deployment Methods

There are two primary methods that you can use with deployment services to update your application stack: *in-place upgrade* and *replacement upgrade*. An *in-place upgrade* involves performing application updates on existing Amazon EC2 instances. A *replacement upgrade*, however, involves provisioning new Amazon EC2 instances, redirecting traffic to the new resources, and terminating older instances.

An *in-place upgrade* is typically useful in a rapid deployment with a consistent rollout schedule. It is designed for stateless applications. You can still use the *in-place upgrade* method for stateful applications by implementing a rolling deployment schedule and by following the guidelines mentioned in the section below on blue/green deployments.

In contrast, *replacement upgrades* offer a simpler way to deploy by provisioning new resources. By deploying a new stack and redirecting traffic from the old to the new one, you don't have the complexity of upgrading existing resource and potential failures. This is also useful if your application has unknown dependencies. The underlying Amazon EC2 instance usage is considered temporary or ephemeral in nature for the period of deployment until the current release is active. During the new release, a new set of Amazon EC2 instances is rolled out by terminating older instances. This type of upgrade technique is more common in an immutable infrastructure.

There are several deployment services that are especially useful for an *in-place upgrade*: AWS CodeDeploy, AWS OpsWorks, and AWS Elastic Beanstalk. *AWS CodeDeploy* is a deployment service that automates application deployments to Amazon EC2 instances or on-premises instances in your own facility. *AWS CodeDeploy* makes it easier for you to release new features rapidly, helps you avoid downtime during application deployment, and handles the complexity of updating your applications without many of the risks associated with error-prone manual deployments. You can also use *AWS OpsWorks* to manage your application deployment and updates. When you deploy an application, *AWS OpsWorks* Stacks triggers a Deploy event, which runs each layer's Deploy recipes. *AWS OpsWorks* Stacks also installs stack configuration and deployment attributes that contain all of the information needed to deploy the application, such as the application's repository and database connection data. *AWS Elastic Beanstalk* provides several options for how deployments are processed, including deployment policies (All at Once, Rolling, Rolling with Additional

Batch, and Immutable) and options that let you configure batch size and health check behavior during deployments.

For replacement upgrades, you provision a new environment with the deployment services, such as AWS Elastic Beanstalk, AWS CloudFormation, and AWS OpsWorks. A full set of new instances running the new version of the application in a separate Auto Scaling group will be created alongside the instances running the old version. Immutable deployments can prevent issues caused by partially completed rolling deployments. Typically, you will use a different Elastic Load Balancing load balancer for both the new stack and the old stack. By using Amazon Route 53 with weighted routing, you can roll traffic to the load balancer of the new stack.

In-Place Upgrade

AWS CodeDeploy is a deployment service that automates application deployments to Amazon EC2 instances or on-premises instances in your own facility. You can deploy a nearly unlimited variety of application content, such as code, web and configuration files, executables, packages, scripts, multimedia files, and so on. AWS CodeDeploy can deploy application content stored in Amazon S3 buckets, GitHub repositories, or Bitbucket repositories. Once you prepare deployment content and the underlying Amazon EC2 instances, you can deploy an application and its revisions on a consistent basis. You can push the updates to a set of instances called a *deployment group* that is made up of tagged Amazon EC2 instances and/or Auto Scaling groups. In addition, AWS CodeDeploy works with various configuration management tools, continuous integration and deployment systems, and source control systems. You can find a complete list of product integration options in the AWS CodeDeploy documentation.

AWS CodeDeploy is used for deployments by AWS CodeStar. AWS *CodeStar* enables you to develop, build, and deploy applications quickly on AWS. AWS CodeStar provides a unified user interface, enabling you to manage your software development activities easily in one place. With AWS CodeStar, you can set up your entire continuous delivery toolchain in minutes, allowing you to start releasing code faster. AWS CodeStar stores your application code securely on AWS *CodeCommit*, a fully managed source control service that eliminates the need to manage your own infrastructure to host Git repositories. AWS CodeStar compiles and packages your source code with AWS *CodeBuild*, a fully managed build service that makes it possible for you to build, test, and integrate code more frequently. AWS CodeStar accelerates software release with the help of AWS *CodePipeline*, a Continuous Integration and Continuous Delivery (CI/CD) service. AWS CodeStar integrates with AWS CodeDeploy and AWS CloudFormation so that you can easily update your application code and deploy to Amazon EC2 and AWS Lambda.

Another service to use for managing the entire lifecycle of an application is AWS OpsWorks. You can use built-in layers or deploy custom layers and recipes to launch your application stack. In addition, numerous customization options are available for configuration and pushing application updates. When you deploy an application, AWS OpsWorks Stacks triggers a Deploy event, which runs each layer's Deploy recipes. AWS OpsWorks Stacks also installs stack configuration and deployment attributes that contain all of the information needed to deploy the application, such as the application's repository and database connection data.

Replacement Upgrade

The replacement upgrade method replaces in-place resources with newly provisioned resources. There are advantages and disadvantages between the in-place upgrade method and replacement upgrade method. You can perform a replacement upgrade in a number of ways. You can use an Auto Scaling policy to define how you want to add (scale out) or remove (scale in) instances. By coupling this with your update strategy, you can control the rollout of an application update as part of the scaling event.

For example, you can create a new Auto Scaling Launch Configuration that specifies a new AMI containing the new version of your application. Then you can configure the Auto Scaling group to use the new launch configuration. The Auto Scaling termination policy by default will first terminate the instance with the oldest launch configuration and that is closest to the next billing hour. This in effect provides the most cost-effective method to phase out all instances that use the previous configuration. If you are using Elastic Load Balancing, you can attach an additional Auto Scaling configuration behind the load balancer and use a similar approach to phase in newer instances while removing older instances.

Similarly, you can configure rolling deployments in conjunction with deployment services such as AWS Elastic Beanstalk and AWS CloudFormation. You can use update policies to describe how instances in an Auto Scaling group are replaced or modified as part of your update strategy. With these deployment services, you can configure the number of instances to get updated concurrently or in batches, apply the updates to certain instances while isolating in-service instances, and specify the time to wait between batched updates. In addition, you can cancel or roll back an update if you discover a bug in your application code. These features can help increase the availability of your application during updates.

Blue/Green Deployments

Blue/green is a method where you have two identical stacks of your application running in their own environments. You use various strategies to migrate the traffic from your current application stack (blue) to a new version of the application (green). This method is used for a replacement upgrade. During a blue/green deployment, the latest application revision is installed on replacement instances and traffic is rerouted to these instances either immediately or as soon as you are done testing the new environment.

This is a popular technique for deploying applications with zero downtime. Deployment services like AWS Elastic Beanstalk, AWS CloudFormation, or AWS OpsWorks are particularly useful for blue/green deployments because they provide a simple way to duplicate your existing application stack.

Blue/green deployments offer a number of advantages over in-place deployments. An application can be installed and tested on the new instances ahead of time and deployed to production simply by switching traffic to the new servers. Switching back to the most recent version of an application is faster and more reliable because traffic can be routed back to the original instances as long as they have not been terminated. With an in-place deployment, versions must be rolled back by redeploying the previous version of the application. Because the instances provisioned for a blue/green deployment are new, they reflect

the most up-to-date server configurations, which helps you avoid the types of problems that sometimes occur on long-running instances.

For a stateless web application, the update process is pretty straightforward. Simply upload the new version of your application and let your deployment service deploy a new version of your stack (green). To cut over to the new version, you simply replace the Elastic Load Balancing URLs in your Domain Name Server (DNS) records. AWS Elastic Beanstalk has a Swap Environment URLs feature to facilitate a simpler cutover process. If you use Amazon Route 53 to manage your DNS records, you need to swap Elastic Load Balancing endpoints for AWS CloudFormation or AWS OpsWorks deployment services.

For applications with session states, the cutover process can be complex. When you perform an update, you don't want your end users to experience downtime or lose data. You should consider storing the sessions outside of your deployment service because creating a new stack will re-create the session database with a certain deployment service. In particular, consider storing the sessions separately from your deployment service if you are using an Amazon RDS database.

If you use Amazon Route 53 to host your DNS records, you can consider using the Weighted Round Robin (WRR) feature for migrating from blue to green deployments. The feature helps to drive the traffic gradually rather than instantly. If your application has a bug, this method helps ensure that the blast radius is minimal, as it only affects a small number of users. This method also simplifies rollbacks if they become necessary by redirecting traffic back to the blue stack. In addition, you only use the required number of instances while you scale up in the green deployment and scale down in the blue deployment. For example, you can set WRR to allow 10 percent of the traffic to go to green deployment while keeping 90 percent of traffic on blue. You gradually increase the percentage of green instances until you achieve a full cutover. Keeping the DNS cache to a shorter Time To Live (TTL) on the client side also ensures that the client will connect to the green deployment with a rapid release cycle, thus minimizing bad DNS caching behavior. For more information on Amazon Route 53, see Chapter 5, "Networking."

Hybrid Deployments

You can also use the deployment services in a hybrid fashion for managing your application fleet. For example, you can combine the simplicity of managing AWS infrastructure provided by AWS Elastic Beanstalk and the automation of custom network segmentation provided by AWS CloudFormation. Leveraging a hybrid deployment model also simplifies your architecture because it decouples your deployment method so that you can choose different strategies for updating your application stack.

Deployment Services

AWS deployment services provide easier integration with other AWS Cloud services. Whether you need to load-balance across multiple Availability Zones by using Elastic Load Balancing or by using Amazon RDS as a back end, the deployment services like AWS Elastic

Beanstalk, AWS CloudFormation, and AWS OpsWorks make it simpler to use these services as part of your deployment.

If you need to use other AWS Cloud services, you can leverage tool-specific integration methods to interact with the resource. For example, if you are using AWS Elastic Beanstalk for deployment and want to use Amazon DynamoDB for your back end, you can customize your environment resources by including a configuration file within your application source bundle. With AWS OpsWorks, you can create custom recipes to configure the application so that it can access other AWS Cloud services. Similarly, several template snippets with a number of example scenarios are available for you to use within your AWS CloudFormation templates.

AWS offers multiple strategies for provisioning infrastructure. You could use the building blocks (for example Amazon EC2, Amazon EBS, Amazon S3, and Amazon RDS) and leverage the integration provided by third-party tools to deploy your application. But for even greater flexibility, you can consider the automation provided by the AWS deployment services.

AWS Elastic Beanstalk

AWS Elastic Beanstalk is the fastest and simplest way to get an application up and running on AWS. It is ideal for developers who want to deploy code and not worry about managing the underlying infrastructure. AWS Elastic Beanstalk reduces management complexity without restricting choice or control. You simply upload your application, and AWS Elastic Beanstalk automatically handles the details of capacity provisioning, load balancing, scaling, and application health monitoring.

AWS Elastic Beanstalk provides platforms for programming languages (such as Java, PHP, Python, Ruby, or Go), web containers (for example Tomcat, Passenger, and Puma), and Docker containers, with multiple configurations of each. AWS Elastic Beanstalk provisions the resources needed to run your application, including one or more Amazon EC2 instances. The software stack running on the Amazon EC2 instances depends on the configuration. In a configuration name, the version number refers to the version of the platform configuration. You can also perform most deployment tasks, such as changing the size of your fleet of Amazon EC2 instances or monitoring your application, directly from the AWS Elastic Beanstalk Console.

Applications

The first step in using AWS Elastic Beanstalk is to create an application that represents your web application in AWS. In AWS Elastic Beanstalk, an application serves as a container for the environments that run your web application and versions of your web application's source code, saved configurations, logs, and other artifacts that you create while using AWS Elastic Beanstalk.

Environment Tiers

When you launch an AWS Elastic Beanstalk environment, you choose an environment tier, platform, and environment type. The environment tier that you choose determines whether

AWS Elastic Beanstalk provisions resources to support a web application that handles HTTP(S) requests or an application that handles background-processing tasks. An environment tier whose web application processes web requests is known as a *web server tier*. An environment tier whose application runs background jobs is known as a *worker tier*. The environment is the heart of the application. When you create an environment, AWS Elastic Beanstalk provisions the resources required to run your application.

If your application performs operations or workflows that take a long time to complete, you can offload those tasks to a dedicated worker environment. Decoupling your web application front end from a process that performs blocking operations is a common way to ensure that your application stays responsive under load. For the worker environment tier, AWS Elastic Beanstalk also creates and provisions an Amazon Simple Queue Service (Amazon SQS) queue if you don't already have one. When you launch a worker environment tier, AWS Elastic Beanstalk installs the necessary support files for your programming language of choice and a daemon on each Amazon EC2 instance in the Auto Scaling group. The daemon is responsible for pulling requests from an Amazon SQS queue and sending the data to the application running in the worker environment tier that will process those messages.

Environment Platforms

AWS Elastic Beanstalk supports Java, .NET, PHP, Node.js, Python, Packer, Ruby, Go, and Docker, and it is ideal for web applications. Due to AWS Elastic Beanstalk's open architecture, non-web applications can also be deployed using AWS Elastic Beanstalk. Additionally, AWS Elastic Beanstalk supports custom platforms that can be based on an AMI that you create from one of the supported operating systems and that can include further customizations. You can choose to have your AWS Elastic Beanstalk environments automatically updated to the latest version of the underlying platform running your application during a specified maintenance window. AWS Elastic Beanstalk regularly releases new versions of supported platforms with operating system, web and application server, language, and framework updates.

Managing Environments

AWS Elastic Beanstalk makes it easy to create new environments for your application. You can create and manage separate environments for development, testing, and production use, and you can deploy any version of your application to any environment. Environments can be long-running or temporary. When you terminate an environment, you can save its configuration to re-create it later.

As you develop your application, you will deploy it often, possibly to several different environments for different purposes. AWS Elastic Beanstalk lets you configure how deployments are performed. You can deploy to all of the instances in your environment simultaneously or split a deployment into batches with rolling deployments.

Managing Application Versions

AWS Elastic Beanstalk creates an application version whenever you upload source code. This usually occurs when you create a new environment or upload and deploy code using

the environment management console or AWS Elastic Beanstalk CLI. You can also upload a source bundle without deploying it from the application console.

A single-environment deployment of AWS Elastic Beanstalk performs an in-place update when you update your application versions, so your application may become unavailable to users for a short period of time. It is possible to avoid this downtime by performing a blue/green deployment with AWS Elastic Beanstalk, where you deploy the new version to a separate environment and then use AWS Elastic Beanstalk to swap the CNAMEs of the two environments to redirect traffic to the new version instantly.

Blue/green deployments require that your application's environments run independently of your production database—if your application uses one. If your AWS Elastic Beanstalk green environment has an Amazon RDS DB instance included in it, the data will not transfer over to the AWS Elastic Beanstalk blue environment and will be lost if you terminate the original environment.

Amazon EC2 Container Service

Amazon EC2 Container Service (Amazon ECS) is a highly scalable, high-performance container management service that makes it easy to run, stop, and manage Docker containers on a cluster of Amazon EC2 instances. *Docker* is a technology that allows you to build, run, test, and deploy distributed applications that are based on Linux containers. Amazon ECS uses Docker images in task definitions to launch containers on Amazon EC2 instances in your clusters.

Amazon ECS lets you launch and stop container-based applications with simple API calls, allowing you to get the state of your cluster from a centralized service and giving you access to many familiar Amazon EC2 features, such as security groups, Amazon EBS volumes, and IAM roles. Amazon ECS is a good option if you are using Docker for a consistent build and deployment experience or as the basis for sophisticated distributed systems. Amazon ECS is also a good option if you want to improve the utilization of your Amazon EC2 instances.

You can use Amazon ECS to schedule the placement of containers across your cluster based on your resource needs, isolation policies, and availability requirements. Amazon ECS eliminates the need for you to operate your own cluster management and configuration management systems or worry about scaling your management infrastructure. It also can be used to create a consistent deployment and build experience, manage and scale batch and Extract-Transform-Load (ETL) workloads, and build sophisticated application architectures on a microservices model.

While AWS Elastic Beanstalk can also be used to develop, test, and deploy Docker containers rapidly in conjunction with other components of your application infrastructure, using Amazon ECS directly provides more fine-grained control and access to a wider set of use cases.

Clusters

Amazon ECS is a regional service that simplifies running application containers in a highly available manner across multiple Availability Zones within a region. You can create

Amazon ECS clusters within a new or existing Amazon VPC. After a cluster is up and running, you can indicate task definitions and services that specify which Docker container images to run across your clusters. Container images are stored in and pulled from container registries, which may exist within or outside of your AWS infrastructure. A *cluster* is a logical grouping of Amazon EC2 instances on which you run tasks. When running a task, Amazon ECS downloads your container images from a registry that you specify and runs those images on the container instances within your cluster.

Instances

An *Amazon ECS container instance* is an Amazon EC2 instance that is running the Amazon ECS container agent and has been registered into a cluster. When you run tasks with Amazon ECS, your tasks are placed on your active container instances. A container instance must be running the Amazon ECS container agent to register into one of your clusters. If you are using the Amazon ECS-optimized AMI, the agent is already installed. To use a different operating system, install the appropriate agent.

The Amazon ECS container agent makes calls to Amazon ECS on your behalf. You must launch container instances with an instance profile that has an appropriate IAM role. The IAM role must have a policy attached that authenticates to your account and provides the required resource permissions. Additionally, the container instances need external network access to allow the agent to communicate with the Amazon ECS service endpoint. If your container instances do not have public IP addresses, then they must use Network Address Translation (NAT) or an HTTP proxy to provide this access. When the Amazon ECS container agent registers an instance into your cluster, the container instance reports its status as ACTIVE and its agent connection status as TRUE. This container instance can accept run task requests.

The Amazon ECS-optimized AMI is the recommended AMI for you to use to launch your Amazon ECS container instances. The Amazon ECS-optimized AMI is preconfigured and has been tested on Amazon ECS by AWS engineers. It provides the latest, tested, minimal version of the Amazon Linux AMI, Amazon ECS container agent, recommended version of Docker, and container services package to run and monitor the Amazon ECS agent. Typically, the Amazon ECS config file on the instance is modified with a user data script to specify parameters such as the cluster with which to register the instance.

Containers

To deploy applications on Amazon ECS, your application components must be architected to run in containers. A Docker container is a standardized unit of software development that contains everything that your software application needs to run including operating system, configuration, code, runtime, system tools, and system libraries. Containers are lighter in weight and have less memory and computational overhead than virtual machines, so they make it easy to support applications that consist of hundreds or thousands of small, isolated microservices. A properly containerized application is easy to scale and maintain and makes efficient use of available system resources.

Building your application as a collection of tight, focused containers allows you to build them in parallel with strict, well-defined interfaces. With better interfaces between microservices, you have the freedom to improve and even totally revise implementations without fear of breaking running code. Because your application's dependencies are spelled out explicitly and declaratively, less time will be lost diagnosing, identifying, and fixing issues that arise from missing or obsolete packages. Using containers allows you to build components that run in isolated environments, thus limiting the ability of one container to disrupt the operation of another accidentally while still being able to share libraries and other common resources cooperatively. This opportunistic sharing reduces memory pressure and leads to increased runtime efficiency.

Images

Containers are created from a read-only template called an *image*. Images are typically built from a *Dockerfile*, a manifest that describes the base image to use for your Docker image and what you want installed and running on it. When you build the Docker image from your Dockerfile, the images are stored in a registry from which they can be downloaded and run on your container instances.

Repository

Docker uses images that are stored in repositories to launch containers. The default repository for the Docker daemon is Docker Hub. Although you don't need a Docker Hub account to use Amazon ECS or Docker, having a Docker Hub account gives you the freedom to store your modified Docker images so that you can use them in your Amazon ECS task definitions.

Another registry option is Amazon EC2 Container Registry (Amazon ECR). Amazon ECR is a managed AWS Docker registry service. Customers can use the familiar Docker CLI to push, pull, and manage images. Amazon ECR supports private Docker repositories with resource-based permissions using IAM so that specific users or Amazon EC2 instances can access repositories and images.

Tasks

A *task definition* is like a blueprint for your application. Every time you launch a task in Amazon ECS, you specify a task definition so that the service knows which Docker image to use for containers, how many containers to use in the task, and the resource allocation for each container.

Amazon ECS provides three task features:

- A service scheduler for long-running tasks and applications
- The ability to run tasks manually for batch jobs or single-run tasks, with Amazon ECS placing tasks on your cluster for you
- The ability to run tasks on the container instance that you specify so that you can integrate with custom or third-party schedulers or place a task manually on a specific container instance

A task is the instantiation of a task definition on a container instance within your cluster. Before you can run Docker containers on Amazon ECS, you must create a task definition. You can define multiple containers and data volumes in a task definition.

Services

Amazon ECS allows you to run and maintain a specified number (the “desired count”) of instances of a task definition simultaneously in an Amazon ECS cluster. This is called a *service*. If any of your tasks should fail or stop for any reason, the Amazon ECS service scheduler launches another instance of your task definition to replace it and maintain the desired count of tasks in the service. In addition to maintaining the desired count of tasks in your service, you can optionally run your service behind a load balancer. The load balancer distributes traffic across the tasks that are associated with the service.

The service scheduler is ideally suited for long-running, stateless services and applications. You can update your services that are maintained by the service scheduler, such as deploying a new task definition or changing the running number of desired tasks. By default, the service scheduler spreads tasks across Availability Zones, but you can use task placement strategies and constraints to customize task placement decisions.

AWS OpsWorks Stacks

AWS OpsWorks is an application management service that makes it easy for developers and operations personnel to deploy and operate applications of all shapes and sizes. AWS OpsWorks works best if you want to deploy your code, have some abstraction from the underlying infrastructure, and have an application more complex than a Three-Tier architecture. AWS OpsWorks is also recommended if you want to manage your infrastructure with a configuration management system such as Chef.

AWS OpsWorks Stacks provides a simple and flexible way to create and manage stacks and applications. It has a rich set of customizable components that you can mix and match to create a stack that satisfies your specific purposes. Additionally, if you have existing computing resources, you can incorporate them into a stack along with instances that you created with AWS OpsWorks Stacks. Such resources can be existing Amazon EC2 instances or even on-premises instances that are running on your own hardware. You can then use AWS OpsWorks Stacks to manage all related instances as a group, regardless of how they were created.

Stacks

The *stack* is the top-level AWS OpsWorks Stacks entity. It represents a set of instances that you want to manage collectively, typically because they have a common purpose such as serving PHP applications. In addition to serving as a container, a stack handles tasks that apply to the group of instances as a whole, such as managing applications and cookbooks. For example, a stack whose purpose is to serve web applications might contain a set of application server instances, a load balancer to direct traffic to the instances, and a database instance that serves as a back end of the application instances. A common practice is to have multiple stacks that represent different environments (such as development, staging, and production stacks).

Layers

A *layer* is a blueprint for a set of instances, such as Amazon EC2 instances. It specifies information such as the instance's settings, resources, installed packages, and security groups. You can add Chef recipes to lifecycle events of your instances; for example, to install and configure any required software. Every stack contains one or more layers, each of which represents a stack component. As you work with AWS OpsWorks layers, keep in mind that each instance in a stack must be a member of at least one layer, except for registered instances. Amazon EC2 instances can optionally be a member of multiple AWS OpsWorks layers, however. In that case, AWS OpsWorks Stacks runs the recipes to install and configure packages, deploy applications, and so on for each of the instance's layers.

Instances

After you create a layer, you usually add at least one instance. You can add more instances later if the current set can't handle the load. You can also use load-based or time-based instances to scale the number of instances automatically. An instance's operating system can have any of several Linux distributions or Windows Server 2012 R2. To add an instance to the stack, you can add it directly within AWS OpsWorks Stacks or add a Linux-backed instance that was created elsewhere, including on-premises instances.

Applications

An AWS OpsWorks Stacks application represents code that you want to run on an application server. The code itself resides in a repository such as an Amazon S3 archive, and the application contains the information required to deploy the code to the appropriate application server instances. When you deploy an application, AWS OpsWorks Stacks triggers a Deploy event, which runs each layer's Deploy recipes. AWS OpsWorks Stacks also installs stack configuration and deployment attributes that contain all of the information needed to deploy the app, such as the application's repository and database connection data.

The first step in deploying an application to your application servers is to add an application to the stack. The stack represents the application and contains a variety of metadata, such as the application's name and type and the information required to deploy the application to the server instances (for example, the repository URL). You can deploy applications from a Git repository, Amazon S3 bundle, HTTP bundle, and other repositories. Individual repository types have their own requirements.

When AWS OpsWorks Stacks runs a command on an instance, it triggers a Deploy command in response to a Deploy lifecycle event. The Deploy command adds a set of attributes to the instance's node object that describes the stack's current configuration. For Deploy events and Execute Recipes stack commands, AWS OpsWorks Stacks installs Deploy attributes, which provide some additional deployment information.

Cookbooks

AWS OpsWorks Stacks uses Chef cookbooks to handle tasks such as installing and configuring packages and deploying applications. Your custom cookbooks must be stored in an online repository, either an archive such as a .zip file or a source control manager such as

Git. A stack can have only one custom cookbook repository, but the repository can contain any number of cookbooks. When you install or update the cookbooks, AWS OpsWorks Stacks installs the entire repository in a local cache on each of the stack’s instances. When an instance needs, for example, to run one or more recipes, it uses the code from the local cache.

Repositories can use Source Control Managers, Git, or Subversion, or the repository can be stored in Amazon S3 or an HTTP location. Cookbooks are organized as directories in the root of the structure. Each cookbook directory has at least one, and typically all, of the standard directories and files, which must use standard names. Standard names of directories are attributes, recipes, templates, and other. The cookbook directory should also include `metadata.rb`, which is the cookbook’s metadata. Templates must be in a subdirectory of the templates directory, which contains at least one, and optionally multiple, subdirectories. Those subdirectories can optionally have further subdirectories as well.

AWS CloudFormation

AWS CloudFormation takes care of provisioning and configuring resources for you. You don’t need to create and configure AWS resources individually and figure out what’s dependent on what. AWS CloudFormation provides the systems operator, architect, and other personnel with the ability to provision and manage stacks of AWS resources based on templates that you create to model your infrastructure architecture. You can manage anything from a single Amazon EC2 instance to a complex multi-tier, multi-regional application. Using templates means that you can impose version control on your infrastructure and easily replicate your infrastructure stack quickly and with repeatability.

This deployment service is ideal for simplifying deploying infrastructure, while providing capabilities to address complex architectures. For a scalable web application that also includes a back end database, you might use an Auto Scaling group, an Elastic Load Balancing load balancer, and an Amazon RDS DB instance. Normally, you might use each individual service to provision these resources, after which you would have to configure them to work together. All of these tasks can add complexity and time before you even get your application up and running. Instead, you can create or modify an existing AWS CloudFormation template. A template describes all of your resources and their properties.

By using AWS CloudFormation, you can quickly replicate your infrastructure. If your application requires additional availability, you might replicate it in multiple regions so that if one region becomes unavailable, your users can still use your application in other regions. The challenge in replicating your application is that it also requires you to replicate your resources. Not only do you need to record all of the resources that your application requires, but you must also provision and configure those resources in each region. You can reuse your AWS CloudFormation template to set up your resources consistently and repeatedly. You can describe your resources once and then provision the same resources in multiple regions by creating a stack in each region with one template. AWS CloudFormation is also useful for moving from development to test to production phases—simply create the stack and name it for its purpose.

AWS CloudFormation is recommended if you want a tool for granular control over the provisioning and management of your own infrastructure. As part of your infrastructure

deployment, you may want to incorporate the components necessary to deploy application updates. For example, AWS CodeDeploy is a recommended adjunct to AWS CloudFormation for managing the application deployments and updates.

Creating Stacks

A *stack* is a collection of AWS resources that you can manage as a single unit. In other words, you can create, update, or delete a collection of resources by creating, updating, or deleting stacks. All of the resources in a stack are defined by the stack's AWS CloudFormation template. A stack, for example, can include all of the resources required to run a web application, such as a web server, a database, and networking rules. If you no longer require that web application, you can simply delete the stack and all of its related resources are deleted.

AWS CloudFormation ensures that all stack resources are created or deleted as appropriate. Because AWS CloudFormation treats the stack resources as a single unit, they must all be created or deleted successfully for the stack to be created or deleted. If a resource cannot be created, by default, AWS CloudFormation rolls back the stack creation and automatically deletes any resources that were created. If a resource cannot be deleted, any remaining resources are retained until the stack can be successfully deleted. You can work with stacks by using the AWS CloudFormation console, API, or AWS CLI.

Before you create a stack, you must have a template that describes what resources AWS CloudFormation will include in your stack. Creating a stack on the AWS CloudFormation console is an easy, wizard-driven process.



After you launch a stack, use the AWS CloudFormation console, API, or AWS CLI to update resources in your stack. Do not make changes to stack resources outside of AWS CloudFormation. Doing so can create a mismatch between your stack's template and the current state of your stack resources, which can cause errors if you update or delete the stack.

Deleting Stacks

After the stack deletion is complete, the stack will be in the `DELETE_COMPLETE` state. Stacks in the `DELETE_COMPLETE` state are not displayed in the AWS CloudFormation console by default. When stacks are in the `DELETE_FAILED` state, because AWS CloudFormation couldn't delete a resource, rerun the deletion with the `RetainResources` parameter and specify the resource that AWS CloudFormation can't delete to bypass or correct the error and rerun. Typically, a delete stack failure most commonly occurs for one of two reasons: There are dependencies external to the stack, or you do not have IAM permissions to delete the resource. Some resources must be empty before they can be deleted. For example, you must delete all objects in an Amazon S3 bucket or remove all instances in an Amazon EC2 security group before you can delete the bucket or security group. In addition to AWS CloudFormation permissions, you must be allowed to use the underlying services, such as Amazon S3 or Amazon EC2.

Updating Stacks

When you need to make changes to a stack's settings or change its resources, you update the stack instead of deleting it and creating a new stack. For example, if you have a stack with an Amazon EC2 instance, you can update the stack to change the instance's AMI ID. When you update a stack, you submit changes, such as new input parameter values or an updated template. AWS CloudFormation compares the changes you submit with the current state of your stack and updates only the changed resources.



When updating a stack, AWS CloudFormation might interrupt resources or replace updated resources, depending on which properties you update. Review how AWS CloudFormation updates a given resource and whether it will affect end users if it is interrupted.

When you update a stack, AWS CloudFormation updates resources based on differences between what you submit and the stack's current template. Resources that have not changed run without disruption during the update process. For updated resources, AWS CloudFormation uses one of these update behaviors: Update with No Interruption, Updates with Some Interruption, or Replacement. The method AWS CloudFormation uses depends on which property you update for a given resource type. The update behavior for each property is described in the AWS Resource Types Reference section of the AWS CloudFormation User Guide.

Depending on the update behavior, you can decide when to modify resources to reduce the impact of these changes on your application. In particular, you can plan when resources must be replaced during an update. For example, if you update the Port property of an `AWS::RDS::DBInstance` resource type, AWS CloudFormation replaces the DB instance by creating a new DB instance with the updated port setting and deleting the old DB instance.

Using Change Sets

AWS CloudFormation provides two methods for updating stacks: direct update or creating and executing change sets. When you directly update a stack, you submit changes, and AWS CloudFormation immediately deploys them. Use direct updates when you want to deploy your updates quickly.

With change sets, you can preview the changes that AWS CloudFormation will make to your stack and then decide whether to apply those changes. *Change sets* are JSON-formatted documents that summarize the changes that AWS CloudFormation will make to a stack. Use change sets when you want to ensure that AWS CloudFormation doesn't make unintentional changes, or when you want to consider several options. For example, you can use a change set to verify that AWS CloudFormation won't replace your stack's database instances during an update. Understanding how your changes will affect running resources before you implement them can help you update stacks with confidence. You can create and manage change sets using the AWS CloudFormation console, AWS CLI, or AWS CloudFormation API.



Change sets don't indicate whether AWS CloudFormation will successfully update a stack. For example, a change set doesn't check if you will surpass an account limit, if you're updating a resource that doesn't support updates, or if you have insufficient permissions to modify a resource, all of which can cause a stack update to fail. If an update fails, AWS CloudFormation attempts to roll back your resources to their original state. After you execute a change, AWS CloudFormation removes all change sets that are associated with the stack because they aren't applicable to the updated stack.

When you directly modify resources in the stack's template to generate a change set, AWS CloudFormation classifies the change as a direct modification, as opposed to changes triggered by an updated parameter value. The following change set, which added a new tag to the Amazon EC2 `i-1abc23d4` instance, is an example of a direct modification. All other input values, such as the parameter values and capabilities, are unchanged, so we'll focus on the `Changes` structure.

```
"Changes": [
    {
        "ResourceChange": {
            "ResourceType": "AWS::EC2::Instance",
            "PhysicalResourceId": "i-1abc23d4",
            "Details": [
                {
                    "ChangeSource": "DirectModification",
                    "Evaluation": "Static",
                    "Target": {
                        "Attribute": "Tags",
                        "RequiresRecreation": "Never"
                    }
                }
            ],
            "Action": "Modify",
            "Scope": [
                "Tags"
            ],
            "LogicalResourceId": "MyEC2Instance",
            "Replacement": "False"
        },
        "Type": "Resource"
    }
]
```

In the Changes structure, there's only one ResourceChange structure. This structure describes information such as the type of resource that AWS CloudFormation will change, the action that AWS CloudFormation will take, the ID of the resource, the scope of the change, and whether the change requires a replacement (where AWS CloudFormation creates a new resource and then deletes the old one). In the example, the change set indicates that AWS CloudFormation will modify the Tags attribute of the i-1abc23d4 Amazon EC2 instance and doesn't require the instance to be replaced.

In the Details structure, AWS CloudFormation labels this change as a direct modification that will never require the instance to be re-created (replaced). You can confidently execute this change, knowing that AWS CloudFormation won't replace the instance.

AWS CloudFormation shows this change as a *Static* evaluation. A *Static* evaluation means that AWS CloudFormation can determine the tag's value before executing the change set. In some cases, AWS CloudFormation can determine a value only after you execute a change set. AWS CloudFormation labels those changes as *Dynamic* evaluations. For example, if you reference an updated resource that is conditionally replaced, AWS CloudFormation can't determine whether the reference to the updated resource will change.

Template Anatomy

To provision and configure your stack resources, you must understand AWS *CloudFormation templates*, which are formatted text files in JSON or YAML. These templates describe the resources that you want to provision in your AWS CloudFormation Stacks. You can use AWS CloudFormation Designer or any text editor to create and save templates.

When you provision your infrastructure with AWS CloudFormation, the AWS CloudFormation template describes exactly what resources are provisioned and their settings. Because these templates are text files, you simply track differences in your templates to track changes to your infrastructure, similar to the way developers control revisions to source code. For example, you can use a version control system with your templates so that you know exactly what changes were made, who made them, and when. If you need to reverse changes to your infrastructure, you can use a previous version of your template.

You can author AWS CloudFormation templates in JSON or YAML formats. AWS supports all AWS CloudFormation features and functions for both formats, including in AWS CloudFormation Designer. When deciding which format to use, pick the format in which you're most comfortable working. Also consider that YAML inherently provides some features, such as commenting, that aren't available in JSON. The following example shows a JSON-formatted template fragment. Be aware that the AWSTemplateFormatVersion section of the template identifies the capabilities of the template. The latest template format version is 2010-09-09, and it is currently the only valid value.

```
{  
  "AWSTemplateFormatVersion" : " 2010-09-09",  
  "Description" : "JSON string",  
  "Metadata" : {  
    template metadata
```

```
},
"Parameters" : {
    set of parameters
},
"Mappings" : {
    set of mappings
},
"Conditions" : {
    set of conditions
},
"Transform" : {
    set of transforms
},
"Resources" : {
    set of resources
},
"Outputs" : {
    set of outputs
}
}
```

Template Metadata

You can use the optional `Metadata` section to include arbitrary JSON or YAML objects that provide details about the template. For example, you can include template implementation details about specific resources. During a stack update, however, you cannot update the `Metadata` section by itself. You can update it only when you include changes that add, modify, or delete resources.

```
"Metadata" : {
    "Instances" : {"Description" : "Information about the instances"},
    "Databases" : {"Description" : "Information about the databases"}
}
```

Some AWS CloudFormation features retrieve settings or configuration information that you define from the `Metadata` section. You define this information in AWS CloudFormation-specific metadata keys.

`AWS::CloudFormation::Init` defines configuration tasks for the `cfn-init` helper script. This script is useful for configuring and installing applications on Amazon EC2 instances.

`AWS::CloudFormation::Interface` is a metadata key that defines how parameters are grouped and sorted in the AWS CloudFormation console. When you create or update stacks in the console, the console lists input parameters in alphabetical order by their logical IDs. By using this key, you can define your own parameter grouping and ordering so that users

can efficiently specify parameter values. In addition to grouping and ordering parameters, you can define labels for parameters. A *label* is a user-friendly name or description that the console displays instead of a parameter's logical ID. Labels are useful for helping users understand the values to specify for each parameter.

```
"Metadata" : {
    "AWS::CloudFormation::Interface" : {
        "ParameterGroups" : [ ParameterGroup, ... ],
        "ParameterLabels" : ParameterLabel
    }
}
```

Template Parameters

You can use the optional `Parameters` section to pass values into your template when you create a stack. With parameters, you can create templates that are customized each time that you create a stack. Each parameter must contain a value when you create a stack. You can specify a default value to make the parameter optional so that you don't need to pass in a value when creating a stack.

```
"Parameters" : {
    "InstanceTypeParameter" : {
        "Type" : "String",
        "Default" : "t2.micro",
        "AllowedValues" : ["t2.micro", "m1.small", "m1.large"],
        "Description" : "Select t2.micro, m1.small, or m1.large. Default is t2.micro."
    }
}
```

Within the same template, you can use the `Ref` intrinsic function to use the parameter value in other parts of the template. The following snippet uses the `InstanceTypeParameter` parameter to specify the instance type for an Amazon EC2 instance resource. Keep in mind that parameter and resource names are set by you and must be named uniquely within the template.

```
"Resources" : {
    "EC2Instance" : {
        "Type" : "AWS::EC2::Instance",
        "Properties" : {
            "InstanceType" : { "Ref" : "InstanceTypeParameter" },
            "ImageId" : "ami-2f726546"
        }
    }
}
```

When using parameter types in your templates, AWS CloudFormation can validate parameter values during the stack creation and update process, saving you time, effort, and cost. Using parameter types also enables AWS CloudFormation to show you a more intuitive user interface in the stack creation and update wizards. For example, a drop-down user interface of valid key-pair names is displayed when using a parameter of type AWS::EC2::KeyPair::KeyName. It is a best practice to use a parameter type that is the most restrictive appropriate type. You could use a string parameter type for an Amazon EC2 instance key-pair name, but that would allow operators who are deploying it to enter any string, even if it may not be a valid key pair within the region for the given account.

```
"Parameters" : {
    "KeyPairParameter" : {
        "Type" : "AWS::EC2::KeyPair::KeyName",
        "Description" : "Key pair used for EC2 instance."
    }
},
"Resources" : {
    "EC2Instance" : {
        "Type" : "AWS::EC2::Instance",
        "Properties" : {
            "InstanceType" : { "Ref" : "InstanceTypeParameter" },
            "KeyName" : { "Ref" : "KeyPairParameter" },
            "ImageId" : "ami-2f726546"
        }
    }
}
```

Template Mapping

The optional Mappings section matches a key to a corresponding set of named values. For example, if you want to set values based on a region, you can create a mapping that uses the region name as a key and contains the values that you want to specify for each specific region. You use the Fn::FindInMap intrinsic function to retrieve values in a map. However, you cannot include parameters, pseudo parameters, or intrinsic functions in the Mappings section.

The Mappings section consists of the key name mappings. The keys and values in mappings must be literal strings. Within a mapping, each map is a key followed by another mapping. The key identifies a map of name/value pairs and must be unique within the mapping. The name can contain only alphanumeric characters.

```
"Mappings" : {
    "Mapping01" : {
        "Key01" : {
            "Name" : "Value01"
        }
    }
},
```

```

"Key02" : {
    "Name" : "Value02"
},
"Key03" : {
    "Name" : "Value03"
}
}
}
}

```

Most commonly, you'll see a `Mappings` section used for AMI IDs. If you want to use the template in multiple regions, you'll need a mapping to use the corresponding ID for the region. An AMI ID is unique for the account within a region. That means that if you hardcode the image value for an Amazon EC2 instance, you would only be able to use the template within the region where that image exists. Even if you copy an image to another region, that copied image will have a different ID, which you will need to map as well.

As mentioned previously, you can use the `Fn::FindInMap` function to return a named value based on a specified key. This is a three-parameter function. First, you specify the name of the map within which to look. Second, you specify the key to find within the map. Third, you specify which attribute you want to return from the item.

In the following example, let's assume that you have two images for an instance, and these images have two different versions of your application. You could test each version by creating a stack for each and selecting the appropriate parameter. Additionally, instead of specifying a parameter for the region, you can use a pseudo parameter. `AWS::Region` is a pseudo parameter. Because AWS CloudFormation is a regional service and you specify the region to which to deploy, you don't have to supply the region name manually.

```

"Parameters": {
    "InstanceType" : {
        "Type" : "String",
        "Default" : "Version1",
        "AllowedValues" : [
            "Version1",
            "Version2"
        ]
    },
    "Mappings" : {
        "RegionMap" : {
            "us-east-1" : { "Version1" : "ami-6411e20d", "Version2" : "ami-7a11e213" },
            "us-west-1" : { "Version1" : "ami-c9c7978c", "Version2" : "ami-cfc7978a" },
            "eu-west-1" : { "Version1" : "ami-37c2f643", "Version2" : "ami-31c2f645" },
            "ap-southeast-1" : { "Version1" : "ami-66f28c34", "Version2" : "ami-60f28c32" },
            "ap-northeast-1" : { "Version1" : "ami-9c03a89d", "Version2" : "ami-a003a8a1" }
        }
    }
},

```

```
"Resources" : {  
    "myEC2Instance" : {  
        "Type" : "AWS::EC2::Instance",  
        "Properties" : {  
            "ImageId" : { "Fn::FindInMap" : [  
                "RegionMap",  
                { "Ref" : "AWS::Region" },  
                { "Ref" : "VersionParameter" }]  
            },  
            "InstanceType" : "t2.micro"  
        }  
    }  
}
```

Template Conditions

The optional Conditions section includes statements that define when a resource is created or when a property is defined. For example, you can compare whether a value is equal to another value. Based on the result of that condition, you can conditionally create resources. If you have multiple conditions, separate them with commas.

You might use conditions when you want to reuse a template that can create resources in different contexts, such as a test environment versus a production environment. In your template, you can add an EnvironmentType input parameter, which accepts either prod or test as inputs. For the production environment, you might include Amazon EC2 instances with certain capabilities. For the test environment, however, you probably want to use reduced capabilities to save money. With conditions, you can define which resources are created and how they're configured for each environment type.

Conditions are evaluated based on input parameter values that you specify when you create or update a stack. Within each condition, you can reference another condition, a parameter value, or a mapping. After you define all of your conditions, you can associate them with resources and resource properties in the Resources and Outputs sections of a template.

At stack creation or stack update, AWS CloudFormation evaluates all of the conditions in your template before creating any resources. Any resources that are associated with a true condition are created. Any resources that are associated with a false condition are ignored.



During a stack update, you cannot update conditions by themselves. You can update conditions only when you include changes that add, modify, or delete resources.

To create resources conditionally, you must include statements in at least three different sections of a template: Parameters, Conditions, and Resources. The Parameters section defines the input values that you want to evaluate in your conditions. Conditions will result

in a true or false result, based on values from these input parameters. The Conditions section is where you define the intrinsic condition functions. These conditions determine when AWS CloudFormation creates the associated resources. The Resources section is where you associate conditions with the resources that you want to create conditionally. Use the condition key and a condition's logical ID to associate it with a resource or output.

You can use the following intrinsic functions to define conditions:

- Fn::And
- Fn::Equals
- Fn::If
- Fn::Not
- Fn::Or

The following sample template includes an EnvType input parameter, where you can specify prod to create a stack for production or test to create a stack for testing. The CreateProdResources condition evaluates to true if the EnvType parameter is equal to prod. In the sample template, the Amazon EC2 instance is associated with the CreateProdResources condition. Therefore, the resources are created only if the EnvType parameter is equal to prod. When the parameter is not set to prod, it does not create the instance.

```
{  
    "AWSTemplateFormatVersion" : "2010-09-09",  
    "Parameters" : {  
        "EnvType" : {  
            "Default" : "test",  
            "Type" : "String",  
            "AllowedValues" : ["prod", "test"],  
        }  
    },  
    "Conditions" : {  
        "CreateProdResources" : {"Fn::Equals" : [{"Ref" : "EnvType"}, "prod"]}  
    },  
    "Resources" : {  
        "EC2Instance" : {  
            "Type" : "AWS::EC2::Instance",  
            "Condition" : "CreateProdResources",  
            "Properties" : {  
                "ImageId" : { "Fn::FindInMap" : [ "RegionMap", { "Ref" : "AWS::Region" } , "AMI" ]}  
            }  
        }  
    }  
}
```

Template Resources

The required Resources section declares the AWS resources that you want to include in the stack, such as an Amazon EC2 instance or an Amazon S3 bucket. You must declare each resource separately. If you have multiple resources of the same type, however, you can declare them together by separating them with commas. The Resources section consists of the key name resources.

Each of the resources within this section has a logical ID. The logical ID must be alphanumeric (A-Za-z0-9) and unique within the template. Use the logical ID to reference the resource in other parts of the template. For example, if you want to map an Amazon EBS volume to an Amazon EC2 instance, you reference the logical IDs to associate the block stores with the instance.

```
"Resources" : {  
    "MyLogicalID" : {  
        "Type" : "Resource type",  
        "Properties" : {  
            Set of properties  
        }  
    }  
}
```

In addition to the logical ID, certain resources also have a physical ID, which is the actual assigned name for that resource, such as an Amazon EC2 instance ID or an Amazon S3 bucket name. Use the physical IDs to identify resources outside of AWS CloudFormation templates, but only after the resources have been created. For example, you might give an Amazon EC2 instance resource a logical ID of MyEC2Instance, but when AWS CloudFormation creates the instance, AWS CloudFormation automatically generates and assigns a physical ID (such as i-28f9ba55) to the instance. You can use this physical ID to identify the instance and view its properties (such as the DNS name) by using the Amazon EC2 Console. Since you do not know the physical ID of a resource until the stack is created, you use references to associate resources where you would normally use a physical ID. In the following example, an Elastic IP resource needs to be associated with an Amazon EC2 instance by providing the instance ID. Since we do not know the instance ID ahead of stack creation, we reference the Amazon EC2 instance also created with the AWS CloudFormation template.

```
"Resources" : {  
    "EC2Instance" : {  
        "Type" : "AWS::EC2::Instance",  
        "Properties" : {  
            "ImageId" : "ami-7a11e213"  
        }  
    },  
    "MyEIP" : {
```

```
"Type" : "AWS::EC2::EIP",
"Properties" : {
    "InstanceId" : { "Ref" : "EC2Instance" }
}
}
```

The resource type identifies the type of resource that you are declaring. For example, AWS::EC2::Instance declares an Amazon EC2 instance. Resource properties are additional options that you can specify for a resource. For example, for each Amazon EC2 instance, you must specify an AMI ID for that instance. Property values can be literal strings, lists of strings, Booleans, parameter references, pseudo references, or the value returned by a function, depending on what the property is used for. The following are the properties available for an AWS::EC2::Instance AWS CloudFormation resource.

```
{
    "Type" : "AWS::EC2::Instance",
    "Properties" : {
        "Affinity" : String,
        "AvailabilityZone" : String,
        "BlockDeviceMappings" : [ EC2 Block Device Mapping, ... ],
        "DisableApiTermination" : Boolean,
        "EbsOptimized" : Boolean,
        "HostId" : String,
        "IamInstanceProfile" : String,
        "ImageId" : String,
        "InstanceInitiatedShutdownBehavior" : String,
        "InstanceType" : String,
        "Ipv6AddressCount" : Integer,
        "Ipv6Addresses" : [ IPv6 Address Type, ... ],
        "KernelId" : String,
        "KeyName" : String,
        "Monitoring" : Boolean,
        "NetworkInterfaces" : [ EC2 Network Interface, ... ],
        "PlacementGroupName" : String,
        "PrivateIpAddress" : String,
        "RamdiskId" : String,
        "SecurityGroupIds" : [ String, ... ],
        "SecurityGroups" : [ String, ... ],
        "SourceDestCheck" : Boolean,
        "SsmAssociations" : [ SSMAssociation, ... ],
        "SubnetId" : String,
```

```
    "Tags" : [ Resource Tag, ... ],
    "Tenancy" : String,
    "UserData" : String,
    "Volumes" : [ EC2 MountPoint, ... ],
    "AdditionalInfo" : String
}
}
```

There are hundreds of different AWS resources supported directly within AWS CloudFormation, each with their own set of properties. While not every AWS resource is directly supported, you can use custom resources where they are not supported directly. Custom resources enable you to write custom provisioning logic in templates that AWS CloudFormation runs any time you create, update (if you changed the custom resource), or delete stacks. Use the `AWS::CloudFormation::CustomResource` or `Custom::String` resource type to define custom resources in your templates. Custom resources require one property, the service token, which specifies where AWS CloudFormation sends requests (for example, an Amazon Simple Notification Service [Amazon SNS] topic).

Template Outputs

The optional `Outputs` section declares output values that you can import into other stacks (to create cross-stack references), return in response (to describe stack calls), or view on the AWS CloudFormation console. For example, you can output the Amazon S3 bucket name from a stack to make the bucket easier to find. The `Outputs` section consists of the key name `outputs` followed by a space and a single colon. The value of the property is returned by the AWS CloudFormation `describe-stacks` command. The value of an output can include literals, parameter references, pseudo parameters, a mapping value, or intrinsic functions.

In the following example, the output named `LoadBalancerDNSName` returns the DNS name for the resource with the logical ID `LoadBalancer` only when the `CreateProdResources` condition is true. The second output shows how to specify multiple outputs and use a reference to a resource to return its physical ID.

```
"Outputs" : {
    "LoadBalancerDNSName" : {
        "Description": "The DNSName of the backup load balancer",
        "Value" : { "Fn::GetAtt" : [ "LoadBalancer", "DNSName" ]},
        "Condition" : "CreateProdResources"
    },
    "InstanceId" : {
        "Description": "The Instance ID",
        "Value" : { "Ref" : "EC2Instance" }
    }
}
```

Outputs also have an export field. Exported values are useful for cross-stack referencing. In the following examples, the output named StackVPC returns the ID of an Amazon VPC and then exports the value for cross-stack referencing with the name VPCID appended to the stack's name.

```
"Outputs" : {  
    "StackVPC" : {  
        "Description" : "The ID of the VPC",  
        "Value" : { "Ref" : "MyVPC" },  
        "Export" : {  
            "Name" : {"Fn::Sub": "${AWS::StackName}-VPCID" }  
        }  
    }  
}
```

Best Practices

Best practices are recommendations that can help you use AWS CloudFormation more effectively and securely throughout its entire workflow. Planning and organizing your stacks, creating templates that describe your resources and the software applications that run on them, and managing your stacks and their resources are all good best practices.

Use the lifecycle and ownership of your AWS resources to help you decide what resources should go in each stack. Normally, you might put all of your resources in one stack. As your stack grows in scale and broadens in scope, however, managing a single stack can be cumbersome and time consuming. By grouping resources with common lifecycles and ownership, owners can make changes to their set of resources by using their own process and schedule without affecting other resources. A layered architecture organizes stacks into multiple horizontal layers that build on top of one another, where each layer has a dependency on the layer directly below it. You can have one or more stacks in each layer, but within each layer your stacks should have AWS resources with similar lifecycles and ownership.

When you organize your AWS resources based on lifecycle and ownership, you might want to build a stack that uses resources that are in another stack. You can hardcode values or use input parameters to pass resource names and IDs. However, these methods can make templates difficult to reuse or can increase the overhead to get a stack running. Instead, use cross-stack references to export resources from a stack so that other stacks can use them. Stacks can use the exported resources by calling them using the Fn::ImportValue function.

You can reuse your templates to replicate your infrastructure in multiple environments. For example, you can create environments for development, testing, and production so that you can test changes before implementing them into production. To make templates reusable, use the Parameters, Mappings, and Conditions sections so that you can customize your stacks when you create them. For example, for your development environments, you can specify a lower-cost instance type compared to your production environment, but all other configurations and settings remain the same. Creating your stacks based on the organizational structure is also useful. For example, the network architects create a template

that provisions the Amazon VPC in a standard way for the organization. By parameterizing the template, they can reuse it for different deployments.

AWS Command Line Interface (AWS CLI)

The AWS CLI can be used to automate systems operations. You can use it to manage an operational environment that is currently in production, automating the provisioning and updating of application deployments.

Generating Skeletons

Most AWS CLI commands support `--generate-cli-skeleton` and `--cli-input-json` parameters that you can use to store parameters in JSON and read them from a file instead of typing them at the command line. You can generate AWS CLI skeleton outputs in JSON that outline all of the parameters that can be specified for the operation. This is particularly useful for generating templates that are used in scripting the deployment of applications. For example, you can use it to generate a template for Amazon ECS task definitions or an AWS CloudFormation resource's Properties section.

To generate an Amazon ECS task definition template, run the following AWS CLI command.

```
aws ecs register-task-definition --generate-cli-skeleton
```

You can use this template to create your task definition, which can then be pasted into the console JSON input area or saved to a file and used with the AWS CLI `--cli-input-json` option.

This process is also useful in generating the properties attributes for an AWS CloudFormation resource. You could use the output generated in the previous command.

```
{
    "Type" : "AWS::ECS::TaskDefinition",
    "Properties" : {
        <paste-generate-cli-skeleton>
    }
}
```

This can be done for other resources in an AWS CloudFormation template, such as an Amazon EC2 instance. Run the following command and paste the output into the Properties section of the resource.

```
aws ec2 run-instances --generate-cli-skeleton
```

```
{
    "Type" : "AWS::EC2::Instance",
    "Properties" : {
        <paste-generate-cli-skeleton>
    }
}
```

Summary

In this chapter, we discussed the following deployment strategies:

- Provisioning Infrastructure
 - Automating the provisioning of building-block services with AWS Cloud deployment services
- Deploying Applications
 - Methods of deploying applications onto your infrastructure
- Configuration Management
 - Using tagging to track resources and manage infrastructure
 - Using custom variables to make deployments flexible and reusable
 - Strategies for creating AMIs
 - Configuring infrastructure for security and troubleshooting
- Scalability Capabilities
 - Including scaling capabilities for the infrastructure of an application
 - Scalability considerations during the upgrade of an application
- Continuous Deployment
 - In-place vs. replacement upgrades
 - Methods for deploying application and infrastructure upgrades

In this chapter, we discussed the following deployment services:

- AWS Elastic Beanstalk
 - Creating and managing AWS Elastic Beanstalk environments
 - Deploying and managing an application in an AWS Elastic Beanstalk environment
- Amazon ECS
 - Building an Amazon ECS cluster
 - Deploying instances used for an Amazon ECS cluster
 - Managing containers with Amazon ECS tasks and services
 - Using a repository for container images
- AWS OpsWorks Stacks
 - Creating an AWS OpsWorks stack
 - Using layers for managing Amazon EC2 instances
 - Deploying applications to a layer of your stack
- AWS CloudFormation
 - Creating, updating, and deleting an AWS CloudFormation Stack
 - Methods and considerations for updating an AWS CloudFormation Stack

- Anatomy of AWS CloudFormation templates
- Strategies and best practices for managing AWS CloudFormation Stacks and templates

It is important to understand the various capabilities of each deployment service, including what they can and cannot do. The exam will not ask you about the actual commands used in deploying applications or how to create a template, however.

Resources to Review

AWS YouTube channel: <https://www.youtube.com/user/AmazonWebServices>

AWS What's New: <https://aws.amazon.com/new>

AWS CloudFormation documentation:

<http://docs.aws.amazon.com/AWSCloudFormation/latest/UserGuide>Welcome.html>

AWS OpsWorks Stacks documentation:

<http://docs.aws.amazon.com/opsworks/latest/userguide/welcome.html>

Amazon ECS documentation:

<http://docs.aws.amazon.com/AmazonECS/latest/developerguide>Welcome.html>

Amazon ECR documentation:

<http://docs.aws.amazon.com/AmazonECR/latest/userguide/what-is-ecr.html>

AWS Elastic Beanstalk documentation:

<http://docs.aws.amazon.com/elasticbeanstalk/latest/dg>Welcome.html>

Blue/Green Deployments on AWS whitepaper:

https://d0.awsstatic.com/whitepapers/AWS_Blue_Green_Deployments.pdf

Creating an Amazon EBS-backed Linux AMI:

<http://docs.aws.amazon.com/AWSEC2/latest/UserGuide/creating-an-ami-ebs.html>

Exam Essentials

Know how to work with AWS CloudFormation Stacks. This includes creating, updating, and deleting an AWS CloudFormation Stack.

Understand the anatomy of an AWS CloudFormation template. Templates include several major sections. These sections are Metadata, Parameters, Mappings, Conditions, Resources, Transform, and Outputs. The Resources section is the only required section.

Understand an AWS CloudFormation change set. Understanding how your changes will affect running resources before you implement them can help you update stacks.

Know how to create an AWS OpsWorks stack. How to create an AWS OpsWorks stack, create layers within that stack, and deploy applications to a layer

Know how to deploy an Amazon ECS cluster. Using the Amazon ECS-optimized AMI for use in a cluster. Assigning an instance profile so that the agent can communicate with the Amazon ECS service

Understand various deployment methodologies. Understand blue/green deployments, in-place upgrades, and replacement upgrades, how they differ, and why you might use one method instead of another.

Know how to create a custom AMI. Understand why baking your own AMI can be useful in deploying applications. You could decide to create only a base image, one that includes application frameworks, or one that includes everything, including your application code. Including everything can accelerate deployments; however, that requires creating a new image with every upgrade.

Understand how to use tagging for configuring resources. Tagging can be used for targeting groups of resources to which you wish to deploy applications. It can also be useful in version control, customizing deployments, managing resources, and so on.

Know how to deploy an application with AWS Elastic Beanstalk. Understand how to create an environment and specify the type of platform it will run.

Know the AWS Elastic Beanstalk environment tiers. There is a web tier and a platform tier. If your application performs operations or workflows that take a long time to complete, you can offload those tasks to a dedicated worker environment.

Know that you can use AWS Elastic Beanstalk for blue/green deployments. AWS Elastic Beanstalk performs an in-place upgrade when you update your application versions. By performing a blue/green deployment, you deploy the new version to a separate environment and then swap CNAMEs of the two environments to redirect traffic to the new version instantly.

Understand how to use Amazon EC2 instance profiles and why they are used. An instance profile is a container for an IAM role that you can use to pass role information to an Amazon EC2 instance when the instance starts. This allows you to provide automatic key rotation without storing credentials on the instance.

Test Taking Tip

The test is exact—do not add any conditions (for example, encrypted at rest) to the questions or presented scenarios.

Exercises

By now you should have set up an account in AWS. If you haven't, now would be the time to do so. It is important to note that these exercises are in your AWS account and thus are not free.

Use the Free Tier when launching resources. The Amazon AWS Free Tier applies to participating services across the following AWS Regions: US East (Northern Virginia), US West (Oregon), US West (Northern California), Canada (Central), EU (London), EU (Ireland), EU (Frankfurt), Asia Pacific (Singapore), Asia Pacific (Tokyo), Asia Pacific (Sydney), and South America (Sao Paulo). For more information, see https://aws.amazon.com/sdm/optimization/server-side-test/free-tier/free_np/.

Remember to delete and terminate resources to minimize usage charges.

EXERCISE 8.1

Create an AWS Elastic Beanstalk Environment.

1. Sign in to the AWS Management Console.
2. Open the AWS Elastic Beanstalk Console.
3. Choose Create New Application.
4. Enter the name of the application and, optionally, a description. Click Next.
5. Choose Create New Environment from the Actions menu of the application.
6. Choose between Web Server and Worker environment types.
7. For Environment Type, choose the platform that matches the language used by your application.
8. For Application Version, choose an existing application version, upload your own, or use an Amazon S3 URL.
9. For Environment Info, set a unique environment URL. Click Next.
10. Set any additional resources needed. Click Next.
11. Set any Configuration Details, Environment Tags, and Permissions as necessary or accept the defaults. Click Next for each section of the Environment Creation wizard.
12. After reviewing the environment to be created, click Launch.

EXERCISE 8.2

Manage Application Versions with AWS Elastic Beanstalk.

1. Sign in to the AWS Management Console.
2. Open the AWS Elastic Beanstalk Console.
3. Choose an application.

EXERCISE 8.2 (*continued*)

4. In the navigation pane, choose Application Versions.
 5. Choose Upload.
 6. Enter a label for this version in the Version label field.
 7. Choose Browse to specify the location of the source bundle.
 8. Choose Upload.
 9. Select the checkbox next to the Version Label. Click Deploy.
 10. Select the Environment. Click Deploy.
-

EXERCISE 8.3**Perform a Blue/Green Deployment with AWS Elastic Beanstalk.**

1. Sign in to the AWS Management Console.
 2. Open the AWS Elastic Beanstalk Console.
 3. From the Actions menu of an existing environment, choose Clone Environment.
 4. Set the new Environment name and Environment URL. Click Clone.
 5. Deploy the new application version to the new environment as detailed in the prior steps.
 6. Test the new version on the new environment.
 7. From the new environment's dashboard, choose Actions and then choose Swap Environment URLs.
 8. From the Environment name drop-down list, select the current environment. Click Swap.
-

EXERCISE 8.4**Create an Amazon ECS Cluster.**

1. Open the Amazon ECS console.
2. In the navigation pane, choose Clusters.
3. Click Create Cluster.

4. On the Create Cluster page, enter the following values:
 - a. Cluster name: **default**
 - b. Amazon EC2 instance type: **t2.micro**
 - c. Amazon VPC: **default**
 - d. Subnets: **public subnet**
 - e. Security group: **Create a new security group**
 - f. Container instance IAM role: **Create new role**
 5. Click Create.
-

EXERCISE 8.5

Launch an Amazon EC2 Instance Optimized for Amazon ECS.

1. Open the Amazon EC2 Console.
2. From the console dashboard, choose Launch Instance.
3. On the Choose an AMI page, choose Community AMIs.
4. To use the Amazon ECS-optimized AMI, type **amazon-ecs-optimized** in the Search community AMIs field. Press the Enter key. Choose the current Amazon ECS-optimized AMI.
5. On the Choose an Instance Type page, you can select the hardware configuration of your instance. Select Next.
6. On the Configure Instance Details page, select the appropriate Amazon VPC configuration, and select an IAM role with a policy to communicate with Amazon ECS.
7. To launch into a non-default cluster, choose the Advanced Details list. Paste the following script into the User data field, replacing `your_cluster_name` with the name of your cluster.

```
#!/bin/bash  
echo ECS_CLUSTER=your_cluster_name >> /etc/ecs/ecs.config
```

8. Choose Next: Add Storage.
 9. Choose Review and Launch.
 10. On the Review Instance Launch page, under Security Groups, select an existing security group or create a new one.
 11. On the Review Instance Launch page, choose Launch.
 12. In the Select an Existing Key Pair or Create a New Key Pair dialog box, select choose an existing key pair. Click Launch Instances.
-

EXERCISE 8.6**Use Amazon ECR.**

1. Open the Amazon ECS console.
2. In the navigation pane, choose Repositories.
3. On the Repositories page, choose Create Repository.
4. For Repository name, enter a unique name for your repository. Choose Next Step.
5. On the Build, Tag, and Push Docker Image page, retrieve the Docker login command that you can use to authenticate your Docker client to your registry.
6. Use the local shell with Docker installed or connect to a container instance via Secure Shell (SSH), and run the following commands.
7. Install Git, and clone the sample application.

```
sudo yum install -y git  
git clone https://github.com/awslabs/ecs-demo-php-simple-app  
cd ecs-demo-php-simple-app
```

8. Run the Docker login command that was returned in the previous step.

```
aws ec2 get-login --region <region>
```

9. Run the docker build command to build the Docker image.

```
docker build -t <repository-name>/amazon-ecs-sample
```

10. Run the docker tag command to tag your image so that you can push it to the repository.

```
docker tag <repository-name>/amazon-ecs-sample:latest <account-number>.dkr  
.ecr.<region>.amazonaws.com/<repository-name>:latest
```

11. Run the Docker push command to push the image to the repository.
12. Return to the Amazon ECS Repositories dashboard under images. Click the refresh icon to see the image in the repository.

EXERCISE 8.7**Work with Amazon ECS Task Definitions.**

1. Open the Amazon ECS console.
2. In the navigation pane, choose Task Definitions.

3. On the Task Definitions page, choose Create New Task Definition.
4. Type **webserver** for the Task Definition Name.
5. Click Add Volume. In the Name field, type **my-volume**, and click Add.
6. Click Add Container. Enter the following values:
 - a. Container name: **httpd**
 - b. Image: **httpd**
 - c. Host port and container port: **80**
 - d. Memory limits (MB): **128**
7. In the Add Container panel under Advanced container configuration in the Storage and Logging section, enter the following values:
 - a. Source volume: **my-volume**
 - b. Container path: **/usr/local/apache2/htdocs**
8. Click Add.
9. Click Add Container again. Enter the following values:
 - a. Container Name: **busybox**
 - b. Image: **busybox**
 - c. Memory Limits (MB): **128**
 - d. Essential: **unchecked**
10. In the Add Container panel under Advanced container configuration in the Environment section, enter the following values:
 - a. Entry point: **sh, -c**
 - b. Command:

```
echo '<html><head></head><body><h1>Hello World!</h1></body></html>' > /usr/local/apache2/htdocs/index.html
```
11. In the Add Container panel under Advanced container configuration in the Storage and Logging section, enter the following values:
 - a. Source container: **httpd**
12. Review the task definition. Click Create.
13. From the Actions menu, click Run Task.
14. Accept the default by clicking Run Task.

EXERCISE 8.8**Work with Amazon ECS Services.**

1. Open the Amazon ECS console.
2. In the navigation pane, choose Clusters.
3. Select the default cluster.
4. Under the Services tab, click Create.
5. On the Create Service page, set the following values:
 - a. Task definition: **webserver:1**
 - b. Service name: **webservice**
 - c. Number of tasks: **1**
6. Click Create Service.

To update your web application, you need to create a new revision of the task definition and update the service to deploy the updated task definition. Perform the following steps to update a task and deploy it by updating the service.

1. Open the Amazon ECS console.
2. In the navigation pane, choose Task Definitions.
3. Click the webserver task definition.
4. Select the checkbox next to webserver:1.
5. Click Create New Revision.
6. Under Container Definitions, click the busybox container.
7. In Edit Container panel under the Environment section, modify the Command to add **Version 2**.

```
echo '<html><head></head><body><h1>Hello World! Version 2 </h1></body></html>' > /usr/local/apache2/htdocs/index.html
```

8. Click Update.
9. Click Create.
10. In the navigation pane, choose Clusters.
11. Select the default cluster.
12. In the Services tab, select webservice.
13. Click Update.
14. For Task Definition, select webserver:2.
15. Click Update Service.

16. Click View Service.
 17. Click the Tasks tab.
 18. Click the refresh icon until the running task shows that webserver:2 is running.
-

EXERCISE 8.9**Create an AWS OpsWorks Stack.**

1. Open the AWS OpsWorks console.
 2. Click Go to AWS OpsWorks Stacks.
 3. On the AWS OpsWorks Stacks page at the top of the screen, click Select Stack, and then click Add Stack.
 4. Select Chef 12 stack.
 5. In the Stack name box, type a name.
 6. Select the Region to which you want the AWS OpsWorks stack deployed.
 7. For Amazon VPC, choose the default Amazon VPC.
 8. For Use Custom Chef cookbooks, choose Yes.
 9. For Repository type, choose Http Archive.
 10. For Repository URL, type **<https://s3.amazonaws.com/opsworks-demo-assets/opsworks-linux-demo-cookbooks-nodejs.tar.gz>**
 11. Choose Advanced.
 12. For IAM role, select aws-opsworks-service-role if available. Otherwise choose New IAM role.
 13. For Default IAM instance profile, select aws-opsworks-ec2-role if available. Otherwise choose New IAM instance profile.
 14. For API endpoint region, choose the regional API endpoint.
 15. Click Add Stack.
-

EXERCISE 8.10**Make a Layer in AWS OpsWorks Stacks.**

1. Open the AWS OpsWorks console.
2. Click Go to AWS OpsWorks Stacks.
3. Click the name of your stack.

EXERCISE 8.10 (continued)

4. In the left navigation pane, click Layers.
 5. For Layers, choose Add Layer.
 6. Type a Name and a Short name.
 7. Choose Add Layer.
 8. On the Layers page, for the layer you create, choose Network.
 9. On the Network tab, under Automatically Assign IP Addresses, verify that Public IP addresses is set to yes.
 10. On the Layers page, choose Security.
 11. Click Edit. In the Security groups drop-down, select AWS-OpsWorks-WebApp.
 12. Click Save.
-

EXERCISE 8.11**Add an Amazon EC2 Instance to an AWS OpsWorks Stacks Layer.**

1. Open the AWS OpsWorks console.
 2. Click Go to AWS OpsWorks Stacks.
 3. Click the name of your stack.
 4. In the left navigation pane, click Layers.
 5. To the right of the Layer, click Add Instance.
 6. For Size, select t2.micro.
 7. Click Advanced. Choose Ubuntu 14.04 LTS for Operating System.
 8. Choose Add Instance.
 9. On the Instances page, next to the instance and under actions, click start.
 10. After a few minutes, the status will change to online.
-

EXERCISE 8.12**Add an Application to AWS OpsWorks Stacks.**

1. Open the AWS OpsWorks console.
2. Click Go to AWS OpsWorks Stacks.
3. Click the name of your stack.

4. In the left navigation pane, click Apps.
 5. Choose Add App. The Add App page is displayed.
 6. For Settings, provide a Name.
 7. For Application Source, Repository URL, type:
<https://github.com/awslabs/opsworks-windows-demo-nodejs.git>
 8. Accept the remaining defaults. Click Add App.
 9. In the navigation pane, choose Layers. The Layers page is displayed.
 10. For the Layer, choose Recipes.
 11. For Custom Chef Recipes, for Deploy, type **nodejs_demo::default**. Press the Enter key.
 12. Choose Save.
 13. In the service navigation pane, choose Apps.
 14. For your App, under Actions, click deploy.
 15. Accept the defaults. Click Deploy.
-

EXERCISE 8.13

Create an AWS CloudFormation Stack.

1. Open the AWS CloudFormation console.
2. Select Create Stack.
3. Choose Select a sample template from the drop-down menu.
4. Provide a unique stack name.
5. Fill in the required parameters, and click Next.
6. On the Options page, click Next.
7. Review the stack to be created. Click Create.

EXERCISE 8.14

Delete an AWS CloudFormation Stack.

1. Open the AWS CloudFormation console.
 2. From the list of stacks in the AWS CloudFormation console, select the stack that you want to delete.
 3. Choose Actions and then Delete Stack.
 4. Click Yes, Delete when prompted.
-

Review Questions

1. Your company uses Chef for its automation. What AWS service can run Chef recipes?
 - A. AWS Elastic Beanstalk
 - B. AWS OpsWorks
 - C. AWS CodeDeploy
 - D. AWS CodeCommit
2. Your company is migrating to AWS and wants to apply DevOps techniques to automate the deployment of new environments. What tool should they use?
 - A. AWS CloudFormation
 - B. AWS Config
 - C. AWS Chef
 - D. AWS Shield
3. AWS CloudFormation template requires which of the following sections?
 - A. Format Version
 - B. Description
 - C. Metadata
 - D. Parameters
 - E. Resources
4. AWS CloudFormation supports the following text formats: (Choose two.)
 - A. YAML
 - B. JSON
 - C. RAW
 - D. XML
5. Which of the following languages does AWS Elastic Beanstalk NOT support?
 - A. Go
 - B. C
 - C. Java
 - D. PHP
6. You can deploy Docker containers on the following: (Choose three.)
 - A. Amazon EC2
 - B. Amazon ECS
 - C. AWS Lambda
 - D. AWS Elastic Beanstalk

7. AWS Elastic Beanstalk supports the following types of deployment methods. (Choose three.)
 - A. Blue/green
 - B. Rolling
 - C. AWS CloudFormation conditional
 - D. AWS OpsWorks layer update
 - E. Immutable
8. Your company has not approved the use of AWS Elastic Beanstalk. You would like to do blue/green deployments on your AWS environments. What tool can be used to automate your deployment and perform a blue/green deployment?
 - A. This is not possible without AWS Elastic Beanstalk.
 - B. AWS CloudFormation
 - C. AWS Datapipeline
 - D. AWS CodeCommit
9. What are the two methods for updating an AWS CloudFormation Stack? (Choose two.)
 - A. Direct update
 - B. Indirect update
 - C. Rolling update
 - D. Creation of change sets
10. You are creating an AWS CloudFormation template and you want to use it for multiple regions. What optional section of an AWS CloudFormation Stack needs to exist?
 - A. Description
 - B. Resources
 - C. Conditions
 - D. Mappings
11. Which of the following statements are true? (Choose two.)
 - A. You can customize your application deployments on AWS using the Java programming language in AWS CloudFormation.
 - B. You can customize your application deployments on AWS using JSON templates in AWS CloudFormation.
 - C. You can customize your application deployments on AWS using JSON templates in AWS OpsWorks.
 - D. You can customize your AWS deployments on AWS using the Java programming language with AWS OpsWorks templates.

- 12.** Your developers require a service that easily provisions an environment and then deploys and runs applications in the AWS cloud. The service needs to be integrated with developer tools and provides a one-stop experience for you to manage the lifecycle of your applications. What service would provide this capability without needing to write code specifically to deploy the infrastructure?
- A. AWS OpsWorks Stacks
 - B. AWS CloudFormation
 - C. AWS Elastic Beanstalk
 - D. Amazon EC2 Container Service
- 13.** True or False. Your application deployment requires an Elastic Load Balancer, Amazon EC2 Instances, Amazon RDS Instance, and an Amazon DynamoDB table. All of these services are integrated and deployable with AWS OpsWorks Stacks.
- A. True
 - B. False
- 14.** You need to make changes to an existing AWS CloudFormation Stack's settings or change its resources. Instead of deleting it and creating a new stack, what two methods can be used to update the stack? (Choose two.)
- A. Directly update a stack by submitting changes, specifying new input parameter values or an updated template.
 - B. Execute a script to modify the resources manually.
 - C. Create and execute a change set so that you can preview the changes.
 - D. Update the configuration of the resource directly within the given services instead of within AWS CloudFormation.
- 15.** You need to deploy a new version of your application to a different environment and then perform a CNAME swap to redirect traffic with zero downtime. This blue/green deployment needs to be automated without the complexity of building a solution yourself. What service supports this capability?
- A. AWS OpsWorks Stacks
 - B. AWS CloudFormation
 - C. AWS Elastic Beanstalk
 - D. Amazon EC2 Container Service
- 16.** The developers within your organization have determined that Docker containers will be used for application deployment. What AWS managed service can run containers and provide auto scaling of containers based on their utilization?
- A. AWS Elastic Beanstalk
 - B. AWS OpsWorks
 - C. Amazon EC2 Container Service
 - D. Amazon Elastic Compute Cloud

- 17.** An AWS CloudFormation template was used to deploy the resources used for your application. The resources are no longer needed, however you need to keep the data for your Amazon RDS instance. How can you clean up the resources while preventing the database from being deleted?
- A.** Manually delete all resources except the Amazon RDS instance.
 - B.** Update the stack directly, and remove all of the resources from the template except the Amazon RDS instance.
 - C.** Set a deletion policy attribute for the Amazon RDS instance resource, specifying to retain on delete and to create a snapshot.
 - D.** It is not possible to retain resources when deleting an AWS CloudFormation Stack.
- 18.** You're working on creating an AWS CloudFormation template based on the specification provided by the applications architect. It requires an Amazon DynamoDB table be provisioned which an Amazon EC2 instance will utilize. Both the DynamoDB table and EC2 instance should be provisioned with CloudFormation. How do you tell CloudFormation to create the DynamoDB table and wait for it to be completed before creating the EC2 instance that will need to connect to it?
- A.** Create multiple stacks and a script that waits for one stack to be completed before creating the next.
 - B.** Add a DependsOn attribute to the Amazon EC2 instance resource, and provide the Amazon DynamoDB resource's logical name.
 - C.** Add a user data script to the instance that polls for the Amazon DynamoDB table.
 - D.** Order the resources in the template so that the Amazon EC2 instance resource is listed after the Amazon DynamoDB table.
- 19.** An Amazon EC2 instance is being created with an AWS CloudFormation template. It needs to have permissions to an Amazon S3 bucket for storing data persistently outside of the instance. How do you provide the instance permissions to the Amazon S3 bucket?
- A.** Save the access key and secret access key in the user data script section of your CloudFormation template.
 - B.** Store credentials in Amazon S3, and when creating the AWS CloudFormation Stack specify a parameter with the location of the credentials.
 - C.** Add an IAM Instance Profile to the AWS CloudFormation template that references a role with the appropriate policy attached, giving it allow permissions to the bucket. Reference the instance profile in the Amazon EC2 instance resource.
 - D.** Create a script that retrieves the keys and stores them on the instance.
- 20.** Your organization would like to implement a DevOps workflow. Application deployments will be frequent, so it was decided to reuse resources to minimize costs. What two services support an in-place upgrade of applications to existing resources?
- A.** AWS OpsWorks
 - B.** AWS CloudFormation
 - C.** AWS Elastic Beanstalk
 - D.** AWS CodeCommit

Chapter 9



Monitoring and Metrics

THE AWS CERTIFIED SYSOPS ADMINISTRATOR - ASSOCIATE EXAM TOPICS COVERED IN THIS CHAPTER MAY INCLUDE, BUT ARE NOT LIMITED TO, THE FOLLOWING:

Domain 1.0 Monitoring and Metrics

- ✓ 1.1 Demonstrate ability to monitor availability and performance
- ✓ 1.2 Demonstrate ability to monitor and manage billing and cost optimization processes

Domain 3.0 Analysis

- ✓ 3.1 Optimize the environment to ensure maximum performance
- ✓ 3.2 Identify performance bottlenecks and implement remedies
- ✓ 3.3 Identify potential issues on a given application deployment

Domain 6.0 Security

- ✓ 6.2 Ensure data integrity and access controls when using the AWS platform
- ✓ 6.4 Demonstrate ability to prepare for security assessment use of AWS



Introduction to Monitoring and Metrics

This chapter covers the collection of metrics from Amazon CloudWatch which, in conjunction with other AWS Cloud services, can assist in the management, deployment, and optimization of workloads in the cloud.

In addition to performance monitoring, services such as Amazon CloudWatch Logs, AWS Config, AWS Trusted Advisor, and AWS CloudTrail can provide a detailed inventory of provisioned resources for security audits and financial accounting.

Amazon CloudWatch was designed to monitor cloud-based computing. Even so, when using Amazon CloudWatch Logs, systems in an existing, on-premises datacenter can send log information to Amazon CloudWatch for monitoring.

Sometimes, it is important to monitor and view the health of AWS in general. To do this, there are two tools: the AWS Service Health Dashboard and the AWS Personal Health Dashboard. These services display the general status of AWS and provide a personalized view of the performance and availability of provisioned resources.

This chapter covers these topics separately and also highlights how they can work together to maintain a robust environment on AWS.

An Overview of Monitoring

Everything fails, all the time.

—Werner Vogels

Computing systems are incredibly complex. Effectively troubleshooting them requires data that is easily understood to be delivered in real time. Service Level Agreements (SLAs) often require high levels of availability, and a lack of meaningful information can lead to loss of time and revenue.

Why Monitor?

Monitoring provides several major benefits:

- Monitoring enables systems operators to catch issues before they become problems. This, in turn, maintains high availability and delivers high-quality customer service.

- Monitoring provides tools for making informed decisions about capacity planning.
- Monitoring is an input mechanism for automation.
- Monitoring provides visibility into the cost, utilization, and security of computing resources.

Monitoring is the process of observing and recording resource utilization in real time. *Alarms* are notifications based on this information in response to a predefined condition. Frequently, this condition involves failure. Alarms can also be configured to send notifications when resources are being underutilized and money is being wasted.

Traditional monitoring tools have been designed around on-premises datacenters with the idea that servers are going to be in place for an extended amount of time. Because of this, these tools have difficulty distinguishing between an Amazon Elastic Compute Cloud (Amazon EC2) instance that has failed and one that was terminated purposely. AWS created its own monitoring service, Amazon CloudWatch, that is integrated with other AWS Cloud services and uses AWS Identity and Access Management (IAM) to keep monitoring data secure.

Amazon CloudWatch

Amazon CloudWatch is a service that monitors the health and status of AWS resources in real time. It provides system-wide visibility into resource utilization, application performance, and operational health by tracking, measuring, reporting, alerting, and reacting to events that occur in an environment.

Amazon CloudWatch Logs collects and monitors log files, can set alarms, and automatically reacts to changes in AWS resources. Logs can be monitored in real time or stored for analysis.

Amazon CloudWatch Alarms monitor a single metric and perform one or more actions based on customer-defined criteria.

Amazon CloudWatch Events delivers a near real-time stream of system events that describe changes in AWS resources. These streams are delivered to Amazon EC2 instances, AWS Lambda functions, Amazon Kinesis Streams, Amazon EC2 Container Service (Amazon ECS) tasks, AWS Step Functions state machines, Amazon Simple Notification Service (Amazon SNS) topics, Amazon Simple Queue Service (Amazon SQS) queues, or built-in targets.

AWS CloudTrail

AWS CloudTrail monitors calls made to the Amazon CloudWatch Events Application Programming Interface (API) for an account. This includes calls made by the AWS Management Console, the AWS Command Line Interface (AWS CLI), and other AWS Cloud services. When AWS CloudTrail logging is turned on, Amazon CloudWatch Events writes log files to an Amazon Simple Storage Service (Amazon S3) bucket.

AWS Config

AWS Config provides a detailed view of the configuration of AWS resources in an AWS account, including how the resources are related to one another. It also provides historical

information to show how configurations and relationships have changed over time. Related to monitoring, AWS Config allows customers to create rules that check the configuration of their AWS resources and check for compliance with an organization's policies. When an AWS Config rule is triggered, it generates an event that can be captured by Amazon CloudWatch Events.

Amazon CloudWatch can monitor AWS resources, such as Amazon EC2 instances, Amazon DynamoDB tables, Amazon Relational Database Service (Amazon RDS) DB instances, custom metrics generated by applications and services, and log files generated by applications and operating systems.

AWS Trusted Advisor

AWS Trusted Advisor is an online resource designed to help reduce cost, increase performance, and improve security by optimizing an AWS environment. It provides real-time guidance to help provision resources following AWS best practices.

AWS Trusted Advisor checks for best practices in four categories:

- Cost Optimization
- Security
- Fault Tolerance
- Performance Improvement

The following four AWS Trusted Advisor checks are available at no charge to all AWS customers to help improve security and performance:

- Service Limits
- Security Groups – Specific Ports Unrestricted
- IAM Use
- Multi-Factor Authentication (MFA) on the Root Account



With a Business or Enterprise support plan, it is possible to retrieve, refresh, and report on AWS Trusted Advisor results using the AWS Support API.

The AWS CLI, the AWS Tools for Windows PowerShell, and the AWS Software Development Kits (SDKs) include support for the AWS Support API.

AWS Service Health Dashboard

The *AWS Service Health Dashboard* provides access to current status and historical data about every AWS Cloud service. If there's a problem with a service, it is possible to expand the appropriate line in the details section to get more information.

In addition to the dashboard, it is also possible to subscribe to the RSS feed for any service.

For anyone experiencing a real-time operational issue with one of the AWS Cloud services currently reporting as being healthy, there is a Contact Us link at the top of the page to report an issue.

The AWS Service Health Dashboard is available at <http://status.aws.amazon.com/>.



All dates and times in the AWS Service Health Dashboard are in Pacific Time (PST/PDT).

AWS Personal Health Dashboard

The *AWS Personal Health Dashboard* provides alerts and remediation guidance when AWS is experiencing events that impact customers. While the AWS Service Health Dashboard displays the general status of AWS Cloud services, the AWS Personal Health Dashboard provides a personalized view into the performance and availability of the AWS Cloud services underlying provisioned AWS resources.

The dashboard displays relevant and timely information to help manage events in progress and provides proactive notification to help plan scheduled activities. Alerts are automatically triggered by changes in the health of AWS resources and provide event visibility and also guidance to help diagnose and resolve issues quickly.

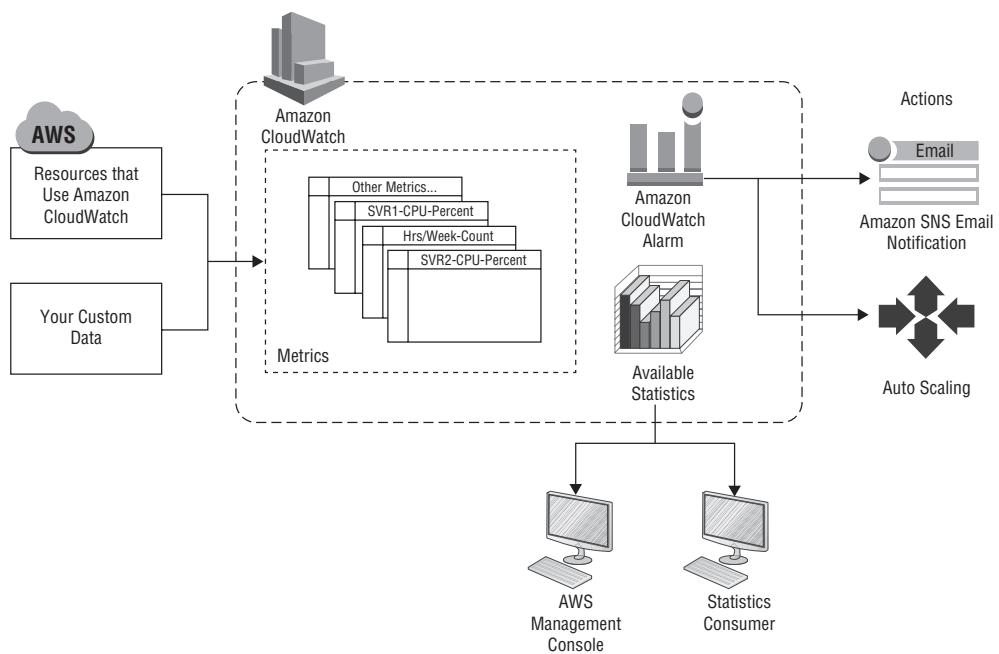
Now let's take a deep dive into each of these services.

Amazon CloudWatch

Amazon CloudWatch monitors, in real time, AWS resources and applications running on AWS. Amazon CloudWatch is used to collect and track metrics, which are variables used to measure resources and applications. Amazon CloudWatch Alarms send notifications and can automatically make changes to the resources being monitored based on user-defined rules. Amazon CloudWatch is basically a metrics repository. An AWS product such as Amazon EC2 puts metrics into the repository and customers retrieve statistics based on those metrics. Additionally, custom metrics can be placed into Amazon CloudWatch for reporting and statistical analysis.

For example, it is possible to monitor the CPU usage and disk reads and writes of Amazon EC2 instances. With this information, it is possible to determine when additional instances should be launched to handle the increased load. Additionally, these new instances can be launched automatically before there is a problem, eliminating the need for human intervention. Conversely, monitoring data can be used to stop underutilized instances automatically in order to save money.

In addition to monitoring the built-in metrics that come with AWS, it is possible to create, monitor, and trigger actions using custom metrics. Amazon CloudWatch provides system-wide visibility into resource utilization, application performance, and operational health. Figure 9.1 illustrates how Amazon CloudWatch connects to both AWS and on-premises environments.

FIGURE 9.1 Amazon CloudWatch integration

Amazon CloudWatch can be accessed using the following methods:

- Amazon CloudWatch console: <https://console.aws.amazon.com/cloudwatch/>
- The AWS CLI
- The Amazon CloudWatch API
- AWS SDKs

Amazon CloudWatch Services

The following services are used in conjunction with Amazon CloudWatch:

Amazon SNS Amazon SNS coordinates and manages the delivery or sending of messages to subscribing endpoints or clients. Use Amazon SNS with Amazon CloudWatch to send messages when an alarm threshold has been reached.

Auto Scaling Auto Scaling automatically launches or terminates Amazon EC2 instances based on user-defined policies, health status checks, and schedules. Use an Amazon CloudWatch Alarm with Auto Scaling to scale Amazon EC2 instances in or out based on demand.

AWS CloudTrail AWS CloudTrail can monitor calls made to the Amazon CloudWatch API for an account. It includes calls made by the AWS Management Console, AWS CLI, and other

AWS Cloud services. When AWS CloudTrail logging is turned on, Amazon CloudWatch writes log files to an Amazon S3 bucket that is specified when AWS CloudTrail is configured.

IAM IAM manages the authentication and authorization for AWS resources such as Amazon CloudWatch and its collected metrics.

Metrics

At the core of Amazon CloudWatch are *metrics*, which are time-ordered sets of data points that contain information about the performance of resources. By default, several services provide some metrics at no additional charge. These metrics include information from Amazon EC2 instances, Amazon Elastic Block Store (Amazon EBS) volumes, and Amazon RDS DB instances. It is also possible to enable detailed monitoring for some resources, such as Amazon EC2 instances.



Amazon CloudWatch has a free tier, and many applications can operate within the free tier limits. The AWS Free Tier applies to participating services across the following AWS Regions: US East (Northern Virginia), US West (Oregon), US West (Northern California), Canada (Central), EU (London), EU (Ireland), EU (Frankfurt), Asia Pacific (Singapore), Asia Pacific (Tokyo), Asia Pacific (Sydney), and South America (Sao Paulo).

- New and existing customers get three dashboards of up to 50 metrics each per month with no additional charge.
- Basic monitoring metrics for Amazon EC2 instances are available at no additional charge.
- All metrics for Amazon EBS volumes, Elastic Load Balancing load balancers, and Amazon RDS DB instances are available at no additional charge.
- Customers can use 10 metrics (applicable to detailed monitoring for Amazon EC2 instances, custom metrics, or Amazon CloudWatch Logs), 10 alarms, and 1 million API requests each month with no additional charges.
- Customers receive 5 GB of data ingestion and 5 GB of archived storage per month with no additional charge.



Metrics exist only in the AWS Region in which they were created.

Custom Metrics

In addition to monitoring AWS resources, Amazon CloudWatch can be used to monitor data produced from applications, scripts, and services. A *custom metric* is any metric provided to Amazon CloudWatch via an agent or an API.

Custom metrics can be used to monitor the time it takes to load a web page, capture request error rates, monitor the number of processes or threads on an instance, or track the amount of work performed by an application.

Ways to create custom metrics include the following:

- The PutMetricData API.
- AWS-provided sample monitoring scripts for Windows and Linux.
- The Amazon CloudWatch *collectd* plugin.
- Applications and tools offered by AWS Partner Network (APN) partners.

Custom metrics come at an additional cost based on usage.



The Amazon CloudWatch collectd plugin is a publishing extension for *collectd*, an open-source statistic gathering daemon. All configured collectd metrics are automatically published to Amazon CloudWatch and allow for the monitoring of servers and applications within and outside of Amazon EC2 instances.

It also provides additional functionality when running on Amazon EC2 instances. These features include the automatic discovery of instance ID and AWS Region.

Amazon CloudWatch Metrics Retention

In November 2016, Amazon CloudWatch changed the length of time metrics are stored inside the service as follows:

- One-minute data points are available for 15 days.
- Five-minute data points are available for 63 days.
- One-hour data points are available for 455 days.

If metrics need to be available for longer than those periods, they can be archived using the GetMetricStatistics API call.

Metrics cannot be deleted. They automatically expire after 15 months if no new data is published to them.



The maximum number of data points returned from a single call is 1,440. If more than 1,440 data points are requested, Amazon CloudWatch returns an error.

To reduce the number of data points, narrow the specified time range, make multiple requests across adjacent time ranges, or increase the specified period.

A period can be as short as 60 seconds.



Data points are not returned from Amazon CloudWatch in chronological order.

Namespaces

A *namespace* is a container for a collection of Amazon CloudWatch metrics. Each namespace is isolated from other namespaces. This isolation ensures that data collected is only from services specified and prevents different applications from mistakenly aggregating the same statistics.

There are no default namespaces. When creating a custom metric, a namespace is required. If the specified namespace does not exist, Amazon CloudWatch will create it.

Namespace names must contain valid XML characters and be fewer than 256 characters in length.

Allowed characters in namespace names are as follows:

- Alphanumeric characters (0-9, A-Z, a-z)
- Period (.)
- Hyphen (-)
- Underscore (_)
- Forward Slash (/)
- Hash (#)
- Colon (:)

AWS namespaces use the following naming convention: AWS/service. For example, Amazon EC2 uses the AWS/EC2 namespace. Sample AWS Namespaces are shown in Table 9.1.



The AWS/ namespace is reserved and cannot be used by customers.

TABLE 9.1 A Small Sample of AWS Namespaces

| AWS Product | Namespace |
|------------------------|--|
| Auto Scaling | AWS/AutoScaling |
| Amazon EC2 | AWS/EC2 |
| Amazon EBS | AWS/EBS |
| Elastic Load Balancing | AWS/ELB (Classic Load Balancers) |
| Elastic Load Balancing | AWS/ApplicationELB (Application Load Balancers) |

For a comprehensive list of namespaces, see <http://docs.aws.amazon.com/AmazonCloudWatch/latest/monitoring/aws-namespaces.html>.

Dimensions

A *dimension* is a name/value pair that uniquely identifies a metric and further clarifies the metric data stored. A metric can have up to 10 dimensions.

Every metric has specific characteristics that describe it. Think of dimensions as categories or metadata for those characteristics. The categories can aid in the design of a structure for a statistics plan. Because dimensions are part of the unique identifier for a metric, whenever a unique name/value pair is added to a metric, a new metric is created.



AWS Cloud services that send data to Amazon CloudWatch attach dimensions to each metric.

Dimensions can be used to filter the results from Amazon CloudWatch. For example, it is possible to get statistics for a specific Amazon EC2 instance by specifying the instance ID dimension when doing a search.



When selecting a metric without specifying a dimension, Amazon CloudWatch will aggregate all data for the specified metric to create the requested statistic.

This does not work for custom metrics.

Dimension Combinations

Amazon CloudWatch treats each unique combination of dimensions as a separate metric, even if the metrics use the same metric name. It is not possible to retrieve statistics using combinations of dimensions that have not been specifically published. When retrieving statistics, specify the same values for the namespace, metric name, and dimension parameters that were used when the metrics were created. The start and end times can be specified for Amazon CloudWatch to use for aggregation.

To illustrate, here are four distinct metrics named `ServerStats` in the `DataCenterMetric` namespace that have the following properties:

```
Dimensions: Server=Prod, Domain=Titusville, Unit: Count, Timestamp: 2017-05-18T12:30:00Z, Value: 105
```

```
Dimensions: Server=Test, Domain=Titusville, Unit: Count, Timestamp: 2017-05-18T12:31:00Z, Value: 115
```

```
Dimensions: Server=Prod, Domain=Rockets, Unit: Count, Timestamp: 2017-05-18T12:32:00Z, Value: 95
```

```
Dimensions: Server=Test, Domain=Rockets, Unit: Count, Timestamp: 2017-05-18T12:33:00Z, Value: 97
```

If those four metrics are the only ones that have been published, statistics can be retrieved for these combinations of dimensions:

- `Server=Prod,Domain=Titusville`
- `Server=Prod,Domain=Rockets`

- Server=Test,Domain=Titusville
- Server=Test,Domain=Rockets

It is not possible to receive statistics for the following dimensions. In this case, dimensions must be specified to retrieve statistics:

- Server=Prod
- Server=Test
- Domain=Titusville
- Domain=Rockets



If no dimensions are specified, no statistics will be returned.

Statistics

Statistics are metric data aggregations over specified periods of time. Amazon CloudWatch provides statistics based on the metric data points provided by custom data or by other services in AWS to Amazon CloudWatch. Aggregations are made using the namespace, metric name, dimensions, and the data point unit of measure within the time period specified. Available CloudWatch statistics are provided in Table 9.2.

TABLE 9.2 Available CloudWatch Statistics

| Statistic | Description |
|-------------|---|
| Minimum | The lowest value observed during the specified period. Use this value to determine low volumes of activity for an application. |
| Maximum | The highest value observed during the specified period. Use this value to determine high volumes of activity for an application. |
| Sum | All values submitted for the matching metric added together. This statistic can be useful for determining the total volume of a metric. |
| Average | The value of Sum/SampleCount during the specified period. By comparing this statistic with Minimum and Maximum, the full scope of a metric can be determined, and it is possible to discover how close average use is to the Minimum and Maximum. |
| SampleCount | The number of data points used for the statistical calculation. |
| pNN.NN | The value of the specified percentile. You can specify any percentile using up to two decimal places. |



The Amazon CloudWatch statistic Average, when compared with Minimum and Maximum, helps gauge when AWS resources need to be scaled.

Pre-calculated statistics can be added to Amazon CloudWatch. Instead of data point values, specify values for SampleCount, Minimum, Maximum, and Sum. Amazon CloudWatch calculates the average for you. Values added in this way are aggregated with any other values associated with the matching metric.

Units

Each statistic has a unit of measure. Example units include bytes, seconds, count, and percent.

A unit can be specified when creating a custom metric. If one is not specified, Amazon CloudWatch uses None as the unit. Units provide conceptual meaning to data.



Though Amazon CloudWatch attaches no significance to a unit internally, other applications can derive semantic information based on the unit. Put another way, by defining a unit, other applications can use the data properly.

Metric data points that specify a unit of measure are aggregated separately. When getting statistics without specifying a unit, Amazon CloudWatch aggregates all data points of the same unit together. If there are two identical metrics that have different units, two separate data streams are returned—one for each unit.

Periods

A *period* is the length of time associated with a specific Amazon CloudWatch statistic. Each statistic represents an aggregation of the metrics data collected for a specified period of time. Although periods are expressed in seconds, the minimum granularity for a period is one minute.

Because of this minimum granularity, period values are expressed as multiples of 60. By varying the length of the period, the data aggregation can be adjusted.



A period can be as short as one minute or as long as one day. One minute is expressed as 60 seconds. One day is 86,400 seconds long. The default value is 60 seconds.

When retrieving statistics, specify a period, start time, and an end time. These parameters determine the overall length of time associated with the collected statistic.

Default values for the start time and end time return statistics from the past hour.

The values specified for the start time and end time determine how many periods Amazon CloudWatch will return.



Retrieving statistics using the default values for the period, start time, and end time will return an aggregated set of statistics for each minute of the previous hour.

- For statistics aggregated in 10-minute blocks, specify a period of 600.
- For statistics aggregated over the entire hour, specify a period of 3,600.

Periods are also important for Amazon CloudWatch Alarms. When creating an alarm to monitor a specific metric, Amazon CloudWatch will compare that metric to a specified threshold value. Customers have extensive control over how Amazon CloudWatch makes comparisons. In addition to the period length, the number of evaluation periods can be specified as well. For example, if three evaluation periods are specified, Amazon CloudWatch compares a window of three data points. Amazon CloudWatch only sends a notification if the oldest data point is breaching and the others are breaching or are missing.



For metrics that are continuously emitted, Amazon CloudWatch sends a notification until three failures are found.

Aggregation

Amazon CloudWatch aggregates statistics according to the period length specified when retrieving statistics. When publishing as multiple data points with the same or similar timestamps, Amazon CloudWatch aggregates them by period length.



Aggregated statistics are only available when using detailed monitoring.

Amazon CloudWatch does not aggregate data across regions.

Data points for a metric that share the same timestamp, namespace, and dimension can be published, and Amazon CloudWatch will return aggregated statistics for them. It is also possible to publish multiple data points for the same or different metrics with any timestamp.

For large datasets, a pre-aggregated dataset can be inserted called a *statistic set*. With statistic sets, give Amazon CloudWatch the Min, Max, Sum, and SampleCount for a number of data points. This is commonly used data that needs to be collected many times in a minute.



Consider a metric for the request latency of a web page. It doesn't make sense to publish metric data for every web page hit. A better option is to collect the latency of all hits to that web page, aggregate them once a minute, and send this statistic set to Amazon CloudWatch.

Amazon CloudWatch doesn't differentiate the source of a metric. If a metric is published with the same namespace and dimensions from different sources, Amazon CloudWatch treats it as a single metric. This can be useful for service metrics in a distributed, scaled system.



If all the hosts in a web server application publish identical metrics representing the latency of requests they are processing, Amazon CloudWatch would treat them as a single metric. This would provide statistics for minimum, maximum, average, and sum of all the requests across an application.

Dashboards

Amazon CloudWatch dashboards are customizable pages in the Amazon CloudWatch console that can be used to monitor resources in a single view. Monitored resources can be in a single region or multiple regions. Use Amazon CloudWatch dashboards to create customized views of the metrics and alarms for AWS resources.

With dashboards, it is possible to create the following:

- A single view for selected metrics and alarms to help assess the health of resources and applications across one or more regions
- An operational playbook that provides guidance for team members during operational events about how to respond to specific incidents
- A common view of critical resource and application measurements that can be shared by team members for faster communication flow during operational events

Percentiles

A percentile indicates the relative standing of a value in a dataset and is often used to isolate anomalies. For example, the 95th percentile means that 95 percent of the data is below this value and 5 percent of the data is above this value. Percentiles help in a better understanding of the distribution of metric data.

Percentiles can be used with the following services:

Amazon EC2

Amazon RDS

Amazon Kinesis
Application Load Balancer
Elastic Load Balancing
Amazon API Gateway



NOTE When monitoring, using an average value can hide events that are drastically different than normal ones. When monitoring a maximum value, however, a single, random spike can skew the results. To avoid this, monitor the 95th percentile of CPU utilization to check for instances with an unusually heavy load.

Monitoring Baselines

Monitoring is an important part of maintaining the reliability, availability, and performance of Amazon EC2 instances. Collect monitoring data from all of the parts of an AWS solution to be able to debug any multi-point failures easily.

In order to monitor an environment effectively, have a plan that answers the following questions:

- What are the goals for monitoring?
- What resources need to be monitored?
- How often will these resources be monitored?
- What monitoring tools will be used?
- Who will perform the monitoring tasks?
- Who should be notified when something goes wrong?

After the monitoring goals have been defined and the monitoring plan has been created, the next step is to establish a baseline for normal Amazon EC2 performance.

Measure Amazon EC2 performance at various times and under different load conditions. While monitoring Amazon EC2, store a history of collected monitoring data. Over time, the historical data can be compared to current data to identify normal performance patterns and also anomalies.

For example, if monitoring CPU utilization, disk I/O, and network utilization for your Amazon EC2 instances and performance falls outside of an established baseline, reconfigure or optimize the instance to reduce CPU utilization, improve disk I/O, or reduce network traffic (see Table 9.3).

Amazon EC2 Status Checks

Inside Amazon EC2, there are two types of status checks: a system status check and an instance status check.

TABLE 9.3 Establishing an Amazon EC2 Baseline

| Item to Monitor | Amazon EC2 Metric |
|--|---------------------------------|
| CPU utilization | CPUUtilization |
| Memory utilization | Requires an agent |
| Memory used | Requires an agent |
| Memory available | Requires an agent |
| Network utilization | NetworkIn NetworkOut |
| Disk performance | DiskReadOps DiskWriteOps |
| Disk Swap utilization (Linux instances) | Requires an agent |
| Swap used (Linux instances) | Requires an agent |
| Page File utilization (Windows instances only) | Requires an agent |
| Page File used (Windows instances only) | Requires an agent |
| Page File available (Windows instances only) | Requires an agent |
| Disk Reads/Writes | DiskReadBytes DiskWriteBytes |
| Disk Space utilization (Linux instances) | Requires an agent |
| Disk Space used (Linux instances) | Requires an agent |
| Disk Space available (Linux instances only) | Requires an agent |

System Status Checks

System status checks monitor the AWS systems required to use your instance in order to ensure that they are working properly. These checks detect problems on the hardware that an instance is using.

When a system status check fails, there are three possible courses of action. One option is to wait for AWS to fix the issue. If an instance boots from an Amazon EBS volume,

stopping and starting the instance will move it to new hardware. If the Amazon EC2 instance is using an instance store volume, terminate it, and start a new one to put it on new hardware.

The following are examples of problems that can cause system status checks to fail:

- Loss of network connectivity
- Loss of system power
- Software issues on the physical host
- Hardware issues on the physical host that impact network reachability

Instance Status Checks

Instance status checks monitor the software and network configuration of individual instances. These checks detect problems that require user involvement to repair. When an instance status check fails, it can often be fixed with a reboot or by reconfiguring the Amazon EC2 instance.

The following are examples of problems that can cause instance status checks to fail:

- Failed system status checks
- Incorrect networking or startup configuration
- Exhausted memory
- Corrupted file system
- Incompatible kernel



To view the status checks for running Amazon EC2 instances, use the `describe-instance-status` AWS CLI command.

To view the status of all instances:

```
aws ec2 describe-instance-status
```

To get the status of all instances with an instance status of impaired:

```
aws ec2 describe-instance-status --filters Name=instance-status
                                .status,Values=impaired
```

To get the status of a single instance, use the following command with a known instance ID:

```
aws ec2 describe-instance-status --instance-ids i-1234567890abcdef0
```

Authentication and Access Control

Access to Amazon CloudWatch requires credentials. These credentials must have permissions to access AWS resources such as the Amazon CloudWatch console or retrieving Amazon CloudWatch metric data.

Every AWS resource is owned by an AWS account, and permissions to create or access a resource are governed by permissions policies. An account administrator can attach permissions policies to IAM identities: users, groups, and roles.

Permissions Required to Use the Amazon CloudWatch Console

For a user to work with the Amazon CloudWatch console, there is a minimum set of permissions required to allow that user to describe other AWS resources in the AWS account.

The Amazon CloudWatch management console requires permissions from the following services:

- Auto Scaling
- AWS CloudTrail
- Amazon CloudWatch
- Amazon CloudWatch Events
- Amazon CloudWatch Logs
- Amazon EC2
- Amazon Elasticsearch Service
- IAM
- Amazon Kinesis
- AWS Lambda
- Amazon S3
- Amazon SNS
- Amazon SQS
- Amazon Simple Workflow Service (Amazon SWF)



If a policy is created that is more restrictive than the minimum required permissions, the console won't function as intended for users with that IAM policy.

To ensure that users with restricted permissions can use the Amazon CloudWatch console, attach the CloudWatchReadOnlyAccess managed policy to the user, group, or role.

AWS Managed Policies for Amazon CloudWatch

AWS provides standalone IAM policies that cover many common use cases. These policies were created and administered by AWS and grant the required permissions for services without customers having to create and maintain their own.

AWS Managed Policies

The following AWS managed policies are specific to Amazon CloudWatch:

CloudWatchFullAccess Grants full access to Amazon CloudWatch

CloudWatchReadOnlyAccess Grants read-only access to Amazon CloudWatch

CloudWatchActionsEC2Access Grants read-only access to Amazon CloudWatch Alarms and metrics and also Amazon EC2 metadata. Grants access to the Stop, Terminate, and Reboot API actions for Amazon EC2 instances

Customers create their own IAM policies to allow permissions for Amazon CloudWatch actions and resources. Attach these custom policies to the IAM users or groups that require those permissions.



When accessing Amazon CloudWatch through the AWS CLI or the Amazon CloudWatch API, the minimum console permissions are not needed.

Amazon CloudWatch Resources and Operations

Amazon CloudWatch has no specific resources that can be controlled. As a result, there are no Amazon CloudWatch Amazon Resource Names (ARNs) to use in an IAM policy.

For example, it is not possible to give a user access to Amazon CloudWatch data for a specific set of Amazon EC2 instances or a specific load balancer.



It is not possible to use IAM roles with the Amazon CloudWatch command-line tools.

When writing an IAM policy, use an asterisk (*) as the resource name to control access to Amazon CloudWatch actions.

```
{  
    "Version": "2012-10-17",  
    "Statement": [  
        {  
            "Effect": "Allow",  
            "Action": ["cloudwatch:GetMetricStatistics", "cloudwatch>ListMetrics"],  
            "Resource": "*",  
            "Condition": {  
                "Bool": {  
                    "aws:SecureTransport": "true"  
                }  
            }  
        }  
    ]  
}
```

AWS Cloud Services Integration

The following AWS Cloud services support Amazon CloudWatch without additional charges. Customers have the option to choose which of the preselected metrics they want to use.

Auto Scaling Groups Seven preselected metrics at a one-minute frequency

Elastic Load Balancing Thirteen preselected metrics at a one-minute frequency

Amazon Route 53 health checks One preselected metric at a one-minute frequency

Amazon EBS IOPS (Solid State Drive [SSD]) volumes Ten preselected metrics at a one-minute frequency

Amazon EBS General Purpose (SSD) volumes Ten preselected metrics at a one-minute frequency

Amazon EBS Magnetic volumes Eight preselected metrics at a five-minute frequency

AWS Storage Gateway Eleven preselected gateway metrics and five preselected storage volume metrics at a five-minute frequency

Amazon CloudFront Six preselected metrics at a one-minute frequency

Amazon DynamoDB tables Seven preselected metrics at a five-minute frequency

Amazon ElastiCache nodes Thirty-nine preselected metrics at a one-minute frequency

Amazon RDS DB instances Fourteen preselected metrics at a one-minute frequency

Amazon EMR job flows Twenty-six preselected metrics at a five-minute frequency

Amazon Redshift Sixteen preselected metrics at a one-minute frequency

Amazon SNS topics Four preselected metrics at a five-minute frequency

Amazon SQS queues Eight preselected metrics at a five-minute frequency

AWS OpsWorks Fifteen preselected metrics at a one-minute frequency

Amazon CloudWatch Logs Six preselected metrics at one-minute frequency

Estimated charges on your AWS bill It is also possible to enable metrics to monitor AWS charges. The number of metrics depends on the AWS products and services used. These metrics are offered at no additional charge.

Amazon CloudWatch Limits

Table 9.4 lists Amazon CloudWatch limits.

TABLE 9.4 Amazon CloudWatch Limits

| Resource | Default Limit |
|----------|--|
| Actions | 5/alarm. This limit cannot be changed. |
| Alarms | 10/month/customer for no additional charge 5,000 per region per account |

| Resource | Default Limit |
|--------------------------------|---|
| API requests | 1,000,000/month/customer for no additional charge |
| Custom metrics | No limit |
| DescribeAlarms | 3 Transactions per Second (TPS) This is the maximum number of operation requests that can be made per second without being throttled. A limit increase can be requested. |
| Dimensions | 10/metric. This limit cannot be changed. |
| GetMetricStatistics | 400 TPS This is the maximum number of operation requests per second before being throttled. A limit increase can be requested. |
| ListMetrics | 25 TPS This is the maximum number of operation requests per second before being throttled. A limit increase can be requested. |
| Metric data | 15 months. This limit cannot be changed. |
| MetricDatum items | 20/PutMetricData request A MetricDatum object can contain a single value or a StatisticSet object representing many values. This limit cannot be changed. |
| Metrics | 10/month/customer for no additional charge |
| Period | One day (86,400 seconds) This limit cannot be changed. |
| PutMetricAlarm request | 3 TPS The maximum number of operation requests you can make per second without being throttled. A limit increase can be requested. |
| PutMetricData request | 40 KB for HTTP POST requests 150 TPS The maximum number of operation requests that you can make per second without being throttled. A limit increase can be requested. |
| Amazon SNS email notifications | 1,000/month/customer for no additional charge |

Amazon CloudWatch Alarms

Amazon CloudWatch Alarms are used to initiate automatically an action in response to a predefined condition. An alarm watches a single metric over a specified time period and, based on the value of that metric relative to a threshold over time, performs one or more specified actions. Those actions include triggering an Auto Scaling policy, publishing to an Amazon SNS topic, and updating a dashboard.

Alarms only trigger actions after sustained state changes. Amazon CloudWatch Alarms are not generated simply because a metric is in a particular state. The state must change and be maintained for a user-specified number of periods.



When creating an alarm, select a period that is greater than or equal to the frequency of the metric to be monitored.

Basic monitoring for Amazon EC2 provides metrics for instances every five minutes. When setting an alarm on a basic monitoring metric, select a period of at least 300 seconds (5 minutes).

Detailed monitoring for Amazon EC2 provides metrics for instances every one minute. When setting an alarm on a detailed monitoring metric, select a period of at least 60 seconds (1 minute).

An Amazon CloudWatch Alarm is always in one of three states: OK, ALARM, or INSUFFICIENT_DATA.

- When the monitored metric is within the range that has been defined as acceptable, it is in the OK state.
- When a metric breaches a user-defined threshold, it transitions to the ALARM state.
- If the data needed to make a decision is missing or incomplete, it is in the INSUFFICIENT_DATA state.

Actions are set to respond to the transition of the metric as it moves into each of the three states. Actions only happen on state transitions and will not be re-executed if the condition persists.

Multiple actions are allowed for an alarm. If an alarm is triggered, Amazon SNS could be used to send a notification email, while at the same time an Auto Scaling policy is updated.

Alarms and Thresholds

Alarms are designed to be triggered when three things happen:

- A monitored metric has reached a particular value or is within a defined range.
- That metric's value stays at that value or within the specified range for a number of times in a row in a period.
- The value or range of the metric has been consistent for a number of user-defined periods.

In short, an Amazon CloudWatch Alarm is triggered when a monitored metric reaches a value, is reported multiple times in a row, and stays at that value for a period of time.



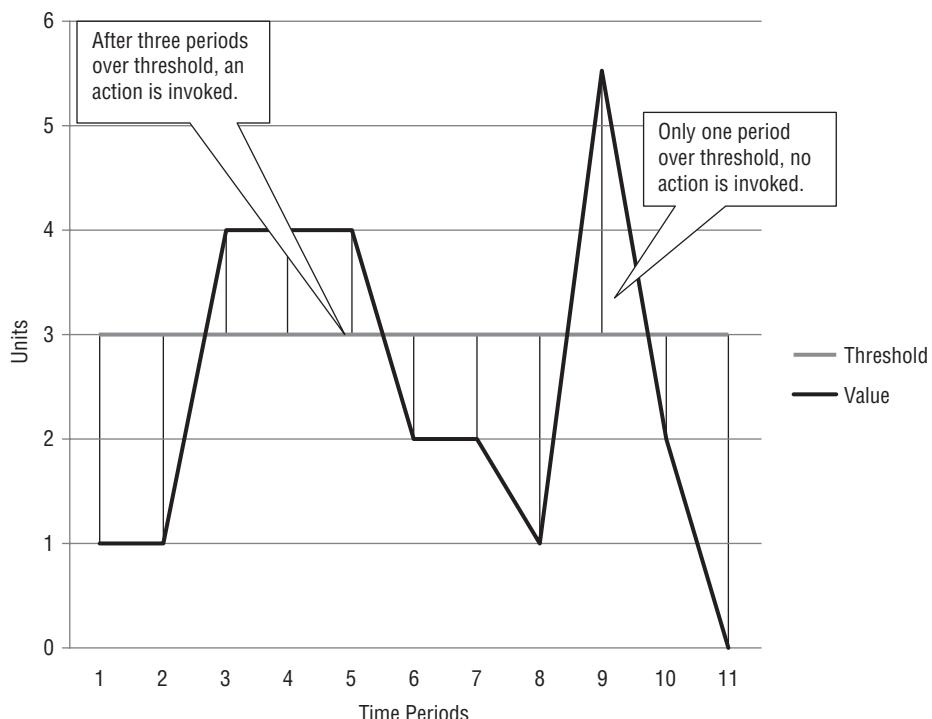
A single data point will *not* trigger an Amazon CloudWatch Alarm.

In Figure 9.2, the alarm threshold is set to three units and the alarm is evaluated over three periods. The alarm goes to ALARM state if the oldest of the three periods evaluated has matched the alarm criteria and the next two periods have met the criteria or are missing.

In the figure, this happens with the third through fifth time periods when the alarm's state is set to ALARM. At period six, the value drops below the threshold and the state reverts to OK.

Later, during the ninth time period, the threshold is breached again, but for only one period. Because of this, the alarm state remains OK. Figure 9.2 shows this as a graph.

FIGURE 9.2 A threshold breach without a change in alarm state





After an alarm is triggered, its subsequent behavior depends on the type of action associated with the alarm.

- For Auto Scaling policy notifications, the alarm continues to invoke the action for every period that the alarm remains unchanged.
- For Amazon SNS notifications, no additional actions are invoked.

Alarms can also be added to dashboards. When an alarm is on a dashboard, it turns red when it is in the ALARM state, making it easier to monitor its status.



Like metrics, alarms—and their corresponding triggers—exist only in the region in which they are created.

Missing Data Points

Similar to how each alarm is always in one of three states, each specific data point reported to Amazon CloudWatch falls under one of three categories:

Good: Within the threshold

Bad: Violating the threshold

Missing: Data for the metric is not available, or not enough data is available for the metric to determine the alarm state.

Customers can specify how alarms handle missing data points. They can be treated as:

Missing: The alarm looks back further in time to find additional data points.

Good: Treated as a data point that is within the threshold

Bad: Treated as a data point that is breaching the threshold

Ignored: The current alarm state is maintained.

The best choice of how to treat missing data points depends on the type of metric. For a metric that continually reports data, such as CPUUtilization of an instance, it might be best to treat missing data points as bad because their absence indicates something is wrong. For a metric that generates data points only when an error occurs, such as ThrottledRequests in Amazon DynamoDB, missing data points should be treated as good.

Choosing the best option for an alarm prevents unnecessary and misleading alarm condition changes and more accurately indicates the health of a system.

Common Amazon CloudWatch Metrics

There are hundreds of metrics available for monitoring on AWS. The common ones are listed here, broken down by service, with a brief explanation. As we have mentioned throughout, this book is designed to do more than just prepare you for an exam—it should serve you well as a day-to-day guide in working with AWS.

Amazon EC2

There are two types of EC2 status checks: system status checks and instance status checks.

System Status Checks

System Status Checks monitor AWS hardware to ensure that instances are working properly. These checks detect problems with an instance that requires AWS involvement to repair. When a system status check fails, customers can choose to wait for AWS to fix the issue, or they can resolve it by either stopping and starting an instance or by terminating and replacing it.



When an instance is stopped and then restarted, it will be moved to new hardware. When an instance is rebooted, it stays on the hardware on which it was provisioned.

The following are examples of problems that can cause system status checks to fail:

- Loss of network connectivity
- Loss of system power
- Software issues on the physical host
- Hardware issues on the physical host that impact network reachability

Instance Status Checks

Instance Status Checks monitor the software and network configuration of an individual instance. These checks detect problems that require customer involvement to repair. When an instance status check fails, typical solutions include rebooting or reconfiguring the instance.

The following are examples of problems that can cause instance status checks to fail:

- Failed system status checks
- Incorrect networking or startup configuration
- Exhausted memory
- Corrupted file system
- Incompatible kernel

The following Amazon CloudWatch metrics offer insight into the usage and utilization of Amazon EC2 instances. For Amazon EC2, common metrics include the following:

- CPUUtilization
- NetworkIn
- NetworkOut
- DiskReadOps
- DiskWriteOps
- DiskReadBytes
- DiskWriteBytes

CPUUtilization This metric is the percentage of allocated Amazon EC2 compute units that are currently in use on an instance. This metric identifies the processing power required to run an application on a selected instance.

NetworkIn This metric represents the number of bytes received on all network interfaces by the instance. This metric identifies the volume of incoming network traffic to a single instance.

Similar to NetworkOut, the number reported is the number of bytes received during the period. When using Basic Monitoring, divide this number by 300 to find bytes/second. With Detailed Monitoring, divide it by 60.

NetworkOut This metric is the number of bytes sent out on all network interfaces by the instance. This metric identifies the volume of outgoing network traffic from a single instance.

Similar to NetworkIn, the number reported is the number of bytes received during the period. When using Basic Monitoring, divide this number by 300 to find bytes/second. With Detailed Monitoring, divide it by 60.

DiskReadOps This metric reports the completed read operations from all instance store volumes available to the instance in a specified period of time.

To calculate the average I/O operations per second (IOPS) for the period, divide the total operations in the period by the number of seconds in that period.

DiskWriteOps This metric reports the completed write operations to all instance store volumes available to the instance in a specified period of time.

To calculate the average I/O operations per second (IOPS) for the period, divide the total operations in the period by the number of seconds in that period.

DiskReadBytes This metric reports the number of bytes read from all instance store volumes available to the instance.

This metric is used to determine the volume of the data the application reads from the hard disk of the instance. It can be used to determine the speed of the application.

The number reported is the number of bytes received during the period. When using Basic Monitoring, divide this number by 300 to find bytes/second. With Detailed Monitoring, divide it by 60.

DiskWriteBytes This metric reports the number of bytes written to all instance store volumes available to the instance.

This metric is used to determine the volume of the data the application writes onto the hard disk of the instance. This can be used to determine the speed of the application.

The number reported is the number of bytes received during the period. When using Basic Monitoring, divide this number by 300 to find bytes/second. With Detailed Monitoring, divide it by 60.

Amazon Elastic Block Store Volume Monitoring

Amazon Elastic Block Store (Amazon EBS) sends data points to Amazon CloudWatch for several metrics. Amazon EBS General Purpose SSD (gp2), Throughput Optimized HDD (st1), Cold HDD (sc1), and Magnetic (standard) volumes automatically send five-minute metrics to CloudWatch. Provisioned IOPS SSD (io1) volumes automatically send one-minute metrics to CloudWatch.

Common Amazon EBS metrics include the following:

- `VolumeReadBytes`
- `VolumeWriteBytes`
- `VolumeReadOps`
- `VolumeWriteOps`
- `VolumeTotalReadTime`
- `VolumeTotalWriteTime`
- `VolumeIdleTime`
- `VolumeQueueLength`
- `VolumeThroughputPercentage`
- `VolumeConsumedReadWriteOps`
- `BurstBalance`

VolumeReadBytes and VolumeWriteBytes These metrics provide information on the I/O operations in a specified period of time. The `Sum` statistic reports the total number of bytes transferred during the period. The `Average` statistic reports the average size of each I/O operation during the period. The `SampleCount` statistic reports the total number of I/O operations during the period. The `Minimum` and `Maximum` statistics are not relevant for this metric.



Data is reported to Amazon CloudWatch only when the volume is active. If the volume is idle, no data is reported to Amazon CloudWatch.

VolumeReadOps and VolumeWriteOps These metrics report the total number of I/O operations in a specified period of time. To calculate the average I/O operations per second (IOPS) for the period, divide the total operations in the period by the number of seconds in that period.

VolumeTotalReadTime and VolumeTotalWriteTime These metrics report the total number of seconds spent by all operations that completed in a specified period of time. If multiple requests are submitted at the same time, this total could be greater than the length of the period.

For example, for a period of 5 minutes (300 seconds): If 700 operations completed during that period, and each operation took 1 second, the value would be 700 seconds.

VolumeIdleTime This metric represents the total number of seconds in a specified period of time when no read or write operations were submitted.

VolumeQueueLength This metric is the number of read and write operation requests waiting to be completed in a specified period of time.



VolumeQueueLength should be at or near 0. If a database running on an Amazon EC2 instance is running slowly, check this metric. If it is higher than 0, increase the number of available IOPS.

VolumeThroughputPercentage This metric is used with Provisioned IOPS SSD volumes only. It is the percentage of I/O operations per second (IOPS) delivered of the total IOPS provisioned for an Amazon EBS volume. Provisioned IOPS SSD volumes deliver within 10 percent of the provisioned IOPS performance 99.9 percent of the time over a given year.



During a write, if there are no other pending I/O requests in a minute, the metric value will be 100 percent.

A volume's I/O performance may become temporarily degraded due to actions such as creating a snapshot of a volume during peak usage, running the volume on a non-EBS-optimized instance, or accessing data on the volume for the first time.

VolumeConsumedReadWriteOps This metric is used with Provisioned IOPS SSD volumes only. The total amount of read and write operations (normalized to 256K capacity units) consumed in a specified period of time

I/O operations that are smaller than 256K each count as 1 consumed IOPS. I/O operations that are larger than 256K are counted in 256K capacity units. For example, a 1024K I/O would count as 4 consumed IOPS.

BurstBalance This metric is only used with General Purpose SSD (gp2), Throughput Optimized HDD (st1), and Cold HDD (sc1) volumes. It provides information about the percentage of I/O credits (for gp2) or throughput credits (for st1 and sc1) remaining in the burst bucket.

Data is reported to Amazon CloudWatch only when the volume is active. If the volume is not attached, no data is reported.

Amazon EBS Status Checks

Volume status checks help customers understand, track, and manage potential inconsistencies in the data on an Amazon EBS volume. They are designed to provide you with the information needed to determine if an Amazon EBS volume is impaired and to help customers control how a potentially inconsistent volume is handled.

Volume status checks are automated tests that run every five minutes and return a pass or fail status. If all checks pass, the status of the volume is `ok`. If a check fails, the status of the volume is `impaired`. If the status is `insufficient-data`, the checks may still be in progress on the volume.

There are four status types for Provisioned IOPS EBS Volumes: `ok`, `warning`, `impaired`, and `insufficient-data`.

ok This status means that the volume is performing as expected.

warning This status means that the volume is either Degraded or Severely Degraded.

Degraded means that the volume performance is below expectations. Severely Degraded means that the volume performance is well below expectations.

impaired Impaired means that a volume has either Stalled or is Not Available. Stalled means that the volume performance is severely impacted. Not Available means that it is unable to determine I/O performance because I/O is disabled.

insufficient-data Insufficient-data means that there have not been enough data points collected but that it is online.

Amazon ElastiCache

The following Amazon CloudWatch metrics offer insight into Amazon ElastiCache performance. In most cases, the recommendation is to set CloudWatch Alarms for these metrics to be able to take corrective action before performance issues occur. For Amazon ElastiCache, common metrics include the following:

- `CPUUtilization`
- `SwapUsage`
- `Evictions`
- `CurrConnections`

CPUUtilization This is a host-level metric reported as a percent.

Memcached Because Memcached is multi-threaded, this metric can be as high as 90 percent. If this threshold is exceeded, scale the cache cluster up by using a larger cache node type, or scale out by adding more cache nodes.

Redis Because Redis is single-threaded, the threshold is calculated as utilization/number of processor cores. For example, when using a `cache.m1.xlarge` node that has four cores, the threshold for a 90 percent `CPUUtilization` would be $90/4$, or 22.5 percent.

Administrators have to determine their own threshold value based on the number of cores in the cache node being used.

If this threshold is exceeded and the main workload is from read requests, scale the cache cluster out by adding read replicas. If the main workload is from write requests, AWS recommends scaling up by using a larger cache instance type.

SwapUsage This is a host-level metric reported in bytes.

Memcached This metric should not exceed 50 MB. If it does, AWS recommends that the ConnectionOverhead parameter value be increased.

Redis At this time, AWS has no recommendation for this parameter; there is not a need to set an Amazon CloudWatch Alarm for it.

Evictions This is a metric published for both Memcached and Redis cache clusters. AWS recommends customers determine their own alarm thresholds for this metric based on application needs.

Memcached If the chosen threshold is exceeded, scale the cache cluster up by using a larger node type or scale out by adding more nodes.

Redis If you exceed your chosen threshold, scale your cluster up by using a larger node type.

CurrConnections This is a cache engine metric, published for both Memcached and Redis cache clusters. AWS recommends customers determine their own alarm thresholds for this metric based on application needs.

Whether running Memcached or Redis, an increasing number of CurrConnections might indicate a problem with an application.

Amazon RDS Metrics

When using Amazon RDS resources, Amazon RDS sends metrics and dimensions to Amazon CloudWatch every minute. Common metrics include the following:

- DatabaseConnections
- DiskQueueDepth
- FreeStorageSpace
- ReplicaLag
- ReadIOPS
- WriteIOPS
- ReadLatency WriteLatency

DatabaseConnections This metric is a count of the number of database connections in use.

DiskQueueDepth This metric is a count of the number of outstanding I/O operations waiting to access the disk.

FreeStorageSpace This metric, measured in bytes, is the amount of available storage space.



If you worry about disk space on an Amazon RDS Instance, set an alert based on the metric FreeStorageSpace. This way, you will never be surprised by disk space issues.

ReplicaLag This metric, measured in seconds, is the amount of time a Read Replica DB instance lags behind the source DB instance. It applies to MySQL, MariaDB, and PostgreSQL Read Replicas.

ReadIOPS This metric is the average number of disk read I/O operations per second.

WriteIOPS This metric is the average number of disk write I/O operations per second.

ReadLatency This metric, measured in seconds, is the average amount of time taken per disk read I/O operation.

WriteLatency This metric, measured in seconds, is the average amount of time taken per disk write I/O operation.

AWS Elastic Load Balancer

AWS ELB reports metrics to Amazon CloudWatch only when requests are flowing through the load balancer. When there are requests flowing through the load balancer, Elastic Load Balancing measures and sends its metrics in 60-second intervals. If there are no requests flowing through the load balancer or no data for a metric, the metric is not reported.

Common metrics reported to Amazon CloudWatch from an ELB include the following:

- BackendConnectionErrors
- HealthyHostCount
- UnHealthyHostCount
- RequestCount
- Latency
- HTTPCode_Backend_2XX
- HTTPCode_Backend_3XX
- HTTPCode_Backend_4XX
- HTTPCode_Backend_5XX
- HTTPCode_ELB_4XX
- HTTPCode_ELB_5XX
- SpilloverCount
- SurgeQueueLength

BackendConnectionErrors This metric is the number of connections that were not successfully established between the load balancer and the registered instances. Because the load balancer retries the connection when there are errors, this count can exceed the request rate. This count also includes any connection errors related to health checks.

HealthyHostCount This metric is the number of healthy instances registered with a load balancer. A newly registered instance is considered healthy after it passes the first health check.

If cross-zone load balancing is enabled, the number of healthy instances for the LoadBalancerName dimension is calculated across all Availability Zones. Otherwise, it is calculated per Availability Zone.

UnHealthyHostCount This metric is the number of unhealthy instances registered with a load balancer. An instance is considered unhealthy after it exceeds the unhealthy threshold configured for health checks. An unhealthy instance is considered healthy again after it meets the healthy threshold configured for health checks.

RequestCount This metric is the number of requests completed or connections made during the specified interval, which is either one or five minutes.

HTTP listener The number of requests received and routed, including HTTP error responses from the registered instances

TCP listener The number of connections made to the registered instances

Latency This metric represents the elapsed time, in seconds, between when a request has been sent to an instance and the reply received.

HTTP listener The total time elapsed, in seconds, from the time the load balancer sent the request to a registered instance until the instance started to send the response headers

TCP listener The total time elapsed, in seconds, for the load balancer to establish a connection to a registered instance successfully.

HTTPCode_Backend_2XX This metric is the number of HTTP 2XX response codes generated by registered instances. 2XX status codes report success. The action was successfully received, understood, and accepted. This count does not include any response codes generated by the load balancer.

HTTPCode_Backend_3XX This metric is the number of HTTP 3XX response codes generated by registered instances. 3XX status codes report redirection. Further action must be taken in order to complete the request.

HTTPCode_Backend_4XX This metric is the number of HTTP 4XX response codes generated by registered instances. 4XX status codes report client errors. The request contains bad syntax or cannot be fulfilled.

HTTPCode_Backend_5XX This metric is the number of HTTP 5XX response codes generated by registered instances. 5XX status codes report server errors. The server failed to fulfill an apparently valid request.

HTTPCode_ELB_4XX This metric is the number of HTTP 4XX client error codes generated by the load balancer. Client errors are generated when a request is malformed or incomplete. This error is generated by the ELB.

HTTPCode_ELB_5XX This metric the number of HTTP 5XX server error codes generated by the load balancer. This count does not include any response codes generated by the registered instances.

The metric is reported if there are no healthy instances registered to the load balancer, or if the request rate exceeds the capacity of the instances (spillover) or the load balancer. This error is generated by the ELB.

SpilloverCount This metric is the total number of requests that were rejected because the surge queue is full.

HTTP listener The load balancer returns an HTTP 503 error code.

TCP listener The load balancer closes the connection.

SurgeQueueLength This metric is the total number of requests that are pending routing. The load balancer queues a request if it is unable to establish a connection with a healthy instance in order to route the request.

The maximum size of the queue is 1,024. Additional requests are rejected when the queue is full.



The surge queue will start to increase when back-end instances are unable to process application requests as fast as they are coming in. This is the result of a number of issues, chief among them are the capacity constraints experienced by back-end instances.

Improve computational capacity by scaling up instances to ensure that the SurgeQueueLength never increases beyond the maximum queue capacity.

Spillover is a direct consequence of a surge queue length increase. When an ELB reaches its maximum queue capacity, it will start dropping new requests. When the requests are dropped, end users are not notified.

Monitor SurgeQueueLength and SpilloverCount to be alerted before a potentially high spillover count occurs that results in dropped requests and poor customer experience.

Amazon CloudWatch Events

Amazon CloudWatch Events delivers a near real-time stream of system events that describes changes in AWS resources. Using relatively simple rules, it is possible to route events to one or more targets for processing.

Think of Amazon CloudWatch Events as the central nervous system for an AWS environment. It is connected to supported AWS Cloud services, and it becomes aware of operational changes as they happen. Then, driven by rules, it sends messages and activates functions in response to the changes.

Events

An event is a change in an AWS environment, and it can be generated in four different ways:

- They arise from within AWS when resources change state, like when an Amazon EC2 instance state changes from pending to running.
- API calls and console sign-ins can generate events and deliver them to Amazon CloudWatch Events via AWS CloudTrail.
- Code can be run to generate application-level events and publish them to Amazon CloudWatch Events for processing.
- They can be issued on a scheduled basis, with options for periodic or Cron-style scheduling.



Here is an Amazon EC2 instance state-change notification event with an instance in the pending state.

```
{  
    "id": "7bf73129-1428-4cd3-a780-95db273d1602",  
    "detail-type": "EC2 Instance State-change Notification",  
    "source": "aws.ec2",  
    "account": "8675309719",  
    "time": "2015-11-11T21:29:54Z",  
    "region": "us-east-1",  
    "resources": [  
        "arn:aws:ec2:us-east-1: 8675309719:instance/i-dec1221"  
    ],  
    "detail": {  
        "instance-id": "i-dec1221",  
        "state": "pending"  
    }  
}
```

Remember, an event indicates there has been a change in an AWS environment. AWS resources can generate events when their state changes. AWS CloudTrail publishes events from API calls. Custom application-level events can be created and published to Amazon CloudWatch Events. Scheduled events are generated on a periodic basis.



Amazon EC2 generates an event when the state of an Amazon EC2 instance changes from pending to running. Auto Scaling generates events when it launches or terminates instances.

- AWS CloudTrail publishes events when API calls are made.
- Custom application-level events can be generated and published to Amazon CloudWatch Events.
- Scheduled events can be generated on a periodic basis.

Amazon CloudWatch Events can be used to schedule actions that trigger at certain times using cron or rate expressions. All scheduled events use the Universal Time (UTC) time zone and a minimum precision of one minute.

Rules

A *rule* matches incoming events and routes them to targets for processing. A single rule can route to multiple targets and are processed in parallel. This enables different parts of an organization to look for and process the events that are of interest to them.

A rule can customize the JSON sent to the target by passing only certain parts or by overwriting it with a constant.



Because rules sent to multiple targets are processed in parallel, their order is lost.

Targets

A *target* processes data in JSON format that has been sent to it from Amazon CloudWatch Events. Amazon CloudWatch Events delivers a near real-time stream of system events to one or more target functions or streams for analysis.

These targets include the following:

- Amazon EC2 instances
- AWS Lambda functions
- Amazon Kinesis Streams
- Amazon ECS tasks
- Amazon Step Functions state machines
- Amazon SNS topics
- Amazon SQS queues
- Built-in targets

Metrics and Dimensions

Amazon CloudWatch Events sends metrics to Amazon CloudWatch every minute.

The AWS/Events namespace includes the metrics shown in Table 9.5.

TABLE 9.5 Amazon CloudWatch Events Metrics

| Metric | Description |
|-------------------|---|
| Invocations | <ul style="list-style-type: none">▪ Measures the number of times a target is invoked for a rule in response to an event. This includes successful and failed invocations, but it does not include throttled or retried attempts until they fail permanently.▪ Amazon CloudWatch Events only sends this metric to Amazon CloudWatch if it has a non-zero value.▪ Valid Dimensions: RuleName▪ Units: Count |
| FailedInvocations | <ul style="list-style-type: none">▪ Measures the number of invocations that failed permanently. This does not include invocations that are retried or that succeeded after a retry attempt.▪ Valid Dimensions: RuleName▪ Units: Count |
| TriggeredRules | <ul style="list-style-type: none">▪ Measures the number of triggered rules that matched with any event.▪ Valid Dimensions: RuleName▪ Units: Count |
| MatchedEvents | <ul style="list-style-type: none">▪ Measures the number of events that matched with any rule.▪ Valid Dimensions: None▪ Units: Count |
| ThrottledRules | <ul style="list-style-type: none">▪ Measures the number of triggered rules that are being throttled.▪ Valid Dimensions: RuleName▪ Units: Count |

Amazon CloudWatch Events metrics use a single dimension: RuleName. As the name implies, it filters available metrics by rule name.

Amazon CloudWatch Logs

Amazon CloudWatch Logs can be used to monitor, store, and access log files from Amazon EC2 instances, AWS CloudTrail, and servers running in an on-premises datacenter. It is then possible to retrieve and report on the associated log data from Amazon CloudWatch Logs.

Amazon CloudWatch Logs can monitor and store application logs, system logs, web server logs, and other custom logs. By setting alarms on these metrics, notifications can be generated for application or web server issues and can take the necessary actions.

Amazon CloudWatch Logs is made up of several components:

- Log agents
- Log events
- Log streams
- Log groups
- Metric filters
- Retention policies

Log Agents A log *agent* directs logs to Amazon CloudWatch. AWS does not have effective visibility above the Hypervisor as part of the shared responsibility model. Because of this, agents have to send data into Amazon CloudWatch.

Log Events A log *event* is an activity reported to the log file by the operating system or application along with a timestamp. Log events support only text format.

Log events contain two properties: the timestamp of when the event occurred and the raw log message.

By default, any line that begins with a non-whitespace character closes the previous log message and starts a new log message.

Log Streams A log *stream* is a group of log events reported by a single source, such as a web server.

Log Groups A log *group* is a group of log streams from multiple resources, such as a group of web servers managing the same content.

Retention policies and metric filters are set on log groups—not log streams.

Metric Filters Metric *filters* tell Amazon CloudWatch how to extract metric observations from ingested log events and turn them into Amazon CloudWatch metrics.

For example, with a metric filter called `404_Error`, it will filter log events to find 404 access errors. An alarm can be created to monitor those 404 errors on different servers.

Retention Policies *Retention policies* determine how long events are retained inside Amazon CloudWatch Logs. Policies are assigned to log groups and applied to all of the log streams in the group.

Retention time can be set from 1 day to 10 years. You can also opt for logs to never expire.

Archived Data

All log events uploaded to Amazon CloudWatch are retained. It is possible to specify the retention duration. Data is compressed, put into an archive, and stored. Charges are incurred for storage of the archived data.



Amazon CloudWatch Logs can ingest logs from sources external to AWS. This means that log data from an on-premises datacenter can be sent to Amazon CloudWatch Logs for reporting.

Log Monitoring

Use Amazon CloudWatch Logs to monitor applications and systems using log data. Amazon CloudWatch Logs can track the number of errors that occur in application logs and send a notification whenever the rate of errors exceeds a threshold.

Because Amazon CloudWatch Logs uses log data for monitoring, no code changes are required. The current time is used for each log event if the `datetime_format` isn't provided. If the provided `datetime_format` is invalid for a given log message, the timestamp from the last log event with a successfully parsed timestamp is used. If no previous log events exist, the current time is used. A warning message is logged when a log event falls back to the current time or the time of a previous log event.



Timestamps are used for retrieving log events and generating metrics. If the wrong format is specified, log events could become non-retrievable and generate inaccurate metrics.

Agents

An *agent* is required to publish log data to Amazon CloudWatch Logs because AWS has no visibility above the Hypervisor. There are agents available for Linux and Windows.

Agents have the following components:

- A plugin to the AWS CLI that pushes log data to Amazon CloudWatch Logs
- A script that runs the Amazon CloudWatch Logs `aws logs push` command to send data to Amazon CloudWatch Logs
- A cron job that ensures that the daemon is always running

Amazon CloudWatch Logs Agent for Linux

The Amazon CloudWatch Logs agent requires Python version 2.6, 2.7, 3.0, or 3.3 and any of the following versions of Linux:

- Amazon Linux version 2014.03.02 or later
- Ubuntu Server version 12.04, 14.04, or 16.04

- CentOS version 6, 6.3, 6.4, 6.5, or 7.0
- Red Hat Enterprise Linux (RHEL) version 6.5 or 7.0
- Debian 8.0

Amazon CloudWatch Logs: Agents and IAM

The Amazon CloudWatch Logs agent requires the `CreateLogGroup`, `CreateLogStream`, `DescribeLogStreams`, and `PutLogEvents` operations.



With the latest agent, `DescribeLogStreams` is no longer needed.

Here is a sample IAM policy for using an agent:

```
{  
  "Version": "2012-10-17",  
  "Statement": [  
    {  
      "Effect": "Allow",  
      "Action": [  
        "logs:CreateLogGroup",  
        "logs:CreateLogStream",  
        "logs:PutLogEvents",  
        "logs:DescribeLogStreams"  
      ],  
      "Resource": [  
        "arn:aws:logs:*:*:  
      ]  
    }  
  ]  
}
```



Starting with Amazon Linux Amazon Machine Image (AMI) 2014.09, the Amazon CloudWatch Logs agent is available as a Red Hat Package Manager (RPM) installation with the `awslogs` package. Earlier versions of Amazon Linux can access the `awslogs` package by updating their instance with the `sudo yum update -y` command. By installing the `awslogs` package as an RPM instead of the using the Amazon CloudWatch Logs installer, instances receive regular package updates and patches from AWS without having to reinstall the Amazon CloudWatch Logs agent manually.



Do not update the Amazon CloudWatch Logs agent using the RPM installation method if Python script was used to install the agent. Doing so may cause configuration issues that prevent the Amazon CloudWatch Logs agent from sending logs to Amazon CloudWatch.

Amazon CloudWatch Logs Agent for Windows

Starting with EC2Config version 2.2.5, it is possible to export all Windows Server log messages from the system log, security log, application log, and Internet Information Services (IIS) log and send them to Amazon CloudWatch Logs. EC2Config version 2.2.10 or later adds the ability to export any event log data, event tracing for Windows data, or text-based log files to Amazon CloudWatch Logs. Windows performance counter data can also be exported to Amazon CloudWatch.



Amazon EC2 instances use an agent to send log data to Amazon CloudWatch. For Microsoft Windows, the agent is either the EC2Config service or the Systems Manager (SSM) Agent.

By default, the EC2Config service is included in AWS Windows Server 2003-2012 R2 AMIs. EC2Config starts when the instance boots and performs tasks during startup and each time an Amazon EC2 instance starts or stops. EC2Config can also perform tasks on demand. Some of these tasks are automatically enabled, while others must be enabled manually.

Windows Server 2016 AMIs do not use EC2Config. Instead, these AMIs use the EC2Launch PowerShell script. Table 9.6 shows which agent types are available on different versions of Microsoft Windows.

TABLE 9.6 Microsoft Windows Agents

| Operating System | Agent | Notes |
|-----------------------------|------------------------|--|
| Windows Server 2016 | SSM Agent | The EC2Config service is not supported on Windows Server 2016. |
| Windows Server 2008-2012 R2 | EC2Config or SSM Agent | If an instance is running EC2Config version 3.x or earlier, then the EC2Config service sends log data to Amazon CloudWatch. If an instance is running EC2Config version 4.x or later, then SSM Agent sends log data to Amazon CloudWatch. |



Log data is encrypted in transit and at rest.

It is also possible to install Amazon CloudWatch Logs agents and create log streams using AWS OpsWorks and Chef. Chef is a third-party systems and cloud infrastructure automation tool. Chef uses “recipes” to install and configure software and “cookbooks,” which are collections of recipes, to perform configuration and policy distribution tasks.



It is possible to monitor application logs for specific literal terms (such as `NullReferenceException`) or count the number of occurrences of a literal term at a particular position in log data. This literally could be a 404 status code in an Apache access log.

When the term is found, Amazon CloudWatch Logs reports the data to a customer-specified Amazon CloudWatch metric.

Searching and Filtering Log Data

After an agent begins publishing logs to Amazon CloudWatch, it is possible both to search for and filter log data by creating one or more filters. Metric filters define the terms and patterns to look for in log data as it is sent to Amazon CloudWatch Logs.

Amazon CloudWatch Logs uses these metric filters to turn log data into Amazon CloudWatch metrics that can be graphed or used to set an alarm condition.



Filters do not retroactively filter data. Filters only publish the metric data points for events that happen after the filter was created.

Metric filters consist of the following key elements:

Filter Pattern A symbolic description of how Amazon CloudWatch Logs should interpret the data in each log event. For example, a log entry could contain timestamps, IP addresses, and strings. Use the pattern to specify what to look for in the log file.

Metric Name The name of the Amazon CloudWatch metric to which the monitored log information should be published

Metric Namespace The destination namespace of the new Amazon CloudWatch metric

Metric Value The data published to the metric. For example, when counting the occurrences of a particular term like “Error,” the value will be “1” for each occurrence. When counting a value like bytes transferred, the published value will be the value in the log event.



Amazon CloudWatch Logs supports the rotation of logs.

The following file rotation mechanisms are supported:

- Rename existing log files with a numerical suffix, then re-create the original empty log file. For example, `/var/log/syslog.log` is renamed `/var/log/syslog.log.1`. If `/var/log/syslog.log.1` already exists from a previous rotation, it is renamed `/var/log/syslog.log.2`.
- Truncate the original log file in place after creating a copy. For example, `/var/log/syslog.log` is copied to `/var/log/syslog.log.1` and `/var/log/syslog.log` is truncated. There might be data loss for this case, so be careful about using this file rotation mechanism.
- Create a new file with a common pattern like the old one. For example, `/var/log/syslog.log.2017-01-01` remains and `/var/log/syslog.log.2017-01-02` is created.

The fingerprint (source ID) of the file is calculated by hashing the log stream key and the first line of file content. To override this behavior, use the `file_fingerprint_lines` option. When file rotation happens, the new file is supposed to have new content, and the old file is not supposed to have content appended; the agent pushes the new file after it finishes reading the old file.

Amazon CloudWatch Logs Metrics and Dimensions

Amazon CloudWatch Logs sends data to Amazon CloudWatch every minute.

Metrics

The AWS/Logs namespace includes the metrics shown in the Table 9.7.

TABLE 9.7 AWS/Logs Namespace Metrics

| Metric | Description |
|---------------|--|
| IncomingBytes | <p>The volume of log events in uncompressed bytes uploaded to Amazon CloudWatch Logs. When used with the LogGroupName dimension, this is the volume of log events in uncompressed bytes uploaded to the log group.</p> <p>Valid Dimensions: LogGroupName</p> <p>Valid Statistic: Sum</p> <p>Units: Bytes</p> |

| Metric | Description |
|--------------------|---|
| IncomingLogEvents | <p>The number of log events uploaded to Amazon CloudWatch Logs. When used with the LogGroupName dimension, this is the number of log events uploaded to the log group.</p> <p>Valid Dimensions: LogGroupName</p> <p>Valid Statistic: Sum</p> <p>Units: None</p> |
| ForwardedBytes | <p>The volume of log events in compressed bytes forwarded to the subscription destination.</p> <p>Valid Dimensions: LogGroupName, DestinationType, FilterName</p> <p>Valid Statistic: Sum</p> <p>Units: Bytes</p> |
| ForwardedLogEvents | <p>The number of log events forwarded to the subscription destination.</p> <p>Valid Dimensions: LogGroupName, DestinationType, FilterName</p> <p>Valid Statistic: Sum</p> <p>Units: None</p> |
| DeliveryErrors | <p>The number of log events for which Amazon CloudWatch Logs received an error when forwarding data to the subscription destination.</p> <p>Valid Dimensions: LogGroupName, DestinationType, FilterName</p> <p>Valid Statistic: Sum</p> <p>Units: None</p> |
| DeliveryThrottling | <p>The number of log events for which Amazon CloudWatch Logs was throttled when forwarding data to the subscription destination.</p> <p>Valid Dimensions: LogGroupName, DestinationType, FilterName</p> <p>Valid Statistic: Sum</p> <p>Units: None</p> |

Dimensions

Amazon CloudWatch Logs supports the filtering of metrics using the dimensions shown in Table 9.8.

TABLE 9.8 Amazon CloudWatch Logs Dimensions

| Dimension | Description |
|--------------|--|
| LogGroupName | The name of the Amazon CloudWatch Logs log group from which to display metrics |

TABLE 9.8 Amazon CloudWatch Logs Dimensions (*continued*)

| Dimension | Description |
|-----------------|--|
| DestinationType | The subscription destination for the Amazon CloudWatch Logs data, which can be AWS Lambda, Amazon Kinesis Streams, or Amazon Kinesis Firehose |
| FilterName | The name of the subscription filter that is forwarding data from the log group to the destination. The subscription filter name is automatically converted by Amazon CloudWatch to ASCII and any unsupported characters get replaced with a question mark (?). |

Monitoring AWS Charges

Customers can monitor AWS costs using Amazon CloudWatch. With Amazon CloudWatch, it is possible to create billing alerts that send notifications when the usage of provisioned services exceeds a customer-defined limit.

When usage exceeds these thresholds, AWS can send an email notification or publish a notification to an Amazon SNS topic. To create billing alerts and register for notifications, enable them in the AWS Billing and Cost Management console.



Using the procedure shown, it is possible to sign up to receive notifications from AWS when prices change.



Design Scenario

Steps to Configure Price Change Notifications

1. Sign in to the AWS Management Console and open the Amazon SNS console:
<https://console.aws.amazon.com/sns/v2/home>.
2. If necessary, change the region on the navigation bar to US East (N. Virginia). All AWS billing metric data is stored in this region, even for resources in other regions.
3. On the navigation pane, choose Subscriptions.
4. Choose Create Subscription.
5. For Topic ARN:

Notifications sent every time a price changes:

arn:aws:sns:us-east-1:278350005181:price-list-api

Notifications about price changes once a day:

arn:aws:sns:us-east-1:278350005181:daily-aggregated-price-list-api

6. For Protocol, use the default HTTP setting.
7. For Endpoint, choose email.
8. Choose Create Subscription.

Detailed Billing

In December 2016, Amazon announced the addition of detailed billing to Amazon CloudWatch Logs. Reports can be generated based on usage and cost per log group.

Tags and Log Groups

You can use tags to classify log groups and give them categories such as purpose, owner, or environment. You can create a custom set of categories to meet specific needs, and you can also use tags to categorize and track AWS costs. When you apply tags to your AWS resources, including log groups, AWS cost allocation reports include usage and costs aggregated by tags.

- Tags can be added to log groups to get a detailed view of costs across business dimensions.
- Up to 50 tags can be added to each log group.
- Tags are added to log groups using the AWS CLI or Amazon CloudWatch Logs API.



Using tags is a simple yet powerful way to manage AWS resources and organize data, including billing data. Tags can be applied to resources that represent business categories such as cost centers, application names, or owners to organize costs across multiple services. It is an AWS best practice for customers to tag as many of their resources as possible.

Log Group Tag Restrictions

The following restrictions apply to tags.

Basic Restrictions

- The maximum number of tags per log group is 50.
- The keys and values of a tag are case sensitive.
- Tags in a deleted log group cannot be changed or edited.

Tag Key Restrictions

- Each tag key must be unique. If a tag is added with a key that's already in use, the new tag overwrites the existing key/value pair.
- Tag keys cannot start with aws: because this prefix is reserved for use by AWS. AWS creates tags that begin with this prefix on customers' behalves, but customers cannot edit or delete them.
- Tag keys must be between 1 and 128 Unicode characters in length.
- Tag keys must consist of the following characters:
 - Unicode letters and digits
 - Whitespace
 - Underscore (_)
 - Period (.)
 - Forward slash (/)
 - Equals sign (=)
 - Plus sign (+)
 - Hyphen (-)
 - At symbol (@)

Tag Value Restrictions

- Tag values must be between 0 and 255 Unicode characters in length.



Tag values can be blank—this is the 0 part of the length.

At the end of the billing cycle, the total charges (tagged and untagged) on the billing report with cost allocation tags reconciles with the total charges on the Bills page total and other billing reports for the same period.

Tags can also be used to filter views in Cost Explorer.



Design Scenario

Steps to Configure Billing Tags

In order for tags to appear on your billing reports, they must be activated in the Billing console.

To activate tags:

1. Sign in to the AWS Management Console, and open the AWS Billing and Cost Management console:

<https://console.aws.amazon.com/billing/home#/>.

2. In the navigation pane, choose Cost Allocation Tags.
3. Select the tag(s) to activate.
4. Choose Save.



If tags are added or changed on a resource partway through a billing period, costs will be split into two separate lines in the Cost Allocation Report. The first line will show costs before the update, and the second line will show costs after the update.

Cost Explorer

Cost Explorer is an AWS tool that can be used to view charts of costs. This spend data can be viewed for up to the past 13 months and used to forecast the spend data for the next 3 months. It can also be used to see patterns in how much is spent on AWS resources over time, identify areas that need further inquiry, and see trends to assist in understanding costs.

Cost Explorer can reveal which service is being used the most and which Availability Zone gets the most network traffic.

With Cost Explorer, there are a variety of filters:

- API operation
- Availability Zone
- AWS Cloud service
- Custom cost allocation tags
- Amazon EC2 instance type
- Linked account(s)
- Platform
- Purchase option
- Region
- Tenancy
- Usage type
- Usage type group



Every time a filter is applied to costs, Cost Explorer creates a new chart. It is possible to use a browser's bookmark feature to save configuration settings for repeated use. When returning to the saved link, Cost Explorer refreshes the page using current cost data for the time range selected and displays the most recent forecast.

This feature makes it easy to save a configuration that is often used, such as "Spend Report – Last Seven Days."

Cost Explorer uses the same dataset used to generate the AWS Cost and Usage Reports and the detailed billing reports. The dataset can also be downloaded as a comma-separated value (CSV) file for detailed analysis.

AWS Billing and Cost Management Metrics and Dimensions

The AWS Billing and Cost Management service sends metrics to Amazon CloudWatch.

Metrics

The AWS/Billing namespace uses the metric in Table 9.9.

TABLE 9.9 AWS/Billing Namespace Metric

| Metric | Description |
|------------------|---|
| EstimatedCharges | The estimated charges for AWS usage. This can be either estimated charges for one service or a roll-up of estimated charges for all services. |

Dimensions

AWS Billing and Cost Management supports filtering metrics using the dimensions in Table 9.10.

TABLE 9.10 AWS/ Billing and Cost Management Metrics

| Dimension | Description |
|---------------|--|
| ServiceName | The name of the AWS Cloud service This dimension is omitted for the total of estimated charges across all services. |
| LinkedAccount | The linked account number This is used for Consolidated Billing only. This dimension is included only for accounts that are linked to a separate paying account in a Consolidated Billing relationship. It is not included for accounts that are not linked to a Consolidated Billing paying account. |
| Currency | The monetary currency to bill the account. This dimension is required. Unit: USD |

AWS CloudTrail

AWS CloudTrail is a service that enables governance, compliance, operational auditing, and risk auditing of an AWS account. With AWS CloudTrail, it is possible to log, continuously monitor, and retain events related to API calls across an AWS infrastructure.

AWS CloudTrail provides a history of AWS API calls for an account. This includes API calls made through the AWS Management Console, AWS SDKs, command-line tools, and other AWS Cloud services. This history simplifies security analysis, resource change tracking, and troubleshooting.

AWS CloudTrail provides visibility into user activity by recording API calls made on an account. It records important information about each API call, including the name of the API, the identity of the caller, the time of the API call, the request parameters, and the response elements returned by the AWS Cloud service. This information can be used in the tracking of changes made to AWS resources and help troubleshoot operational issues. This makes it easier to ensure compliance with internal policies and regulatory standards.

What Are Trails?

A *trail* is a configuration that enables logging of the AWS API activity and related events in an account. AWS CloudTrail delivers the logs to an Amazon S3 bucket and, optionally, to an Amazon CloudWatch Logs log group.

It is possible to specify an Amazon SNS topic that receives notifications of log file deliveries. For a trail that applies to all regions, the trail configuration in each region is identical.

Types of Trails

You can create trails with the AWS CloudTrail console, the AWS CLI, or the AWS CloudTrail API. There are two types of trails: those that apply to all regions and those that apply to one region.

Trails that Apply to All Regions

When creating a trail that applies to all regions, AWS CloudTrail creates the same trail in each region. It then records the log files in each region and delivers the log files to an Amazon S3 bucket that is user-specified. This is the default option when you create a trail in the AWS CloudTrail console.

A trail that applies to all regions has the following advantages:

- The configuration settings for the trail apply consistently across all regions.
- Log files from all regions are sent to a single Amazon S3 bucket and, optionally, to an Amazon CloudWatch Logs log group.
- Trail configurations for all regions are managed from one location.

- Events are immediately received from new regions. When a new region launches, AWS CloudTrail automatically creates a trail in the new region with the same settings as your original trail.
- Trails can be created in regions that are not used often to monitor for unusual activity. When applying a trail to all regions, AWS CloudTrail uses the trail created in a particular region to create trails with identical configurations in all other regions in an account. This has the following effects:
 - If an Amazon SNS topic has been configured for the trail, Amazon SNS notifications about log file deliveries in all regions are sent to that single Amazon SNS topic.
 - Global service events will be delivered from a single region to the specified Amazon S3 bucket and to, if one has been configured, the Amazon CloudWatch Logs log group.
 - If log file integrity validation has been activated, log file integrity validation is enabled in all regions for the trail.

A Trail that Applies to One Region

When creating a trail that applies to one region, AWS CloudTrail records the log files in that region only and delivers log files to a user-specified Amazon S3 bucket. When creating additional individual trails that apply to specific regions, those trails can be set to deliver log files to a single Amazon S3 bucket regardless of region.



For both types of trails, it is possible to use an Amazon S3 bucket from any region.

Multiple Trails per Region

If there are different but related user groups such as developers, security personnel, and IT auditors that need access to AWS CloudTrail, create multiple trails per region. This allows each group to receive its own copy of the log files.

AWS CloudTrail supports five trails per region. A trail that applies to all regions counts as one trail in every region. To see a list of the trails in all regions, open the Trails page of the AWS CloudTrail console.

Encryption

By default, log files are encrypted using Amazon S3 Server-Side Encryption (SSE). Log files can be stored in an Amazon S3 bucket for as long as they are needed. Amazon S3 lifecycle rules can be defined to archive or delete log files automatically.

AWS CloudTrail Log Delivery

AWS CloudTrail typically delivers log files within 15 minutes of an API call. In addition, AWS CloudTrail publishes log files multiple times an hour—approximately every five minutes. These log files contain API calls from services that support AWS CloudTrail.



AWS CloudTrail captures API calls made directly by a user or on behalf of a user by an AWS Cloud service. Services that make API calls on behalf of users include AWS CloudFormation, AWS Elastic Beanstalk, AWS OpsWorks, and Auto Scaling.

For example, an AWS CloudFormation CreateStack call can result in additional API calls to Amazon EC2, Amazon RDS, Amazon EBS, or other services as required by the AWS CloudFormation template. This behavior is normal and expected.

To identify whether an API call was made by an AWS Cloud service, review the `invokedby` field in the AWS CloudTrail event.

Overview: Creating a Trail

When creating or updating a trail with the AWS CloudTrail console or using the AWS CLI, the same steps need to be followed.

1. Turn on AWS CloudTrail by creating a trail. By default, when you create a trail in a region in the AWS CloudTrail console, the trail applies to all regions.
2. Create an Amazon S3 bucket or specify an existing bucket where the log files are to be delivered. By default, log files from all regions in an account are delivered to the specified bucket.
3. Configure the trail to log the types of events desired. The choices are read-only, write-only, or all management and data events. By default, trails log all management events.
4. Create an Amazon SNS topic to receive notifications when log files are delivered. Delivery notifications from all regions are sent to the Amazon SNS topic specified.
5. Configure Amazon CloudWatch Logs to receive logs from AWS CloudTrail so that they can be monitored for specific log events.
6. Turn on log file encryption. This encrypts files for added security.
7. Turn on integrity validation for log files. This enables the delivery of digest files that you can use to validate the integrity of log files after AWS CloudTrail has delivered them.
8. Add tags to the trail.

Monitoring with AWS CloudTrail

Amazon CloudWatch is a web service that collects and tracks metrics to monitor AWS resources and the applications that run on it. *Amazon CloudWatch Logs* is a feature of Amazon CloudWatch used specifically to monitor log data. Integration with Amazon CloudWatch Logs enables AWS CloudTrail to send events containing API activity in an AWS account to an Amazon CloudWatch Logs log group.

AWS CloudTrail events that are sent to Amazon CloudWatch Logs can trigger alarms according to the metric filters defined by customers. Optionally, you can configure Amazon

CloudWatch Alarms to send notifications or make changes to the resources being monitored based on log stream events that metric filters extract.

Using Amazon CloudWatch Logs, you can track AWS CloudTrail events alongside events from the operating system, applications, or other AWS Cloud services that are sent to Amazon CloudWatch Logs.

AWS CloudTrail vs. Amazon CloudWatch

AWS CloudTrail adds depth to the monitoring capabilities already offered by AWS. Amazon CloudWatch focuses on performance monitoring and system health, and AWS CloudTrail focuses on API activity. While AWS CloudTrail does not report on system performance or health, you can use AWS CloudTrail in combination with Amazon CloudWatch Logs alarms to create notifications to gain a deeper understanding of AWS resources and their utilization.

AWS CloudTrail: Trail Naming Requirements

AWS CloudTrail trail names must meet the following requirements:

- Contain only ASCII letters (a-z, A-Z)
- Numbers (0-9)
- Periods (.)
- Underscores (_)
- Dashes (-)
- They must start with a letter or number and end with a letter or number.
- Be between 3 and 128 characters long.
- Have no adjacent periods, underscores, dashes, or combinations of these characters. Names like my-_namespace and my-\-namespace are invalid.
- Not be in IP address format. For example, 10.9.28.68 is invalid.

Getting and Viewing AWS CloudTrail Log Files

AWS CloudTrail delivers log files to an Amazon S3 bucket specified during the creation of the trail. Typically, log files appear in the bucket within 15 minutes of the recorded AWS API call or other AWS event. Log files are generally published every five minutes.

Finding AWS CloudTrail Log Files

AWS CloudTrail publishes log files to the Amazon S3 bucket in a gzip archive. In the Amazon S3 bucket, the log file has a formatted name that includes the following elements:

- The bucket name specified when you created the trail
- The (optional) prefix that you specified when the trail was created

- The string "AWSLogs"
- The account number
- The string "CloudTrail"
- A region identifier
- The year the log file was published in YYYY format
- The month the log file was published in MM format
- The day the log file was published in DD format
- An alphanumeric string that separates the file from others that cover the same time period

This is what a complete log file object name looks like:

```
bucket/prefix/AWSLogs/AccountID/CloudTrail/region/YYYY/MM/DD/file_name.json.gz
```

Retrieve Log Files

To retrieve a log file, use the Amazon S3 console, the AWS CLI, or the Amazon S3 API.

To find your log files with the Amazon S3 console, do the following:

- Open the Amazon S3 console.
- Choose the bucket specified for the trails.
- Navigate through the object hierarchy to find the correct log.

All log files have a .gz extension.



You can find the bucket name on the Trails page of the AWS CloudTrail console.



Log files are written in JSON format. There are a number of options available for viewing JSON-formatted files, including browser plugins, text editors, and Integrated Development Environments (IDEs). Use your preferred search engine to research more details.

Configuring Amazon SNS Notifications for AWS CloudTrail

It is possible to be notified when AWS CloudTrail publishes new log files to an Amazon S3 bucket. You manage notifications using Amazon SNS.

Notifications are optional. To activate them, configure AWS CloudTrail to send update information to an Amazon SNS topic whenever a new log file has been sent. To receive these notifications, subscribe to the topic. To handle notifications programmatically, subscribe an Amazon SQS queue to the topic.

Controlling User Permissions for AWS CloudTrail

AWS CloudTrail integrates with IAM, which controls access to AWS CloudTrail and other AWS resources that AWS CloudTrail requires, including Amazon S3 buckets and Amazon SNS topics. Use IAM to control which AWS users can create, configure, or delete AWS CloudTrail trails, start and stop logging, and access the buckets that contain log information.

Granting Permissions for AWS CloudTrail Administration

To administer an AWS CloudTrail trail, grant explicit permissions to IAM users to perform the actions associated with the AWS CloudTrail tasks. For most scenarios, you can accomplish this by using an AWS managed policy that contains predefined permissions.

A typical approach is to create an IAM group that has the appropriate permissions and then add individual IAM users to that group. For example, you can create one IAM group for users that should have full access to AWS CloudTrail actions and a separate group for users who should be able to view trail information but not create or change trails.

These are the AWS Managed Policies for AWS CloudTrail:

AWSCloudTrailFullAccess This policy gives users in the group full access to AWS CloudTrail actions and permissions to manage the Amazon S3 bucket, the log group for Amazon CloudWatch Logs, and an Amazon SNS topic for a trail.

AWSCloudTrailReadOnlyAccess This policy lets users in the group view trails and buckets.

Log Management and Data Events

When creating a trail, the trail logs read-only and write-only management events for your account. If desired, update the trail to specify whether or not the trail should log data events. Data events are object-level API operations that access Amazon S3 object resources, such as `GetObject`, `DeleteObject`, and `PutObject`. Only events that match the trail settings are delivered to the Amazon S3 bucket and Amazon CloudWatch Logs log group. If the event doesn't match the settings for a trail, the trail doesn't log the event.

Amazon SNS Topic Policy for AWS CloudTrail

To send notifications to an Amazon SNS topic, AWS CloudTrail must have the required permissions. AWS CloudTrail automatically attaches the required permissions to the topic when the following occurs:

- Create an Amazon SNS topic as part of creating or updating a trail in the AWS CloudTrail console.
- Create an Amazon SNS topic with the AWS CLI `create-subscription` and `update-subscription` commands.

AWS CloudTrail adds the following fields in the policy automatically:

- The allowed SIDs
- The service principal name for AWS CloudTrail
- The Amazon SNS topic, including region, account ID, and topic name

The following policy allows AWS CloudTrail to send notifications about log file delivery from supported regions:

Amazon SNS Topic Policy

```
{  
    "Version": "2012-10-17",  
    "Statement": [  
        {"Sid": "AWSCloudTrailSNSPolicy20131101",  
        "Effect": "Allow",  
        "Principal": {"Service": "cloudtrail.amazonaws.com"},  
        "Action": "SNS:Publish",  
        "Resource": "arn:aws:sns:Region:SNSTopicOwnerAccountId:SNSTopicName"  
    ]  
}
```

AWS Config

AWS Config is a fully managed service that provides AWS resource inventory, configuration history, and configuration change notifications to enable security and governance. AWS Config can discover existing AWS resources, export a complete inventory of AWS resources with all configuration details, and determine how a resource was configured at any point in time. These capabilities enable compliance auditing, security analysis, resource change tracking, and troubleshooting.

AWS Config makes it easy to track resource configuration without the need for upfront investments and to avoid the complexity of installing and updating agents for data collection or maintaining large databases. After AWS Config is enabled, continuously updated details can be viewed of all configuration attributes associated with AWS resources.

Amazon SNS can be configured to provide notifications of every configuration change.

AWS Config provides a detailed view of the configuration of AWS resources in an AWS account. This includes how resources are related to one another and how they were configured in the past to show how configurations and relationships change over time.

An AWS resource is an entity in AWS such as an Amazon EC2 instance, an Amazon EBS volume, a security group, or an Amazon Virtual Private Cloud (Amazon VPC).

With AWS Config, you can do the following:

- Evaluate AWS resource configurations for desired settings.
- Get a snapshot of the current configurations of the supported resources that are associated with an AWS account.
- Retrieve configurations of one or more resources that exist in an account.
- Retrieve historical configurations of one or more resources.
- Receive a notification whenever a resource is created, modified, or deleted.
- View relationships between resources, such as those that use a particular security group.

Ways to Use AWS Config

When running applications on AWS, resources must be created and managed collectively. As the demand for an application grows, so too does the need to keep track of the addition of AWS resources. AWS Config is designed to help oversee application resources in the following scenarios.

Resource Administration

To exercise better governance over resource configurations and to detect resource misconfigurations, fine-grained visibility is needed into what resources exist and how these resources are configured at any time. Use AWS Config to automatically send notifications whenever resources are created, modified, or deleted. There is no need to monitor these changes by polling calls made to each individual resource.

Use AWS Config rules to evaluate the configuration settings of AWS resources. When AWS Config detects that a resource violates the conditions in one of the established rules, AWS Config flags the resource as noncompliant and sends a notification. AWS Config continuously evaluates resources as they are created, changed, or deleted.

Auditing and Compliance

Some data requires frequent audits to ensure compliance with internal policies and best practices. To demonstrate compliance, access is needed to the historical configurations of the resources. This information is provided by AWS Config.

Managing and Troubleshooting Configuration Changes

When using multiple AWS resources that depend on one another, a change in the configuration of one resource might have unintended consequences on related resources. With AWS Config, it is possible to view how one resource is related to other resources and assess the impact of the proposed change.

The historical configurations of resources provided by AWS Config can assist troubleshooting issues by providing access to the last known good configuration of a problem resource.

Security Analysis

To analyze potential security weaknesses, detailed historical information about AWS resource configurations is required. This information could include the IAM permissions that are granted to your users or the Amazon EC2 security group rules that control access to your resources.

Use AWS Config to view the IAM policy that was assigned to an IAM user, group, or role at any time in which AWS Config was recording. This information can help determine the permissions that belonged to a user at a specific time.

Use AWS Config to view the configuration of Amazon EC2 security groups and the port rules that were open at a specific time. This information can help determine whether a security group was blocking incoming TCP traffic to a specific port.

AWS Config Rules

An AWS Config rule represents desired configurations for a resource, and it is evaluated against configuration changes on the relevant resources, as recorded by AWS Config. The results of evaluating a rule against the configuration of a resource are available on a dashboard. Using AWS Config rules, customers can assess their overall compliance and risk status from a configuration perspective, view compliance trends over time, and pinpoint which configuration change caused a resource to drift out of compliance with a rule.

A rule represents desired Configuration Item (CI) attribute values for resources, which are evaluated by comparing those attribute values with CIs recorded by AWS Config. There are two types of rules: AWS managed rules and customer managed rules.

AWS Managed Rules

AWS managed rules are prebuilt and managed by AWS. Choose the rule to enable and then supply a few configuration parameters to get started.

Customer Managed Rules

It is possible to develop custom rules and add them to AWS Config. Associate each custom rule with an AWS Lambda function. This Lambda function contains the logic that evaluates whether AWS resources comply with the rule.

Associate this function with a rule, and the rule invokes the function either in response to configuration changes or periodic intervals. The function then evaluates whether resources comply with the rule, and it sends its evaluation results to AWS Config.

Using AWS Config rules, customers can assess their overall compliance and risk status from a configuration perspective, view compliance trends over time, and pinpoint which configuration change caused a resource to drift out of compliance with a rule.



Customers can create up to 50 AWS Config rules in an AWS account by default. This is a soft limit, and it can be increased by contacting AWS Support.

How Rules Are Evaluated

Any rule can be set up as a change-triggered rule or as a periodic rule.

A change-triggered rule is executed when AWS Config records a configuration change for any of the resources specified. Additionally, one of the following must be specified:

Tag Key A tag key:value implies any configuration changes recorded for resources with the specified tag key:value will trigger an evaluation of the rule.

Resource Type(s) Any configuration changes recorded for any resource within the specified resource type(s) will trigger an evaluation of the rule.

Resource ID Any changes recorded to the resource specified by the resource type and resource ID will trigger an evaluation of the rule.

A periodic rule is triggered at a specified frequency. Available frequencies are 1 hour, 3 hours, 6 hours, 12 hours, or 24 hours. A periodic rule has a full snapshot of current CIs for all resources available to the rule.

Configuration Items

A CI is the configuration of a resource at a given point in time. A CI consists of five sections:

- Basic information about the resource that is common across different resource types (for example, ARNs, tags)
- Configuration data specific to the resource (such as an Amazon EC2 instance type)
- Map of relationships with other resources (for example, Amazon EC2::Volume vol-6886ff28 is “attached to instance” Amazon EC2 instance i-24601abc)
- AWS CloudTrail event IDs that are related to this state
- Metadata that helps identify information about the CI, such as the version of the CI and when the CI was captured

Rule Evaluation

Evaluation of a rule determines whether a rule is compliant with a resource at a particular point in time. It is the result of evaluating a rule against the configuration of a resource. AWS Config rules will capture and store the result of each evaluation. This result will include the resource, rule, time of evaluation, and a link to the CI that caused noncompliance.

Rule Compliance

A resource is compliant if it conforms with all rules that apply to it. Otherwise, it is non-compliant. Similarly, a rule is compliant if all resources evaluated by the rule comply with the rule. Otherwise, it is noncompliant.

In some cases, such as when inadequate permissions are available to the rule, an evaluation may not exist for the resource, leading to a state of insufficient data. This state is excluded from determining the compliance status of a resource or rule.

AWS Config and AWS CloudTrail

AWS CloudTrail records user API activity on an account and allows access to information about this activity. You can use AWS CloudTrail to get full details about API actions, such as identity of the caller, the time of the API call, the request parameters, and the response elements returned by the AWS Cloud service.

AWS Config records point-in-time configuration details for AWS resources as CIs. You can use a CI to answer “What did my AWS resource look like?” at a point in time. You can use AWS CloudTrail to answer “Who made an API call to modify this resource?”

In practice, you can use the AWS Config console to detect that a security group was incorrectly configured in the past. With the integrated AWS CloudTrail information, you can find the user that misconfigured the security group and learn when it happened.

Each custom rule is simply an AWS Lambda function. When the function is invoked in order to evaluate a resource, it is provided with the resource's CI. The function can inspect the item and make calls to other AWS API functions as desired. After the AWS Lambda function makes its decision about compliance, it calls the PutEvaluations function to record the decision.

Pricing

With AWS Config, customers are charged based on the number of CIs recorded for supported resources in an AWS account and are charged only once for recording the CI. There is no additional fee or any upfront commitment for retaining the CI. Users can stop recording CIs at any time and continue to access the CIs previously recorded. Charges per CI are rolled up into the monthly bill.

If you are using AWS Config rules, charges are based on active AWS Config rules in that month. When a rule is compared with an AWS resource, the result is recorded as an evaluation. A rule is active if it has one or more evaluations in a month.

Configuration snapshots and configuration history files are delivered to an Amazon S3 bucket. Configuration change notifications are delivered via Amazon SNS. Standard rates for Amazon S3 and Amazon SNS apply. Customer-managed rules are authored using AWS Lambda. Standard rates for AWS Lambda apply.

Summary

Amazon CloudWatch monitors AWS resources and the applications run on AWS in real time. Use CloudWatch to collect and track metrics, which are variables used to measure resources and applications.

Amazon CloudWatch Alarms send notifications or automatically make changes to the resources being monitored based on customer-defined rules. This monitoring data can be used to determine whether additional instances should be launched in order to handle the increased load or stop under-utilized instances to save money.

In addition to monitoring the built-in metrics that come with AWS, custom metrics can be imported and monitored. Custom metrics can include detailed information about an Amazon EC2 Instance or data from servers running in an on-premises datacenter.

Amazon CloudWatch provides system-wide visibility into resource utilization, application performance, and operational health.

AWS CloudTrail is a service that enables governance, compliance, operational auditing, and risk auditing of an AWS account. With CloudTrail, customers can log, continuously monitor, and retain events related to API calls across an AWS infrastructure.

AWS CloudTrail provides a history of AWS API calls for an account. This includes API calls made through the AWS Management Console, AWS SDKs, command-line tools, and other AWS services. This history simplifies security analysis, resource change tracking, and troubleshooting.

AWS Config is a service that enables customers to assess, audit, and evaluate the configurations of their AWS resources. Config continuously monitors and records your AWS resource configurations and allows the automation of the evaluation of recorded configurations against desired configurations.

With AWS Config, review changes in configurations and relationships between AWS resources, dive into detailed resource configuration histories, and determine the overall compliance against the configurations specified in internal guidelines. This simplifies compliance auditing, security analysis, change management, and operational troubleshooting.

Resources to Review

The Status Health Dashboard: <http://status.aws.amazon.com/>

Amazon CloudWatch: <https://aws.amazon.com/cloudwatch/>

Amazon CloudWatch Logs: <http://docs.aws.amazon.com/AmazonCloudWatch/latest/logs/WhatIsCloudWatchLogs.html>

Amazon CloudWatch Events: <http://docs.aws.amazon.com/AmazonCloudWatch/latest/events/WhatIsCloudWatchEvents.html>

AWS CloudTrail: <http://docs.aws.amazon.com/awscloudtrail/latest/userguide/cloudtrail-user-guide.html>

Understanding Your Usage with Billing Reports: <http://docs.aws.amazon.com/awsaccountbilling/latest/aboutv2/billing-reports.html>

Analyzing Your Costs with Cost Explorer: <http://docs.aws.amazon.com/awsaccountbilling/latest/aboutv2/cost-explorer-what-is.html>

Using AWS Trusted Advisor as a Web Service: <https://docs.aws.amazon.com/awssupport/latest/user/trustedadvisor.html>

AWS Config: <https://aws.amazon.com/config/>

Amazon CloudWatch Plugin for collectd: <https://aws.amazon.com/blogs/aws/new-cloudwatch-plugin-for-collectd/>



Yes, this book is about systems operations. However, everyone should be aware of the principle behind the Well-Architected Framework, as it is part of all three associate level exams.

The AWS Well-Architected Framework: http://d0.awsstatic.com/whitepapers/architecture/AWS_Well-Architected_Framework.pdf

Troubleshoot a Classic Load Balancer—Response Code Metrics: <http://docs.aws.amazon.com/elasticloadbalancing/latest/classic/ts-elb-http-errors.html>

Security at Scale—Logging in AWS: <https://d0.awsstatic.com/whitepapers/aws-security-at-scale-logging-in-aws.pdf>

Exam Essentials

Be familiar with Amazon CloudWatch. Amazon CloudWatch is a monitoring service for AWS cloud resources and the applications that you run on AWS. You can use Amazon CloudWatch to collect and track metrics, collect and monitor log files, and set alarms.

Amazon CloudWatch can monitor AWS resources such as Amazon EC2 instances, Amazon DynamoDB tables, and Amazon RDS DB instances, as well as custom metrics generated by your applications and services and any log files your applications generate.

You can use Amazon CloudWatch to gain system-wide visibility into resource utilization, application performance, and operational health. You can use these insights to react and keep your application running smoothly.

Amazon CloudWatch Events has three components: Events, Rules, and Targets. Events indicate a change in an AWS environment. Targets process events. Rules match incoming events and route them to targets for processing.

Understand what Amazon CloudWatch Alarms is and what it can do. Amazon CloudWatch Logs lets you monitor and troubleshoot systems and applications using existing system, application, and custom log files.

With Amazon CloudWatch Logs, monitor logs in near real time for specific phrases, values, or patterns. Log data can be stored and accessed indefinitely in highly durable, low-cost storage without filling hard drives.

Be able to create or edit an Amazon CloudWatch Alarm. You can choose specific metrics to trigger the alarm and specify thresholds for those metrics. You can then set your alarm to change state when a metric exceeds a threshold that you have defined.

Know how to create a monitoring plan. Creating a monitoring plan involves answering some basic questions. What are your goals for monitoring? What resources will you monitor? How often will you monitor these resources? What monitoring tools will you use? Who will perform the monitoring tasks? Who should be notified when something goes wrong?

Know and understand custom metrics. You can now store your business and application metrics in Amazon CloudWatch. You can view graphs, set alarms, and initiate automated actions based on these metrics, just as you can for the metrics that Amazon CloudWatch already stores for your AWS resources.

Visibility for metrics above the Hypervisor requires an agent. Amazon CloudWatch can tell you CPU and RAM utilization at the Hypervisor level but it has no way of knowing what specific tasks or processes are affecting performance. CloudWatch can see disk I/O but cannot see disk usage. To do this requires an agent.

Be familiar with what an Amazon CloudWatch Alarm is and how it works. You can create a CloudWatch Alarm that watches a single metric. The alarm performs one or more actions based on the value of the metric relative to a threshold over a number of time periods. The action can be an Amazon EC2 action, an Auto Scaling action, or a notification sent to an Amazon SNS topic.

Know the three states of an Amazon CloudWatch Alarm. These are OK, ALARM, and INSUFFICIENT_DATA. If an alarm is in the OK state, a monitored metric is within the range

you have defined as acceptable. If the alarm is in the ALARM state, the metric has breached a threshold. If data is missing or incomplete, it is in the INSUFFICIENT_DATA state.

There are two levels of monitoring: Basic and Detailed. Basic Monitoring for Amazon EC2 sends CPU load, disk I/O, and network I/O metric data to Amazon CloudWatch in five-minute periods by default. To send metric data for an instance to CloudWatch in one-minute periods, enable Detailed Monitoring on the instance. Some services, like Amazon RDS, have Detailed Monitoring on by default.

Amazon CloudWatch performs two types of Amazon EC2 status checks: System and Instance. A System Status Check monitors the AWS systems required to use your instance to ensure that they are working properly. These checks detect problems with your instance that require *AWS involvement* to repair. An Instance Status Check monitors the software and network configuration of your individual instance. These checks detect problems that require *your involvement* to repair.

Be familiar with some common metrics used for monitoring. There are many metrics available, and not all of them are tested on the exam. Some are tested, however, and it is a good idea to know the common ones: VolumeQueueLength, DatabaseConnections, DiskQueueDepth, FreeStorageSpace, ReplicaLag, ReadIOPS, WriteIOPS, ReadLatency, WriteLatency, SurgeQueueLength, and SpilloverCount.

Know how to set up an Amazon CloudWatch Event subscription for Amazon RDS.

Amazon RDS uses the Amazon Simple Notification Service (Amazon SNS) to provide notification when an Amazon RDS event occurs. These notifications can be in any notification form supported by Amazon SNS for an AWS Region, such as an email, a text message, or a call to an HTTP endpoint. Details can be found here: http://docs.aws.amazon.com/AmazonRDS/latest/UserGuide/USER_Events.html.

Amazon ElastiCache has two engines available: Redis and Memcached. Memcached is multi-threaded, and Redis is single-threaded.

Know the Amazon EBS Volume Status Checks. Degraded and Severely Degraded performance means that the EBS Volume Status Check is in a Warning state and there is some I/O. Stalled or Not Available means that the EBS Volume is in the Impaired state, and there is no I/O.

Learn the metrics SurgeQueueLength and SpilloverCount. When the SurgeQueueLength is exceeded, spillover occurs. Requests are dropped without notifying end users. Customer experience is negatively impacted.

Test Taking Tip

The more a learner focuses on the meaning of information being presented, the more elaborately he or she will process the information. This principle is so obvious that it is easy to miss. What it means is this: When you are trying to drive a piece of information into your brain's memory systems, make sure you understand exactly what that information means.

Medina, John. *Brain Rules (Updated and Expanded): 12 Principles for Surviving and Thriving at Work, Home, and School* (p. 139). Pear Press. Kindle Edition.

As you prepare for your exam, make sure that you are worried less about facts and figures and more about how to accomplish tasks associated with systems operations. Worry less about the tool and more about what the tool does and how it integrates with other tools.

Yes, facts and figures are important. You will need to know things like what an Amazon SQS queue is and how it differs from an Amazon SNS topic. Instead of trying to memorize all of the things an Amazon SQS queue can do, study it in terms of how it solves a problem that you have.

If you understand how something like how Amazon SQS can improve your environment by decoupling computing resources, the details will fall into place on their own. It will no longer be a fact that you've memorized, but rather something that makes your life easier.

Exercises

By now you should have set up an account in AWS. If you haven't, now would be the time to do so. It is important to note that these exercises are in your AWS account and thus are not free.

Use the Free Tier when launching resources. For more information, see https://aws.amazon.com/sdm/optimization/server-side-test/free-tier/free_np/.

If you have not yet installed the AWS Command Line utilities, refer to Chapter 2, “Working with AWS Cloud Services,” Exercise 2.1 (Linux) or Exercise 2.2 (Windows).

The reference for the AWS CLI can be found at <http://docs.aws.amazon.com/cli/latest/reference/>.

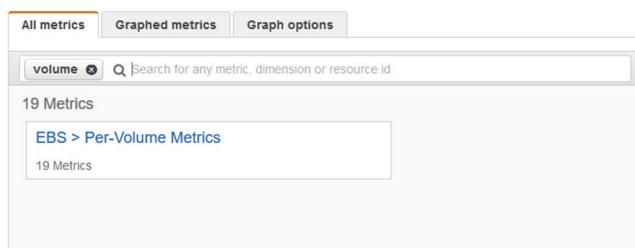
EXERCISE 9.1

Search for Available Metrics.

It is possible to search within all of the metrics in an account using targeted search terms. Metrics are returned that have matching results within their namespace, metric name, or dimensions.

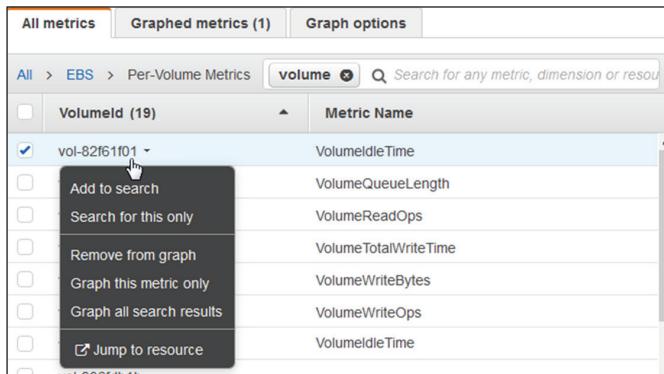
This exercise assumes the existence of an Amazon EC2 instance backed with an Amazon EBS volume.

1. Open the Amazon CloudWatch console: <https://console.aws.amazon.com/cloudwatch/>.
2. In the navigation pane, choose Metrics.
3. In the search field on the All Metrics tab, type **volume**, and press Enter. Refer to the following graphic. This shows all of the namespaces with metrics using this search term.

EXERCISE 9.1 (*continued*)

Select a namespace with the results of the search to view the metrics. Perform the following:

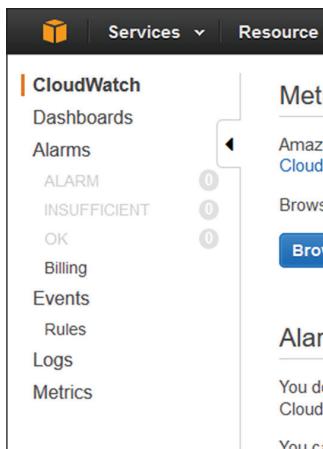
4. To graph one or more metrics, select the checkbox next to each metric. (To select all metrics, select the checkbox in the heading row of the table.)
5. To view one of the resources in its console, choose the resource ID, and then select Jump to Resource. The following graphic illustrates this step.
6. To view help for a metric, choose the metric name, and then select What Is This?

**EXERCISE 9.2****View Available Metrics for Running Amazon EC2 Instances by Namespace and Dimension Using the Amazon CloudWatch Console.**

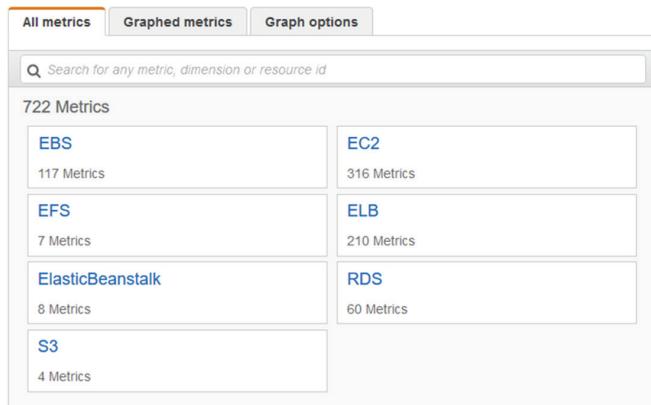
This exercise requires at least one running Amazon EC2 instance.

1. Open the Amazon CloudWatch console: <https://console.aws.amazon.com/cloudwatch/>.

2. Ensure that the region you are in has at least one running Amazon EC2 instance. The following graphic shows the Amazon CloudWatch navigation pane.



3. In the navigation pane, choose Metrics. The following graphic shows an example. Available metrics will vary.



4. Select Amazon EC2. This will open a window of available dimensions. See the following graphic for an example. Available dimensions will vary.

EXERCISE 9.2 (continued)

The screenshot shows the AWS CloudWatch Metrics console. At the top, there are three tabs: 'All metrics' (selected), 'Graphed metrics', and 'Graph options'. Below the tabs, the path 'All > EC2' is shown, followed by a search bar. The main area displays '316 Metrics' and is divided into two sections: 'By Auto Scaling Group' containing '124 Metrics' and 'Per-Instance Metrics' containing '192 Metrics'.

5. Select Per-Instance Metrics.

The All Metrics tab displays all metrics for that dimension in the namespace. In this window, the following can be performed:

6. To sort the table, use the column heading.
7. To graph a metric, select the checkbox next to the metric. (To select all metrics, select the checkbox in the heading row of the table.)
8. To filter by resource, choose the resource ID, and click Add to Search.
9. To filter by metric, choose the metric name, and click Add to Search.

The following graphic shows an example of this window. Available metrics will vary.

The screenshot shows the AWS CloudWatch Metrics console with the 'Per-Instance Metrics' table selected. The table has columns: 'Instance Name (192)', 'InstanceId', and 'Metric Name'. A context menu is open over the first row of the table, with the 'Add to search' option highlighted. Other options in the menu include 'Search for this only', 'Add to graph', 'Graph this metric only', 'Graph all search results', and 'Jump to resource'.

| Instance Name (192) | InstanceId | Metric Name |
|---------------------|------------|-------------------|
| my-instance | i-abbc12a7 | CPUUtilization |
| my-instance | | DiskReadBytes |
| my-instance | | DiskReadOps |
| my-instance | | DiskWriteBytes |
| my-instance | | DiskWriteOps |
| my-instance | | NetworkIn |
| my-instance | | NetworkOut |
| my-instance | | NetworkPacketsIn |
| my-instance | | NetworkPacketsOut |

EXERCISE 9.3**View Available Metrics by Namespace, Dimension, or Metric Using the AWS CLI.**

To view all of the metrics for Amazon EC2 in the AWS/EC2 namespace, use the following command from the AWS CLI:

```
aws cloudwatch list-metrics --namespace AWS/EC2
```

The JSON output will look something like this:

```
{
  "Metrics": [
    {
      "Namespace": "AWS/EC2",
      "Dimensions": [
        {
          "Name": "InstanceId",
          "Value": "i-0ab76393e31b62ec2"
        }
      ],
      "MetricName": "DiskWriteBytes"
    },
    {
      "Namespace": "AWS/EC2",
      "Dimensions": [
        {
          "Name": "InstanceId",
          "Value": "i-0ab76393e31b62ec2"
        }
      ],
      "MetricName": "DiskReadOps"
    },
    ...
  ]
}
```

The following graphic shows the output in table format instead of JSON. Here is the command:

```
aws cloudwatch list-metrics --namespace AWS/EC2 --output table
```

EXERCISE 9.3 (continued)

| ListMetrics | |
|------------------|---------------------|
| Metrics | |
| MetricName | Namespace |
| CPUCreditBalance | AWS/EC2 |
| Dimensions | |
| Name | Value |
| InstanceId | i-0ab76393e31b62ec2 |
| Metrics | |
| MetricName | Namespace |
| CPUCreditUsage | AWS/EC2 |
| Dimensions | |
| Name | Value |
| InstanceId | i-0ab76393e31b62ec2 |
| Metrics | |
| MetricName | Namespace |
| NetworkPacketsIn | AWS/EC2 |
| Dimensions | |
| Name | Value |
| InstanceId | i-0ab76393e31b62ec2 |
| Metrics | |

EXERCISE 9.4**List All Available Metrics for a Specific Resource.**

Specify the AWS/EC2 namespace and the InstanceId dimension to view the results for a single instance. In this example, the instance ID is i-0ab76393e31b62ec2.

```
aws cloudwatch list-metrics --namespace AWS/EC2 --dimensions Name=InstanceId,Value= i-0ab76393e31b62ec2
```

EXERCISE 9.5**List all Resources that Use a Single Metric.**

Specify the AWS/EC2 namespace and the metric CPUUtilization to view the results for the single metric.

```
aws cloudwatch list-metrics --namespace AWS/EC2 --metric-name CPUUtilization
```

EXERCISE 9.6**Get Statistics for a Specific Resource.**

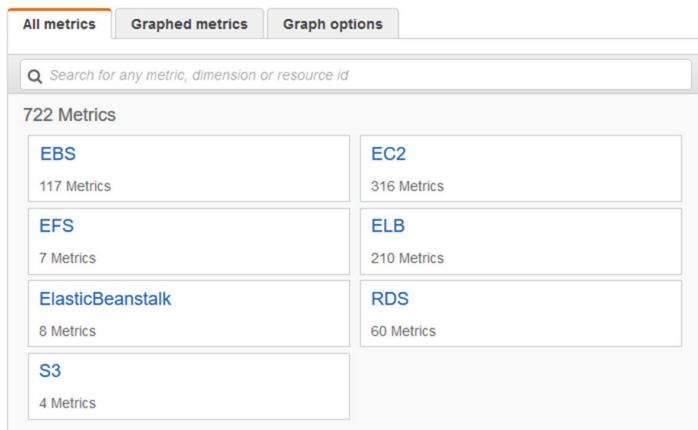
The following exercise displays the maximum CPU utilization of a specific Amazon EC2 instance using the Amazon CloudWatch console.

Requirements

You must have an ID of an Amazon EC2 instance. Retrieve the instance ID using the AWS Management Console or the `describe-instances` command.

By default, basic monitoring is enabled and collects metrics at five-minute intervals. Detailed monitoring can be enabled to capture data points at one-minute intervals. This will incur additional charges.

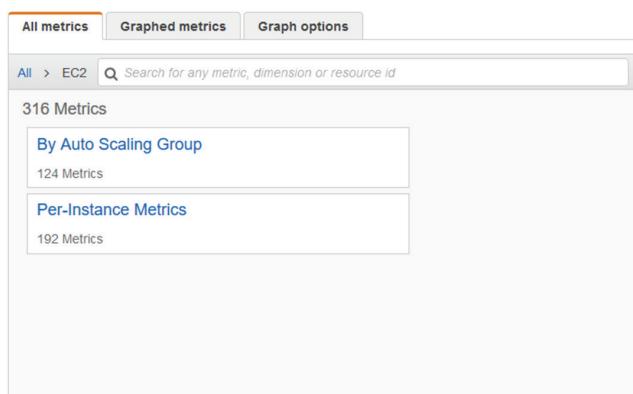
1. Open the Amazon CloudWatch console:
<https://console.aws.amazon.com/cloudwatch/>
2. In the navigation pane, choose Metrics.
3. Select the Amazon EC2 metric namespace. The following graphic shows an example of the Metrics window. Available metrics will vary.



The screenshot shows the 'Graphed metrics' tab selected in the CloudWatch Metrics console. A search bar at the top is empty. Below it, a section titled '722 Metrics' displays various service namespaces with their respective metric counts:

| Service Namespace | Metrics |
|-------------------|-------------|
| EBS | 117 Metrics |
| EFS | 7 Metrics |
| ElasticBeanstalk | 8 Metrics |
| S3 | 4 Metrics |
| EC2 | 316 Metrics |
| ELB | 210 Metrics |
| RDS | 60 Metrics |

4. After choosing the Amazon EC2 metric namespace, a new window will open. See the following graphic for an example. Available dimensions will vary. Select the Per-Instance Metrics dimension.

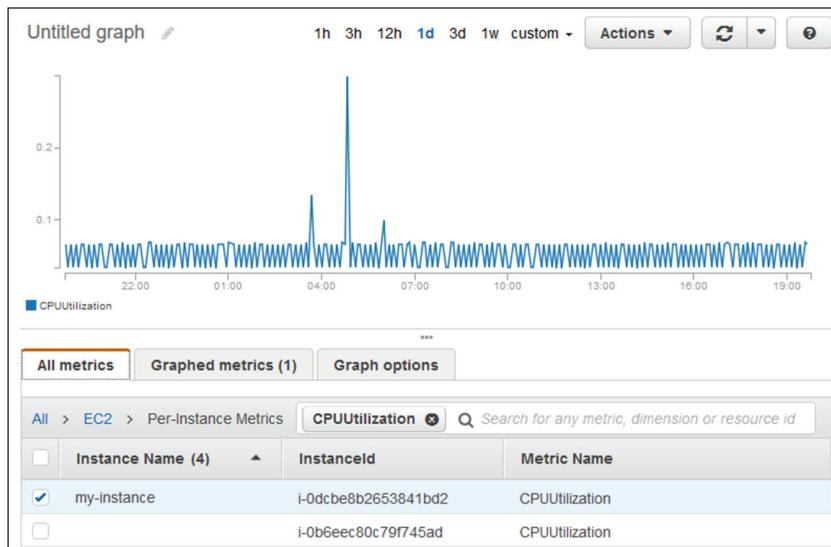


The screenshot shows the 'Graphed metrics' tab selected in the CloudWatch Metrics console. The path 'All > EC2' is visible in the top left. A search bar is empty. Below it, a section titled '316 Metrics' displays two categories:

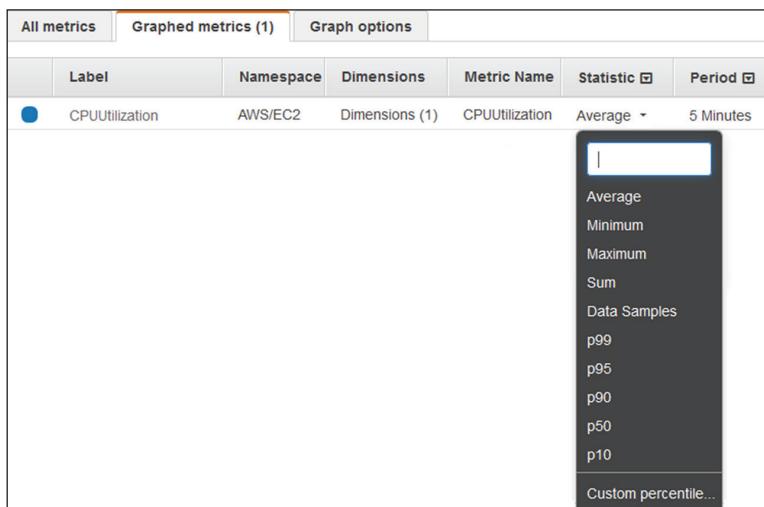
| Dimension | Metrics |
|-----------------------|-------------|
| By Auto Scaling Group | 124 Metrics |
| Per-Instance Metrics | 192 Metrics |

EXERCISE 9.6 (continued)

5. In the search field, type **CPUUtilization** and press Enter.
6. Select the row for the specific instance that displays a graph for the CPUUtilization metric for the instance. The following graphic shows a graph for an instance named my-instance. Available instance names and IDs will vary.



7. Change the statistic by choosing the Graphed Metrics tab.
8. Choose either the column heading, Statistic, or an individual value in the Statistic column and then select Maximum. The following graphic shows an example of this menu.



EXERCISE 9.7**Get CPU Utilization for a Single Amazon EC2 Instance from the Command Line.****Requirements**

You must have an ID of an Amazon EC2 instance. Retrieve the instance ID using the management console or the describe-instances command.

By default, basic monitoring is enabled and collects metrics at five-minute intervals. Detailed monitoring can be enabled to capture data points at one-minute intervals. This will incur additional charges.

From the AWS CLI, you will use the command `get-metric-statistics` and the metric name `CPUUtilization`.

The returned statistics are six-minute values for the requested 24-hour time interval. Each value represents the maximum CPU utilization percentage for the specified instance for a particular six-minute time period.



Data points are not returned in chronological order.

In Linux, the backslashes (\) allow commands to span more than one line and are presented here for readability.

```
aws cloudwatch get-metric-statistics \
--namespace AWS/EC2 \
--metric-name CPUUtilization \
--dimensions Name=InstanceId,Value=i-0e022540492655edd \
--statistics Maximum \
--start-time 2017-05-01T09:28:00 \
--end-time 2017-05-02T09:28:00 \
--period 360
--output json
```

The output in JSON looks like this:

```
{
  "Datapoints": [
    {
      "Timestamp": "2017-05-01T13:58:00Z",
      "Maximum": 0.67,
      "Unit": "Percent"
```

EXERCISE 9.7 (continued)

```
 },
{
    "Timestamp": "2017-05-02T01:46:00Z",
    "Maximum": 0.83,
    "Unit": "Percent"
},
{
    "Timestamp": "2017-05-02T05:52:00Z",
    "Maximum": 0.67,
    "Unit": "Percent"
},
{
    "Timestamp": "2017-05-01T16:22:00Z",
    "Maximum": 0.49,
    "Unit": "Percent"
},
...
{
    "Timestamp": "2017-05-02T06:52:00Z",
    "Maximum": 0.67,
    "Unit": "Percent"
}
],
"Label": "CPUUtilization"
}
```

The following graphic shows the output in table format instead of JSON.

```
aws cloudwatch get-metric-statistics \
--namespace AWS/EC2 \
--metric-name CPUUtilization \
--dimensions Name=InstanceId,Value=i-0e022540492655edd \
--statistics Maximum \
--start-time 2017-05-01T09:28:00 \
--end-time 2017-05-02T09:28:00 \
--period 360
--output table
```

| GetMetricStatistics | | |
|---------------------|----------------------|------------|
| Label | CPUUtilization | Datapoints |
| Maximum | Timestamp | Unit |
| 0.67 | 2017-05-01T13:58:00Z | Percent |
| 0.83 | 2017-05-02T01:46:00Z | Percent |
| 0.67 | 2017-05-02T05:52:00Z | Percent |
| 0.49 | 2017-05-01T16:22:00Z | Percent |
| 0.67 | 2017-05-01T11:04:00Z | Percent |
| 0.68 | 2017-05-01T20:28:00Z | Percent |
| 0.67 | 2017-05-02T08:16:00Z | Percent |
| 0.67 | 2017-05-01T20:58:00Z | Percent |
| 0.66 | 2017-05-01T15:40:00Z | Percent |
| 1.8 | 2017-05-02T08:46:00Z | Percent |
| 0.66 | 2017-05-01T11:34:00Z | Percent |
| 0.67 | 2017-05-01T23:22:00Z | Percent |

EXERCISE 9.8

Create a Billing Alert.

In order to create a billing alarm, billing alerts must first be enabled. This only needs to be done once and, once enabled, billing alerts cannot be disabled.

1. Sign in to the AWS Management Console and open the AWS Billing and Cost Management console: <https://console.aws.amazon.com/billing/home#/>.
2. On the navigation pane, choose Preferences.
3. Select the Receive Billing Alerts checkbox.
4. Choose Save Preferences.

EXERCISE 9.9

Create a Billing Alarm.

To create a billing alarm, billing alerts must be turned on first. These steps send alerts to a single email address.

1. Open the Amazon CloudWatch console: <https://console.aws.amazon.com/cloudwatch/>.
2. If necessary, change the region on the navigation bar to US East (N. Virginia). All billing metric data is stored in this region, even for resources in other regions.
3. On the navigation pane under Metrics, choose Billing.
4. In the list of billing metrics, select the checkbox next to Currency USD for the metric named EstimatedCharges, as shown in the following graphic.

| Billing > Total Estimated Charge | | |
|---|------------------|--|
| Currency | Metric Name | |
| <input checked="" type="checkbox"/> USD | EstimatedCharges | |

EXERCISE 9.9 (continued)

5. Choose Create Alarm.

Define the alarm as follows:

6. For the alarm to trigger as soon as an account goes over the free tier, set When my total AWS charges for the month exceed to \$.01. The AWS Free Tier applies to participating services across the following AWS Regions: US East (Northern Virginia), US West (Oregon), US West (Northern California), Canada (Central), EU (London), EU (Ireland), EU (Frankfurt), Asia Pacific (Singapore), Asia Pacific (Tokyo), Asia Pacific (Sydney), and South America (Sao Paulo). This will result in a notification being sent as soon as the first charge is incurred. Otherwise, set it to a custom amount, and a notification will be sent when the chosen amount's threshold is reached. Refer to the following graphic:

When my total AWS charges for the month
exceed: \$.01 USD
send a notification to: Select a notification list New list

7. Choose the New List link next to the Send a Notification To box.
8. When prompted, enter a valid email address.
9. Choose Create Alarm.
10. In the Confirm New Email Addresses dialog box, confirm the email address. When choosing I will do it later, the alarm will remain in the Pending confirmation status until the email address is confirmed. Alarms will not send an alert until the email address is confirmed. To view the status of the alarm, choose Alarms in the navigation pane. Refer to the following graphic to see an alarm with a Config Status of Pending confirmation.

| State | Name | Threshold | Config Status |
|-------|--------------|----------------------------------|----------------------|
| ALARM | BillingAlarm | EstimatedCharges > 0 for 6 hours | Pending confirmation |

EXERCISE 9.10**Create an Amazon CloudWatch Dashboard.**

To get started with Amazon CloudWatch dashboards, first create a dashboard.

1. Open the Amazon CloudWatch console: <https://console.aws.amazon.com/cloudwatch/>.
2. In the navigation pane, choose Dashboards.

3. Choose Create Dashboard.
 4. In the Create New Dashboard dialog box, type a name for the dashboard, and then choose Create Dashboard.
 5. Add a graph to the dashboard by choosing Metric Graph, and then click Configure. Then, in the Add Metric Graph dialog box, select the metrics to graph, and then choose Create Widget.
 6. Add a text block to the dashboard by choosing Text Widget, and then click Configure. In the New Text Widget dialog box, for Markdown, add and format text using Markdown, and then choose Create Widget.
 7. Choose Save Dashboard.
-

Review Questions

1. Which of the following requires a custom Amazon CloudWatch metric to monitor?
 - A. Amazon EC2 CPU Utilization
 - B. Amazon EC2 Disk IO
 - C. Amazon EC2 Memory Utilization
 - D. Amazon EC2 Network IO
2. While using Auto Scaling with an ELB in front of several Amazon EC2 instances, you want to configure Auto Scaling to remove one instance when CPU Utilization is below 20 percent. How is this accomplished?
 - A. Configure Amazon CloudWatch Logs to send a notification to the Auto Scaling Group when CPU utilization is less than 20 percent, and configure the Auto Scaling policy to remove the instance.
 - B. Configure Amazon CloudWatch to send a notification to the Auto Scaling Group when the aggregated CPU Utilization is less than 20 percent, and configure the Auto Scaling policy to remove the instance.
 - C. Monitor the Amazon EC2 instances with Amazon CloudWatch, and use Auto Scaling to remove an instance with scheduled actions.
 - D. Configure Amazon CloudWatch to generate an email using Amazon SNS when CPU utilization is less than 20 percent. Log into the console, and lower the desired capacity number inside Auto Scaling to remove the instance.
3. Your company has configured the custom metric upload with Amazon CloudWatch, and it has authorized employees to upload data using AWS CLI as well as AWS SDK. How can you track API calls made to CloudWatch?
 - A. Use AWS CloudTrail to monitor the API calls.
 - B. Create an IAM role to allow users who assume the role to view the data using an Amazon S3 bucket policy.
 - C. Enable logging with Amazon CloudWatch to capture metrics for the API calls.
 - D. Enable detailed monitoring with Amazon CloudWatch.
4. Of the services listed here, which provide detailed monitoring without extra charges being incurred? (Choose two.)
 - A. AWS Auto Scaling
 - B. Amazon Route 53
 - C. Amazon Elastic Map Reduce
 - D. Amazon Relational Database Service
 - E. Amazon Simple Notification Service

5. You have configured an ELB Classic Load Balancer to distribute traffic among multiple Amazon EC2 instances. Which of the following will aid troubleshooting efforts related to back-end servers?
 - A. HTTPCode_Backend_2XX
 - B. HTTPCode_Backend_3XX
 - C. HTTPCode_Backend_4XX
 - D. HTTPCode_Backend_5XX
6. What is the minimum time interval for data that Amazon CloudWatch receives and aggregates?
 - A. Fifteen seconds
 - B. One minute
 - C. Three minutes
 - D. Five minutes
7. Using the Free Tier, what is the frequency of updates received by Amazon CloudWatch?
 - A. Fifteen seconds
 - B. One minute
 - C. Three minutes
 - D. Five minutes
8. The type of monitoring automatically available in five-minute periods is called what?
 - A. Elastic
 - B. Simple
 - C. Basic
 - D. Detailed
9. You have created an Auto Scaling Group using the AWS CLI. You now want to enable Detailed Monitoring for this group. How is this accomplished?
 - A. Enable Detailed Monitoring from the AWS console.
 - B. When creating an alarm on the Auto Scaling Group, Detailed Monitoring is automatically activated.
 - C. When creating Auto Scaling Groups using the AWS CLI or API, Detailed Monitoring is enabled for Auto Scaling by default.
 - D. Auto Scaling Groups do not support Detailed Monitoring.

- 10.** There are 10 Amazon EC2 instances running in multiple regions using an internal memory management tool to capture log files and send them to Amazon CloudWatch in US-West-2. Additionally, you are using the AWS CLI to configure CloudWatch to use the same namespace and metric in all regions. Which of the following is true?
- A.** Amazon CloudWatch will receive and aggregate statistical data based on the namespace and metric.
 - B.** Amazon CloudWatch will process the data only for the server that responds first and ignore the other regions.
 - C.** Amazon CloudWatch will process the statistical data for the most recent response regardless of region and overwrite other data.
 - D.** Amazon CloudWatch cannot receive data across regions.
- 11.** You have misconfigured an Amazon EC2 instance's clock and are sending data to Amazon CloudWatch via the API. Because of the misconfiguration, logs are being sent 60 minutes in the future. Which of the following is true?
- A.** Amazon CloudWatch will process the data.
 - B.** It is not possible to send data from the future.
 - C.** It is not possible to send data manually to Amazon CloudWatch.
 - D.** Agents cannot send data for more than 60 minutes in the future.
- 12.** You have a system that sends data to Amazon CloudWatch every five minutes for tracking/monitoring. Which of these parameters is required as part of the put-metric-data request?
- A.** Key
 - B.** Namespace
 - C.** Metric Name
 - D.** Timestamp
- 13.** To monitor API calls against AWS use _____ to capture the history API requests and use _____ to respond to operational changes in real time.
- A.** AWS Config; Amazon Inspector
 - B.** AWS CloudTrail; AWS Config
 - C.** AWS CloudTrail; Amazon CloudWatch Events
 - D.** AWS Config; AWS Lambda

Chapter 10



High Availability

THE AWS CERTIFIED SYSOPS ADMINISTRATOR - ASSOCIATE EXAM TOPICS COVERED IN THIS CHAPTER MAY INCLUDE, BUT ARE NOT LIMITED TO, THE FOLLOWING:

Domain 1.0 Monitoring and Metrics

- ✓ **1.1 Demonstrate ability to monitor availability and performance**

Content may include the following:

- Monitoring endpoints health checks and routing traffic to healthy endpoints
- Enabling Amazon Elastic Compute Cloud (Amazon EC2) auto-recovery

Domain 2.0 High Availability

- ✓ **2.1 Implement scalability and elasticity based on scenario**

Content may include the following:

- Scalability of Amazon Elastic Compute Cloud (Amazon EC2) with Auto Scaling
- Scaling Amazon RDS database for performance
- Elastic Load Balancing
- Session state management

- ✓ **2.2 Ensure level of fault tolerance based on business needs**

Content may include the following:

- Using Amazon Simple Queue Service (Amazon SQS) to assist in making applications fault tolerant
- Using Amazon Simple Notification Service (Amazon SNS) to fan-out an application



- Delivering high availability to your Amazon RDS with Multi-AZ deployments
- Enabling cross-region replication for Amazon Simple Storage Service (Amazon S3)
- Making AWS Direct Connect highly available
- AWS Direct Connect failover to backup VPN

Domain 5.0 High Availability

✓ 5.3 Manage backup and disaster recovery processes

Content may include the following:

- Different backup and restore methods available on AWS

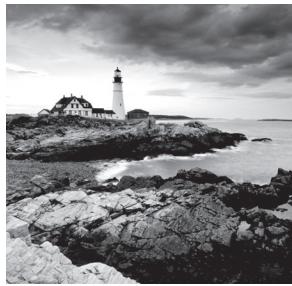
Domain 7.0 Networking

✓ 7.1 Demonstrate ability to implement networking features of AWS

✓ 7.2 Demonstrate ability to implement connectivity features of AWS

Content may include the following:

- Different Failover scenarios using Amazon Route 53



Introduction to High Availability

This chapter covers high availability on AWS. In previous chapters, you were introduced to compute, networking, databases, and storage on AWS. What you haven't been exposed to are some of the fully managed services that can help you deploy and maintain a highly available and scalable application.

What is *high availability*? Availability refers to the amount of time your system is in a functioning condition. In general terms, your availability is referred to as 100 percent minus your system's downtime. Since events that may disrupt your system's availability are never entirely predictable, there are always ways to make an application more available. Improving availability typically leads to increased cost, however. When considering how to make your environment more available, it's important to balance the cost of the improvement with the benefit to your users. Does high availability mean that you ensure your application is always alive/reachable, or does it mean that the application is servicing requests within an acceptable level of performance? A way to make your application highly available and scalable is to decouple components so that they scale independently and can survive minimal disruptions. See Table 10.1 for varying high availability performance levels.

TABLE 10.1 Levels of High Availability

| Percent of Uptime | Max Downtime per Year | Equivalent Downtime per Day |
|-------------------|-----------------------|-----------------------------|
| 90% (1 nine) | 36.5 days | 2.4 hours |
| 99% (2 nines) | 3.65 days | 14 minutes |
| 99.9% (3 nines) | 8.76 hours | 86 seconds |
| 99.99% (4 nines) | 52.6 minutes | 8.6 seconds |
| 99.999% (5 nines) | 5.25 minutes | .86 seconds |

In this chapter, you will be introduced to Amazon Simple Queue Services (Amazon SQS) and Amazon Simple Notification Services (Amazon SNS), which can be used to decouple applications and retain transactions. In addition to decoupling tiers within an application, these services can help deliver high availability and fault tolerance to your application. This chapter will also cover additional strategies to keep your infrastructure highly available, redundant, and fault tolerant.

Amazon Simple Queue Service

Amazon Simple Queue Service (Amazon SQS) is a web service that gives you access to message queues that store messages waiting to be processed. With Amazon SQS, you can quickly build message queuing applications that can run on any computer. You can use the service to move data between diverse, distributed application components without losing messages and without requiring each component always to be available.

Amazon SQS can help you build a distributed application with decoupled components by working closely with the Amazon Elastic Compute Cloud (Amazon EC2) and other AWS infrastructure services. You can access the service via the Amazon SQS console, the AWS Command Line Interface (AWS CLI), a generic web services Application Programming Interface (API), and any programming language that the AWS Software Development Kit (SDK) supports. Amazon SQS supports both standard and First-In, First-Out (FIFO) queues.

Using Amazon Simple Queue Service to Decouple an Application

Think of a queue as a temporary repository for messages that are awaiting processing. Using Amazon SQS, you can decouple the components of an application so that they run independently of each other, with Amazon SQS easing message management between components. The queue acts as a buffer between the component producing and saving data and the component receiving the data for processing. This means that the queue resolves issues that arise if the producer (for example, a web front end) is producing work faster than the consumer (such as the application worker) can process it or if the producer or consumer is only intermittently connected to the network.

Amazon SQS is designed to deliver your message at least once and supports multiple readers and writers interacting with the same queue. A single queue can be used simultaneously by many distributed application components with no need for those components to coordinate with each other to share the queue.

Amazon SQS is engineered always to be available and deliver messages. This is achieved by the system being distributed between multiple machines and multiple facilities. Due

to this highly distributed architecture, there is a trade-off—when using standard queues, Amazon SQS does not guarantee FIFO delivery of messages. This may be okay for many distributed applications, as long as each message can stand on its own and as long as all messages are delivered. In that scenario, the order is not important. If your system requires that order be preserved, you can place sequencing information in each message so that you can reorder the messages when the queue returns them.

You can have as many queues with as many messages as you like in the Amazon SQS system. A queue can be empty if you haven't sent any messages to it, or if you have deleted all of the messages from it.

You assign a name to each of your queues. You can get a list of all of your queues or a subset of your queues that share the same initial characters in their names (for example, you could get a list of all of your queues whose names start with Q3).

Use Cases for Amazon SQS

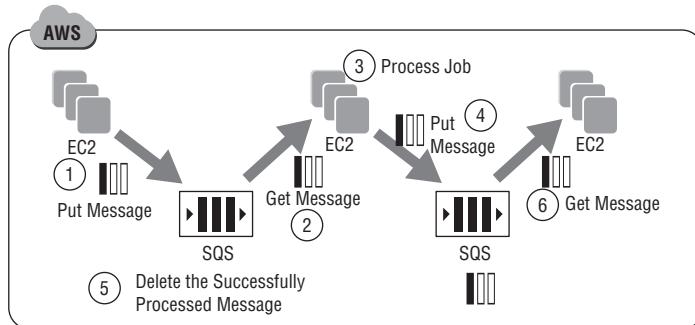
- Integrate Amazon SQS with other AWS infrastructure web services to make applications more reliable and flexible.
- Use an Amazon SQS queue as a queue of work, where each message is a task that needs to be completed by a process. One or many computers can read tasks from the queue and perform them.
- Have Amazon SQS help a browser-based application receive notifications from a server. The application server can add the notifications to a queue, which the browser can poll even if there is a firewall between them.
- Keep notifications of significant events in a business process in an Amazon SQS queue. Each event can have a corresponding message in a queue, and applications that need to be aware of the event can read and process the messages.

Amazon SQS is engineered always to be available and deliver messages. You can delete a queue at any time, whether it is empty or not. Amazon SQS queues retain messages for a set period of time. The default setting is four days; however, you can configure a queue to retain messages for up to 14 days.

The Queuing Chain Pattern

You can achieve loose coupling of systems by using queues between systems and exchanging messages that transfer jobs. This enables asynchronous linking of systems. This method lets you increase the number of virtual servers that receive and process the messages in parallel. If there is no image to process, you can configure Auto Scaling to terminate the servers that are in excess. See Figure 10.1 for an example of a queuing chain pattern.

Although you can use this pattern without cloud technology, the queue itself is provided as an AWS Cloud service (Amazon SQS), which makes it easier for you to use this pattern.

FIGURE 10.1 Demonstrates a queuing chain pattern

The Problem to Be Solved

Taking image processing as an example, the sequential operations of uploading, storing, and encoding the image, creating a thumbnail, and copyrighting are tightly linked. This tight linkage complicates the recovery operations when there has been a failure.

How This Benefits Your Application

- Use asynchronous processing to return responses quickly.
- Structure the system through loose coupling of Amazon EC2 instances.
- Handle performance and service requirements through merely increasing or decreasing the number of Amazon EC2 instances used in job processing.
- A message remains in the queue service even if an Amazon EC2 instance fails, which enables processing to be continued immediately upon recovery of the Amazon EC2 instance and facilitates a system that is resistant to failure.



AWS reserves the right to delete a queue without notification if one of the following actions hasn't been performed on it for 30 consecutive days: SendMessage, ReceiveMessage, DeleteMessage, GetQueueAttributes, SetQueueAttributes, AddPermission, or RemovePermission.

Message Order

A standard queue makes a best effort to preserve order in messages, but due to the distributed nature of the queue, AWS cannot guarantee that you will receive messages in the exact order that you sent them. If your system requires that order be preserved, we recommend using a FIFO queue.

At-Least-Once Delivery

Amazon SQS stores copies of your messages on multiple servers for redundancy and high availability. On rare occasions, one of the servers storing a copy of a message might be

unavailable when you receive or delete the message. If that occurs, the copy of the message will not be deleted on that unavailable server, and you might get that message copy again when you receive messages. Because of this, you must design your application to be idempotent (that is, it must not be adversely affected if it processes the same message more than once). The point being made is that Amazon SQS will deliver a message at least once, and may in some cases deliver the message again. For more information, refer to visibility timeout in this chapter.

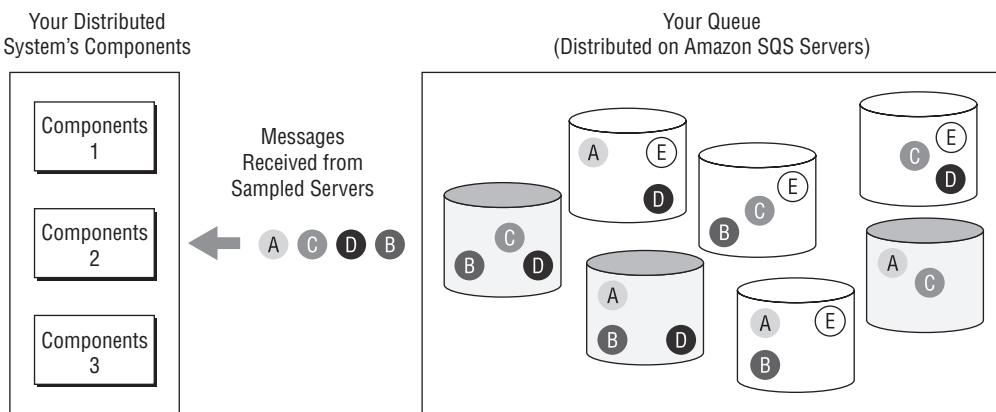
Message Sample

When you retrieve messages from the queue, Amazon SQS samples a subset of the servers and returns messages from just those servers. Amazon SQS supports two types of polling methods:

- *Short polling* means a sample of queues is polled and a particular receive request might not return all of your messages, whereas a subsequent request will. If you keep retrieving from your queues, Amazon SQS will sample all of the servers and you will receive all of your messages. This can be CPU intensive for your application.
- *Long polling* reduces the number of empty responses by allowing Amazon SQS to wait until a message is available in the queue before sending a response. Unless the connection times out, the response to the `ReceiveMessage` request contains at least one of the available messages, up to the maximum number of messages specified in the `ReceiveMessage` action.

Figure 10.2 shows an example of a short poll of the queue and messages being returned after a receive request. Amazon SQS samples several of the servers (in gray) and returns the messages from those servers (Message A, C, D, and B). Message E is not returned to this particular request, but it would be returned to a subsequent request.

FIGURE 10.2 Short poll



Visibility Timeout

Visibility timeout is the period of time that a message is invisible to the rest of your application after an application component gets it from the queue. During the visibility timeout, the component that received the message usually processes it and then deletes it from the queue. This prevents multiple components from processing the same message.

Here is how it works:

1. When a message is received, it becomes “locked” while being processed. This prevents it from being processed by other components.
2. The component receiving the message processes it and then deletes it from the queue.
3. If message processing fails, the lock expires and the message becomes available again (fault tolerance).

When the application needs more time for processing, the visibility timeout can be changed dynamically via the `ChangeMessageVisibility` operation.

Standard Queues

Amazon SQS offers standard queues as the default queue type. A *standard queue* allows you to have a nearly unlimited number of transactions per second. Standard queues support at-least-once message delivery. Occasionally (because of its highly-distributed architecture), however, more than one copy of a message might be delivered out of order. Standard queues provide best-effort ordering, versus a First-In, First-Out (FIFO) queue. This ensures that messages are delivered in the same order as they’re sent.

You can use standard message queues in many scenarios, as long as your application can process messages that arrive more than once and out of order. For example:

- **Decouple live user requests from intensive background work:** Let users upload media while resizing or encoding it.
- **Allocate tasks to multiple worker nodes:** Process a high number of credit card validation requests.
- **Batch messages for future processing:** Schedule multiple entries to be added to a database.

First-In, First-Out Queues

A *First-In, First-Out (FIFO)* queue has all of the capabilities of the standard queue. FIFO queues are designed to enhance messaging between applications when the order of operations and events are critical or where duplicates can’t be tolerated. FIFO queues also provide exactly-once processing and are limited to 300 Transactions Per Second (TPS).

FIFO queues are designed to enhance messaging between applications when the order of operations and events is critical. For example:

- Ensuring that user-entered commands are executed in the right order
- Displaying the correct product price by sending price modifications in the right order
- Preventing a student from enrolling in a course before registering for an account

Dead Letter Queues

A *Dead Letter Queue (DLQ)* is an Amazon SQS queue that you configure to receive messages from other Amazon SQS queues, referred to as “source queues.” Typically, you set up a DLQ to receive messages after a maximum number of processing attempts has been reached. A DLQ provides the ability to isolate messages that could not be processed.

A DLQ is just like any other Amazon SQS queue—messages can be sent to it and received from it like any other Amazon SQS queues. You can create a DLQ from the Amazon SQS API and the Amazon SQS area in the AWS Management Console.



The name of a FIFO queue must end with the `.fifo` suffix. The suffix counts towards the 80-character queue name limit. To determine whether a queue is FIFO, you can check whether the queue name ends with the suffix.

Shared Queues

A developer associates an access policy statement (specifying the permissions being granted) with the queue to be shared. Amazon SQS provides APIs to create and manage the access policy statements: `AddPermission`, `RemovePermission`, `SetQueueAttributes`, and `GetQueueAttributes`. Refer to the latest API specification for more details.

Amazon SQS in each region is totally independent in message stores and queue names; therefore, the messages cannot be shared between queues in different regions.

Queues can be shared in the following ways:

- With other AWS accounts
- Anonymously
- A *permission* gives access to another person to use your queue in some particular way.
- A *policy* is the actual document that contains the permissions you granted.



Keep in mind that the queue owner is responsible for all costs related to the queue. Therefore, you probably want to limit anonymous access in some other way (for example, by time or IP address).

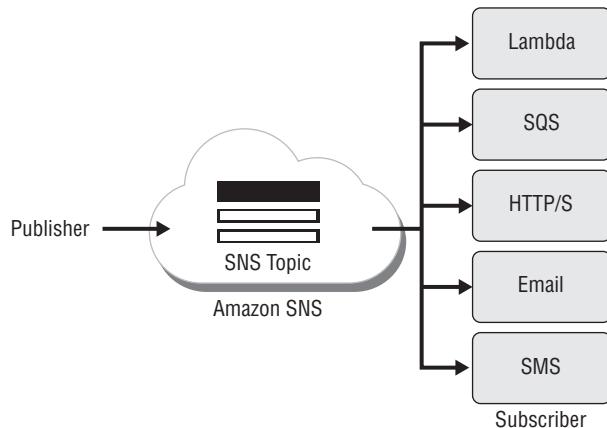
Characteristics of Amazon SQS

- Configurable settings per queue
- Message order is not guaranteed with standard queues.
- Message order is preserved with FIFO queues.
- Messages can be deleted while in queue.
- Messages can contain up to 256 KB of text data, including XML, JSON, and unformatted text.
- FIFO and standard queues support server-side encryption (as of April 2017).

Amazon Simple Notification Service

Amazon Simple Notification Service (Amazon SNS) is a web service that coordinates and manages the delivery or sending of messages to subscribing endpoints or clients. In Amazon SNS, there are two types of clients—*publishers* and *subscribers*, which are also referred to as *producers* and *consumers*. Publishers communicate asynchronously with subscribers by producing and sending a message to a topic, which is a logical access point and communication channel. Subscribers (for example web servers, email addresses, Amazon SQS queues, and AWS Lambda functions) consume or receive the message or notification over one of the supported protocols (such as Amazon SQS, HTTP/S, email, Short Message Service [SMS], or AWS Lambda) when they are subscribed to the topic. See Figure 10.3 for details on the supported protocols.

FIGURE 10.3 Protocols supported by Amazon SNS



When using Amazon SNS, you create a topic and control access to it by defining resource policies that determine which publishers and subscribers can communicate with the topic. Refer to Chapter 3, “Security and AWS Identity and Access Management (IAM),” for more details on resource policies.

A publisher sends messages to topics that they have created or to topics to which they have permission to publish. Instead of including a specific destination address in each message, a publisher sends a message to the topic. Amazon SNS matches the topic to a list of subscribers that have subscribed to that topic and delivers the message to each of those subscribers. Each topic has a unique name that identifies the Amazon SNS endpoint for publishers to post messages and subscribers to register for notifications. Subscribers receive all messages published to the topics to which they subscribe, and all subscribers to a topic receive the same messages. The service is accessed via the Amazon SQS console, the AWS CLI, a generic web services API, and any programming language that the AWS SDK supports.



In contrast to Amazon SQS, which is a one-to-one relationship of messages to a worker, Amazon SNS has a one-to-many relationship between a topic and its subscribers.

Mobile Push Messaging

Amazon SNS lets you push messages to mobile devices or distributed services via API or an easy-to-use management console. You can seamlessly scale from a handful of messages per day to millions of messages or more.

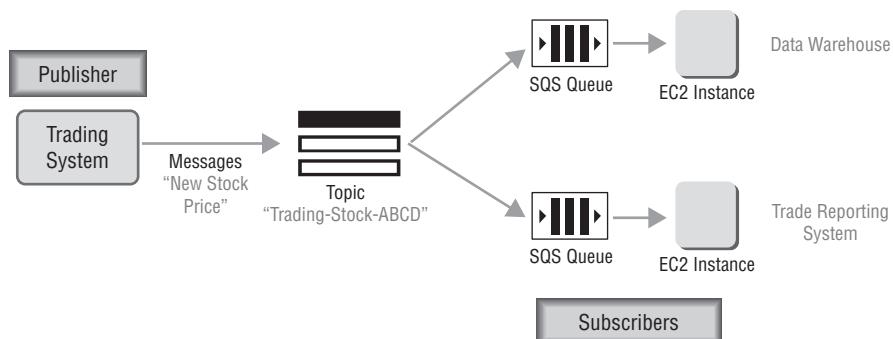
With Amazon SNS you can publish a message once and deliver it one or more times. You can choose to direct unique messages to individual Apple, Android, or Amazon devices or broadcast deliveries to many mobile devices with a single publish request.

Amazon SNS allows you to group multiple recipients using topics. A *topic* is an “access point” for allowing recipients to subscribe dynamically to identical copies of the same notification. One topic can support deliveries to multiple endpoint types; for example, you can group together iOS, Android, and SMS recipients. When you publish once to a topic, Amazon SNS delivers appropriately formatted copies of your message to each subscriber.

Amazon SNS Fan-Out Scenario

In a *fan-out scenario*, an Amazon SNS message is sent to a topic and then replicated and pushed to multiple Amazon SQS queues, HTTP/S endpoints, or email addresses. This allows for parallel asynchronous processing. For example, you could develop an application that sends an Amazon SNS message to a topic whenever a change in a stock price occurs. Then the Amazon SQS queues that are subscribed to that topic would receive identical notifications for the new stock price. The Amazon EC2 server instance attached to one of the queues could handle the trade reporting, while the other server instance could be attached to a data warehouse for analysis and the trending of the stock price over a specified period of time. Figure 10.4 describes the fan-out scenario.

FIGURE 10.4 Fan-out scenario



To have Amazon SNS deliver notifications to an Amazon SQS queue, a developer should subscribe to a topic specifying “Amazon SQS” as the transport and a valid Amazon SQS queue as the endpoint. In order to allow the Amazon SQS queue to receive notifications from Amazon SNS, the Amazon SQS queue owner must subscribe the Amazon SQS queue to the topic for Amazon SNS. If the user owns both the Amazon SNS topic being subscribed to and the Amazon SQS queue receiving the notifications, nothing further is required. Any message published to the topic will automatically be delivered to the specified Amazon SQS queue. If the owner of the Amazon SQS queue is not the owner of the topic, Amazon SNS will require an explicit confirmation to the subscription request.

Characteristics of Amazon SNS

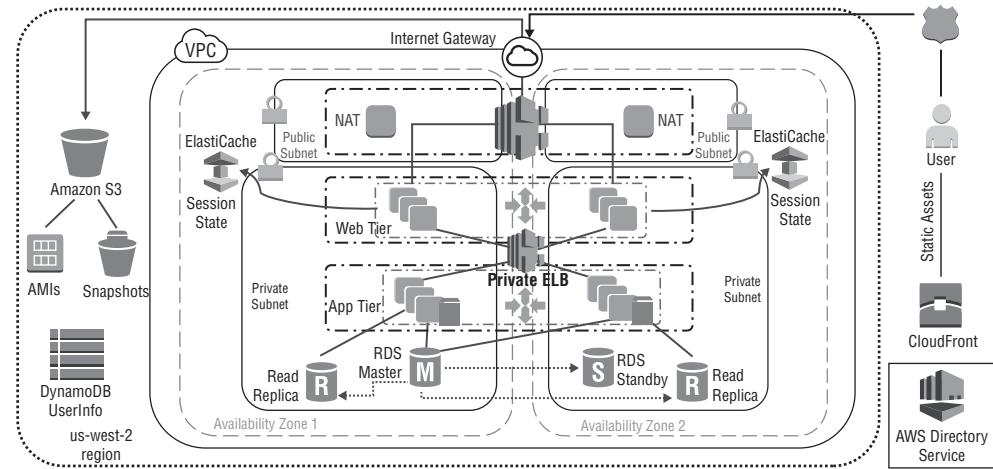
- Each notification message contains a single published message.
- Message order is not guaranteed.
- A message cannot be deleted after it has been published.
- Amazon SNS delivery policy can be used to control retries in case of message delivery failure.
- Messages can contain up to 256 KB of text data, including XML, JSON, and unformatted text.

Highly Available Architectures

In Chapter 5, “Networking,” you learned that the boundary of an Amazon Virtual Private Cloud (Amazon VPC) is at the region, and that regions comprise Availability Zones. In Chapter 4, “Compute,” you learned how Amazon EC2 instances are bound to the Availability Zone. What happens if a natural disaster takes out your Amazon EC2 instance running in an affected Availability Zone? What if your application goes viral and starts taking on more traffic than expected? How do you continue to serve your users?

So far in this chapter, you have learned that services like Amazon SNS and Amazon SQS are highly available, fault-tolerant systems, which can be used to decouple your application. Chapter 7, “Databases,” introduced Amazon DynamoDB and Amazon ElastiCache as NoSQL databases. Now we are going to put all these parts together to deliver a highly available architecture.

Figure 10.5 illustrates a highly available three-tier architecture (the concept of which was discussed in Chapter 2, “Working with AWS Cloud Services”).

FIGURE 10.5 Highly available three-tier architecture

Network Address Translation (NAT) Gateways

Network Address Translation (NAT) gateways were covered in Chapter 5. NAT servers allow traffic from private subnets to traverse the Internet or connect to other AWS Cloud services. Individual NAT servers can be a single point of failure. The NAT gateway is a managed device, and each NAT gateway is created in a specific Availability Zone and implemented with redundancy in that Availability Zone. To achieve optimum availability, use NAT gateways in each Availability Zone.



If you have resources in multiple Availability Zones, they share one NAT gateway. When the NAT gateway's Availability Zone is down, resources in the other Availability Zones will lose Internet access. To create an Availability Zone-independent architecture, create a NAT gateway in each Availability Zone and configure your routing to ensure that resources use the NAT gateway in the same Availability Zone.

Elastic Load Balancing

As discussed in Chapter 5, Elastic Load Balancing comes in two types: Application Load Balancer and Classic Load Balancer. *Elastic Load Balancing* allows you to decouple an application's web tier (or front end) from the application tier (or back end). In the event of a node failure, the Elastic Load Balancing load balancer will stop sending traffic to the affected Amazon EC2 instance, thus keeping the application available, although in a deprecated state. Self-healing can be achieved by using Auto Scaling. For a refresher on Elastic Load Balancing, refer to Chapter 5.

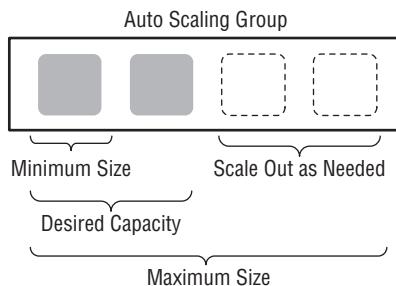
Auto Scaling

Auto Scaling is a web service designed to launch or terminate Amazon EC2 instances automatically based on user-defined policies, schedules, and health checks. Application Auto Scaling automatically scales supported AWS Cloud services with an experience similar to Auto Scaling for Amazon EC2 resources. Application Auto Scaling works with Amazon EC2 Container Service (Amazon ECS) and will not be covered in this guide.

Auto Scaling helps to ensure that you have the correct number of Amazon EC2 instances available to handle the load for your application. You create collections of Amazon EC2 instances, called *Auto Scaling groups*. You can specify the minimum number of instances in each Auto Scaling group, and Auto Scaling ensures that your group never goes below this size. Likewise, you can specify the maximum number of instances in each Auto Scaling group, and Auto Scaling ensures that your group never goes above this size. If you specify the desired capacity, either when you create the group or at any time thereafter, Auto Scaling ensures that your group has that many instances. If you specify scaling policies, then Auto Scaling can launch or terminate instances on demand as your application needs increase or decrease.

For example, the Auto Scaling group shown in Figure 10.6 has a minimum size of one instance, a desired capacity of two instances, and a maximum size of four instances. The scaling policies that you define adjust the number of instances, within your minimum and maximum number of instances, based on the criteria that you specify.

FIGURE 10.6 Auto Scaling group



Auto Scaling is set in motion by Amazon CloudWatch. Amazon CloudWatch is covered in Chapter 9, “Monitoring and Metrics.”

Auto Scaling Components

- Launch configuration
- Group
- Scaling policy (optional)
- Scheduled action (optional)

Launch configuration Launch configuration defines how Auto Scaling should launch your Amazon EC2 instances. Auto Scaling provides you with an option to create a new launch configuration using the attributes from an existing Amazon EC2 instance. When you use this option, Auto Scaling copies the attributes from the specified instance into a template from which you can launch one or more Auto Scaling groups.

Auto Scaling group Your Auto Scaling group uses a launch configuration to launch Amazon EC2 instances. You create the launch configuration by providing information about the image that you want Auto Scaling to use to launch Amazon EC2 instances. The information can be the image ID, instance type, key pairs, security groups, and block device mapping.

Auto Scaling policy An Auto Scaling group uses a combination of policies and alarms to determine when the specified conditions for launching and terminating instances are met. An alarm is an object that watches over a single metric (for example, the average CPU utilization of your Amazon EC2 instances in an Auto Scaling group) over a time period that you specify. When the value of the metric breaches the thresholds that you define over a number of time periods that you specify, the alarm performs one or more actions. An action can be sending messages to Auto Scaling. A *policy* is a set of instructions for Auto Scaling that tells the service how to respond to alarm messages. These alarms are defined in Amazon CloudWatch (refer to Chapter 9 for more details).

Scheduled action Scheduled action is scaling based on a schedule, allowing you to scale your application in response to predictable load changes. To configure your Auto Scaling group to scale based on a schedule, you need to create scheduled actions. A *scheduled action* tells Auto Scaling to perform a scaling action at a certain time in the future. To create a scheduled scaling action, you specify the start time at which you want the scaling action to take effect, along with the new minimum, maximum, and desired size you want for that group at that time. At the specified time, Auto Scaling will update the group to set the new values for minimum, maximum, and desired sizes, as specified by your scaling action.

Session State Management

In order to scale out or back in, your application needs to be stateless. Consider storing session-related information off of the instance. Amazon DynamoDB or Amazon ElastiCache can be used for session state management. For more information on Amazon ElastiCache and Amazon DynamoDB, refer to Chapter 7.

Amazon Elastic Compute Cloud Auto Recovery

Auto Recovery is an Amazon EC2 feature that is designed to increase instance availability. It allows you to recover supported instances automatically when a system impairment is detected.

To use Auto Recovery, create an Amazon CloudWatch Alarm that monitors an Amazon EC2 instance and automatically recovers the instance if it becomes impaired due to an underlying hardware failure or a problem that requires AWS involvement to repair. Terminated instances cannot be recovered. A recovered instance is identical to the original instance, including the instance ID, private IP addresses, elastic IP addresses, and all instance metadata.

When the StatusCheckFailed_System alarm is triggered and the recover action is initiated, you will be notified by the Amazon SNS topic that you selected when you created the alarm and associated the recover action. During instance recovery, the instance is migrated during an instance reboot, and any data that is in-memory is lost. When the process is complete, information is published to the Amazon SNS topic that you configured for the alarm. Anyone who is subscribed to this Amazon SNS topic will receive an email notification that includes the status of the recovery attempt and any further instructions. You will notice an instance reboot on the recovered instance.

Examples of problems that cause system status checks to fail include the following:

- Loss of network connectivity
- Loss of system power
- Software issues on the physical host
- Hardware issues on the physical host that impact network reachability

The recover action can also be triggered when an instance is scheduled by AWS to stop or retire due to degradation of the underlying hardware.



The recover action is supported only on instances with the following characteristics:

- C3, C4, M3, M4, R3, R4, T2, or X1 instance type
- Amazon VPC only (not Amazon EC2-Classic)
- Shared tenancy (the tenancy attribute is set to default)
- Amazon EBS volumes only (do not configure instance store volumes)

If your instance has a public IPv4 address, it retains the public IPv4 address after recovery.

Scaling Your Amazon Relational Database Service Deployment

In Chapter 7, we covered how to achieve high availability of your database by deploying Amazon Relational Database Service (Amazon RDS) across multiple Availability Zones. You can also improve the performance of a read-heavy database by using Read Replicas to

scale your database horizontally. Amazon RDS MySQL, PostgreSQL, and Maria DB can have up to 5 Read Replicas, and Amazon Aurora can have up to 15 Read Replicas.

You can also place your Read Replica in a different AWS Region closer to your users for better performance. Additionally, you can use Read Replicas to increase the availability of your database by promoting a Read Replica to a master for faster recovery in the event of a disaster. Read Replicas are not a replacement for the high availability and automatic failover capabilities that Multi-AZ architectures provide, however. For a refresher on Amazon RDS high availability and Read Replicas, refer to Chapter 7.

Multi-Region High Availability

In addition to building a highly available application that runs in a single region, your application may require regional fault tolerance. This can be delivered by placing and running infrastructure in another region and then using Amazon Route 53 to load balance the traffic between the regions.

Amazon Simple Storage Service

If you need to keep Amazon Simple Storage Service (Amazon S3) data in multiple regions, you can use cross-region replication. *Cross-region replication* is a bucket-level feature that enables automatic, asynchronous copying of objects across buckets in different AWS Regions. More information on Amazon S3 and cross-region replication is available in Chapter 6, “Storage Systems.”

Amazon DynamoDB

Amazon DynamoDB uses DynamoDB Streams to replicate data between regions. An application in one AWS Region modifies the data in an Amazon DynamoDB table. A second application in another AWS Region reads these data modifications and writes the data to another table, creating a replica that stays in sync with the original table.

Amazon Route 53

When you have more than one resource performing the same function (for example, more than one HTTP/S server or mail server), you can configure Amazon Route 53 to check the health of your resources and respond to Domain Name System (DNS) queries using only the healthy resources. For example, suppose your website, Example.com, is hosted on 10 servers, 2 each in 5 regions around the world. You can configure Amazon Route 53 to

check the health of those servers and to respond to DNS queries for Example.com using only the servers that are currently healthy.

You can set up a variety of failover configurations using Amazon Route 53 alias, weighted, latency, geolocation routing, and failover resource record sets.

Active-active failover Use this failover configuration when you want all of your resources to be available the majority of the time. When a resource becomes unavailable, Amazon Route 53 can detect that it is unhealthy and stop including it when responding to queries.

Active-passive failover Use this failover configuration when you want a primary group of resources to be available the majority of the time and a secondary group of resources to be on standby in case all of the primary resources become unavailable. When responding to queries, Amazon Route 53 includes only the healthy primary resources. If all of the primary resources are unhealthy, Amazon Route 53 begins to include only the healthy secondary resources in response to DNS queries.

Active-active-passive and other mixed configurations You can combine alias and non-alias resource record sets to produce a variety of Amazon Route 53 behaviors. More information on record types can be found in Chapter 5.

In order for these failover configurations to work, health checks will need to be configured. There are three types of health checks: health checks that monitor an endpoint, health checks that monitor Amazon CloudWatch Alarms, and health checks that monitor other health checks.

The following sections discuss simple and complex failover configurations.

Health Checks for Simple Failover

The simplest failover configuration of having two or more resources performing the same function can benefit from health checks. For example, you might have multiple Amazon EC2 servers running HTTP server software responding to requests for your Example.com website. In Amazon Route 53, you create a group of resource record sets that have the same name and type, such as weighted resource record sets or latency resource record sets of type A. You create one resource record set for each resource, and you configure Amazon Route 53 to check the health of the corresponding resource. In this configuration, Amazon Route 53 chooses which resource record set will respond to a DNS query for Example.com and bases the choice in part on the health of your resources.

As long as all of the resources are healthy, Amazon Route 53 responds to queries using all of your Example.com weighted resource record sets. When a resource becomes unhealthy, Amazon Route 53 responds to queries using only the healthy resource record sets for Example.com.

Following are the steps for how you configure Amazon Route 53 to check the health of your resources in this simple configuration and how Amazon Route 53 responds to queries based on the health of your resources.

Configuring Amazon Route 53 to Check the Health of Your Resources

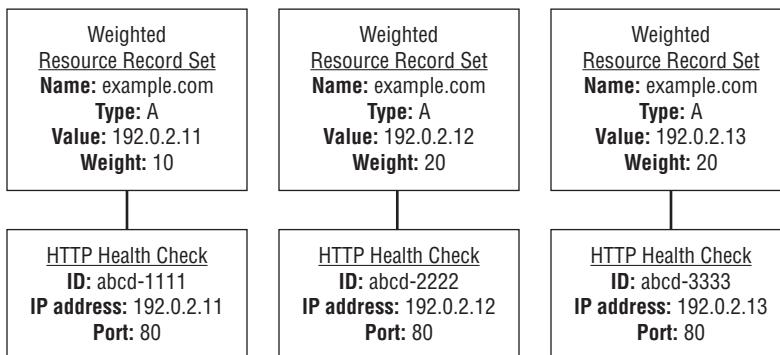
1. You identify the resources whose health you want Amazon Route 53 to monitor. For example, you might want to monitor all of the HTTP servers that respond to requests for Example.com.

2. You create health checks for your resources. A health check tells Amazon Route 53 how to send requests to the endpoint whose health you want to check: which protocol (HTTP, HTTPS, or Transmission Control Protocol [TCP]) and which IP address and port to use and also a domain name and path for HTTP/HTTPS health checks.

A common configuration is to create one health check for each resource and to use the same IP address for the health check endpoint for the resource. If the IP address for your HTTP server is 192.0.2.117, you create a health check for which the IP address is 192.0.2.117.

3. You might need to configure router and firewall rules so that Amazon Route 53 can send regular requests to the endpoints that you specified in your health checks.

4. You create a group of resource record sets for your resources (for example, a group of weighted resource record sets that all have a type of A). You associate the health checks that you created in Step 2 with the corresponding resource record sets. The graphic illustrates how these health checks will operate.



5. Amazon Route 53 periodically sends a request to each endpoint that you specified when you created your health checks; it doesn't perform the health check when it receives a DNS query. Based on the responses, Amazon Route 53 decides whether the endpoints are healthy and uses that information to determine how to respond to queries.

6. When Amazon Route 53 receives a query for Example.com:

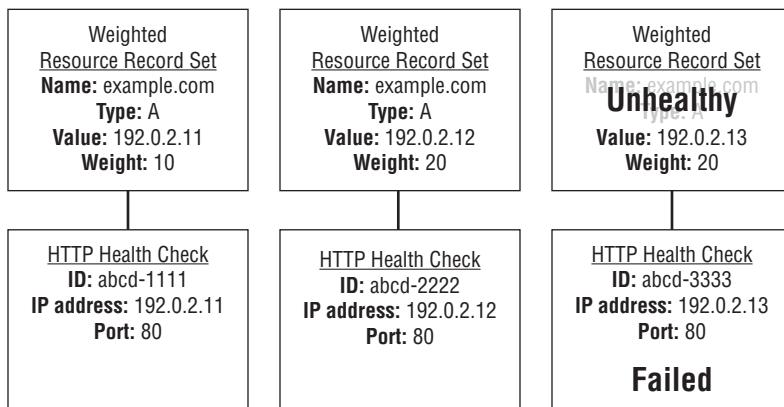
- Amazon Route 53 chooses a resource record set based on the routing policy. In this case, it chooses a resource record set based on weight.

- b. It determines the current health of the selected resource record set by checking the status of the health check for that resource record set.
- c. If the selected resource record set is unhealthy, it repeats the process of choosing a resource record set based on the routing policy. This time, the unhealthy resource record set isn't considered.
- d. It responds to the query with the selected healthy resource record set.

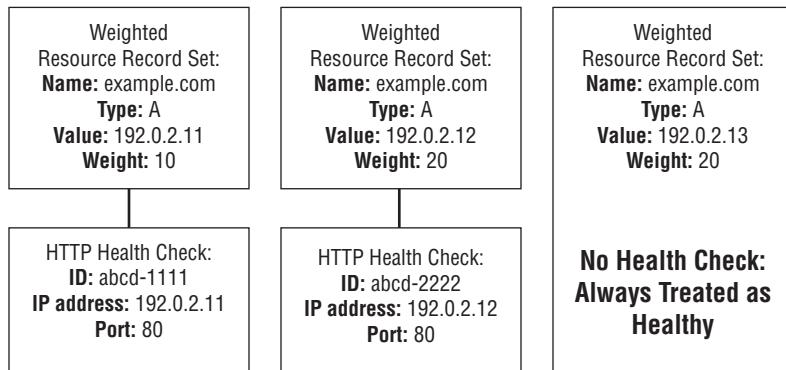
The following example shows a group of weighted resource record sets in which the third resource record set is unhealthy. Initially, Amazon Route 53 selects a resource record set based on the weights of all three resource record sets. If it happens to select the unhealthy resource record set the first time, Amazon Route 53 selects another resource record set, but this time it omits the weight of the third resource record set from the calculation. See Figure 10.7 for an example of an unhealthy health check.

- When Amazon Route 53 initially selects from among all three resource record sets, it responds to requests using the first resource record set about 20 percent of the time: $10/(10 + 20 + 20)$.
- When Amazon Route 53 determines that the third resource record set is unhealthy, it responds to requests using the first resource record set about 33 percent of the time: $10/(10 + 20)$.

FIGURE 10.7 Unhealthy health check



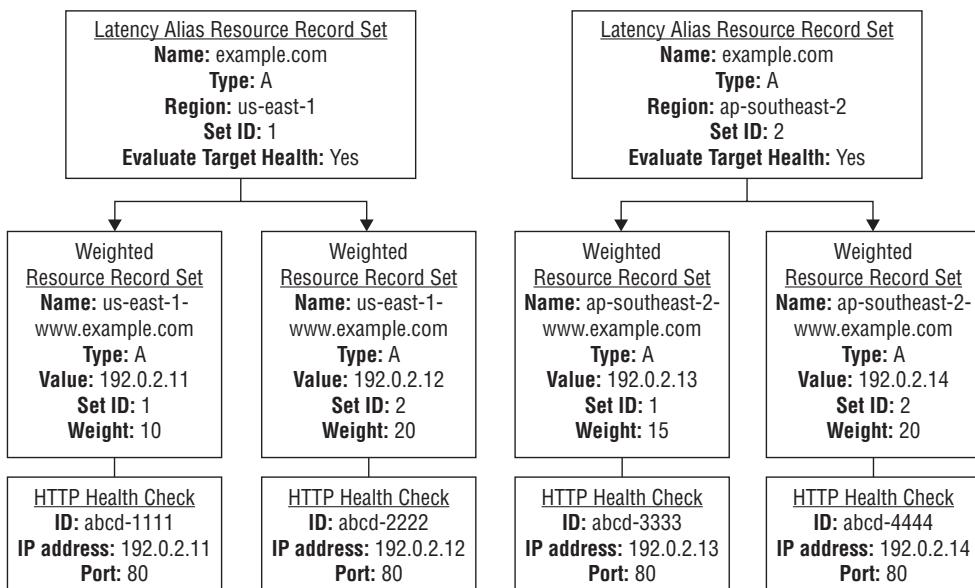
If you omit a health check from one or more resource record sets in a group of resource record sets, Amazon Route 53 treats those resource record sets as healthy. Amazon Route 53 has no basis for determining the health of the corresponding resource without an assigned health check and might choose a resource record set for which the resource is unhealthy. See Figure 10.8 for more information.

FIGURE 10.8 No health check enabled

Health Checks for Complex Failover

Checking the health of resources in complex configurations works much the same way as in simple configurations. In complex configurations, however, you use a combination of alias resource record sets (including weighted alias, latency alias, and failover alias) and non-alias resource record sets to build a decision tree that gives you greater control over how Amazon Route 53 responds to requests.

For example, you might use latency alias resource record sets to select a region close to a user and use weighted resource record sets for two or more resources within each region to protect against the failure of a single endpoint or an Availability Zone. Figure 10.9 shows this configuration.

FIGURE 10.9 Health check for a complex failover

An overview of how Amazon EC2 and Amazon Route 53 are configured follows:

- You have Amazon EC2 instances in two regions: us-east-1 and ap-southeast-2. You want Amazon Route 53 to respond to queries by using the resource record sets in the region that provides the lowest latency for your customers, so you create a latency alias resource record set for each region. (You create the latency alias resource record sets after you create resource record sets for the individual Amazon EC2 instances.)
- Within each region, you have two Amazon EC2 instances. You create a weighted resource record set for each instance. The name and the type are the same for both of the weighted resource record sets in each region.

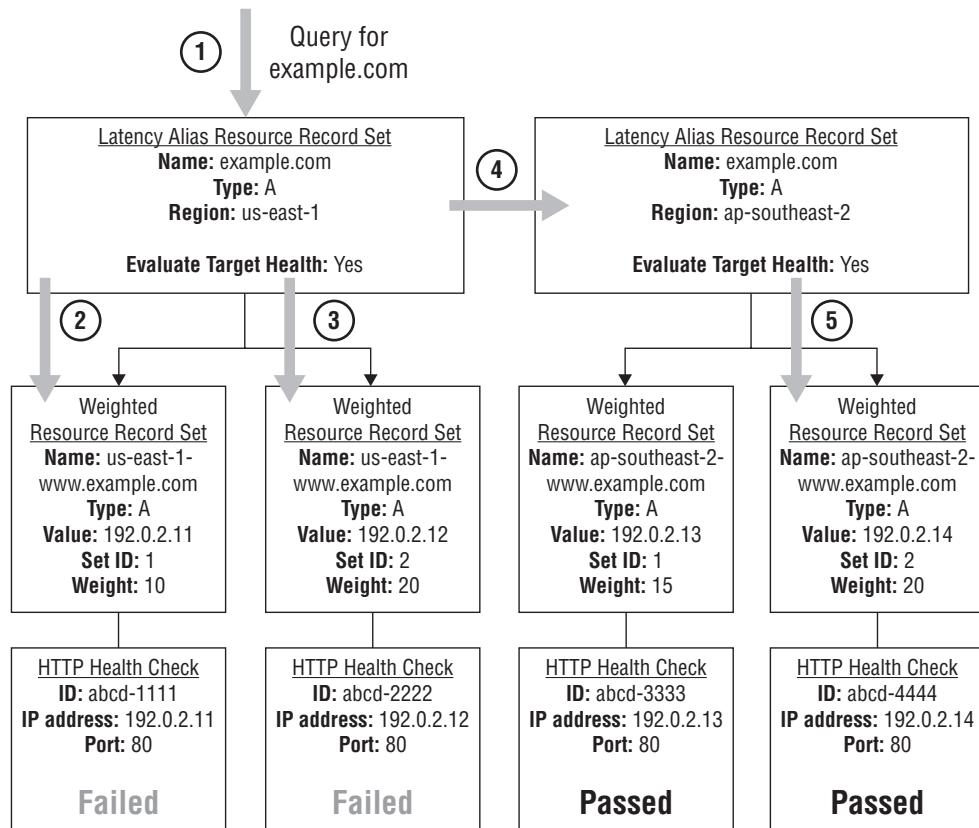
When you have multiple resources in a region, you can create weighted or failover resource record sets for your resources. You can also make even more complex configurations by creating weighted alias or failover alias resource record sets that, in turn, refer to multiple resources.

- Each weighted resource record set has an associated health check. The IP address for each health check matches the IP address for the corresponding resource record set. This isn't required, but it is the most common configuration.
- For both latency alias resource record sets, you set the value of Evaluate Target Health to Yes.

You use the Evaluate Target Health setting for each latency alias resource record set to make Amazon Route 53 evaluate the health of the alias targets—the weighted resource record sets—and respond accordingly.

Figure 10.10 demonstrates the sequence of events that follows:

1. Amazon Route 53 receives a query for Example.com. Based on the latency for the user making the request, Amazon Route 53 selects the latency alias resource record set for the us-east-1 region.
2. Amazon Route 53 selects a weighted resource record set based on weight. Evaluate Target Health is Yes for the latency alias resource record set, so Amazon Route 53 checks the health of the selected weighted resource record set.
3. The health check failed, so Amazon Route 53 chooses another weighted resource record set based on weight and checks its health. That resource record set also is unhealthy.
4. Amazon Route 53 backs out of that branch of the tree, looks for the latency alias resource record set with the next-best latency, and chooses the resource record set for ap-southeast-2.
5. Amazon Route 53 again selects a resource record set based on weight and then checks the health of the selected resource record set. The health check passed, so Amazon Route 53 returns the applicable value in response to the query.

FIGURE 10.10 Evaluate target health

Highly Available Connectivity Options

In this section of this chapter, we discuss how to make your connections to AWS redundant. In Chapter 5, we discussed the various connectivity options to connect to AWS and, more specifically, your Amazon VPC. We will discuss the Virtual Private Network (VPN) and AWS Direct Connect connectivity options and how to make them highly available.

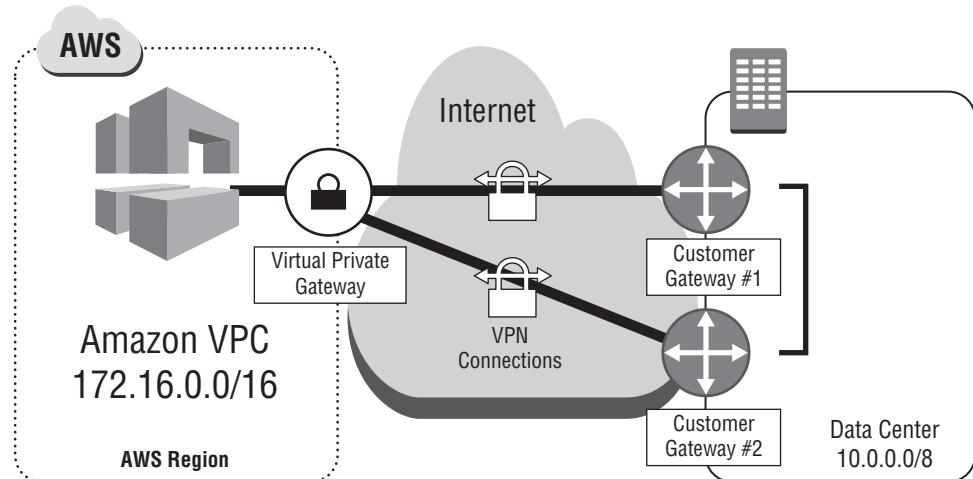
Redundant Active-Active VPN Connections

To get your connectivity up and running quickly, you can implement VPN connections because they are a quick, easy, and cost-effective way to set up remote connectivity to your Amazon VPC. To enable redundancy, each AWS Virtual Private Gateway (VGW) has two VPN endpoints with capabilities for static and dynamic routing. Although statically routed

VPN connections from your router (also known as the *customer gateway*) are sufficient for establishing remote connectivity to an Amazon VPC, this is not a highly available configuration.

The best practice for making VPN connections highly available is to use redundant routers (customer gateways) and dynamic routing for automatic failover between AWS and customer VPN endpoints. Figure 10.11 demonstrates each VPN connection, consisting of two IPsec tunnels to both VGW endpoints, as a single line. See Figure 10.11 for a sample active-active VPN connection.

FIGURE 10.11 Redundant active-active VPN connections



Configuration Details

The configuration in Figure 10.11 consists of four fully-meshed, dynamically-routed IPsec tunnels between both VGW endpoints and two customer gateways. AWS provides configuration templates for a number of supported VPN devices to assist in establishing these IPsec tunnels and configuring Border Gateway Protocol (BGP) for dynamic routing.

In addition to the AWS-provided VPN and BGP configuration details, customers must configure Amazon VPCs to route traffic efficiently to their datacenter networks. In the example in Figure 10.11, the VGW will prefer to send 10.0.0.0/16 traffic to Data Center 1 through Customer Gateway 1 and only reroute this traffic through Data Center 2 if the connection to Data Center 1 is down. Likewise, 10.1.0.0/16 traffic will prefer the VPN connection originating from Data Center 2.

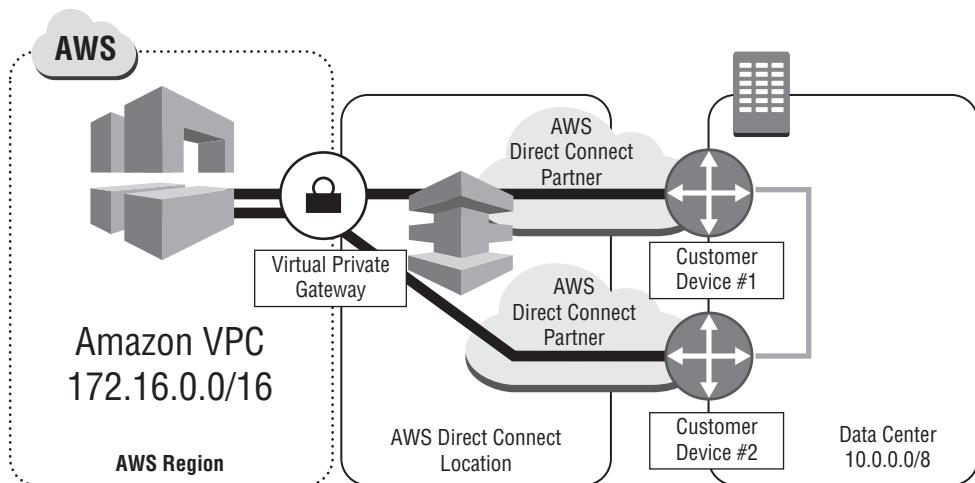
This configuration relies on the Internet to carry traffic between on-premises networks and Amazon VPC. Although AWS leverages multiple Internet Service Providers (ISPs), and even if the customer leverages multiple ISPs, an Internet service disruption can still affect the availability of VPN network connectivity due to the interdependence of ISPs and Internet routing. The only way to control the exact network path of your traffic is to provision private network connectivity with AWS Direct Connect.

Redundant Active-Active AWS Direct Connect Connections

If you need to establish private connectivity between AWS and your datacenter, office, or colocation environment, use AWS Direct Connect. More information on AWS Direct Connect can be found in Chapter 5.

AWS Direct Connect uses a dedicated, physical connection in one AWS Direct Connect location. Therefore, to make AWS Direct Connect highly available, consider implementing multiple dynamically-routed AWS Direct Connect connections. Architectures with the highest levels of availability will leverage different AWS Direct Connect partner networks to ensure network-provider redundancy. Refer to Figure 10.12 for an illustration of an AWS Direct Connect connection, consisting of a physical connection that contains logical connections, as one continual line.

FIGURE 10.12 AWS Direct Connect with redundant connections



Configuration Details

The configuration depicted in Figure 10.12 consists of AWS Direct Connect connections to separate AWS Direct Connect routers in two locations from two independently configured customer devices. AWS provides example router configurations to assist in establishing AWS Direct Connect connections and configuring BGP for dynamic routing.

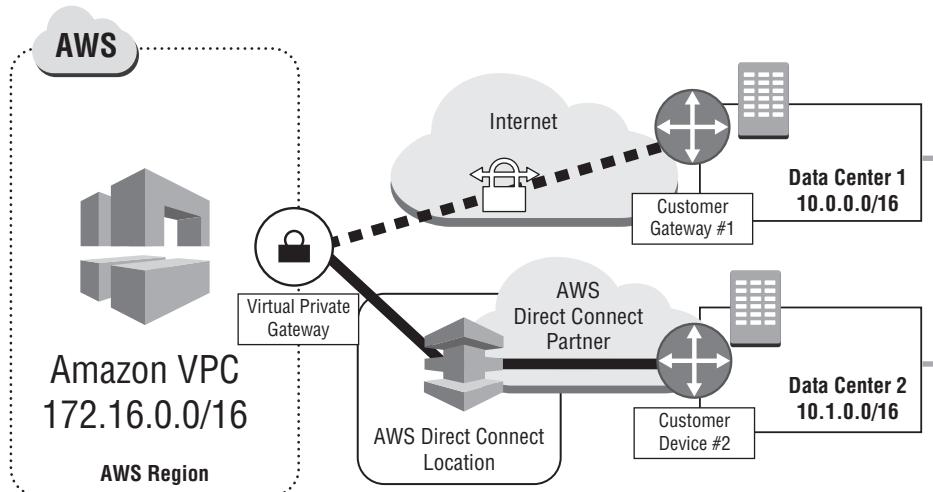
In addition to the AWS-provided configuration details, you must configure your Amazon VPCs to route traffic efficiently to the datacenter networks. In the example in Figure 10.12, the VGW will prefer to send 10.0.0.0/16 traffic to Data Center 1 and only reroute this traffic to Data Center 2 if connectivity to Customer Device 1 is down. Likewise, 10.1.0.0/16 traffic will prefer the AWS Direct Connect connection from Data Center 2.

AWS Direct Connect allows you to create resilient connections to AWS because you have full control over the network path and network providers between your remote networks and AWS.

AWS Direct Connect with Backup VPN Connection

Some AWS customers would like the benefits of one or more AWS Direct Connect connections for their primary connectivity to AWS, combined with a lower-cost backup connection. To achieve this objective, they can establish AWS Direct Connect connections with a VPN backup, as depicted in Figure 10.13.

FIGURE 10.13 AWS Direct Connect with backup VPN connection



Configuration Details

The configuration depicted in Figure 10.13 consists of two dynamically-routed connections—one using AWS Direct Connect and the other using a VPN connection from two different customer gateways. AWS provides example router configurations to assist in establishing both AWS Direct Connect and VPN connections with BGP for dynamic routing. By default, AWS will always prefer to send traffic over an AWS Direct Connect connection, so no additional configuration is required to define primary and backup connections. In the example in Figure 10.13, both Customer Gateway 1 (VPN) and Customer Device 2 (AWS Direct Connect) advertise a summary route of 10.0.0.0/16. AWS will send all traffic to Customer Device 2 as long as this network path is available.

Although BGP prefers AWS Direct Connect to VPN connections by default, you still have the ability to influence AWS routing decisions by advertising more specific (or static) routes. If, for example, you want to leverage your backup VPN connection for a subset of traffic (for example, developer traffic versus production traffic), you can advertise specific routes from Customer Gateway 1. Refer to Chapter 5 for more information on route preference.

Disaster Recovery

Disaster recovery (DR) is about preparing for and recovering from a disaster. Any event that has a negative impact on your company's business continuity or finances could be termed a disaster. This includes hardware or software failure, a network outage, a power outage, physical damage to a building like fire or flooding, human error, or some other significant event. You should plan on how to minimize the impact of a disaster by investing time and resources to plan and prepare, train employees, and document and update processes.

The amount of investment for DR planning for a particular system can vary dramatically depending on the cost of a potential outage. Companies that have traditional physical environments typically must duplicate their infrastructure to ensure the availability of spare capacity in the event of a disaster. The infrastructure needs to be procured, installed, and maintained so that it is ready to support the anticipated capacity requirements. During normal operations, the infrastructure typically is under-utilized or over-provisioned. With Amazon Web Services (AWS), your company can scale up its infrastructure on an as-needed, pay-as-you-go basis. You get access to the same highly secure, reliable, and fast infrastructure that Amazon uses to run its own global network of websites. AWS also gives you the flexibility to change and optimize resources quickly during a DR event, which can result in significant cost savings.

We will be using two common industry terms for disaster planning:

Recovery time objective (RTO) This represents the time it takes after a disruption to restore a business process to its service level, as defined by the operational level agreement (OLA). For example, if a disaster occurs at 12:00 PM (noon) and the RTO is eight hours, the DR process should restore the business process to the acceptable service level by 8:00 PM.

Recovery point objective (RPO) This is the acceptable amount of data loss measured in time. For example, if a disaster occurs at 12:00 PM (noon) and the RPO is one hour, the system should recover all data that was in the system before 11:00 AM. Data loss will span only one hour, between 11:00 AM and 12:00 PM (noon).

A company typically decides on an acceptable RTO and RPO based on the financial impact to the business when systems are unavailable. The company determines financial impact by considering many factors, such as the loss of business and damage to its reputation due to downtime and the lack of systems availability.

IT organizations then plan solutions to provide cost-effective system recovery based on the RPO within the timeline and the service level established by the RTO.

In this final section of the chapter, we will discuss the various types of disaster recovery scenarios on Amazon Web Services.

Backup and Restore Method

In most traditional environments, data is backed up to tape and sent off-site regularly. If you use this method, it can take a long time to restore your system in the event of a

disruption or disaster. Amazon S3 is an ideal destination for backup data that might be needed quickly to perform a restore. Transferring data to and from Amazon S3 is typically done through the network, and it is therefore accessible from any location. There are many commercial and open-source backup solutions that integrate with Amazon S3. You can use AWS Import/Export Snowball to transfer very large datasets by shipping storage devices directly to AWS. For longer-term data storage where retrieval times of several hours are adequate, there is Amazon Glacier, which has the same durability model as Amazon S3. Amazon Glacier is a low-cost alternative to Amazon S3. Amazon Glacier and Amazon S3 can be used in conjunction to produce a tiered backup solution.

AWS Storage Gateway enables snapshots of your on-premises data volumes to be transparently copied into Amazon S3 for backup. You can create local volumes or Amazon EBS volumes from these snapshots. *Storage-cached volumes* allow you to store your primary data in Amazon S3, but keep your frequently accessed data local for low-latency access. As with AWS Storage Gateway, you can snapshot the data volumes to give highly durable backup. In the event of DR, you can restore the cache volumes either to a second site running a storage cache gateway or to Amazon EC2.

You can use the gateway-VTL (Virtual Tape Library) configuration of AWS Storage Gateway as a backup target for your existing backup management software. This can be used as a replacement for traditional magnetic tape backup. For systems running on AWS, you also can back up into Amazon S3. Snapshots of Amazon EBS volumes, Amazon RDS databases, and Amazon Redshift data warehouses can be stored in Amazon S3. Alternatively, you can copy files directly into Amazon S3, or you can choose to create backup files and copy those to Amazon S3. There are many backup solutions that store data directly in Amazon S3, and these can be used from Amazon EC2 systems as well.

Key Steps for the Backup and Restore Method of Disaster Recovery

1. Select an appropriate tool or method to back up your data into AWS.
2. Ensure that you have an appropriate data retention policy in place.
3. Ensure that appropriate security measures are in place for this data, including encryption of the data and the appropriate data access policies.
4. Regularly test the recovery of your data and, more importantly, the restoration of your systems.

Pilot Light Method

The term *pilot light* is often used to describe a DR scenario in which a minimal version of an environment is always running in AWS. The idea of the pilot light is an analogy that comes from a natural gas heater. In a natural gas heater, a small flame or pilot light is always on so it can quickly ignite the entire furnace to heat up the house.

This scenario is similar to a backup-and-restore scenario. For example, with AWS you can maintain a pilot light by configuring and running the most critical core elements of your system in AWS. When the time comes for recovery, you can rapidly provision a full-scale production environment around the critical core. Infrastructure elements for the pilot

light itself typically include your database servers, which would replicate data to Amazon EC2 or Amazon RDS.

Depending on the system, there might be other critical data outside of the database that needs to be replicated to AWS. This is the critical core of the system (the pilot light) around which all other infrastructure pieces in AWS (the rest of the furnace) can quickly be provisioned to restore the complete system.

Preparation Phase

Unlike the Backup-Restore phase, the Pilot-Light method requires some upfront configuration and deployment prior to the occurrence of a disaster.

Key Preparation Steps for the Pilot Light Method of Disaster Recovery

1. Set up Amazon EC2 instances to replicate or mirror data.
2. Ensure that you have all supporting custom software packages available in AWS.
3. Create and maintain Amazon Machine Images (AMIs) of key servers where fast recovery is required.
4. Regularly run these servers, test them, and apply any software updates and configuration changes.
5. Automate the provisioning of AWS resources.

Recovery Phase

To recover the rest of your environment around the pilot light, you can start your systems from the Amazon Machine Images (AMIs) within minutes on the appropriate instance types. For your dynamic data servers, you can resize them to handle production volumes as needed or add capacity accordingly. Horizontal scaling often is the most cost-effective and scalable approach to add capacity to a system. For example, you can add more web servers at peak times. However, you can also choose larger Amazon EC2 instance types, and thus scale vertically for applications such as relational databases. From a networking perspective, any required DNS updates can be done in parallel.

After recovery, you should ensure that redundancy is restored as quickly as possible. A failure of your DR environment shortly after your production environment fails is unlikely; however it is possible, so be prepared. Continue to take regular backups of your system and consider additional redundancy at the data layer. Remember that your DR environment is now your production environment.

Key Recovery Steps for the Pilot-Light Method of Disaster Recovery

1. Start your application Amazon EC2 instances from your custom AMIs.
2. Resize existing database/data store instances to process the increased traffic.
3. Add additional database/data store instances to give the DR site resilience in the data tier; if you are using Amazon RDS, turn on Multi-AZ to improve resilience.
4. Change DNS to point to the Amazon EC2 servers.
5. Install and configure any non-AMI based systems, ideally in an automated way.

Warm-Standby Method

The term *warm-standby* is used to describe a DR scenario in which a scaled-down version of a fully-functional environment is always running in the cloud. A warm-standby solution extends the pilot light elements and preparation. It further decreases the recovery time because some services are always running. By identifying your business-critical systems, you can fully duplicate these systems on AWS and have them always on.

These servers can be running on a minimum-sized fleet of Amazon EC2 instances on the smallest sizes possible. This solution is not scaled to take a full-production load; however, it is fully functional. In a disaster, the system is scaled up quickly to handle the production load. In AWS, this can be done by adding more instances to the load balancer and by resizing the small capacity servers to run on larger Amazon EC2 instance types.

Preparation Phase

Just like the Pilot Light Method of disaster recovery, the warm standby method needs some upfront deployment and preparation prior to the occurrence of a disaster.

Key Preparation Steps for the Warm-Standby Solution for Disaster Recovery

1. Set up Amazon EC2 instances to replicate or mirror data.
2. Create and maintain AMIs.
3. Run your application using a minimal footprint of Amazon EC2 instances or AWS infrastructure.
4. Patch and update software and configuration files in line with your live environment.

Recovery Phase

In the case of failure of the production system, the standby environment will be scaled up for production load, and DNS records will be changed to route all traffic to AWS.

Key Recovery Steps for the Warm-Standby Method of Disaster Recovery

1. Increase the size of the Amazon EC2 fleets in service with the load balancer.
2. Start applications on larger Amazon EC2 instance types as needed.
3. Either manually change the DNS records or use Amazon Route 53 automated health checks so that all traffic is routed to the AWS environment.
4. Use Auto Scaling to right-size the fleet or accommodate the increased load.
5. Scale up your database (if needed).

Multi-Site Solution Method

A *multi-site solution* runs in AWS, as well as on your existing on-site infrastructure, in an active-active configuration. The data replication method that you employ will be determined by the recovery point that you choose. Refer to the RPO definition at the beginning of this section.

You should use a DNS service that supports weighted routing, such as Amazon Route 53, to route production traffic to different sites that deliver the same application or service. A proportion of traffic will go to your infrastructure in AWS, and the remainder will go to your on-site infrastructure.

In an on-site disaster situation, you can adjust the DNS weighting and send all traffic to the AWS servers. The capacity of the AWS service can be rapidly increased to handle the full production load. You can use Amazon EC2 Auto Scaling to automate this process.

You might need some application logic to detect the failure of the primary database services and cut over to the parallel database services running in AWS. The cost of this scenario is determined by how much production traffic is handled by AWS during normal operation.

In the recovery phase, you pay only for what you use for the duration that the DR environment is required at full scale. You can further reduce cost by purchasing Amazon EC2 Reserved Instances for your “always on” AWS servers. Refer to Amazon EC2 purchasing options in Chapter 4.

Preparation Phase

In a multi-site disaster recovery solution, another site must be a duplicate of your current production site, and it will take half of the production traffic.

Key Preparation Steps for the Multi-Site Solution Method for Disaster Recovery

1. Set up your AWS environment to duplicate your production environment.
2. Set up DNS weighting, or similar traffic routing technology, to distribute incoming requests to both sites.
3. Configure automated failover to re-route traffic away from the affected site.

Recovery Phase

When failing over to the available site, DNS records will need to be changed to direct all production traffic to the available production site.

Key Recovery Steps for the Multi-Site Solution Method of Disaster Recovery

1. Either manually or by using DNS failover, change the DNS weighting so that all requests are sent to the AWS site.
2. Have application logic for failover to use the local AWS database servers for all queries.
3. Consider using Auto Scaling to right-size the AWS fleet automatically. You can further increase the availability of your multi-site solution by designing Multi-AZ architectures.

Failing Back from a Disaster

Once you have restored your primary site to a working state, you will need to restore your normal service, which is often referred to as a “fail back.” Depending on your DR strategy, this typically means reversing the flow of data replication so that any data updates received

while the primary site was down can be replicated back, without the loss of data. Let's discuss failback from the various methods of failover.

Failback the Backup and Restore Method

To failback from the disaster recovery site to the now working production site, you will need to follow these key steps:

Key Steps for the Failback the Backup and Restore Method of Disaster Recovery

1. Freeze data changes to the DR site.
2. Take a backup.
3. Restore the backup to the primary site.
4. Re-point users to the primary site.
5. Unfreeze the changes.

Failback Pilot Light, Warm-Standby, and Multi-Site Methods

Failback from the disaster recovery site when Pilot Light, Warm-Standby, and Multi-Site are used follows the same methodology to restore service to the former production site.

Failback Pilot Light, Warm-Standby, and Multi-Site Methods of Disaster Recovery

1. Establish reverse mirroring/replication from the DR site back to the primary site, once the primary site has caught up with the changes.
2. Freeze data changes to the DR site.
3. Re-point users to the primary site.
4. Unfreeze the changes.

Summary

In this chapter, we discussed how to decouple applications for scalability and high availability. The techniques discussed were using Amazon SQS and Amazon SNS to decouple an application's front end from its back end, thus reducing their dependencies on one another (so they can scale independently, for example). Regional applications such as Amazon S3 and Amazon DynamoDB can replicate their data into other regions to provide a regional highly available solution. When building systems, it is a best practice to use the AWS managed services as they are inherently fault tolerant. For workloads that cannot take advantage of managed services, it is up to you to make them highly available. AWS provides the parts and pieces—you put them together to deliver a highly available or fault tolerant system. Lastly, prepare for a disaster and develop a recovery plan using the various DR failover scenarios and the failback methods for each failover method.

Resources to Review

Amazon Simple Queue Service: <https://aws.amazon.com/sqs/>

Amazon Simple Notification Service: <https://aws.amazon.com/sns/>

Amazon DynamoDB Cross-Region Replication: <http://docs.aws.amazon.com/amazondynamodb/latest/developerguide/Streams.CrossRegionRepl.html>

Disaster Recovery on AWS: http://d36cz9buwru1tt.cloudfront.net/AWS_Disaster_Recovery.pdf

Fault Tolerance and High Availability on AWS: https://media.amazonwebservices.com/architecturecenter/AWS_ac_ra_ftha_04.pdf

Exam Essentials

Understand Amazon SQS polling. Amazon SQS uses *short polling* by default, querying only a subset of the servers (based on a weighted random distribution) to determine whether any messages are available for inclusion in the response. Long polling eliminates false empty responses by querying all (rather than a limited number) of the servers.

Understand visibility timeout and where it is applied. Immediately after the message is received, it remains in the queue. To prevent other consumers from processing the message again, Amazon SQS sets a *visibility timeout*, which is a period of time during which Amazon SQS prevents other consuming components from receiving and processing the message.

Understand message order with Amazon SQS standard queues. Standard queues provide a loose-FIFO capability that attempts to preserve the order of messages.

Understand First-In, First-Out (FIFO) queues. FIFO queues preserve the exact order in which messages are sent and received.

Understand the various protocols that Amazon SNS supports for its subscribers. HTTP/HTTPS, Email/Email-JSON, Amazon SQS, SMS, and AWS Lambda

Know the difference between Amazon SQS and Amazon SNS. Amazon SQS uses a pull method to receive messages; Amazon SNS uses a push method.

Understand how Amazon DynamoDB replicates data between regions. DynamoDB streams can be used to replicate data from one Amazon DynamoDB table in one region to another Amazon DynamoDB table in another region.

Understand how to scale an Amazon RDS deployment for high availability. Amazon RDS supports a multi-AZ deployment. Know the Amazon RDS databases that support Read Replicas in other regions.

Have a working knowledge of how Amazon Route 53 can be used to failover to another region. Understand what failover routing is and how it can be used to failover to other regions and to static websites hosted on Amazon S3.

Understand how health checks work with Amazon Route 53. Know the types of health checks and how to set alarms that trigger actions for a given metric.

Know how to set up a highly available VPN. The Virtual Private Gateway (VPG) supports two tunnels. Most routers can function in an active/active pattern with both tunnels. Older routers can handle active/passive configurations.

Understand how to make AWS Direct Connect highly available. To make AWS Direct Connect highly available, deploy two AWS Direct Connects.

Understand how to use a VPN as a backup to AWS Direct Connect. If you are unable to deploy a second AWS Direct Connect or have a requirement for additional availability, then keep your existing VPN. If you do not have one, then you can deploy a VPN for backup connectivity to your Amazon Virtual Private Cloud (Amazon VPC).

Understand the various types of disaster recovery methods available on AWS. The various types of failover are Backup/Restore, Pilot Light, Warm-Standby, and Multi-Site. Understand how to failover and how to fail back.

Test Taking Tip

Be prepared for the testing center! Bring two forms of a government-issued ID (including one photo ID). Bring your confirmation letter. Leave all computers, phones, wearable devices, and so forth at home. You don't want to miss your exam because of physical issues.

And good luck on your exam!

Exercises

By now you should have set up an account in AWS. If you haven't, now would be the time to do so. It is important to note that these exercises are in your AWS account and thus are not free.

Use the Free Tier when launching resources. The Amazon AWS Free Tier applies to participating services across the following AWS Regions: US East (Northern Virginia), US West (Oregon), US West (Northern California), Canada (Central), EU (London), EU (Ireland), EU (Frankfurt), Asia Pacific (Singapore), Asia Pacific (Tokyo), Asia Pacific (Sydney), and South America (Sao Paulo). For more information, see https://aws.amazon.com/sdm/optimization/server-side-test/free-tier/free_np/.

If you have not yet installed the AWS Command Line utilities, refer to Chapter 2, Exercise 2.1 (Linux) or Exercise 2.2 (Windows).

The reference for the AWS CLI can be found at <http://docs.aws.amazon.com/cli/latest/reference/>.

These exercises will walk you through some high-availability scenarios, and will have you research the "how to" in order to complete the exercise. By looking up how to perform

a specific task, you will be on your way to mastering the task. The goal of this guide isn't just to prepare to pass the AWS Certified SysOps Administrator – Associate exam. It is designed to serve you as a reference companion in your day-to-day duties as an AWS Certified SysOps Administrator.

EXERCISE 10.1

Create an Amazon SNS Topic.

In this exercise, you will create an Amazon SNS message.

1. Sign in to the AWS Management Console.
2. Navigate to Messaging, and then to Amazon SNS, to load the Amazon SNS dashboard.
3. Create a new topic. Use **MyTopic** for both the topic name and the display name.
4. Notice that the ARN is automatically assigned.

You have just created your first Amazon SNS topic.

EXERCISE 10.2

Create a Subscription to Your Topic.

In this exercise, you will create a subscription to your topic using your email address, and then you will confirm your email address.

1. In the Amazon SNS dashboard of the AWS Management Console, navigate to Topics.
2. Select the topic that you just created. Change the subscription protocol to Email, and enter your email address.
3. Create the subscription.
4. You will get an email from Amazon Simple Notification Service (Amazon SNS). You must open that email and confirm your email address before going any further.

You have just created a subscription to your topic.

EXERCISE 10.3

Publish to Your Topic.

In this exercise, you will publish a message to your topic.

1. In the Amazon SNS dashboard of the AWS Management Console, navigate to Topics.
2. Navigate to your topic (and select Publish to Topic).

EXERCISE 10.3 (continued)

3. Update the subject with **My Test Message**. Leave the message format to set a Raw and a Time to Live (TTL) field to 300. In the message, type **This is a test, only a test**.
4. Publish the message.

Check your email, and you should have received your first Amazon Simple Notification (Amazon SNS) message. Do not delete this topic until you have finished Exercises 10.4 and 10.5.

EXERCISE 10.4**Create an Amazon Simple Queue Service (Amazon SQS).**

In this exercise, you will create an Amazon Simple Queue Service (Amazon SQS) queue.

1. Sign in to the AWS Management Console.
2. Navigate to Messaging, and then to Amazon Simple Queue Service, to load the Amazon SQS Dashboard.
3. Create a new queue with input 30 seconds for the default visibility timeout and 1 day for the message retention period. Leave the minimum message size at 256 KB, and leave the rest of the settings at their defaults. Then create the queue.

You should now have a queue called **input**.

EXERCISE 10.5**Subscribe the Queue to Your Amazon SNS Topic.**

Now you will subscribe **MyTopic** from Exercise 10.1.

1. In the AWS Management Console, navigate to Messaging and then to Amazon Simple Queue Service.
 2. Select your input queue, and subscribe to your **MyTopic** Amazon SNS Topic.
 3. Return to the Amazon SNS dashboard, and choose Topics.
 4. Publish to **Topic**, and choose the defaults.
 5. Navigate to the Amazon SQS dashboard.
 6. You will notice you have one message in the input queue (and an email should have popped up on your screen if you have email notifications turned on).
 7. Select the input queue, and select Queue Actions to poll the queue.
 8. Select more details to see the message. Scroll through the message body, and notice the different fields.
 9. Delete your message, delete your queue, and then delete your Amazon SNS topic.
-

EXERCISE 10.6**Deploy Amazon RDS in a Multi-AZ Configuration.**

In this exercise, you will create an Amazon Relational Database Service (Amazon RDS) MySQL database in a multi-AZ configuration and simulate a failover.

1. Log in to the AWS Management Console, and navigate to the Amazon RDS console.
2. Launch a new Amazon RDS DB instance, and select MySQL as your Database type and then select Production MySQL Multi-AZ Deployment.
3. In the Instance Specifications dashboard, change the storage type to General Purpose SSD. Change the allocated storage to 100 GB.
4. In the Settings, set DB Instance Identifier = **certbook**, master username = **admin**, and password is **password**. These settings are for lab purposes only. Use complex names and passwords in production.
5. Create a new security group called **certbookdb** and database name = **certbookdb**. Leave everything else at its default setting.
6. Launch DB instance. (This step will take a few minutes to complete, so take a break.)
7. To simulate a failover, select your **certbookdb** Amazon RDS instance, and reboot it.
8. Confirm the reboot with failover.
9. Navigate to the Details tab (the little icon with a magnifying glass).
10. Make note of your Availability Zone, reboot again, and then notice your Availability Zone.
11. Now let's create a Read Replica of **certbookdb**.
12. Select Actions, and then select Create Read Replica Instance.
13. Enter **certbookdb-rr** as the DN identifier, and leave the settings at their defaults.
14. Navigate back to the Details page, and you will see your Read Replica.

That's it. You have created a highly available Amazon RDS MySQL database with a Read Replica. You have also simulated a failover. Now delete your Read Replica (**certbookdb-rr**) and your DB instance (**certbookdb**).

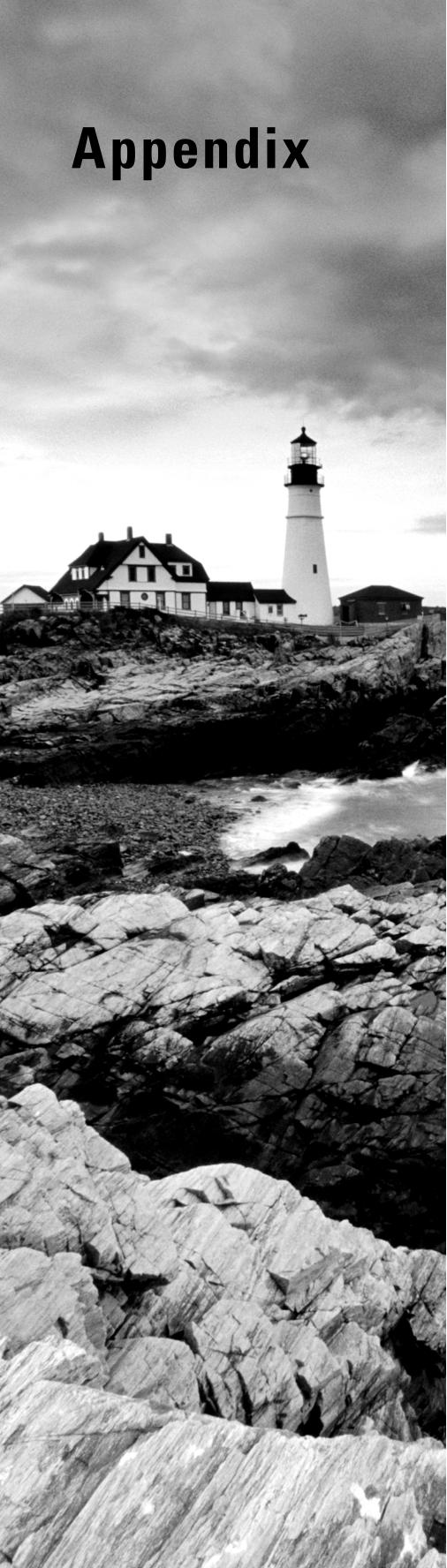
Review Questions

1. You are looking to use Amazon Simple Notification Service (Amazon SNS) to send alerts to members of your operations team. What protocols does Amazon SNS support? (Choose all that apply.)
 - A. HTTP/HTTPS
 - B. SQL
 - C. Email-JSON
 - D. Amazon Simple Queue Service (Amazon SQS)
2. What is the first step that you must take to set up Amazon Simple Notification Service?
 - A. Create a topic.
 - B. Subscribe to a topic.
 - C. Publish to a topic.
 - D. Define the Amazon Resource Name (ARN).
3. You have just finished setting up your AWS Direct Connect to connect your datacenter to AWS. You have one Amazon Virtual Private Cloud (Amazon VPC). You are worried that the connection is not redundant. Your company is going to implement a redundant AWS Direct Connect connection in the coming months. What can you suggest to management as an intermediate backup to a redundant AWS Direct Connect connection?
 - A. Recommend that your management team speed up the implementation of the AWS Direct Connect.
 - B. Nothing. AWS Direct Connect is redundant as long as you use two routers.
 - C. Create a VPN between your datacenter and your Amazon VPC.
 - D. Create another Amazon VPC.
4. Your company has just implemented their redundant AWS Direct Connect connection. What must you do in order to ensure that in case of failure, traffic will route over the redundant AWS Direct Connect link?
 - A. Traffic will failover automatically.
 - B. Manually configure your traffic to route over the redundant link.
 - C. Open a support ticket with AWS.
 - D. Open a support ticket with your telecom provider.
5. You manage an Amazon Simple Storage Service (Amazon S3) bucket for your company's Legal department. The bucket is called **Legal**. You were approached by one of the lawyers who is concerned about the availability of their Legal bucket. What could you do to ensure a higher level of availability for the Legal bucket?
 - A. Nothing. Amazon S3 is designed for eleven nines of durability.
 - B. Enable versioning on the Amazon S3 bucket.

- C. Create an Amazon S3 bucket in another region, and create an AWS Lambda function that copies data to the Amazon S3 bucket in the other region.
 - D. Create an Amazon S3 bucket in another region. Enable versioning on the bucket, and enable cross-region replication to the Amazon S3 bucket in the other region.
6. You are working with your application development team and have created an Amazon Simple Queue Service (Amazon SQS) standard queue. Your developers are complaining that messages are being processed more than once. What can you do to limit or stop this behavior?
- A. Have the developers make their application idempotent.
 - B. Deploy a First-In, First-Out (FIFO) queue and have the developers point their application to it.
 - C. Increase the visibility timeout on the queue.
 - D. Adjust the size of the Amazon Elastic Compute Cloud fleet.
7. You have a test version of a production application in an Amazon Virtual Private Cloud. The test application is using Amazon Relational Database Service (Amazon RDS) for the database. You have just been mandated to move your production version of the application to AWS. The application owner is concerned about the availability of the database. How can you make Amazon RDS highly available?
- A. Deploy Amazon RDS in one Availability Zone (AZ), and deploy a Read Replica in another Availability Zone.
 - B. Deploy Amazon RDS in a multi-Availability Zone model.
 - C. Use Amazon DynamoDB, as it is a regional service.
 - D. Deploy a Read Replica in another region.
8. You are managing a multi-region application deployed on AWS. You currently use your own DNS. What types of failover does Amazon Route 53 support? (Choose three.)
- A. Active-active failover
 - B. Active-active-passive failover
 - C. Latency-based failover
 - D. Weighted Round Robin (WRR) failover
 - E. Active-passive failover
9. Amazon Route 53 supports the following types of health checks. (Choose all that apply.)
- A. Health of a specific Amazon Virtual Private Cloud (VPC)
 - B. The health of a specified resource, such as a web server
 - C. The status of an Amazon CloudWatch Alarm
 - D. The status of other health checks
10. How do you make your hardware VPN redundant?
- A. Deploy your own software VPN solution on AWS, ensuring that the software-based VPN is deployed to two separate Availability Zones. Take down the hardware VPN.

- B. Deploy two customer gateways. Configure your customer gateways to take advantage of the two tunnels provided by the hardware VPN.
 - C. Deploy Cloud Hub.
 - D. The hardware VPN is redundant in itself, and no additional configuration is required.
11. You are part of the disaster recovery team, and you are working to create a DR policy. What is the bare minimum type of disaster recovery that you can perform with AWS?
- A. Multi-site between your datacenter and AWS
 - B. Pilot-light method
 - C. Warm-standby method
 - D. Backup-restore method
12. As you are getting more familiar with AWS, you are looking to respond better to disasters that may happen in your datacenter. Traditional backup and restore operations take too long. The longest part of the restore process is the database. What would be a better option?
- A. Pilot-light method
 - B. Active-active sites
 - C. AWS Snowball
 - D. AWS Storage Gateway stored mode
13. You are not seeing network traffic flow on the AWS side of your VPN connection to your Amazon VPC. How do you check the status?
- A. Deploy a third-party tool to monitor your VPN.
 - B. Go to the AWS management tool, and navigate VPN connections.
 - C. Check your router's logs.
 - D. Turn on route propagation in the Amazon VPC's main routing table.
14. You have just inherited an application that has a recovery point objective (RPO) of zero and a recovery time objective (RTO) of zero. What type of disaster recovery method should you deploy?
- A. Multi-Site active-active method
 - B. Warm-standby method
 - C. Backup-restore method
 - D. Pilot Light
15. You have an application with a recovery point objective of zero and a recovery time objective of 20 minutes. What type of disaster recovery method should you use?
- A. Multi-Site active-active
 - B. Warm-standby method
 - C. Backup-Restore
 - D. Pilot Light

Appendix



Answers to the Review Questions

Chapter 1: Introduction to Systems Operations on AWS

1. A. Amazon CloudWatch is a monitoring service for AWS cloud resources and the applications you run on AWS. You can use Amazon CloudWatch to collect and track metrics, collect and monitor log files, set alarms, and automatically react to changes in your AWS resources.
2. D. Amazon VPC Flow Logs is a feature that enables you to capture information about the IP traffic going to and from network interfaces in your VPC. Flow log data is stored using Amazon CloudWatch Logs.
3. B. AWS CloudTrail is a web service that records API calls made on your account and delivers log files to your Amazon S3 bucket.
4. C. Hybrid cloud architecture is the integration of on-premises resources with cloud resources.
5. B. A security group acts as a virtual firewall that controls the traffic for one or more instances.
6. A, B, D. A Three-Tier architecture on AWS may consist of a web front end, an application layer, and a database layer.
7. D. Amazon EC2 is hosted in multiple locations world-wide. These locations are composed of regions and Availability Zones. Each region is a separate geographic area. Each region has multiple, isolated locations known as Availability Zones. Amazon EC2 provides you with the ability to place resources, such as instances and data, in multiple locations. Resources aren't replicated across regions unless you do so specifically.
8. D. With AD Connector, you can streamline identity management by sourcing and managing all of your user identities from Active Directory. It also enables you to reuse your existing Active Directory security policies, such as password expiration, password history, and account lockout policies. Also, your users will no longer need to remember yet another user name and password combination.
9. A. A virtual private cloud (VPC) is a virtual network dedicated to your AWS account. It is logically isolated from other virtual networks in the AWS cloud. You can launch your AWS resources, such as Amazon EC2 instances, into your Amazon VPC.
10. B. You can host a static website on Amazon S3. On a static website, individual web pages include static content. They may also contain client-side scripts. By contrast, a dynamic website relies on server-side processing, including server-side scripts such as PHP, JSP, or ASP.NET. Amazon S3 does not support server-side scripting.

Chapter 2: Working with AWS Cloud Services

1. B. The AWS CLI for Mac and Linux requires Python to be installed to operate. For Windows, the installer contains all of the required dependencies.
2. A. JSON is a structured format that is handy for creating loosely-formatted data to be processed programmatically. Text is most useful for the query command or text-processing utilities. Table format is intended to be used by operators to create tables of output that can be read by humans.
3. C. You can use the backslash key to indicate that you wish to continue a line after you move to a new line. At the end of the last line, simply press Enter without a backslash to then execute the full command line.
4. A, C, D. AWS SDKs are available for Ruby, Perl, Python, and many other languages, but Pascal and Basic aren't among them.
5. C. While A will return the general AWS CLI help screen, only C will return the syntax and options specific to the AWS IoT service.
6. C. The waiter in Boto provides a handler for your code to deal with the short wait that occurs when you perform tasks like launching an instance or creating a new Amazon Redshift cluster.
7. D. Like the AWS Mobile SDK, the AWS IoT Device SDK provides prebuilt libraries that ease the process of connecting your devices to the AWS IoT Service.
8. B. Some AWS CLI inputs can be encoded in JSON format and passed through to using the --cli-input-json parameter.
9. D. Querying (--query) output data, in text format, is a powerful tool for making sense of complex cloud deployments.
10. B. Your keys are stored in the credentials file, located in either `~/.aws/credentials` on Mac or Linux or in `%UserProfile%\aws\credentials` on Windows.

Chapter 3: Security and AWS Identity and Access Management (IAM)

1. D. Both Amazon Web Services and you are responsible for securing workloads running in the AWS Cloud. AWS Secures the infrastructure of the cloud. You secure the way they use and configure the services provided by AWS.
2. D. AWS is responsible for the physical and environmental security in its datacenters. This includes fire detection, fire suppression, electrical power redundancy, climate, and temperature control.

3. A. The AWS network provides protection against traditional network security issues. AWS API endpoints are hosted on large, Internet-scale, world-class infrastructure that is continuously monitored.
4. B. AWS Identity and Access Management Service provides centralized management of access and authentication of users to the AWS services in an AWS account. AWS Directory Service provides a directory for your application-specific resources. Amazon Cognito provides services for users to sign up and sign in to your apps, federate identities from social identity providers, secure access to AWS resources, and synchronize data across multiple devices, platforms, and applications. AWS Config enables the assessment, auditing, and evaluation of the configurations of AWS resources.
5. B, D. An IAM user uses their IAM user name and a password to access AWS services via the AWS Management Console. An IAM user uses access keys to access AWS services via the AWS Command Line Interface.
6. B. IAM can be used to set a password policy that requires at least one lowercase letter and at least one non-alphanumeric character.
7. C. Access keys consist of an Access Key ID and a Secret Access Key.
8. D. All of the devices listed can be used to provide Multi-Factor Authentication for an AWS account.
9. C. An IAM user can be a member of multiple groups and inherit the permissions from those groups. The other options are not available. There are no default groups in IAM, groups cannot be nested, and IAM roles cannot be members of a group.
10. D. Best practice is to use IAM roles to provide cross-account access. You can delegate access to others by using roles instead of sharing credentials.
11. B. Keys are protected by access control policies that are defined and managed by you. All API requests to AWS KMS must be made using TLS. No single Amazon employee can gain access to Customer Managed Keys (CMKs). There is no mechanism to export CMKs into plaintext. Key usage is isolated to a defined AWS Region. Keys held by KMS in one region cannot be used in another region.
12. A. AWS CloudTrail logs the API requests to AWS resources within your account. Amazon VPC Flow Logs capture information about the IP traffic going to and from network interfaces. Amazon CloudWatch monitors the utilization of AWS resources in your account. AWS Config captures information on configuration changes to AWS resources in your AWS account.
13. D. Customers can run an agent on their Amazon EC2 instances to pass operating system logs to Amazon CloudWatch Logs. Additionally, AWS CloudTrail logs, Amazon VPC, ELB, and ENI Flow Logs can be passed to Amazon CloudWatch Logs.
14. B. AWS Config records configuration changes within an AWS account. Using AWS Config logs in conjunction with AWS CloudTrail logs allows changes to be identified and details about those changes (who, when, how) to be seen.

15. B. Amazon Inspector is an automated security assessment service that helps improve the security and compliance of applications and operating systems deployed on Amazon EC2 instances. AWS Trusted Advisor compares how AWS services are being used in your account against the Cost Optimization, Performance Efficiency, Security, and Reliability pillars of the Well Architected Framework.
16. D. Network Access Control Lists can allow and deny network traffic based on the source or target IP address and the port number. Security Groups deny traffic by default and can be configured to allow traffic based on IP address or CIDR range and port into an Amazon EC2 instance or group of EC2 instances. For the exam, remember that subnets use Network Access Control Lists to control network traffic entering and leaving the subnet. EC2 instances use Security Groups to control network traffic entering and leaving an instance.
17. C. A virtual private gateway attached to an Amazon VPC will enable you to pass the traffic from your on-premises network to the VPC over a VPN connection. Your on-premises network will use a VPN appliance, also known as the customer gateway, to complete your side of the VPN connection.
18. A. To protect data at rest in Amazon DynamoDB, customers should use client-side encryption. Amazon DynamoDB does not provide server-side encryption to protect the data at rest.
19. D. The database instance can be protected using security groups to restrict network traffic reaching the instance. Network ACLs can be used to restrict the network traffic flowing into the subnets containing the database instance. Placing the database instance in a private subnet ensures that it only has a private IP address so that network traffic is not directly routable to or from the Internet to the database instance.
20. C. Amazon Simple Queue Service (Amazon SQS) encrypts data at rest and Amazon Simple Notification Service (Amazon SNS) doesn't encrypt data at rest.

Chapter 4: Compute

1. C. Resizing an instance is easily done, but it does involve a service interruption as you need to stop the instance and then restart it with the correct instance size.
2. B. On-Demand, while not guaranteeing that the job will complete, is the best option of the choices given. Spot pricing, by its nature, is used for bidding on excess capacity; if no excess capacity is available, then the job will not run. Doubling the number of instances needed will not guarantee job completion either.
3. A. In AWS Batch, you are able to specify minimum, maximum, and desired number of CPUs.
4. C. Roles control the ability to execute Application Programming Interfaces (APIs). Logging in to an Amazon EC2 instance via SSH does not involve the execution of an API.
5. D. Amazon EBS-backed Amazon EC2 instances boot faster than instance store-backed Amazon EC2 instances. Having user data also slows down the boot process.

6. B. The My Billing Dashboard will not only give you your current monthly spending, but it will also provide you with an estimate of your total monthly bill.
7. A. The operating systems offered in Amazon Lightsail are Amazon Linux and Ubuntu.
8. D. AWS Lambda has a runtime of 5 minutes, so 35 minutes is too long for a Lambda function; the code has to run on an Amazon EC2 instance. Choosing a Reserved instance allows you to minimize your cost while maximizing the availability of the Amazon EC2 instance.
9. D. AWS Lambda needs to receive a notification, and Amazon SQS does not provide one.
10. B. You can use AWS Elastic Beanstalk to spin up infrastructure that is both externally facing and internally facing.
11. B. Most databases need a guaranteed level of I/O. An instance that is Amazon EBS-optimized (and is using Provisioned IOPS) helps you meet that need.
12. E. SSD-based Amazon EBS gives you low latency. The Provisioned IOPS, however, gives you a greater level of control regarding input and output.
13. E. Instance-storage backed Amazon EC2 instances cannot be stopped; they can only be running or deleted.
14. B. When an Amazon EC2 instance is stopped, the public IP address is removed. If you need the IP to persist, use an Elastic IP.
15. D. You can copy an AMI across AWS Regions using the CopyImage API. You can also use the AWS Management Console, the AWS Command Line Interface (AWS CLI), or AWS Software Development Kits (SDKs). Remember that the AMI ID will change from region to region.
16. C. You can spin up an Apache Web Server by putting the necessary software (or link to the software) in the user data field.
17. B. You want to assign a role instead of making the Amazon EC2 instance a user. Route tables are not assigned to Amazon EC2 instances.
18. C. Running your compute instances in multiple Availability Zones makes the application highly available. Production environments should never run in a single Availability Zone.
19. D. CPU family availability is controlled by AMI choice. Not all compute families are available for all AMIs.
20. B. The number following the letter of the instance family indicates the generation of that family.

Chapter 5: Networking

1. D. Access logs contain more detailed information than Amazon CloudWatch, including information on the IP address of the client making the request.
2. C. When you change the port used for the health check, you also need to update the security group to allow traffic both into and out of that port.

3. D. In order to make things highly available, you need to eliminate single points of failure. Answers A and C have single points of failure. Answer B does not give you the scaling that you may need.
4. E. The latency measurement from your Las Vegas location will not be based on Las Vegas but will be based on the AWS Region that you associate with that record. In this case, it is recommended that you associate the us-west-1 Region located in Northern California. AWS can only create latency record sets in an AWS Region, not in a customer location.
5. D. All these conditions need to be met in order to be able to SSH successfully into an Amazon EC2 instance.
6. C. Placing an instance in a private subnet prevents anyone from being able to access that instance directly from the Internet. It can only be accessed from within the Amazon VPC. This makes the instance more secure.
7. D. The only routing protocol that AWS Direct Connect supports is BGP. Hand-off is a fiber-supported hand-off, and you need a public Autonomous System Number for public VIF and a private Autonomous System Number for private VIF.
8. A. AWS Direct Connect, because it is on a managed network, offers reduced latency. It also reduces your data transfer out costs, and you can use it to connect Amazon VPCs not in the same region.
9. B. A private VIF is the logical interface between your on-premises network and AWS resources located in an Amazon VPC.
10. E. Inbound rules apply to packets entering a particular subnet to which the network ACL is applied.
11. D. You need an Internet gateway to connect to services outside of the Amazon VPC. While some services (Amazon S3, for example) have endpoints that eliminate the need for an Internet gateway, not all AWS services have endpoints.
12. B. Network ACLs can be used both to allow and to block traffic. While it is a good practice to inform AWS of any attacks that you may be experiencing, blocking the whole subnet would also potentially block good traffic.
13. D. You must establish a direct Amazon VPC peering relationship between each VPC. CloudHub doesn't work between VPCs, and there is no such thing as VPN peering on AWS.
14. D, E. The nodes of an Internet-facing load balancer have public IP addresses. The DNS name of an Internet-facing load balancer is publicly resolvable to the public IP addresses of the nodes. Therefore, Internet-facing load balancers can route requests from clients over the Internet. The nodes of an internal load balancer have only private IP addresses. The DNS name of an internal load balancer is publicly resolvable to the private IP addresses of the nodes. Therefore, internal load balancers can only route requests from clients with access to the Amazon VPC for the load balancer.
15. C. A default Amazon VPC has a netmask of /20. It comes with an Internet gateway and a route table reflecting that Internet gateway. NAT and VPN can be added to the default VPC.

16. A. Once enabled, AWS CloudTrail records all API calls made in your account. Amazon CloudWatch is a monitoring service for AWS Cloud resources and the applications you run on AWS. Amazon VPC Flow Logs allow you to monitor traffic into and out of VPCs, subnets, and interfaces.
17. A. Amazon CloudFront will forward a file to a user as soon as it gets its first bytes. It does no error checking.
18. B, C. Origin Access Identity (OAI) is used to restrict access to your Amazon S3 content. Policies that restrict access by IP address would be difficult both to establish and maintain.
19. C. The subnet inherits the route table from the Amazon VPC.
20. D. The HTTP server can be either an Amazon EC2 instance configured as an HTTP server, a server that you manage, or an Amazon S3 bucket. For media files, it has to be an S3 bucket.

Chapter 6: Storage Systems

1. D. While Amazon Glacier is the cheapest storage option available, content cannot be searched or served from Amazon Glacier. Every other option is significantly more expensive to maintain and would not provide more availability.
2. C. The magnetic volumes have a max IOPS of 100 each, and provisioned IOPS are extremely expensive given the fact that the baseline performance of a 2 TB volume is already at 6,000 IOPS. Having two volumes does not improve the performance and instead adds to the complexity of the operation.
3. C. Any Amazon EBS volume can only be mounted to one Amazon EC2 instance at a time. Amazon S3 volumes and Amazon EBS m1 volumes do not exist.
4. D. Amazon EBS is only cost effective when read/write actions are involved. Amazon S3-RSS is cheaper than standard Amazon S3 but significantly less durable than other Amazon S3 methods. Amazon S3-IA is less expensive than Amazon-RSS as long as the number of retrievals is low. Amazon Glacier is the cheapest option, but only if the content is almost never retrieved or the customer can wait three to five hours for retrieval—the response time requirement invalidates this option.
5. A, B, E. A and B are both static files and should be immediate choices. C, D, and F are classic anti-patterns, as they are heavy write objects. E is actually a great use case for Amazon S3, and AWS supports many big data workflows with Amazon S3 as the data repository. If you didn't know that, you could get there by the process of elimination.
6. B, D. IAM logging does not exist. All Application Programming Interface (API) calls are logged using AWS CloudTrail. Server Access Logs are used to track outside access to Amazon S3. While ClearCut logging exists, it is not a function of IT.

7. A. Amazon Glacier Vault Lock is the only option that is unchangeable by any user, even the root account. Amazon S3 Bucket Lock does not exist.
8. B. These are all four of the options available for AWS Storage Gateway as of this writing. The tape interface is for static backups (point in time). The file interface might be an option, except that the system is a Network File System (NFS) mount point, whereas the volume interface is an iSCSI block storage endpoint. Of the two volume interface modes, cached allows you to grow your storage in AWS, whereas the stored mode is 1:1.
9. D. Scratch disk is a perfect use case for the ephemeral nature of instance storage.
10. B. All other options do not apply to Amazon EFS. To get started with EFS, you need to create EFS mount points in each Availability Zone.
11. C, D. Amazon S3 does not use a file system to store its data. Objects are stored in buckets. Objects can be no larger than 5TB, and they can contain data and metadata.
12. A, C. There are no tapes in the AWS infrastructure. Amazon EBS volumes persist when the instance is stopped. The data is automatically replicated within an Availability Zone. Amazon EBS volumes can be encrypted upon creation and used by an instance in the same manner as if they were not encrypted.
13. B. There is no delay in processing when commencing a snapshot.
14. B. The volume is created immediately, but the data is loaded lazily. This means that the volume can be accessed upon creation, and if the data being requested has not yet been restored, it will be restored upon first request.
15. C, E. Versioning protects data against inadvertent or intentional deletion by storing all versions of the object, and MFA Delete requires a one-time code from a Multi-Factor Authentication (MFA) device to delete objects. Cross-region replication and migration to the Amazon Glacier storage class do not protect against deletion. Vault locks are a feature of Amazon Glacier, not a feature of Amazon S3.
16. A, C. Amazon Glacier is optimized for long-term archival storage and is not suited to data that needs immediate access or short-lived data that is erased within 90 days.
17. B. Currently, Amazon EFS supports Linux.
18. B. There is no minimum limit to the amount of data that can be stored in Amazon Glacier, and individual archives can be from 1 byte to 40 terabytes.
19. B. Instance storage doesn't support snapshots. This storage type would need to be backed up via traditional means.
20. C. Instance volumes cannot be attached after an Amazon EC2 instance has been launched. Instance storage doesn't support snapshots. There is no re-size command. The correct command is the `modify-volume` command, and that command applies to Amazon Elastic Block Storage, not instance storage.

Chapter 7: Databases

1. D. Internally, there are no transfer charges. Chargers are only incurred when data leaves AWS.

2. C, D. There is no such thing as an SSD-Backed Instance.

In this question, Amazon EBS was chosen for the persistence of data outside the lifecycle of an Amazon EC2 instance. Data on an instance store-backed instance is lost when that instance is terminated.

Improving performance comes from using Amazon EBS optimized instances and Provisioned IOPS.

3. C. The Point-In-Time-Restore and Snapshot Restore features of Amazon RDS for MySQL require a crash-recoverable storage engine and are supported for the InnoDB storage engine only. While MySQL supports multiple storage engines with varying capabilities, not all of them are optimized for crash recovery and data durability.

4. C. FreeStorageSpace is the name of the metric that returns available storage space. It is reported in bytes.

5. B. If an application's read or write requests exceed the provisioned throughput for a table, Amazon DynamoDB might throttle that request. When this happens, the request fails with an HTTP 400 code (Bad Request), accompanied by a ProvisionedThroughputExceededException. The AWS SDKs have built-in support for retrying throttled requests.

6. C. This is the amount of time a Read Replica DB instance lags behind its source DB instance. It applies to MySQL, MariaDB, and PostgreSQL Read Replicas.

7. A. Use SSL from an application to encrypt a connection to a DB instance running MySQL, MariaDB, Amazon Aurora, SQL Server, Oracle, or PostgreSQL. Each DB engine has its own process for implementing SSL.

8. D. Amazon RDS creates storage volume snapshots of DB instances, backing up entire DB instances and not just individual databases. Creating a DB snapshot on a Single-AZ DB instance results in a brief I/O suspension that can last from a few seconds to a few minutes, depending on the size and class of the DB instance. Multi-AZ DB instances are not affected by this I/O suspension since the backup is taken on the standby.

When creating a DB snapshot, identify which DB instance to back up, and then give the DB snapshot a name so that it can be restored later. If IAM database authentication is enabled, this setting is inherited from the source DB instance.

9. C. Amazon RDS creates automated backups of your DB instance during the backup window of your DB instance. Amazon RDS saves the automated backups of your DB instance according to the backup retention period that you specify up to 35 days.

If necessary, you can recover your database to any point in time during the backup retention period.

10. B. In Amazon DynamoDB, tables, items, and attributes are the core components with which you work. A table is a collection of items, and each item is a collection of attributes. Amazon DynamoDB uses primary keys to identify each item uniquely in a table and secondary indexes to provide more querying flexibility.

11. B. The maximum item size in Amazon DynamoDB is 400 KB, which includes both attribute name binary length (UTF-8 length) and attribute value lengths (again binary length).

The attribute name counts toward the size limit. For more information, see this documentation:

<http://docs.aws.amazon.com/amazondynamodb/latest/developerguide/Limits.html>

12. B. Standby instances are automatically provisioned in a different AZ of the same region as the primary DB instance.

13. C. You can create a second-tier Read Replica from an existing first-tier Read Replica. By creating a second-tier Read Replica, you may be able to move some of the replication load from the primary database instance to a first-tier Read Replica.

Please note that a second-tier Read Replica may lag further behind the primary because of additional replication latency introduced as transactions are replicated from the primary to the first-tier replica and then to the second-tier replica.

14. B. Amazon RDS automatically provisions and maintains a synchronous standby replica in a different AZ. The primary DB instance is synchronously replicated to a standby replica to provide data redundancy, eliminate I/O freezes, and minimize latency spikes during system backups.

Running a DB instance with high availability can enhance availability during planned system maintenance and help protect your databases against DB instance failure and Availability Zone disruption.

The high availability feature is not a scaling solution for read-only scenarios; you cannot use a standby replica to serve read traffic. To service read-only traffic, you should use a Read Replica.

15. A. The promotion of a Read Replica to a primary DB has no effect on other Read Replicas. It causes two things to happen:

- Existing replication stops.
- The Read Replica becomes a standalone DB instance.

16. B. The failover mechanism automatically changes the DNS record of the DB instance to point to the standby DB instance. As a result, you will need to re-establish any existing connections to your DB instance.

<https://aws.amazon.com/blogs/aws/amazon-rds-multi-az-deployment/>

17. C. `DescribeEvents` returns events related to DB instances, DB security groups, DB snapshots, and DB parameter groups for the past 14 days.

Events specific to a particular DB instance, DB security group, database snapshot, or DB parameter group can be obtained by providing the name as a parameter.

By default, the past hour of events is returned.

<http://docs.aws.amazon.com/cli/latest/reference/rds/describe-events.html>

- 18.** A. If your DB instance is a Multi-AZ deployment, you can force a failover from one Availability Zone to another when you select the Reboot option.

When you force a failover of your DB instance, Amazon RDS automatically switches to a standby replica in another Availability Zone and updates the DNS record for the DB instance to point to the standby DB instance.

As a result, you will need to clean up and re-establish any existing connections to your DB instance.

Reboot with failover is beneficial when you want to simulate a failure of a DB instance for testing, or restore operations to the original AZ after a failover occurs.

http://docs.aws.amazon.com/AmazonRDS/latest/UserGuide/USER_RebootInstance.html

- 19.** A. In a Multi-AZ deployment, Amazon RDS automatically provisions and maintains a synchronous standby replica in a different Availability Zone.

The primary DB instance is synchronously replicated across Availability Zones to a standby replica to provide data redundancy, eliminate I/O freezes, and minimize latency spikes during system backups.

<http://docs.aws.amazon.com/AmazonRDS/latest/UserGuide/Concepts.MultiAZ.htm>

- 20.** B. Amazon ElastiCache is a web service that makes it easy to deploy, operate, and scale an in-memory data store or cache in the cloud.

<https://aws.amazon.com/elasticsearch/>

- 21.** B, D, E. The Amazon Relational Database Service, Amazon ElastiCache, and Amazon DynamoDB are all database services. Amazon SQS is a message queuing service, Amazon CloudWatch is an event monitoring service, and Amazon CloudFront is a content distribution service.

- 22.** D. You can use this caching to improve latency and throughput significantly for many read-heavy application workloads, such as social networking, gaming, media sharing, and Q&A portals.

<https://d0.awsstatic.com/whitepapers/performance-at-scale-with-amazon-elasticsearch.pdf>

- 23.** C, F. Both Amazon RDS and Amazon Redshift include the ability to take snapshots of data automatically and store them, durably in Amazon S3.

Chapter 8: Application Deployment and Management

- 1.** B. AWS OpsWorks is a configuration management service that uses Chef, an automation platform that treats server configurations as code. OpsWorks uses Chef to automate how servers are configured, deployed, and managed across your Amazon Elastic Compute Cloud (Amazon EC2) instances or on-premises compute environments.

2. A. AWS CloudFormation is a service that helps you model and set up your Amazon Web Services resources so that you can spend less time managing those resources and more time focusing on your applications that run in AWS. You create a template that describes all of the AWS resources that you want (like Amazon EC2 instances or Amazon RDS DB instances), and AWS CloudFormation takes care of provisioning and configuring those resources for you.
3. E. Format, Version, Description, Metadata, and Parameters are optional sections of an AWS CloudFormation template. Only the Resources section is required.
4. A, B. An AWS CloudFormation template is a JSON- or YAML-formatted text file that describes your AWS infrastructure.
5. B. AWS Elastic Beanstalk does not support the C language. It provides platforms for other programming languages (such as Java, PHP, Python, Ruby, and Go), web containers (Tomcat, Passenger, and Puma), and Docker containers, with multiple configurations of each.
6. A, B, D. You can deploy Docker on Amazon EC2, Docker containers on the Amazon EC2 Container Service, and deploy single Docker containers on AWS Elastic Beanstalk.
7. A, B, E. There is no such thing as AWS CloudFormation conditional deployments. AWS OpsWorks is its own service and doesn't apply to AWS Elastic Beanstalk.
8. B. AWS CloudFormation, AWS OpsWorks, and AWS CodeDeploy support blue/green deployments.
9. A, D. AWS CloudFormation provides two methods for updating stacks: direct update or creating and executing change sets.
10. D. The optional Mappings section matches a key to a corresponding set of named values. For example, if you want to set values based on a region, you can create a mapping that uses the region name as a key and contains the values you want to specify for each specific region. You use the Fn::FindInMap intrinsic function to retrieve values in a map.
11. B, C. AWS CloudFormation and AWS OpsWorks both support customizing applications via JSON. Java is not a supported format for templates.
12. C. AWS Elastic Beanstalk makes it even easier for developers to deploy and manage applications quickly in the AWS Cloud. Developers simply upload their application, and Elastic Beanstalk automatically handles the deployment details of capacity provisioning, load balancing, Auto Scaling, and application health monitoring. The other services listed would require developers to build templates specifically for deploying infrastructure.
13. B. AWS OpsWorks Stacks can be used to connect to a back-end database such as Amazon DynamoDB. However, OpsWorks will not deploy the DynamoDB table for you. Another deployment service would be required to provision the DynamoDB table.
14. A, C. With AWS CloudFormation you can update a stack directly but submitting a new template and CloudFormation will apply the deltas. Additionally you can create and execute a change set against a stack.

15. C. AWS Elastic Beanstalk supports this functionality natively by creating a new environment and utilizing the swap environment URLs feature.
16. C. Amazon EC2 Container Service (ECS) is a managed, highly scalable, high performance container management service that supports Docker containers and allows you to run applications easily on a managed cluster of Amazon EC2 instances.
17. C. An AWS CloudFormation template resource with the `DeletionPolicy` attribute can preserve or (in some cases) back up a resource when its stack is deleted. You specify a `DeletionPolicy` attribute for each resource that you want to control. If a resource has no `DeletionPolicy` attribute, AWS CloudFormation deletes the resource by default.
18. B. An AWS CloudFormation template resource with the `DependsOn` attribute can specify that the creation of a specific resource follows another. When you add a `DependsOn` attribute to a resource, that resource is created only after the creation of the resource specified in the `DependsOn` attribute.
19. C. You can and should use an IAM role to manage temporary credentials for applications that run on an Amazon EC2 instance. The AWS IAM Instance Profile resource creates an AWS Identity and Access Management (IAM) instance profile that can be used with IAM roles for EC2 instances. With the appropriate policy attached to the role, the instance will be able to access the Amazon S3 service.
20. A, C. AWS OpsWorks and AWS Elastic Beanstalk allow for deploying in-place upgrades of applications without replacing the instances. AWS CloudFormation does not support this directly. AWS CodeCommit is a managed repository for storing your application code, not a deployment service.

Chapter 9: Monitoring and Metrics

1. C. AWS has no visibility above the Hypervisor. Create an Amazon CloudWatch custom metric to monitor memory usage. However, threads or processes will not be reported. A third-party application is required for that exact memory reporting.
2. B. You can create a scaling policy that uses Amazon CloudWatch Alarms to determine when your Auto Scaling Group should scale out or scale in. Each CloudWatch Alarm watches a single metric and sends messages to Auto Scaling when the metric breaches a threshold that you specify in your policy. You can use alarms to monitor any of the metrics that the services in AWS which you're using send to CloudWatch, or you can create and monitor your own custom metrics.
http://docs.aws.amazon.com/autoscaling/latest/userguide/policy_creating.html
3. A. AWS CloudTrail is a service that enables governance, compliance, operational auditing, and risk auditing of your AWS account. With CloudTrail, you can log, continuously monitor, and retain events related to API calls across your AWS infrastructure. CloudTrail provides a history of AWS API calls for your account, including API calls made through the AWS Management Console, AWS SDKs, command-line tools, and other AWS services.
<https://aws.amazon.com/cloudtrail/>

4. B, D. Amazon Route 53 sends metrics to Amazon CloudWatch. CloudWatch provides detailed monitoring of Amazon Route 53 by default. Amazon Route 53 sends one-minute metrics to CloudWatch.

http://docs.aws.amazon.com/AmazonCloudWatch/latest/monitoring/r53-metricscollected_shared.html

Amazon Relational Database Service sends metrics to Amazon CloudWatch for each active database instance every minute. Detailed monitoring is enabled by default.

<http://docs.aws.amazon.com/AmazonCloudWatch/latest/monitoring/rds-metricscollected.html>

5. D. *Cause:* A server error response sent from the registered instances.

Solution: View the access logs or the error logs on your instances to determine the cause. Send requests directly to the instance (bypass the load balancer) to view the responses.

http://docs.aws.amazon.com/elasticloadbalancing/latest/classic/ts-elb-http-errors.html#ts-elb-error-metrics-Backend_5XX

6. B. Amazon CloudWatch Basic Monitoring runs at a frequency of five-minute intervals. Detailed Monitor's frequency is at one-minute intervals.

<https://aws.amazon.com/cloudwatch/details/>

7. D. Amazon CloudWatch Basic Monitoring runs at a frequency of five-minute intervals. Detailed Monitor's frequency is at one-minute intervals.

<https://aws.amazon.com/cloudwatch/details/>

8. C. Basic Monitoring metrics (at five-minute frequency) for Amazon EC2 instances are available at no additional charge, as are all metrics for Amazon EBS volumes, Elastic Load Balancers, and Amazon RDS DB instances.

<https://aws.amazon.com/cloudwatch/details/>

9. C. By default, basic monitoring is enabled when you create a launch configuration using the AWS Management Console, and detailed monitoring is enabled when you create a launch configuration using the AWS CLI or an API.

<http://docs.aws.amazon.com/autoscaling/latest/userguide/as-instance-monitoring.html>

10. A. Amazon CloudWatch doesn't differentiate the source of a metric. If you publish a metric with the same namespace and dimensions from different sources, CloudWatch treats this as a single metric. This can be useful for service metrics in a distributed, scaled system. For example, all the hosts in a web server application could publish identical metrics representing the latency of requests that they are processing. CloudWatch treats these as a single metric, allowing you to get the statistics for minimum, maximum, average, and sum of all requests across your application.

<http://docs.aws.amazon.com/AmazonCloudWatch/latest/monitoring/cloudwatch-concepts.html>

11. A. Each metric data point must be marked with a timestamp. The timestamp can be up to two weeks in the past and up to two hours into the future. If you do not provide a

timestamp, Amazon CloudWatch creates a timestamp for you based on the time the data point was received.

http://docs.aws.amazon.com/AmazonCloudWatch/latest/monitoring/cloudwatch_concepts.html

- 12.** B. `put-metric-data` publishes metric data points to Amazon CloudWatch. Amazon CloudWatch associates the data points with the specified metric. If the specified metric does not exist, Amazon CloudWatch creates the metric. When Amazon CloudWatch creates a metric, it can take up to 15 minutes for the metric to appear.

http://docs.aws.amazon.com/AmazonCloudWatch/latest/APIReference/API_PutMetricData.html

- 13.** C. With CloudTrail, you can log, continuously monitor, and retain events related to API calls across your AWS infrastructure. Amazon CloudWatch Events delivers a near real-time stream of system events that describe changes in AWS resources to Amazon EC2 instances, AWS Lambda functions, Amazon Kinesis Streams, Amazon ECS tasks, Step Functions state machines, Amazon SNS topics, Amazon SQS queues, or built-in targets.

<https://aws.amazon.com/cloudtrail/>

<http://docs.aws.amazon.com/AmazonCloudWatch/latest/events/WhatIsCloudWatchEvents.html>

Chapter 10: High Availability

- 1.** A, C, D. Amazon Simple Notification Service (Amazon SNS) supports these protocols: HTTP/HTTPS, Email/Email-JSON, Amazon SQS, and SMS.
- 2.** A. To get started with Amazon SNS, first create your topic, select your protocol(s), subscribe to the topic, and then publish to your topic.
- 3.** C. Some AWS customers would like the benefits of one or more AWS Direct Connect connections for their primary connectivity to AWS, combined with a lower-cost backup connection. To achieve this objective, they can establish AWS Direct Connect connections with a VPN backup.
- 4.** A. If you have established a second AWS Direct Connect connection, traffic will failover to the second link automatically. It is recommended that you enable Bidirectional Forwarding Detection (BFD) when configuring your connections to ensure fast detection and failover.
- 5.** D. Cross-region replication is a bucket-level feature that enables automatic, asynchronous copying of objects across buckets in different AWS Regions. To activate this feature, you add a replication configuration to your source bucket. In the configuration, you provide information, such as the destination bucket for where you want objects replicated. You can request Amazon S3 to replicate all or a subset of objects with specific key name prefixes. For example, you can configure cross-region replication to replicate only objects with the key name prefix `Tax/`. This causes Amazon S3 to replicate objects with a key such as `Tax/doc1` or `Tax/doc2`, but not an object with the key `Legal/doc3`.

6. C. Visibility timeout is the period of time that a message is invisible to the rest of your application after an application component gets it from the queue. During the visibility timeout, the component that received the message usually processes it and then deletes it from the queue.
7. B. Amazon RDS Multi-AZ deployments provide enhanced availability and durability for Database (DB) Instances, making them a natural fit for production database workloads. When you provision a Multi-AZ DB Instance, Amazon RDS automatically creates a primary DB Instance and synchronously replicates the data to a standby instance in a different Availability Zone (AZ).
8. A, B, E. You can set up a variety of failover configurations using Amazon Route 53 alias, weighted, latency, geolocation routing, and failover resource record sets. You can choose Active-active failover, Active-passive failover, Active-active-passive, and other mixed configurations.
9. B, C, D. Amazon Route 53 health checks monitor the health and performance of your web applications, web servers, and other resources. Each health check that you create can monitor one of the following: the health of a specified resource, such as a web server; the status of an Amazon CloudWatch Alarm; and the status of other health checks.
10. B. While A is correct, the question asks how to make the hardware VPN redundant. The software VPN supports two tunnels. Simply download the router configuration from the AWS Management Console and configure two customer gateways with the endpoints for the hardware VPN's Virtual Private Gateway.
11. D. Amazon S3 is an ideal destination for backup data that might be needed quickly to perform a restore. Transferring data to and from Amazon S3 is typically done through the network, and it is therefore accessible from any location. There are many commercial and open source backup solutions that integrate with Amazon S3 and Amazon Glacier.
12. A. While AWS Storage Gateway stored mode would make data restorations quicker, it is the database that takes the longest to restore. Therefore, pilot-light would be the better option as you keep a small database in sync with your production database located in your datacenter. If a disaster strikes, you resize your database to handle production load.
13. D. If you've attached a virtual private gateway to your Amazon VPC and enabled route propagation on your route table, routes representing your VPN connection automatically appear as propagated routes in your route table.
14. A. A multi-site solution runs in AWS as well as on your existing on-site infrastructure in an active-active configuration. The data replication method that you employ will be determined by the recovery point that you choose. With a recovery point objective of zero, you may want a fully-sized DR site that will be running in tandem with your production workload. When disaster strikes, all traffic will be routed to a site that is scaled to handle 100 percent of your application's expected load.
15. B. The term warm-standby is used to describe a DR scenario in which a scaled-down version of a fully-functional environment is always running in the cloud. A warm-standby solution extends the pilot light elements and preparation. It further decreases the recovery time because some services are always running. By identifying your business-critical systems, you can fully duplicate these systems on AWS and have them always on.

Index

A

access control, 484. *See also* AWS Identity and Access Management
Amazon CloudWatch, 379–381
Amazon RDS, 81
Amazon Redshift clusters, 293–294
fine-grained, 81
for load balancers, 178
network isolation for, 82
Route 53, 185
Access Control Lists (ACLs), 3, 8
Amazon S3, 75
Amazon VPC, 157, 158
network, 71, 72, 157, 158, 485, 487
network security and, 48, 71
access key, 25, 55, 484
access logs, 486
Amazon CloudFront, 192
Amazon S3, 13, 77
Elastic Load Balancing, 63, 177
ENIs, 63
ACID, 252–253
ACLS. *See* Access Control Lists
ACM. *See* AWS Certificate Manager
Active Directory, 94, 482
Active Directory Connector (AD Connector), 13, 482
active-active failover, 458
active-active-passive failover, 458
active-passive failover, 458
AD Connector. *See* Active Directory Connector
Address Resolution Protocol (ARP), 71
administrative access, controlling, 123
Adobe Flash Media Server, 186
Adobe Media Server, 186
Adobe RTMP. *See* Real-Time Messaging Protocol
agents, 400, 401–403, 484
aggregation, of Amazon CloudWatch statistics, 375–376
alarms, 192, 365. *See also* Amazon CloudWatch Alarms
Alias records, 183, 497
Amazon API Gateway, 17

Amazon Aurora, 81, 255, 490
PostgreSQL edition, 259
primary instance, 256
read replicas, 267
reliability, 257
replication, 257
scaling up, 273
security, 258
volumes, 256–257
Amazon CloudFront, 3, 9, 11, 17, 153, 185, 234, 488, 492
AWS WAF and, 64
console reports, 194
implementation, 186–194
key pairs, 59
management, 194
monitoring, 192
security, 74–75, 194
TLS for, 191
Amazon CloudWatch, 10, 13, 63, 119, 177, 184, 192, 365, 367, 482, 492, 494, 495
Amazon EBS monitoring, 389–391
Amazon EC2 monitoring, 387–388, 421
Amazon ElastiCache monitoring, 391–392
Amazon RDS metrics, 280, 392–393
authentication and access control, 379–381
AWS charge monitoring, 406–410
AWS Cloud services integration, 382
AWS CloudTrail and, 414, 421
dashboards, 376
data reporting, 390, 403
dimensions, 372–373, 405–406
Elastic Load Balancer metrics, 393–395
events, 396
limits, 382–383
metrics, 404
common, 386–395
custom, 369–370
free, 369
retention of, 370
missing data points, 386
namespaces, 371
RDS metrics, 280
RDS monitoring, 278, 279

- services, 368–369
- statistics, 373
 - aggregation, 375–376
 - percentiles, 376–377
 - periods, 374–375
 - units, 374
- VPC flow logs, 165
- Amazon CloudWatch Alarms, 279, 365, 384–386
- Amazon CloudWatch Events, 365, 395
 - events, 396–397
 - metrics and dimensions, 398
 - rules, 397
 - targets, 397
- Amazon CloudWatch Logs, 13, 62, 279, 318, 365, 399
 - agents, 400, 401–403, 484
 - archived data, 400
 - deployment and, 317
 - dimensions, 405–406
 - log monitoring, 400–401
 - metrics, 404–405
 - searching and filtering data, 403–404
 - VPC flow logs, 165
- Amazon Cognito, 93–94
- Amazon DynamoDB, 9, 12, 251, 283, 472, 485, 490, 491, 492
 - Amazon VPC endpoints, 292
 - core components, 284–285
 - data types, 285–288
 - metadata tags, 291
 - multi-region replication, 457
 - provisioned throughput, 288–290
 - read and write consistency, 290–291
 - reserve capacity, 290
 - secondary indexes, 290
 - security, 80–81
- Amazon EBS. *See* Amazon Elastic Block Store
- Amazon EC2. *See* Amazon Elastic Compute Cloud
 - Amazon EC2 Container Registry (Amazon ECR), 327
 - Amazon EC2 Container Service (Amazon ECS), 109, 123–125, 174
 - clusters, 325–326
 - containers, 326–327
 - images, 327
 - instances, 326
 - repository, 327
 - services, 328
 - tasks, 327–328
- Amazon EC2 Dedicated Host, 74
- Amazon EC2 Systems Manager, 120
- Amazon ECR. *See* Amazon EC2 Container Registry
- Amazon EFS. *See* Amazon Elastic File System
- Amazon Elastic Block Store (Amazon EBS), 7, 9, 12, 210, 212, 486, 488
 - Amazon EFS vs., 222–223
 - bursting IOPS and throughput, 217
 - decommissioning, 220
 - EC2 capacity and, 217–218
 - EC2 configuration and, 115
 - EC2 instance types and, 113
 - EC2 instances backed by, 116–118, 116, 117
 - instance store vs., 221
 - IOPS and throughput calculation, 216
 - maximum IOPS, 215
 - maximum throughput, 215–216
 - monitoring, 220, 389–390
 - mounting volumes, 218
 - pricing, 214
 - provisioned IOPS, 217
 - provisioning, 216–218
 - security, 68–69, 220
 - snapshots, 218–219
 - status checks, 390–391
 - throughput, 215–216, 220
 - volumes, 68, 111
 - types, 213–216
- Amazon Elastic Compute Cloud (Amazon EC2), 8, 12, 16, 109, 174, 195, 388, 390, 494
 - Amazon EBS provisioning and capacity of, 217–218
 - Amazon EMR and, 89–90
 - AMI selection, 112
 - Auto Recovery, 455–456
 - AWS CLI with, 27, 116
 - block storage volumes, 210
 - configuration, 114–117
 - customer security responsibilities, 44
 - EBS volumes attached to, 212–213, 213
 - EBS-backed instances, 116–118, 117
 - features, 111
 - health checks, 175
 - implementation, 111–117

- instance purchasing options, 120–121
- instance type selection, 112–113
- key pairs, 59
- load balancing and, 153, 171
- locations, 482
- management, 117–123
- maximum IOPS, 217–218
- public IP addresses, 114
- security, 65–69, 66
- security groups, 271
- snapshots, 118
- Spot Pricing, 114
- status checks, 378–379
- storage configuration, 115–116
- store-backed instances, 116–118, 117
- Amazon Elastic File System (Amazon EFS),
 - 12, 218, 222–223, 489
- Amazon ElastiCache, 12, 87–88,
 - 296–297, 492
 - monitoring, 391–392
- Amazon EMR, security, 89–90
- Amazon Glacier, 7, 12, 230, 488
 - backup storage with, 468
 - lifecycle policies, 232
 - log archiving to, 62
 - MFA Delete, 231
 - replication, 210
 - retrieval times, 211
 - security, 78–79, 231
 - Vault Lock, 231, 489
- Amazon Infrastructure Group, 47
- Amazon Inspector, 13, 64, 123, 485
- Amazon Kinesis, 17, 90–91
- Amazon Lightsail, 109, 130–133, 486
- Amazon Machine Image (AMI), 8, 111
 - creating, 121–122, 316–317
 - creation from snapshot, 219
 - IDs, 338
 - on instance store, 222
 - key pairs, 59
 - selecting, 112
 - SSH host certificate generation, 49
 - storage formatting and, 210
 - updates to, 67
- Amazon RDS. *See* Amazon Relational Database Service
- Amazon RDS tag, 277
- Amazon Redshift, 44, 292
 - Amazon EC2-Classic and, 294
- Amazon EC2-VPC and, 294
- Amazon RDS vs., 296
- backups, 85
- billing, 295–296
- cluster access and security, 293–294
- cluster management, 293
- clusters, 293
- databases, 294
- encryption, 85–86, 294
- events and notifications, 295
- high availability considerations, 293
- monitoring, 295
- performance, 295
- security, 84–87
- Amazon Redshift Spectrum, 292, 293
- Amazon Relational Database Service (Amazon RDS), 9, 12, 44, 251, 492, 495
 - adding storage and changing storage type, 260–261
- Amazon CloudWatch metrics, 280, 392–393
- Amazon Redshift vs., 296
- automated monitoring tools, 278–280
- automatic patching, 84
- backup, 83, 268–270
- DB instances, 255–256
- encryption, 82–83, 272
- failover, 264–265, 267–268
- features and benefits, 254–255
- high availability, 265–266, 456–457
- I/O credits and burst performance, 261–263
- manual monitoring tools, 281
- monitoring, 278–281
- multi-AZ deployments, 263–264, 497
- MySQL security and, 270–271
- option groups, 272–273
- overview, 255
- pricing, 282–283
- read replicas, 266–267
- replication, 265
- restore options, 268–270, 490
- scaling, 273–276, 456–457
- security, 81–84
- security groups, 271
- snapshot copy, 269
- storage, 259–260
- supported storage engines, 263
- tagging resources, 276–277

- Amazon Resource Name (ARN), 165, 277
- Amazon Route 53, 11, 153, 322, 495, 497
- authentication and access control, 185
 - DNS record types, 183–184
 - health checks, 184, 459–460
 - for complex failover, 461–462
 - for simple failover, 458, 460
 - implementation, 180–184
 - management, 185
 - multi-region high availability, 457–458
 - routing policies, 181–183
- Amazon S3. *See* Amazon Simple Storage Service
- Amazon S3 Infrequent Access (Amazon S3-IA), 227
- Amazon S3 - Reduced Redundancy Storage (Amazon S3-RRS), 226–227
- Amazon S3 Server Side Encryption (SSE), 77
- Amazon S3-IA. *See* Amazon S3 Infrequent Access
- Amazon S3-RRS. *See* Amazon S3 - Reduced Redundancy Storage
- Amazon Simple Email Service (Amazon SES), 18
- Amazon Simple Notification Service (Amazon SNS), 18, 87, 89, 130, 368, 496
- AWS CloudTrail and, 415, 416–417
 - fan-out scenario, 451–452
 - mobile push messaging, 451
 - protocols supported by, 450, 450
 - topic policies, 416–417
 - VPC endpoints, 161
- Amazon Simple Queue Service (Amazon SQS), 17, 88, 109, 130, 492
- application decoupling with, 444–445
 - at-least-once delivery, 446–447
 - AWS Elastic Beanstalk and, 324
 - DLQs, 449
 - message order, 446
 - message sampling, 447
 - shared queues, 449
 - standard queues, 448
 - use cases, 445
 - visibility timeout, 448, 497
- Amazon Simple Storage Service (Amazon S3), 7, 9, 12, 109, 187, 488–489, 497
- access logs, 13, 77
 - accessing, 226
- ACLs, 75
- Amazon EMR and, 89
- AWS Lambda authentication and, 16
- backup storage with, 468
- buckets, 224
- CORS, 77–78
- cross-region replication, 229, 457, 472, 496
- durability vs. availability, 225–226
- Infrequent Access, 227
- log storage to, 62
- MFA Delete, 229
- Reduced Redundancy Storage, 226–227
- replication, 210, 229
- retrieval times, 211
- security, 75, 78, 228–229
 - data access, 76
 - data storage, 77
 - data transfer, 76
- URLs, 225
- versioning, 228
- web hosting with, 17, 482
- Amazon SNS. *See* Amazon Simple Notification Service
- Amazon SQS. *See* Amazon Simple Queue Service
- Amazon Virtual Private Cloud (Amazon VPC), 7, 11, 49, 111, 153
- ACLs, 157, 158
 - Amazon DynamoDB endpoints, 292
 - Classic Load Balancer security and, 171, 173
 - DB instances in, 82
 - dedicated instances, 74
 - default, 156
 - elastic network interfaces, 163
 - implementation, 154–164
 - Internet gateway, 160
 - management, 164–166
 - NAT gateway, 160
 - network architecture, 72
 - network topologies, 73
 - peering, 162, 487
 - route tables, 157
 - security, 70–74
 - security groups, 38, 158, 158–160, 271
 - in serverless architectures, 15
 - size ranges, 155
 - subnets, 15, 156, 159

- VPC endpoint, 161
- VPN gateway, 160–161, 161
- Amazon VPC. *See* Amazon Virtual Private Cloud
- Amazon VPC Flow Logs, 10, 13, 62–63, 151, 165, 482, 484
- Amazon WorkSpaces, security, 94–95
- AMI. *See* Amazon Machine Image
- Aminator, 317
- analytics services, security, 89–91
- Ansible, 317
- API. *See* Application Programming Interface
- API logging, 4
- application load balancers, 64, 171, 174–176
- Application Programming Interface (API), 4, 485
 - Amazon EC2, 67
 - Amazon VPC, 71
 - Authorize Cache Security Group Ingress, 88
 - block storage, 210
 - client, 32
 - customer security responsibilities and, 44
 - invalidation, 191
 - resource, 32, 33
- application services, security, 88–89
- Application Tier, 6
- applications
 - AWS Elastic Beanstalk, 323
 - AWS OpsWorks Stacks, 329
 - decoupling with Amazon SQS, 444–445
 - deploying, 315, 492–494
 - managing versions, 324–325
- ARN. *See* Amazon Resource Name
- ARP. *See* Address Resolution Protocol
- ARP spoofing, 71
- ASN. *See* Autonomous System Number
- at-least-once delivery, 446–447
- attributes, 285
- audit logging, 295
- Aurora Replica, 256, 257
- authentication, 484. *See also* Multi-Factor Authentication
 - Amazon Aurora, 258
 - Amazon CloudWatch, 379–381
 - Amazon RDS, 271
 - Google Authenticator, 56
 - for load balancers, 178
- MySQL, 271
- PAP, 95
- RADIUS, 94–95
- Route 53, 185
- serverless architecture, 15–17
- SSH, 15
- Authorize Cache Security Group Ingress API, 88
- Auto Recovery, 455–456
- Auto Scaling, 8, 12, 127, 318, 368, 494
 - groups, 454
 - high availability and, 454–455
- autocomplete, in AWS CLI, 27
- automatic software patching, 84, 86
- Autonomous System Number (ASN), 167, 487
- Availability Zones, 3, 7, 11, 45, 47, 69, 111, 482, 491
 - Amazon Aurora volumes, 256–257
 - Amazon RDS deployment to multiple, 263–264, 492, 497
 - Amazon Redshift, 293
 - Classic Load Balancer and, 172
 - DB instance replication and, 83
 - failover and, 265–266
 - high availability and, 265–266
 - object storage replication across, 210
 - subnets and, 154
- AWS
 - availability, 47
 - block storage on, 212–223
 - datacenters, 46–47
 - detailed billing, 407–409
 - global infrastructure security, 44–45, 45
 - internal communication, 47–48
 - managed policies, 380–381
 - monitoring charges, 406–410
 - network monitoring and protection, 49–50
 - network security, 48–49
 - object storage on, 224–232
 - open source and, 31
 - physical and environmental security, 46
 - security responsibilities, 43, 44
 - security standard compliance, 50–51
 - services, 4–5
- AWS accounts, best practices for securing, 58–59
- AWS applications, security, 94–95

- AWS Batch, 109, 133–135, 485
- AWS Billing and Cost Management service, 410
- AWS Certificate Manager (ACM), 64, 188–189
- AWS Certification, paths for, 30
- AWS Certified SysOps Administrator - Associate certification, 2
 - test domains, 4
- AWS CLI. *See* AWS Command Line Interface
- AWS Cloud services
 - Amazon CloudWatch integrations, 382
 - monitoring status of, 119
 - securing, 59–62
 - service-specific security, 65–95
 - working with, 483
- AWS CloudFormation, 10, 13, 91, 320, 321, 330, 494
 - best practices, 344–345
 - change sets, 332–334
 - stacks
 - creating, 331
 - deleting, 331
 - updating, 332
 - templates, 334–335, 493
 - conditions, 339–340
 - mapping, 337–339
 - metadata, 335–336
 - outputs, 343–344
 - parameters, 336–337
 - resources, 341–343
- AWS CloudHSM, 61–62
- AWS CloudTrail, 10, 13, 44, 62, 63, 164, 174, 177, 365, 411, 482, 484, 488, 494, 496
 - Amazon CloudWatch and, 369, 414
 - Amazon EC2 and, 119–120
 - Amazon SNS configuration for, 415, 416–417
- AWS Config and, 420–421
 - encryption, 412
 - log delivery, 412–413
 - log management and data events, 416
 - monitoring with, 413–414
 - user permissions, 416
- AWS CloudTrail Log, 280
 - finding files, 414–415
- AWS CodeBuild, 320
- AWS CodeCommit, 320
- AWS CodeDeploy, 319, 493
- AWS CodePipeline, 320
- AWS CodeStar, 320
- AWS Command Line Interface (AWS CLI), 97, 317, 444, 483, 486
 - autocomplete, 27
 - AWS KMS and, 60
 - configuration, 25–26
 - environment variables, 26–27
 - filtering results, 29
 - help, 27
 - IAM users, 25, 52
 - installing, 24
 - internet gateway through, 160
 - output types, 28
 - query option, 29
 - skeleton generation with, 345
 - source code, 27
 - unwieldy lines, 28
 - upgrading, 25
 - VPC creation, 155
 - working with services, 27
- AWS Compliance, 50–51
- AWS Config, 63, 365–366, 417, 484
 - AWS CloudTrail and, 420–421
 - pricing, 421
 - rules, 419–420
 - using, 418
- AWS Config Rules, 63
 - `aws configure`, 25–26
- AWS content delivery network service (CDN), 9
- AWS Database Migration Service, 250
- AWS Direct Connect, 153, 161, 178, 487, 496
 - with backup VPN connection, 466
 - features, 166
 - getting started with, 167–169
 - high availability, 169
 - implementation, 167–169
 - management, 169–170
 - redundant active-active connections, 465–466
 - route preference, 170
 - security, 170
 - VIFs, 167
- AWS Directory Service, 10
- AWS Elastic Beanstalk, 14, 109, 125–127, 319–320, 321, 323–325, 486, 493, 494

- AWS GovCloud, 48, 119
AWS Greengrass, 236
AWS Identity and Access Management (IAM), 10, 13, 51, 56–59, 483–485, 494
access key management, 55
Amazon Aurora, 258
Amazon RDS, 270
AWS CLI configuration and, 25, 52
AWS CloudTrail user permissions, 416
AWS Lambda functions, 17, 129
credentials, 52, 53
customer security responsibilities and, 44
instance profiles and, 317–318
load balancers and, 178
MFA, 56
password management, 53
password policy options, 53–54
security, 91–92
AWS Inspector, 10
AWS Key Management Service (AWS KMS), 16, 17, 60–61
Amazon RDS resource encryption and, 272
AWS Lambda, 109, 486
Amazon VPCs and, 15
authentication, 15–17
implementation, 128–129
invoking, 129
management, 130
security, 130
AWS Lambda@Edge, 17
AWS Management Console, 24, 486, 494, 497
AWS KMS and, 60
IAM users, 52
VPC wizard, 155
AWS Marketplace, 52
AMIs in, 112
VPN options, 179
AWS OpsWorks, 319–320, 321, 492, 493
AWS OpsWorks Stacks, 14, 320, 493
applications, 329
cookbooks, 329–330
instances, 329
layers, 329
stacks, 328
AWS Personal Health Dashboard, 119, 367
AWS Regions, 7, 11, 45, 111
S3 replication and, 229
VGWs and, 178
VPCs and, 154
AWS Security Center, 48
AWS Security Token Service (AWS STS), 81
AWS Service Catalog, 123
AWS Service Health Dashboard, 48, 119, 366–367
AWS Shield, 3, 18
AWS Simple Monthly Calculator, 121
AWS Snowball, 79, 235–236
AWS Snowmobile, 7, 236
AWS Software Development Kits (SDKs), 24, 30, 35, 483, 486, 494
Boto, 31–33
IoT, 33–34
mobile, 33–34
AWS Storage Gateway, 235, 489
security, 80
snapshots from, 468
AWS STS. *See* AWS Security Token Service
AWS Tools for PowerShell, 24, 30, 54
AWS Total Cost of Ownership Calculator, 121
AWS Trusted Advisor, 64–65, 121, 123, 366
AWS VPN CloudHub, 178, 179
AWS WAF. *See* AWS Web Application Firewall
AWS Web Application Firewall (AWS WAF), 18, 64, 176
Amazon CloudFront with, 192
-
- B**
- backups
Amazon RDS, 83, 268–270
Amazon Redshift, 85
Amazon Redshift pricing and, 296
automated, 270
disaster recovery with, 467–468, 472
retention period, 270
storage of, 268–269, 282, 468
VPN connection, 466
window for, 269–270
baking AMIs, 316–317
baselines, monitoring, 377, 378

best practices, AWS CloudFormation, 344–345
BFD. *See* Bidirectional Forwarding Detection
BI. *See* Business Intelligence
Bidirectional Forwarding Detection (BFD), 496
billing
Amazon Redshift, 295–296
AWS Billing and Cost Management Service, 410
detailed, 407–409
dimensions, 410
My Billing Dashboard, 121, 486
Bitbucket, 320
block devices, 69
block storage
on AWS, 212–223
basics, 210
cost efficiency, 211
object storage vs., 209–210
retrieval times, 211
blue/green deployments, 321–322
bootstrapping instances, 316
Border Gateway Protocol (BGP), 161, 167, 464, 487
Boto, 31–33, 483
BPG. *See* Border Gateway Protocol
buckets, 224
burst balance, 220
burst performance, 261–263
business continuity management, 47–48
Business Intelligence (BI), 292

C

cache behaviors, 186
cache cluster, 87
cache invalidation, 191–192
cache nodes, 87
cache security groups, 87
cache warming, 257
CD/CD. *See* Continuous Integration and Continuous Deliver
CDN. *See* AWS content delivery network service; content delivery network
Challenge-Handshake Authentication Protocol (CHAP), 95

change sets, 332–334
CHAP. *See* Challenge-Handshake Authentication Protocol
Chef, 317, 329, 492
CIDR. *See* Classless Inter-Domain Routing
Classic Load Balancers, 171, 172–174
Classless Inter-Domain Routing (CIDR), 156
CLI. *See* command-line interface
Client APIs, 32
climate control, 46
Cloud Services Provider (CSP), 43
CloudFront. *See* Amazon CloudFront
CloudFront Usage Charts, 192
CloudHub. *See* AWS VPN CloudHub
cluster, cache, 87
clusters
access and security, 293–294
Amazon ECS, 325–326
management, 293
monitoring, 295
CMK. *See* Customer Master Key
CMS. *See* Content Management System
command-line interface (CLI), 5. *See also*
AWS Command Line Interface
compute nodes, 293, 295
compute services, security, 65–69
configuration. *See also* AWS Config
active-active AWS Direct Connect connections, 465–466
active-active VPN connections, 463–464
AWS CLI, 25–26
AWS Config and, 418
AWS Direct Connect with VPN backup, 466
management of, 315
container services, 14. *See also* Amazon EC2 Container Service
containers
deploying, 124
Docker, 326–327, 493
images and, 326
content delivery, 193–194
content delivery network (CDN), 9, 153, 234
Content Management System (CMS), 232–234
continuous deployment, 319
Continuous Integration and Continuous Deliver (CI/CD), 320
cookbooks, 329

cookies, 190
CORS. *See* Cross-Origin Resource Sharing
Cost Explorer, 409–410
crash recovery, Amazon Aurora, 257
credentials
 file for, 483
 IAM, 52, 53
Cross-Origin Resource Sharing (CORS),
 77–78, 188, 191
cross-region replication, 229, 457, 496
CSP. *See* Cloud Services Provider
custom variables, 316
Customer Master Key (CMK), 60
 snapshot sharing and, 219
customer security responsibilities, 44
Customer Support Team, 48

D

data transfer
 Amazon RDS pricing and, 283
 Amazon Redshift billing and, 296
 S3 security and, 76
data types, Amazon DynamoDB, 285–288
data warehouses, 292
Database Management System (DBMS), 250
Database Tier, 6
databases, 6, 9, 250, 490–492. *See also*
 relational databases; *specific databases*
 ACID, 252–253
 Amazon Redshift, 294
 instance sizes and scaling, 273
 log files, 280
 non-relational, 253–254
 schemas, 252
 security, 80–88
 snapshots, 83
 SQL vs. NoSQL, 251
datacenters
 availability, 47
 physical and environmental security,
 46–47
DB instances, 255–256
DB security groups, 81
DB snapshots, 83
DBMS. *See* Database Management System
DDoS. *See* Distributed Denial of Service

Dead Letter Queue (DLQ), 130, 449
decoupling, 444–448
 queueing chain pattern for, 445, 446
default root objects, 191
default route table, 157
deploying systems, 2
deployment, 2, 10, 492–494
 of applications, 315
 blue/green, 321–322
 continuous, 319
 custom variables, 316
 hybrid, 322
 logging and, 317
 methods for, 319–322
 monitoring, 318
 rolling, 321
 scaling and, 318
 services for, 315, 322–345
 strategies, 314–322
 tagging, 315
deployment and management services,
 security, 91–92
deployment groups, 320
device detection, 187–188
DHCP. *See* Dynamic Host Configuration
 Protocol
dimensions
 Amazon CloudWatch, 372–373
 Amazon CloudWatch Events, 398
 Amazon CloudWatch Logs, 405–406
 billing, 410
direct invocation, 129
disaster recovery (DR), 269
 backup and restore method, 467–468,
 472
 defining, 467
 fail back, 471–472
 multi-site solution method, 470–471, 472
 pilot light method, 468–469, 472
 warm-standby method, 470, 472, 497
disk task virtualization, 66
Distributed Denial of Service (DDoS), 18, 49
DLQ. *See* Dead Letter Queue
DNS. *See* Domain Name System
DNS record types, 183–184
Docker, 325–328, 493
Dockerfile, 327
document stores, 254

Domain Name System (DNS), 69, 159. *See also* Amazon Route 53
in multi-site disaster recovery, 471
record types, 183–184
domain names, 180
domains, 61
DR. *See* disaster recovery
Dynamic Host Configuration Protocol (DHCP), 159
DynamoDB Stream, 285

E

EBS. *See* Amazon Elastic Block Store
EBS-backed EC2 instances, 116–118, 117
EC2. *See* Amazon Elastic Compute Cloud
ECDHE. *See* Elliptic Curve Diffie-Hellman Ephemeral
ECS. *See* Amazon EC2 Container Service
Edge Locations, 11
EFS. *See* Amazon Elastic File System
Elastic IP addresses (EIPs), 111, 159, 160, 164
Elastic Load Balancing, 12, 153, 171, 275
access logs, 63, 177
Amazon CloudWatch metrics, 393–395
in blue/green deployments, 322
high availability and, 453
security, 69–70
elastic network interfaces (ENIs), 163
access logs, 63
Elliptic Curve Diffie-Hellman Ephemeral (ECDHE), 87
encryption
in Amazon EBS, 69
Amazon RDS, 82–83, 272
Amazon Redshift, 85–86, 294
Amazon SQS, 88
AWS CloudTrail, 412
DB connections, 82
end-to-end traffic, 69–70
S3 server side, 77
snapshot sharing and, 219
end-to-end traffic encryption, 69–70
ENIs. *See* elastic network interfaces
environment platforms, 324
environment tiers, 323–324

environment variables, AWS CLI, 26–27
ephemeral storage. *See* instance store
eth0. *See* primary network interface
events, 129. *See also* Amazon CloudWatch Events
Amazon Redshift, 295
AWS CloudTrail data, 416

F

fail back, 471–472
failover, 491, 497
Amazon RDS, 264–265, 267–268
complex, 461–462
Route 53 configurations, 458
simple, 458, 460
failover routing, 182–183
fan-out scenario, 451, 451–452
fault tolerance, 297
Federal Information Processing Standard (FIPS), 48
federated users, 91
FIFO queues. *See* first-in, first-out queues
file interface, 235
fine-grained access controls, 81
FIPS. *See* Federal Information Processing Standard
FIPS 140-2 standard, 48
fire detection and suppression, 46
firewalls
in Amazon EC2, 67, 68
in Amazon VPC, 71–72
Application Load Balancers and, 176
first-in, first-out queues (FIFO queues), 448
flow logs, VPC, 10, 13, 62–63, 165, 482, 484
fortifying systems, 3
forward headers to origin, 190

G

Gemalto, 56
General Purpose volumes, 261
General Purpose (SSD) volumes, 259
geo restriction, 189
geo targeting, 188

geoblocking, 189
geolocation routing, 182
Git, 320, 327
GitHub, 320
global infrastructure security, 44–45, 45
Google Authenticator, 56
graph stores, 254
GZIP compression, 190

H

Hadoop, 89
HAProxy, 275
Hard Disk Drives (HDDs), 214
Hardened Security Appliance (HSA), 60
Hardware Security Module (HSM), 61–62
Hardware Virtual Machine (HVM), 113
HashiCorp, 317
HDDs. *See* Hard Disk Drives
health checks, 486, 497
 Application Load Balancer, 175
 Classic Load Balancer, 173
 for failover
 complex, 461–462
 simple, 458, 460
 Route 53, 184
high availability, 487, 496–497
 Amazon ElastiCache and, 297
 Amazon RDS, 265–266, 456–457
 Amazon Redshift, 293
 Auto Scaling and, 454–455
 Availability Zones and, 265–266
 AWS Direct Connect, 169
 connectivity options, 463–466
 defining, 443
 Elastic Load Balancing and, 453
 levels of, 443
 multi-region, 457–463
 NAT gateways and, 453
 session state management and, 455
High Performance Computing (HPC), 122
highly available architectures, 452, 453
HLS. *See* HTTP Live Streaming
horizontal scaling, 273, 275
hosted zones, 180
HPC. *See* High Performance Computing
HSA. *See* Hardened Security Appliance

HSM. *See* Hardware Security Module
HTTP. *See* Hypertext Transfer Protocol
HTTP cookies, 190
HTTP Live Streaming (HLS), 193
HTTP streaming, 193
HTTP/2, 175
HTTPS, 69–70, 176
HTTPS-only connections, 191
HVM. *See* Hardware Virtual Machine
hybrid cloud architecture, 11, 482
hybrid deployments, 322
Hypertext Transfer Protocol (HTTP), 175
hypervisor, 65, 66, 494

I

IaaS. *See* Infrastructure as a Service
IAM. *See* AWS Identity and Access Management
 IAM access keys, managing, 55
 IAM groups, 56
 IAM Policies, 56–57
 IAM Roles, 57–58, 91–92
 EC2 instance configuration, 115
 IAM users, 52, 58, 484
identity federation, 13
identity pool, 93
Identity Provider (IdP), 92, 94
IGW. *See* Internet Gateway
images, containers and, 327
incident response, 47
infrastructure
 provisioning, 314–315
 security of, 44–45, 45
Infrastructure as a Service (IaaS), 65
inline policies, 57
in-place upgrade, 319, 320
instance store, 221–222
instance store volumes, 111
instances, 8, 379, 387–388
 Amazon ECS, 326
 Amazon RDS pricing and, 282
 AWS OpsWorks Stacks, 329
bootstrapping, 316
database, 255–256
database scaling and, 273
dedicated, 74

isolating, 66
 metadata, 118
 primary, 256
 profiles, 317–318
 purchasing options, 120–121
 resizing, 485
 standby, 491
 storage, 115–116, 489
 store-backed, 116–118, 117
 type selection, 112–113
 types, 111
 internal attacks, 4
 Internet Gateway (IGW), 8, 11, 73, 160
 Internet of Things (IoT), AWS SDKs for,
 33–34
 invalidation API, 191
 I/O credits, 261–263
 IOPS, 215
 Amazon EC2 maximum, 217–218
 bursting, 217
 calculating, 216
 provisioned, 217, 259
 IoT. *See* Internet of Things
 IP addresses
 Elastic, 111, 159, 160, 164
 private, 120, 155
 public, 114, 120
 IP spoofing, 49
 IPsec, 49, 160, 178, 464
 IPv4, 155
 IPv6, 155
 Application Load Balancer support, 175
 Classic Load Balancer support, 173
 IT security standards, 50–51
 items, 284

J

JMESPath query language, 29
 JSON, 483
 jumbo frames, 113

K

key pairs, 59, 111
 key/value stores, 254

L

labels, AWS CloudFormation template
 parameters, 336
 latency-based routing, 182, 182
 Layer 2 security attacks, 71
 leader nodes, 293
 least privilege approach, 57
 Letter of Authorization and Connecting
 Facility Assignment (LOA-CFA), 168
 lifecycle policies, 232
 LOA-CFA. *See* Letter of Authorization and
 Connecting Facility Assignment
 load balancing, 171, 487. *See also* Elastic
 Load Balancing
 application, 64, 174–176
 authentication and access control, 178
 classic, 172–174
 database scaling and, 275
 implementation, 172–176
 management, 176–177
 security, 178
 log groups, 407–409
 log monitoring, 400–401
 logging. *See also* access logs; Amazon
 CloudWatch Logs; AWS CloudTrail;
 AWS CloudTrail Log
 Amazon Redshift databases, 295
 API, 4
 Application Load Balancer, 176
 audit, 295
 Classic Load Balancers, 173–174
 database, 280
 deployment and, 317
 TCP, 70
 user activity, 44
 long polling, 447

M

MAC. *See* Machine Access Control
 MAC spoofing, 71
 Machine Access Control (MAC), 49
 managed policies, 57
 Man-in-the-Middle (MITM), 49
 MariaDB, 81, 255, 259, 490
 Read Replicas, 266–267

- Massively Parallel Processing (MPP), 84
MaxScale, 275
Memcached, 87, 297
message order, 446
metadata
 Amazon DynamoDB tags, 291
 AWS CloudFormation templates, 335–336
 cookbooks, 330
 instance, 118
metrics, 494–496
 Amazon CloudWatch, 280, 368
 common, 386–395
 custom, 369–370
 free, 369
 retention of, 370
 Amazon CloudWatch Events, 398
 Amazon CloudWatch Logs, 404–405
 for Amazon Lightsail, 132
 Amazon RDS, 280, 392–393
 billing, 410
 custom, 369–370
 Elastic Load Balancer, 393–395
MFA. *See* Multi-Factor Authentication
MFA Delete, 229, 231, 489
Microsoft CHAP (MS-CHAP), 95
Microsoft Silverlight, 193
Microsoft Smooth Streaming Format, 193
Microsoft SQL Server, 81, 255
middleware, 6
missing data points, 386
MITM. *See* Man-in-the-Middle
mobile push messaging, 451
mobile services, security, 92–94
mobile software, AWS SDKs for, 33–34
MongoDB, 254
monitoring, 10, 482, 494–496
 Amazon CloudFront, 192
 Amazon EBS, 220, 389–390
 Amazon ElastiCache, 391–392
 Amazon RDS, 278–281
 Amazon Redshift clusters, 295
 Application Load Balancer, 176
 AWS Batch, 135
 AWS charges, 406–410
 AWS Cloud service status, 119
 with AWS CloudTrail, 413–414
 AWS Elastic Beanstalk, 127
 baselines, 377, 378
 benefits, 364–365
 Classic Load Balancers, 173
 CloudFront, 192
 clusters, 295
 defining, 365
 deployment, 318
 deployment and, 318
 ECS, 125
 logs, 400–401
 networking, 49–50
 security and, 62–65
 systems, 2
mount targets, 223
MPP. *See* Massively Parallel Processing
MS-CHAP. *See* Microsoft CHAP
Multi-AZ deployments, 263–264, 492, 497
Multi-Factor Authentication (MFA), 44, 56, 489
 for Amazon EC2 hosts, 66
 Amazon S3, 229
multi-region high availability, 457–463
multi-site solution, 470–471, 472
My Billing Dashboard, 121, 486
MySQL, 9, 81, 254, 255, 259, 490
 Read Replicas, 266–267
 security, 270–271
MySQL Proxy, 275
-
- N**
- nameservers, 180
namespaces, 371
NAT. *See* Network Address Translation
NAT gateways, 160, 453
Netflix, 317
network access, controlling, 122–123
network ACLs, 71, 72, 157, 158, 485, 487
Network Address Translation (NAT), 73, 159
Network File System (NFS), 489
network isolation, access control with, 82
network security, 48–49
networking, 486–488. *See also* Virtual Private Network
 Amazon Lightsail, 132

EC2 instance types, 113
 monitoring and protection, 49–50
 security, 69–75
 serverless architectures, 15
 three-tier architecture, 7–8
 virtual, 154
NFS. *See* Network File System
 non-relational databases, 253–254, 283. *See also* Amazon DynamoDB
 normalization, 252
 NoSQL, 9, 251, 253–254, 283

O

OAII. *See* Origin Access Identity
OAuth, 93, 94
 object storage, 9. *See also* Amazon Glacier;
 Amazon Simple Storage Service
 on AWS, 224–232
 basics, 210–211
 block storage vs., 209–210
 cost efficiency, 212
 retrieval times, 211
 object versioning, 191–192
 on-demand media, 193
 open source, AWS and, 31
 OpenID Connect, 93, 94
 optimizing systems, 3
 option groups, 272–273
 Oracle, 9, 81, 255, 259, 490
 Origin Access Identity (OAI), 488
 origin servers, 187
 output formats, for AWS CLI, 28

P

Packer, 317
 packet sniffing, 50
 Password Authentication Protocol (PAP), 95
 passwords
 managing, 53
 policy options, 53–54
 TOTP, 56
 PC peering, 162

PC-over-IP (PCoIP), 94
 performance, Amazon Redshift, 295
 physical and environmental security,
 46–47, 483
 pilot light method, 468–469, 472
pip
 AWS CLI installation with, 24
 AWS CLI upgrade with, 25
 Boto installation with, 32
 pip install, 24, 25, 32
 placement groups, 122
 Point-In-Time-Restore, 490
 port scanning, 49–50
 PostgreSQL, 9, 81, 255, 490
 Amazon Aurora, 259
 Read replicas, 266–267
 power systems, 46
 primary instance, 256
 primary network interface (eth0), 163
 private content, 187
 private IP addresses, 120, 155
 private networks, 3
 private subnets, 487
 protocol detection, 188
 provisioned IOPS, 217, 259
 provisioned throughput, 288–290
 provisioning infrastructure, 314–315
 ProxySQL, 275
 public IP addresses, 114, 120
 publishers, 450
 pull event model, 129
 Puppet, 317
 push event model, 129
 push messaging, 451
 Push-Button Compute Scaling, 258
 Python, 24
 Boto SDK, 31–33

Q

--query option, 29, 483
 query string parameters, 190
 Query tool, 29, 483
 queue depth, 259
 queueing chain pattern, 445, 446

R

RADIUS. *See* Remote Authentication Dial In User Service

RDBMS. *See* Relational Database Management System

RDP. *See* Remote Desktop Protocol

RDS. *See* Amazon Relational Database Service

- read capacity units, 288–289
- read consistency, 290
- read replicas, 266–267, 275, 490, 491
- Real-Time Messaging Protocol (RTMP), 186
- Redis, 88, 297
- redundant active-active AWS Direct Connect connections, 465–466
- redundant active-active VPN connections, 463–464
- reference architecture
 - serverless design, 14–18
 - three-tier design, 5–14
- Regional Edge Caches, 187
- regions, 111. *See also* AWS Regions
 - high availability and, 457–463
 - replication across, 229, 457, 496
- Relational Database Management System (RDBMS), 252
- relational databases. *See also specific databases*
 - ACID, 252–253
 - scaling, 273–276
 - sharding, 276
- Remote Authentication Dial In User Service (RADIUS), 94–95
- Remote Desktop Protocol (RDP), 67
 - security groups and, 117
- replacement upgrade, 319, 321
- replication
 - Amazon Aurora, 257
 - Amazon RDS, 265
 - cross-region, 229, 457, 496
- repositories
 - Amazon ECR, 327
 - cookbook, 329–330
 - Docker, 327
 - Git, 320, 329

request headers

- adding or modifying forwarded, 191
- forwarding, 190–191
- request tracing, 176, 177
- reserved capacity, Amazon DynamoDB, 290
- resource administration, 418
- Resource APIs, 32, 33
- restore
 - Amazon RDS, 268–270, 490
 - in disaster recovery, 467–468, 472
 - Point-In-Time, 490
 - Snapshot, 490
- roles, 91–92. *See also* IAM Roles
- rolling deployments, 321
- route tables, 11, 71, 157
- routing, 180, 487. *See also* Amazon Route 53
 - DNS record types, 183–184
 - failover, 182–183
 - geolocation, 182
 - latency-based, 182, 182
 - simple, 181
 - weighted, 181, 181, 471
- routing policies, 181
- RTMP. *See* Real-Time Messaging Protocol
- RTMP distributions, 193
- rules, 397
 - AWS Config, 419–420

S

S3. *See* Amazon Simple Storage Service

Salt, 317

SAML. *See* Security Assertion Markup Language

scaling. *See also* Auto Scaling

- deployment and, 318
- horizontal, 273, 275
- vertical, 273–275

scaling diagonally, 273

scaling out, 273, 275

scaling up, 273–274

schemas, 252

SDKs. *See* Software Development Kits

secret key, 25

secure access points, 48

- Secure Shell (SSH)
 - Amazon EC2 and, 67
 - AMI host certificates for, 49
 - key pairs for, 59
 - security groups and, 117
- Secure Sockets Layer (SSL), 43, 49
 - Amazon Aurora, 258
 - Amazon Redshift, 86–87, 294
 - Classic Load Balancers and, 173
 - custom certificates, 188–189
- securing systems, 3–4
- security, 66, 483–485
 - Amazon Aurora, 258
 - Amazon CloudFront, 74–75, 194
 - Amazon DynamoDB, 80–81
 - Amazon EBS, 68–69, 220
 - Amazon EC2, 65–69, 122–123
 - Amazon EFS, 223
 - Amazon Glacier, 78–79, 231
 - Amazon Lightsail, 133
 - Amazon RDS, 81–84
 - Amazon Redshift, 84–87
 - Amazon Redshift clusters, 293–294
 - Amazon S3, 75–78, 228–229
 - Amazon VPC, 70–74
 - Amazon WorkSpaces, 94–95
 - analytics services, 89–91
 - application services, 88–89
 - AWS applications, 94–95
 - AWS Batch, 135
 - AWS Config and analysis of, 418
 - AWS Direct Connect, 170
 - AWS Elastic Beanstalk, 127
 - AWS Lambda, 130
 - AWS responsibilities, 43, 44
 - AWS Snowball, 79
 - AWS Storage Gateway, 80
 - Classic Load Balancers, 173
 - compute services, 65–69
 - customer responsibilities, 44
 - databases, 80–88
 - deployment and management services, 91–92
 - ECS, 125
 - Elastic Load Balancing, 69–70
 - IAM, 91–92
 - instance store, 221
 - load balancing, 178
- mobile services, 92–94
- monitoring, 62–65
- MySQL, 270–271
- networking, 69–75
- serverless architecture, 15–17
- service-specific, 65–95
- shared responsibility model, 43, 45, 483
- standard compliance, 50–51
- storage services, 75–79
- three-tier architecture, 10
- Security Assertion Markup Language (SAML), 91–92
- security groups, 3, 8, 12, 71, 482, 485
 - Amazon Aurora, 258
 - Amazon EC2, 271
 - Amazon RDS, 271
 - Amazon Redshift, 293–294
 - Amazon VPC, 271
 - cache, 87
 - DB, 81
 - EC2 configuration, 117
 - MySQL, 270
 - VPC, 158, 158–160
- serverless architectures, 14
 - key products, 17–18
 - networking, 15
 - security and authentication, 15–17
- Service Health Dashboard, 48
- Service Level Agreement (SLA), S3, 225
- services. *See also specific services*
 - container, 14
 - deployment, 315, 322–345
 - security of
 - analytics, 89–91
 - application, 88–89
 - compute, 65–69
 - deployment and management, 91–92
 - mobile, 92–94
 - storage, 75–79
 - session state management, 455
 - sharding, 276
 - shared AMIs, 112
 - shared queues, 449
 - shared responsibility model, 43, 45, 483
 - shared tenancy, 115
 - short polling, 447, 447
 - shutdown behavior, 115
- Simple Message Service (SMS), 56

- simple routing, 181
Single Root I/O Virtualization (SR-IOV), 113
skeletons, generating, 345
SLA. *See* Service Level Agreement
SMS. *See* Simple Message Service
Snapshot Restore, 490
snapshots, 489, 490
 Amazon EBS, 218–219
 Amazon EC2, 118
 AMI creation from, 219
 AWS Storage Gateway for, 468
 database, 83
 RDS copy with, 269
 sharing, 219
Software Development Kits (SDKs), 24, 30, 483, 486, 494
 Boto, 31–33
 IoT, 33–34
 mobile, 33–34
software VPN, 178, 179
Spot Pricing, 114
SQL, 251. *See also* Microsoft SQL Server; MySQL; PostgreSQL
SQL Server, 83, 255
SR-IOV. *See* Single Root I/O Virtualization
SSDs, 214, 259, 261
SSE. *See* Amazon S3 Server Side Encryption
SSH. *See* Secure Shell
SSH authentication, 15
SSL. *See* Secure Sockets Layer
SSL/TLS, 44
stacks
 AWS OpsWorks, 14, 320, 328–330, 493
 creating, 331
 deleting, 331
 updating, 332
stale reads, 283
standard queues, 448
standby instances, 491
static websites, 17, 482
statistics, 373–374
 aggregation, 375–376
 percentiles, 376–377
 periods, 374–375
 units, 374
status checks
 Amazon EBS, 390–391
Amazon EC2, 378–379, 387–388
 instance, 379, 387–388
 system, 378–379, 387
storage, 488–489. *See also* Amazon Simple Storage Service
 Amazon RDS, 259–260
 Amazon RDS pricing and, 282
 Amazon Redshift backups, 296
 auto-repair, 257
 of backups, 268–269, 282
 block, 209–223
 block vs. object, 209–210
 cost efficiency, 211–212
 EC2 instance types, 113
 instance, 115–116, 489
 object, 9, 209–212, 224–232
 retrieval times, 211
storage device decommissioning, 47
storage products, 7, 9
storage services, security, 75–79
storage-cached volumes, 468
store-backed EC2 instances, 116–118, 117
stores, 254. *See also* Amazon Elastic Block Store
stream record, 285
streaming, 193
streams, 285
subnets, 7, 11, 71
 Amazon VPC, 156, 159
 Availability Zones and, 154
 private, 487
subscribers, 450
system status checks, 378–379, 387
systems
 deploying, 2
 fortifying, 3
 monitoring, 2
 optimizing, 3
 securing, 3–4
systems operations, 24–34, 482
systems operator, 2
-
- T**
- tab completion, 27
tables, 284
route, 11, 71, 157

tag sets, 277
tags
 in deployment services, 315
 EC2 configuration, 116
 log groups and, 407–409
 metadata, 291
 RDS resources, 276–277
tape interface, 235
targets
 Amazon CloudWatch Events, 397
 geo, 223
 mount, 223
task definition, 327
tasks, 327–328
TCO. *See* Total Cost of Ownership
TCP load balancing, 70
TDE. *See* Transparent Data Encryption
termination protection, 115
three-tier design, 482
 compute methods, 8
 database, 9
 environment, 7
 highly available, 452, 453
 key products, 11–14
 networking, 7–8
 reference architecture, 5–14
 security, 10
thresholds, in Amazon CloudWatch Alarms, 384–386
throughput
 of Amazon EBS, 215–216
 Amazon EBS monitoring and, 220
 bursting, 217
 calculating, 216
 provisioned, 288–290
Time-based One Time Password (TOTP), 56
time-to-live (TTL), 189
TLS. *See* Transport Layer Security
topic policies, 416–417
topics, 451
Total Cost of Ownership (TCO), 121
TOTP. *See* Time-based One Time Password
trails
 creating, 413
 defining, 411
 naming requirements, 414
 types of, 411–412
transactions, 253

transmission protection, 49
Transparent Data Encryption (TDE), 83
Transport Layer Security (TLS), 43, 69–70
 between CloudFront and origin web server, 191
TTL. *See* time-to-live

U

user activity logging, 44
User Agent headers, 187
user data, EC2 configuration and, 115
user management, 10

V

Vault Lock, 231, 489
versioning, 489
 Amazon S3, 228
 applications, 324–325
 object, 191–192
vertical scaling, 273, 274–275
VGW. *See* Virtual Private Gateway
viewer connection protocol, 188
Virtual Interfaces (VIFs), 167, 169, 487
Virtual Local Area Network (VLAN), 167
virtual MFA applications, 56
virtual networks, 154
virtual private cloud (VPC), 482, 497. *See also* Amazon Virtual Private Cloud
Virtual Private Gateway (VGW), 73, 178, 179, 463, 485, 497
Virtual Private Network (VPN), 49, 153, 497
 as backup connection, 466
 in highly available connectivity, 463–464
 installation, 178–179
 management, 179
 redundant active-active, 463–464
Virtual Private Server (VPS), 130
Virtual Tape Library (VTL), 468
visibility timeout, 448, 497
VLAN. *See* Virtual Local Area Network
Vogels, Werner, 253
volume interface, 235

VPC. *See* Amazon Virtual Private Cloud;
virtual private cloud
VPC endpoints, 161
Amazon DynamoDB, 292
VPC Flow Logs, 10, 13, 62–63, 151, 165,
482, 484
VPC peering connection, 162, 487
VPN. *See* Virtual Private Network
VPN gateways, 160–161, 161
VPS. *See* Virtual Private Server
VTL. *See* Virtual Tape Library

W

waiters, 32
warm-standby, 470, 472, 497
web identity federation, 13
web server tier, 324
Web Tier, 5–6

WebSockets, 175
Weighted Round Robin (WRR), 322
weighted routing, 181, 181, 471
wide column stores, 254
worker tier, 324
write capacity units, 289–290
write consistency, 290–291
WRR. *See* Weighted Round Robin

X

X.509 certificates, 59–60
Xen hypervisor, 65, 66

Z

zone apex, 192

Comprehensive Online Learning Environment

Register on Sybex.com to gain access to the comprehensive online interactive learning environment and test bank to help you study for your AWS Certified SysOps Administrator - Associate exam.

The online test bank includes:

- **Assessment Test** to help you focus your study to specific objectives
- **Chapter Tests** to reinforce what you've learned
- **Practice Exams** to test your knowledge of the material
- **Digital Flashcards** to reinforce your learning and provide last-minute test prep before the exam
- **Searchable Glossary** to define the key terms you'll need to know for the exam

Go to <http://www.wiley.com/go/sybextestprep> to register and gain access to this comprehensive study tool package.

WILEY END USER LICENSE AGREEMENT

Go to www.wiley.com/go/eula to access Wiley's ebook EULA.