



training and
certification

AWS Academy Cloud Architecting
Module 01 Student Guide

Version 2.0.13

200-ACACAD-20-EN-SG

© 2023, Amazon Web Services, Inc. or its affiliates. All rights reserved.

This work may not be reproduced or redistributed, in whole or in part,
without prior written permission from Amazon Web Services, Inc.
Commercial copying, lending, or selling is prohibited.

All trademarks are the property of their owners.

Contents

Module 1: Welcome to AWS Academy Cloud Architecting

4



Module 1: Welcome to AWS Academy Cloud Architecting

AWS Academy Cloud Architecting

© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

Module 1: Welcome to AWS Academy Cloud Architecting.

Module overview

Sections

1. Course objectives and overview
2. Café business case introduction
3. Roles in cloud computing
4. What's new at AWS



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

2

This module includes the following sections:

1. Course objectives and overview
2. Café business case introduction
3. Roles in cloud computing
4. What's new at AWS

Module objectives

At the end of this module, you should be able to:

- Identify course prerequisites and objectives
- Recognize the café business case
- Indicate the role of cloud architects
- Explore new technologies at AWS



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

3

At the end of this module, you should be able to:

- Identify course prerequisites and objectives
- Recognize the café business case
- Indicate the role of cloud architects
- Explore new technologies at AWS

Section 1: Course objectives and overview

Module 1: Welcome to AWS Academy Cloud Architecting



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

Introducing Section 1: Course objectives and overview.

Course prerequisites

- Completion of the [AWS Academy Cloud Foundations](#) course
- Or similar level of knowledge of Amazon Web Services (AWS), such as –
 - Passed the AWS Certified Cloud Practitioner certification exam *or*,
 - Completed the AWS Cloud Practitioner Essentials course *or*,
 - Completed the AWS Technical Essentials course
- Additionally, it is assumed that you have –
 - General IT technical knowledge
 - General IT business knowledge



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

5

To begin, it is important to have an understanding of the prerequisites for this course.

The curriculum in this course assumes that you previously attended the *AWS Academy Cloud Foundations* course. Alternatively, you might have a level of familiarity with the AWS Cloud that is similar to what the AWS Academy Cloud Foundations course provides.

It is also assumed that you have general *IT technical knowledge*. You must have some foundational computer literacy skills to be successful in this course. These skills include a knowledge of basic computer concepts, file management, and a good understanding of the internet.

You should also have general *IT business knowledge*, which includes insight into how businesses and other organizations use information technology.

Course objectives

After completing this course, you should be able to:

- Make architectural decisions based on AWS architectural principles and best practices
- Use AWS services to make your infrastructure scalable, reliable, and highly available
- Use AWS managed services to enable greater flexibility and resiliency in an infrastructure
- Indicate how to increase the performance efficiency and reduce costs of infrastructures built on AWS
- Use the AWS Well-Architected Framework to improve architectures that use AWS solutions



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

6

After completing this course, you should be able to:

- Make architectural decisions based on AWS architectural principles and best practices
- Use AWS services to make your infrastructure scalable, reliable, and highly available
- Use AWS managed services to enable greater flexibility and resiliency in an infrastructure
- Indicate how to increase the performance efficiency and reduce costs of infrastructures built on AWS
- Use the AWS Well-Architected Framework to improve architectures that use AWS solutions

Course outline

- Module 1 – Welcome to AWS Academy Cloud Architecting (this module)
- Module 2 – Introducing Cloud Architecting
- Module 3 – Adding a Storage Layer
- Module 4 – Adding a Compute Layer
- Module 5 – Adding a Database Layer
- Module 6 – Creating a Networking Environment
- Module 7 – Connecting Networks
- Module 8 – Securing User and Application Access
- Module 9 – Implementing Elasticity, High Availability, and Monitoring
- Module 10 – Automating Your Architecture
- Module 11 – Caching Content
- Module 12 – Building Decoupled Architectures
- Module 13 – Building Microservices and Serverless Architectures
- Module 14 – Planning for Disaster
- Module 15 – Bridging to Certification



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

7

To achieve the course objectives, the course consists of the 15 modules that are shown. The next slides provide more detail on what subtopics are covered in each module.

Module 2: Introducing Cloud Architecting

Module sections:

1. What is cloud architecting?
2. The Amazon Web Services (AWS) Well-Architected Framework
3. Best practices for building solutions on AWS
4. AWS global infrastructure



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

8

Module 2 introduces cloud architecting concepts. The sections in Module 2 include:

- What is cloud architecting?
- The Amazon Web Services (AWS) Well-Architected Framework
- Best practices for building solutions on AWS
- AWS global infrastructure

Module 3: Adding a Storage Layer

Module sections:

1. The simplest architecture
2. Using Amazon S3
3. Storing data in Amazon S3
4. Moving data to and from Amazon S3
5. Choosing Regions for your architecture



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

9

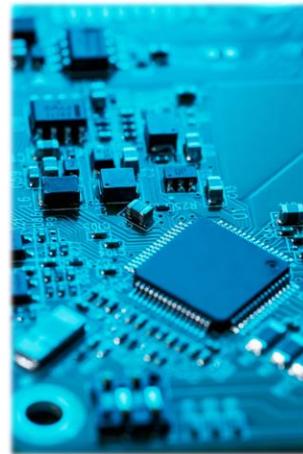
Module 3 focuses on storage and Amazon S3 in particular. The sections in Module 3 include:

- The simplest architecture
- Using Amazon S3
- Storing data in Amazon S3
- Moving data to and from Amazon S3
- Choosing Regions for your architecture

Module 4: Adding a Compute Layer

Module sections:

1. Architectural need
2. Adding compute with Amazon EC2
3. Choosing an Amazon Machine Image (AMI) to launch an Amazon Elastic Compute Cloud (Amazon EC2) instance
4. Selecting an Amazon EC2 instance type
5. Using user data to configure an Amazon EC2 instance
6. Adding storage to an Amazon EC2 instance
7. Amazon EC2 pricing options
8. Amazon EC2 considerations



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

10

Module 4 focuses on the compute layer. The sections in Module 4 include:

- Architectural need
- Adding compute with Amazon Elastic Compute Cloud (Amazon EC2)
- Choosing an Amazon Machine Image (AMI) to launch an Amazon EC2 instance
- Selecting an Amazon EC2 instance type
- Using user data to configure an Amazon EC2 instance
- Adding storage to an Amazon EC2 instance
- Amazon EC2 pricing options
- Amazon EC2 considerations

Module 5: Adding a Database Layer

Module sections:

1. Architectural need
2. Database layer considerations
3. Amazon Relational Database Service (Amazon RDS)
4. Amazon DynamoDB
5. Database security controls
6. Migrating data into AWS databases



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

11

Module 5 focuses on the database layer. The sections in Module 5 include:

- Architectural need
- Database layer considerations
- Amazon Relational Database Service (Amazon RDS)
- Amazon DynamoDB
- Database security controls
- Migrating data into AWS databases

Module 6: Creating a Networking Environment

Module sections:

1. Architectural need
2. Creating an AWS networking environment
3. Connecting your AWS networking environment to the internet
4. Securing your AWS networking environment



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

12

Module 6 focuses on the network layer. The sections in Module 6 include:

- Architectural need
- Creating an AWS networking environment
- Connecting your AWS networking environment to the internet
- Securing your AWS networking environment

Module 7: Connecting Networks

Module sections:

1. Architectural need
2. Connecting to your remote network with AWS Site-to-Site VPN
3. Connecting to your remote network with AWS Direct Connect
4. Connecting virtual private clouds (VPCs) in AWS with VPC peering
5. Scaling your VPC network with AWS Transit Gateway
6. Connecting your VPC to supported AWS services



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

13

Module 7 focuses on connecting networks. The sections in Module 7 include:

- Architectural need
- Connecting to your remote network with AWS Site-to-Site VPN
- Connecting to your remote network with AWS Direct Connect
- Connecting virtual private clouds (VPCs) in AWS with VPC peering
- Scaling your VPC network with AWS Transit Gateway
- Connecting your VPC to supported AWS services

Module 8: Securing User and Application Access

Module sections:

1. Architectural need
2. Account users and AWS Identity and Access Management (IAM)
3. Organizing users
4. Federating users
5. Multiple accounts



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

14

Module 8 focuses on securing user and application access. The sections in Module 8 include:

- Architectural need
- Account users and AWS Identity and Access Management (IAM)
- Organizing users
- Federating users
- Multiple accounts

Module 9: Implementing Elasticity, High Availability, and Monitoring

Module sections:

1. Architectural need
2. Scaling your compute resources
3. Scaling your databases
4. Designing an environment that's highly available
5. Monitoring



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

15

Module 9 focuses on implementing elasticity, high availability, and monitoring. The sections in Module 9 include:

- Architectural need
- Scaling your compute resources
- Scaling your databases
- Designing an environment that's highly available
- Monitoring

Module 10: Automating Your Architecture

Module sections:

1. Architectural need
2. Reasons to automate
3. Automating your infrastructure
4. Automating deployments
5. AWS Elastic Beanstalk



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

16

Module 10 focuses on automating your architecture. The sections in Module 10 include:

- Architectural need
- Reasons to automate
- Automating your infrastructure
- Automating deployments
- AWS Elastic Beanstalk

Module 11: Caching Content

Module sections:

1. Architectural need
2. Overview of caching
3. Edge caching
4. Caching web sessions
5. Caching databases



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

17

Module 11 focuses on caching content. The sections in Module 11 include:

- Architectural need
- Overview of caching
- Edge caching
- Caching web sessions
- Caching databases

Module 12: Building Decoupled Architectures

Module sections:

1. Architectural need
2. Decoupling your architecture
3. Decoupling with Amazon Simple Queue Service (Amazon SQS)
4. Decoupling with Amazon Simple Notification Service (Amazon SNS)
5. Sending messages between cloud applications and on-premises with Amazon MQ



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

18

Module 12 focuses on building decoupled architectures. The sections in Module 12 include:

- Architectural need
- Decoupling your architecture
- Decoupling with Amazon Simple Queue Service (Amazon SQS)
- Decoupling with Amazon Simple Notification Service (Amazon SNS)
- Sending messages between cloud applications and on-premises with Amazon MQ

Module 13: Building Microservices and Serverless Architectures

Module sections:

1. Architectural need
2. Introducing microservices
3. Building microservice applications with AWS container services
4. Introducing serverless architectures
5. Building serverless architectures with AWS Lambda
6. Extending serverless architectures with Amazon API Gateway
7. Orchestrating microservices with AWS Step Functions



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

19

Module 13 focuses on building microservices and serverless architectures. The sections in Module 13 include:

- Architectural need
- Introducing microservices
- Building microservice applications with AWS container services
- Introducing serverless architectures
- Building serverless architectures with AWS Lambda
- Extending serverless architectures with Amazon API Gateway
- Orchestrating microservices with AWS Step Functions

Module 14: Planning for Disaster

Module sections:

1. Architectural need
2. Disaster planning strategies
3. Disaster recovery patterns



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

20

Module 14 focuses on disaster planning. The sections in Module 14 include:

- Architectural need
- Disaster planning strategies
- Disaster recovery patterns

Module 15: Bridging to Certification

Module sections:

1. Certification exam resources
2. Additional resources



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

21

Module 15 focuses on bridging your learning in this course to attaining an AWS certification. The sections in Module 15 include:

- Certification exam resources
- Additional resources

Section 2: Café business case introduction

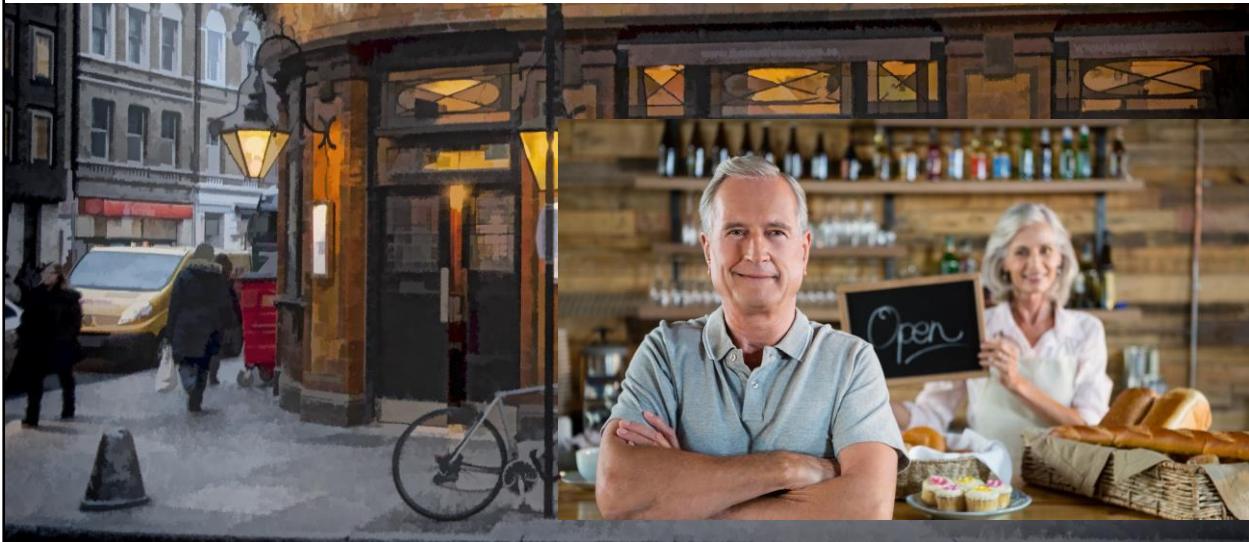
Module 1: Welcome to AWS Academy Cloud Architecting



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

Introducing Section 2: Café business case introduction.

Café and bakery



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

23

The challenge labs in this course are built around a fictional business case. The business case provides a way to explore cloud-computing topics in the context of relatable business needs. This scenario is intended to provide an example of the real-world applicability of technical concepts that you will learn.

Frank and Martha opened a café and bakery that had its start from a retirement dream. Frank and Martha were not yet ready to live out their retirement around the house. Instead, they wanted to do something that included their love of baking and supplemented their income. They enjoy interacting with the people in their neighborhood. They also like to support community events across town with their baked goods and coffees.

To make their dreams a reality, Frank and Martha recently decided to open their café and bakery at the base of their flat. Their daughter, Sofía, and a local school student, Nikhil, help out and work at the café. Since they opened the café, they have experienced an increase in local business. They also sometimes receive inquiries from people who travel through the area either for business or as tourists.

The café owners and staff

Frank

- Co-owner of café
- Retired from Navy
- Likes to bake
- Non-technical



Martha

- Co-owner of café
- Retired accountant
- Knows how to use spreadsheets, otherwise non-technical



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

Sofía

- Daughter of Frank and Martha
- Manages the café's supply chain
- Technical skills, including programming, future business administration student
- Started to use AWS



Nikhil

- Café employee, visual design skills
- Interested in learning cloud computing
- Might take on more responsibilities at the café when Sofía starts her studies at the university



24

Sofía is the daughter of Frank and Martha. She runs the café, which includes managing the supply chain for sourcing ingredients and tracking inventory. She took some programming classes in secondary school and plans to start working toward a university degree in business administration later this year.

Sofía just learned about Amazon Web Services. She has talked to her parents about how they could use AWS services to automate some aspects of the café business, reduce manual administrative work, and improve the customer experience.

Nikhil works at the café part time. He is going to finish secondary school in the spring. He works behind the counter, serving customers and doing other tasks under Sofía's supervision. He has some experience with visual design, and wants to learn more about web development and cloud computing. He plans to get a university degree that will build his existing design skills, and also enable him to learn cloud computing skills.

AWS consultants, café visitors

Olivia

- An AWS solutions architect
- Technical, with a specialty in databases and network technologies



Mateo

- Systems administrator and engineer
- Likes to find ways to automate and to create repeatable solutions
- Knows the importance of backups and disaster recovery in solution design



Faythe

- Developer, experienced with AWS programming interfaces
- Knowledgeable about cloud security



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

25

Olivia is an AWS solutions architect (SA) who recently moved to the downtown area. She frequently visits the café and enjoys talking with Sofía. Through their conversations, Sofía has learned that Olivia is an expert in AWS and cloud technologies. Olivia used to be a network engineer, and she also has a strong background in database technologies.

Faythe is an AWS developer who recently completed an AWS internship program. She likes to use her programming skills to apply the appropriate technology to a business problem. She recently achieved the AWS Certified Security – Specialty and is interested in developing big data solutions. Faythe is friends with Olivia and Mateo, and she also frequently visits the café.

Mateo is an experienced AWS SysOps engineer. He is skilled at bringing automation and fault tolerance to the solutions that he builds. He also likes to design for backup and disaster recovery. Mateo previously worked as a developer, and has been mentoring Faythe since she started as an intern with AWS. Mateo is happy to help Faythe and anyone who is interested in learning. Mateo is friends with Nikhil, Sofía, Olivia, and Faythe, and he frequently grabs a coffee at the café on his way to work.

You will build solutions for the café in the challenge labs in this course

The café has business needs that can be solved with cloud computing architectures.



The café employees and the consultants often socialize and share cloud architecture ideas.

© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

26

In the hands-on labs in this course, you gain hands-on practice building on AWS. Two types of labs are available: guided labs and challenge labs.

Guided labs provide you with step-by-step instructions, to help you gain experience in creating and configuring AWS resources in the different AWS service areas. The guided labs do not mention the café business; however, the skills that you gain in these guided labs prepare you for the challenge labs.

The *challenge labs* present new business requirements that are based on the evolving needs of the café. These labs contain sections where the instructions do not provide full click-level guidance or detailed step-by-step instructions. Rather, you will be challenged to apply the skills that you gained from the guided labs and the concepts that are presented in the lectures.

In the challenge labs, you take on the role of Sofía or Nikhil. With the assistance of the AWS consultants who occasionally pass through the café offering advice, you will architect cloud solutions that help fulfill the business needs of the café.

Section 3: Roles in cloud computing

Module 1: Welcome to AWS Academy Cloud Architecting



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

Introducing Section 3: Roles in cloud computing.

Roles in computing: IT professional

IT professional



- Generalist, might manage an application
- Often manages a production environment
- Highly technical
- Might have significant or limited experience in cloud technologies
- Might specialize in one area (such as security or storage)

Job titles: IT Administrator, Systems Administrator, Network Administrator



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

28

In this section, you will learn about five common roles in cloud computing.

You might want to start your career in cloud computing, or to transition your career to a cloud computing role. Maybe you would like to work in an organization where some employees have cloud computing responsibilities. For any of these reasons, it is helpful to understand the common job titles or roles that individuals, teams, or departments perform.

IT professionals are generalists. They typically have a broad range of skills. For example, they might manage the infrastructure for an entire application and have a strong understanding of the components that make up the solution. However, they might not always have detailed knowledge of any one service that is part of the application. IT professionals are typically highly technical.

Common job titles include IT Administrator, Systems Administrator, or Network Administrator.

Roles in computing: IT leader

IT leader



- Leads a team of IT professionals
- Responsible for day-to-day operations
- Manages a budget, stays informed about and chooses new technologies
- Hands on during early stages of a project, then delegates the team to take over

Job titles: IT Manager, IT Director, IT Supervisor



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

29

IT leaders are managers. They typically lead a team of IT professionals, and decide on the type of technology that will be used for a project. They might be significantly involved in the implementation details early on in the project lifecycle. Then, they delegate the team to handle the details as the project gets closer to completion.

Typical job titles include IT Manager, IT Director, and IT Supervisor.

Roles in computing: Developer

Developer



- Writes, tests, and fixes code
- Thinks about projects at the application level
- Likes sample code
- Works with APIs, SDKs

Job titles: Software Developer, System Architect, Software Development Manager



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

30

Another common role in cloud computing is the developer role. *Developers* like to code. They work with the details—writing, testing, and fixing the code that makes an application work. Developers are likely to borrow ideas from sample code. They work with application programming interfaces (APIs) and software development kits (SDKs).

Common job titles include Software Developer, Systems Architect, or Software Development Manager.

Roles in computing: DevOps engineer

DevOps engineer



- Builds out the infrastructure that applications run on, often in the cloud
- Follow the guidelines of the cloud architect
- Prefer experimenting and trying things out rather than lots of reading

Job titles: DevOps Engineer, Reliability Engineer, Build Engineer



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

31

DevOps engineers spend their time building out the infrastructure that applications run on. They often create or improve on the code that installs and configures servers and application deployments. DevOps staff prefer to experiment and learn by doing. They create repeatable deployment solutions, and they work to apply engineering skills to the business needs of operations teams.

Common job titles include DevOps Engineer, Build Engineer, or Reliability Engineer.

Roles in computing: Cloud architect

Cloud architect



- Stays up-to-date with new technologies, helps decide which to use
- Provides documentation, processes, and tooling to developers
- Gives developers freedom to innovate
- Common challenges include –
 - Resource management
 - Cost optimization
 - Defining best practices for performance, reliability, and security

Job titles: Cloud Architect, Systems Engineer, Systems Analyst



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

32

The final cloud computing role that you will learn about in this section is the cloud architect role.

Cloud architects spend their time reading and staying up-to-date with the latest developments and trends in cloud computing. They are responsible for the design architecture of applications and selecting which technologies should be used to meet the needs of a technical business objective. They should be aware of the capabilities of the many cloud service options that are available. Thus, they can decide which ones should be adopted, given a specific set of business requirements. Cloud architects provide guidance to developers through architectural diagrams and documentation. They also provide tooling, but they give the development team room to innovate if they meet the success criteria.

Common challenges for the cloud architect role include resource management, cost optimization, and defining best practices for performance, reliability, and security.

The responsibilities of cloud architects closely align with the pillars of the AWS Well-Architected Framework, which is discussed in detail in this course.

AWS Certification exams

This course helps prepare you for the **AWS Certified Solutions Architect – Associate** exam

The final module in this course discusses AWS certification in more detail.

Available AWS Certifications

Professional
Two years of comprehensive experience designing, operating, and troubleshooting solutions using the AWS Cloud

Associate
One year of experience solving problems and implementing solutions using the AWS Cloud

Foundational
Six months of fundamental AWS Cloud and industry knowledge

Cloud Practitioner

Specialty
Technical AWS Cloud experience in the Specialty domain as specified in the exam guide

aws certified
Updated May 2019

© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

33

AWS Certification helps learners build credibility and confidence by validating their cloud expertise with an industry-recognized credential. It helps organizations identify skilled professionals who can lead cloud initiatives by using AWS.

You must earn a passing score on a proctored exam to earn an AWS Certification.

AWS Certification does not publish a list of all services or features that are covered in a certification exam. However, the exam guide for each exam lists the current topic areas and objectives that are covered in the exam. Exam guides can be found on the [Prepare for Your AWS Certification Exam](#) webpage.

You are required to update your certification (or recertify) every 3 years. View the [AWS Certification Recertification](#) page for more details.

The information on this slide is current as of June 2020. However, exams are frequently updated, and the details regarding which exams are available—and what is tested by each exam—are subject to change. For the latest AWS certification exam information, view the details on the [AWS Certification](#) webpage.

The final module in this course provides more information to supplement what you learn in this course. It will also guide you in how to apply this knowledge toward achieving the AWS Solutions Architect – Associate certification.



Section 4: What's new at AWS

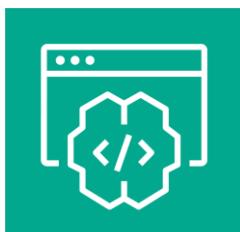
Module 1: Welcome to AWS Academy Cloud Architecting



©2023, Amazon Web Services, Inc. or its affiliates. All rights reserved.

The “What’s new at AWS” section introduces new and emerging technologies at AWS.

What is Amazon CodeWhisperer?



CodeWhisperer



AI-powered code generator for IDEs and code editors

- AI coding companion:

- Generates code suggestions based on comments and existing code
- Offers real-time support for code authoring directly within your integrated development environment (IDE)

- AI security scanner:

- Helps identify hard-to-find vulnerabilities
- References multiple standards and best practices

Content processed by CodeWhisperer Professional is not stored or used for service improvement.

©2023, Amazon Web Services, Inc. or its affiliates. All rights reserved.

35

Amazon CodeWhisperer analyzes your comments and code as you write them in your integrated development environment (IDE). It goes beyond code completion by using natural language processing to comprehend the comments in your code. By understanding English comments, CodeWhisperer generates complete functions and code blocks that align with your descriptions. CodeWhisperer also analyzes the surrounding code, ensuring the generated code matches your style and naming conventions and seamlessly integrates into the existing context.

When scanning for security vulnerabilities, CodeWhisperer assesses your code against multiple sets of standards and best practices. This includes the following:

- Open Worldwide Application Security Project (OWASP) standards
- Crypto library best practices
- AWS security standards

The security scan feature is continuously updated to help keep applications free from new security vulnerabilities.

Compatibility: CodeWhisperer integrates with popular tools such as Visual Studio Code, JetBrains IDEs (IntelliJ IDEA, PyCharm, etc.), Amazon SageMaker Studio, JupyterLab, AWS Cloud9, and AWS Lambda console.

Support: CodeWhisperer supports a wide range of programming languages and development environments, including Python, Java, JavaScript, TypeScript, C#, Go, Rust, PHP, Ruby, Kotlin, C, C++,

shell scripting, structured query language (SQL), and Scala.

Installation: You can access CodeWhisperer by downloading and installing the AWS Toolkit IDE extension or plugin. You can also activate CodeWhisperer from directly within the AWS Lambda and AWS Cloud9 console code editors.

Installation instructions vary depending on the environment. For more information, see “Getting started” in the *CodeWhisperer User Guide* at <https://docs.aws.amazon.com/codewhisperer/latest/userguide/getting-started.html>.

Code generation

- Code suggestions
- Code completion
- Code generation from comments
- Alternate code suggestions
- Option to accept or reject
- Reference tracking for code that resembles open-source training data

```
1 import boto3
```

```
1 import boto3
2 # create an s3 bucket named cw95323
```

```
2 # create an s3 bucket named cw95323
s3 = boto3.resource('s3')
s3.create_bucket(Bucket='cw95323')
# upload a file to the bucket
```



©2023, Amazon Web Services, Inc. or its affiliates. All rights reserved.

36

The code generation feature of CodeWhisperer offers code suggestions in real time in your development environment. It automatically offers code completion and code generation suggestions. It uses natural language processing of English comments in your code and an understanding of surrounding code to suggest whole lines of code, complete functions, and logical blocks of code. The generated code is aligned with your coding style and naming conventions. CodeWhisperer prioritizes secure coding and responsible artificial intelligence (responsible AI) practices. It's optimized for Amazon APIs and trained extensively on Amazon and open-source code. You have the option to accept the first suggestion, explore more suggestions, or continue writing your own code. It's important to review each code suggestion before accepting it because you might need to make edits to ensure that the suggestion aligns with your intended functionality.

User actions

- Previous and next suggestion: Use the left arrow and right arrow.
- Accept a suggestion: Press Tab.
- Reject a suggestion: Press Esc.
- Manually start code generation when typing a comment: On MacOS, press Option+C, and on Windows, press Alt+C.

Open Code Reference Log

CodeWhisperer learns from open-source projects and the code it suggests might occasionally resemble code samples from the training data. With the reference log, you can view references to code suggestions that are similar to the training data. When such occurrences happen, CodeWhisperer notifies you and provides repository and licensing information. Use this information to make decisions about whether to use the code in your project and properly attribute the source code as desired.

Security scan

```
1 import boto3
2
3 def upload_file(bucket_name, file_path):
4     s3 = boto3.client('s3')
5     with open(file_path, 'rb') as file:
6         s3.upload_fileobj(file, bucket_name, file_path)
7         print("File uploaded.")
8
9 bucket_name = input("Enter bucket name: ")
10 file_path = input("Enter file path to upload: ")
11 upload_file(bucket_name, file_path)
```

PROBLEMS



CWE-22 – Path traversal: Constructing path names with unsanitized user input can lead to...



©2023, Amazon Web Services, Inc. or its affiliates. All rights reserved.

37

The security scanning feature of CodeWhisperer detects security vulnerabilities in both CodeWhisperer-generated code and developer-written code. It scans the code to identify potential vulnerabilities and provides suggestions for remediation. This includes scanning for hard-to-find vulnerabilities that might be overlooked. The security scan is compatible with popular IDEs such as VS Code and JetBrains. It supports Python, Java, and JavaScript.

Benefits of Amazon CodeWhisperer



Value to developers

- Increase velocity.
- Spend less time writing code.
- Receive help directly within your IDE.
- Find security vulnerabilities in your code.

Value to organizations

- Use at all experience levels.
- Support open-source attribution.
- Reduce the risk of security vulnerabilities.
- Increase code quality and developer productivity.



©2023, Amazon Web Services, Inc. or its affiliates. All rights reserved.

38

Automated code generation automates repetitive tasks and saves you time. It eliminates the need for you to invest excessive hours in exploring and learning new technologies. Instead, you can rely on high-quality code suggestions that match your coding style. This approach enhances your productivity so you can focus on critical tasks, which encourages innovation and progress in software development. With automated code generation, you can streamline your workflows and achieve significant time savings while ensuring the delivery of code that meets your standards.

CodeWhisperer code generation offers many benefits for software development organizations. It accelerates application development for faster delivery of software solutions. By automating repetitive tasks, it optimizes the use of developer time, so developers can focus on more critical aspects of the project. Additionally, code generation helps mitigate security vulnerabilities, safeguarding the integrity of the codebase.

CodeWhisperer also protects open source intellectual property by providing the open source reference tracker. CodeWhisperer enhances code quality and reliability, leading to robust and efficient applications. And it supports an efficient response to evolving software threats, keeping the codebase up to date with the latest security practices. CodeWhisperer has the potential to increase development speed, security, and the quality of software.

Resources:

- Visit “Getting started” at <https://aws.amazon.com/codewhisperer/resources/>.
- Dive deep with the “Amazon CodeWhisperer – Getting Started” course on AWS Skill Builder at <https://explore.skillbuilder.aws/learn/course/external/view/elearning/16405/amazon-codewhisperer-getting-started>.
- Learn how to build an event-driven serverless app at <https://catalog.us-east-1.prod.workshops.aws/workshops/a33a5d69-1417-4d5f-acc9-ae5c7fba665b/en-US/>.

Module wrap-up

Module 1: Welcome to AWS Academy Cloud Architecting



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

It's now time to review the module.

Module summary

In summary, in this module, you learned how to:

- Identify course prerequisites and objectives
- Recognize the café business case
- Indicate the role of cloud architects
- Explore new technologies at AWS



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

40

In summary, in this module, you learned how to:

- Identify course prerequisites and objectives
- Recognize the café business case
- Indicate the role of cloud architects
- Explore new technologies at AWS

Thank you

Corrections, feedback, or other questions?
Contact us at <https://support.aws.amazon.com/#/contacts/aws-academy>.
All trademarks are the property of their owners.

© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

41

Thank you for completing this module.



training and
certification

AWS Academy Cloud Architecting
Module 02 Student Guide

Version 2.0.12

200-ACACAD-20-EN-SG

© 2023, Amazon Web Services, Inc. or its affiliates. All rights reserved.

This work may not be reproduced or redistributed, in whole or in part,
without prior written permission from Amazon Web Services, Inc.
Commercial copying, lending, or selling is prohibited.

All trademarks are the property of their owners.

Contents

Module 2: Introducing Cloud Architecting

4



Module 2: Introducing Cloud Architecting

AWS Academy Cloud Architecting

© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

Welcome to Module 2: Introducing Cloud Architecting.

Module overview

Sections

1. What is cloud architecting?
2. The Amazon Web Services (AWS) Well-Architected Framework
3. Best practices for building solutions on AWS
4. AWS global infrastructure



Knowledge check



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

2

This module includes the following sections:

1. What is cloud architecting?
2. The Amazon Web Services (AWS) Well-Architected Framework
3. Best practices for building solutions on AWS
4. AWS global infrastructure

At the end of this module, you will be asked to complete a knowledge check that will test your understanding of key concepts covered in this module.

Module objectives

At the end of this module, you should be able to:

- Define cloud architecture
- Describe how to design and evaluate architectures using the AWS Well-Architected Framework
- Explain best practices for building solutions on AWS
- Describe how to make informed decisions on where to place AWS resources



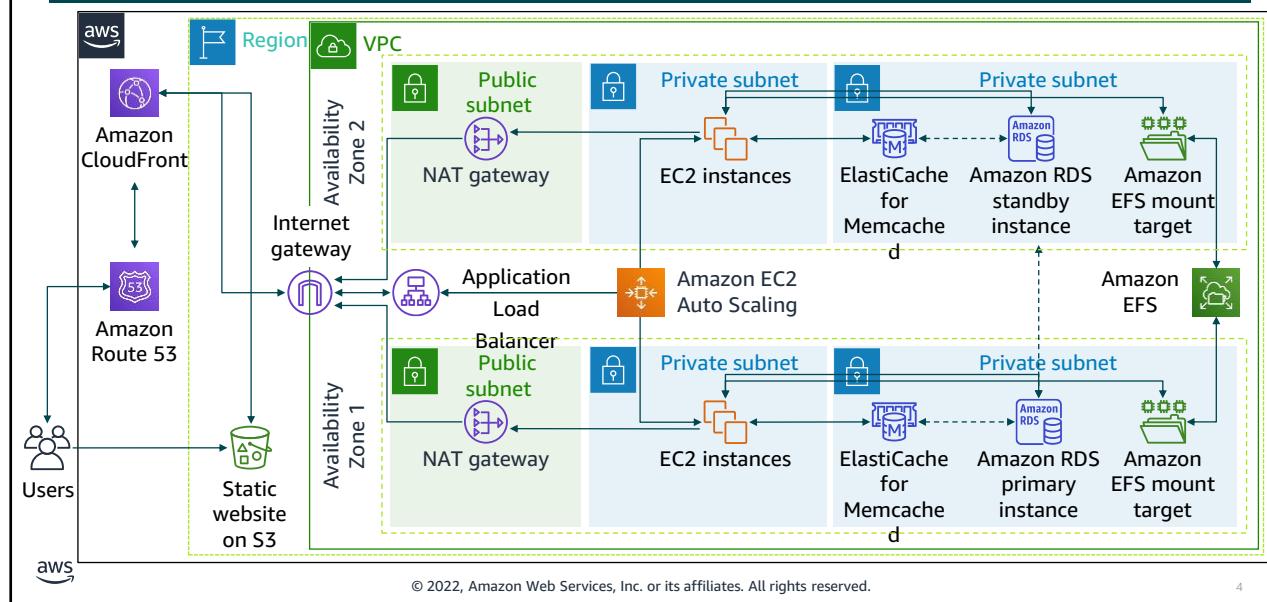
© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

3

At the end of this module, you should be able to:

- Define cloud architecture
- Describe how to design and evaluate architectures using the AWS Well-Architected Framework
- Explain best practices for building solutions on AWS
- Describe how to make informed decisions on where to place AWS resources

A large architecture



By the end of this course, you will have learned about all the components in this architecture diagram. You should also be able to construct your own solution architectures that are as large and robust as this example. You will see this diagram repeated at the start of most modules in the course. New components in the diagram will be revealed as they are introduced in the course.

Section 1: What is cloud architecting?

Module 2: Introducing Cloud Architecting



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

Introducing Section 1: What is cloud architecting?

Architectural need

Around 2000, Amazon was struggling to make its new shopping website highly available and scalable.



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

6

To understand what cloud architecting is and why it's important, first consider an example of what software development is like in its absence.

Around 2000, Amazon was trying to create an ecommerce service that would enable third-party sellers to build their own online shopping sites on top of the Amazon ecommerce engine. The company was struggling to make its new shopping website highly available and scalable.

Origins of AWS

- According to AWS CEO Andy Jassy, at the time, Amazon ecommerce tools were “a jumbled mess”
 - Applications and architectures were **built without proper planning**
 - It was **difficult to separate services** from each other
- Solution: Amazon created a set of well-documented APIs, which became the company’s standard for service development



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

7

In a [TechCrunch interview about the genesis of AWS](#), AWS Chief Executive Officer (CEO) Andy Jassy said that in the beginning, the Amazon ecommerce tools were a “jumbled mess.” Applications and architectures were being built without proper planning. Jassy also said that it was “a huge challenge to separate the various services to make a centralized development platform.”

The solution to this problem was to create a set of well-documented application programming interfaces (APIs) to organize the development environment.

Problems persisted

- Amazon still struggled to build applications quickly.
 - Database, compute, and storage components took **3 months** to build.
 - Each team built their own resources, **with no planning for scalability or reusability**.
- Solution: Amazon built internal services to create highly available, scalable, and reliable architectures on top of its infrastructure. In 2006, Amazon started selling these services as AWS.



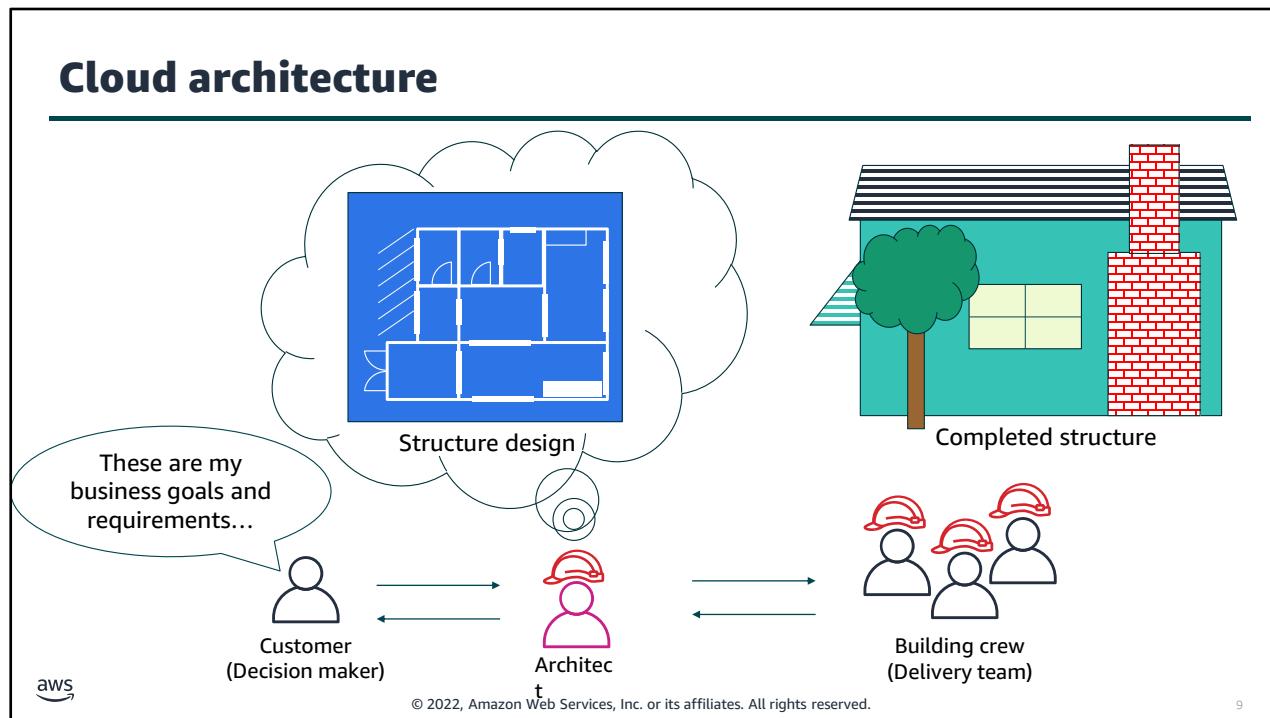
© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

8

However, Amazon still struggled to build applications quickly as the company grew and more software engineers were hired:

- It took 3 months to build database, compute, and storage components for an entire project that was expected to take 3 months.
- Each team built their own resources without planning for scalability or reusability.

The solution was to build internal services to create highly available, scalable, and reliable architectures on top of the Amazon infrastructure. In 2006, Amazon started selling these services as AWS.



So, what is cloud architecture? Cloud architecture is the practice of applying cloud characteristics to a solution that uses cloud services and features to meet an organization's technical needs and business use cases. A solution is similar to a blueprint for a building.

Software systems require architects to manage their size and complexity.

Cloud architects:

- Engage with decision makers to identify the business goals and the capabilities that need improvement.
- Ensure alignment between technology deliverables of a solution and the business goals.
- Work with delivery teams that are implementing the solution to ensure that the technology features are appropriate.

Having well-architected systems increases the likelihood that the technology deliverables will help meet business goals.

Section 1 key takeaways



- Cloud architecture is the practice of applying cloud characteristics to a solution that uses cloud services and features to meet an organization's technical needs and business use cases
- You can use AWS services to create highly available, scalable, and reliable architectures

© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

10

Some key takeaways from this section of the module include:

- Cloud architecture is the practice of applying cloud characteristics to a solution that uses cloud services and features to meet an organization's technical needs and business use cases
- You can use AWS services to create highly available, scalable, and reliable architectures

Section 2: The AWS Well-Architected Framework

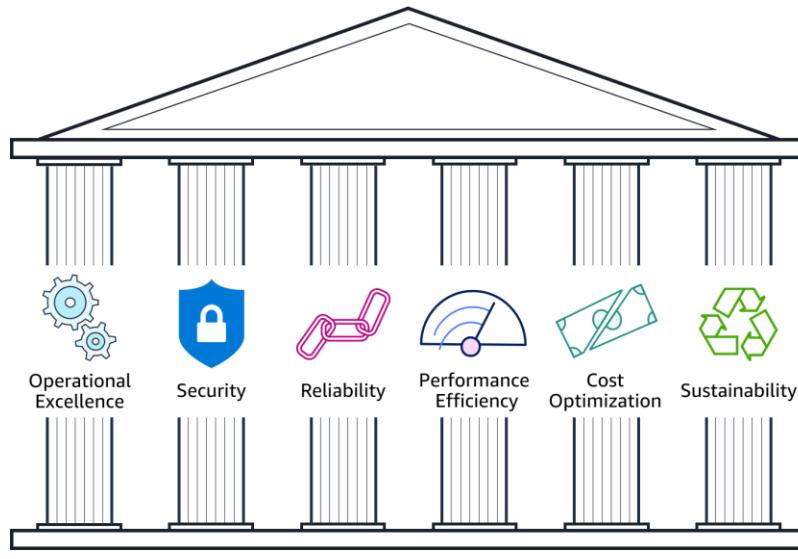
Module 2: Introducing Cloud Architecture



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

Introducing Section 2: The AWS Well-Architected Framework.

Pillars of the AWS Well-Architected Framework

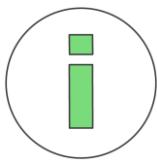


12

The AWS Well-Architected Framework is a guide that provides a consistent approach to evaluate cloud architectures, and guidance to help implement designs. It documents a set of foundational questions and best practices that enable you to understand if a specific architecture aligns well with cloud best practices. AWS developed this framework after reviewing thousands of customer architectures on AWS.

The AWS Well-Architected Framework is organized into six pillars: Operational Excellence, Security, Reliability, Performance Efficiency, Cost Optimization, and Sustainability. The first five pillars have been part of the framework since the framework's introduction in 2015. The sustainability pillar was added as the sixth pillar in 2021 to help organizations learn how to minimize the environmental impacts of running cloud workloads.

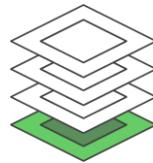
Security pillar



Identity foundation



Traceability



Security at all layers



Risk assessment and mitigation strategies



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

13

The Security pillar addresses the ability to protect information, systems, and assets while delivering business value through risk assessments and mitigation strategies.

Your architecture will present a much stronger security presence if you implement a strong identity foundation, enable traceability, apply security at all layers, automate security best practices, and protect data in transit and at rest.

For more information about security best practices, see the [Security Pillar whitepaper](#).

Operational Excellence pillar

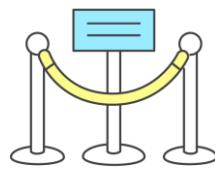
- The ability to run and monitor systems
- To continuously improve supporting process and procedures



Deployed



Updated



Operated



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

14

The Operational Excellence pillar addresses the ability to run systems and gain insight into their operations to deliver business value. It also addresses the ability to continuously improve supporting processes and procedures.

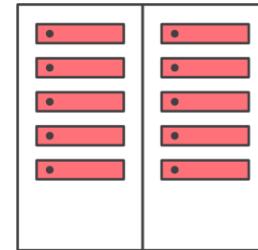
When you design a workload for operations, you must be aware of how it will be deployed, updated, and operated. Implement engineering practices that align with defect reductions and quick, safe fixes. Enable observation with logging, instrumentation, and business and technical metrics so that you can gain insight into what is happening inside your architecture.

In AWS, you can view your entire workload (applications, infrastructure, policy, governance, and operations) as code. It can all be defined in and updated using code. This means that you can apply the same engineering discipline that you use for application code to every element of your stack.

For more information about operational excellence best practices, see the [Operational Excellence Pillar whitepaper](#).

Reliability pillar

- Recover quickly from infrastructure or service disruptions
- Dynamically acquire computing resources to meet demand
- Mitigate disruptions such as:
 - Misconfigurations
 - Transient network issues



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

15

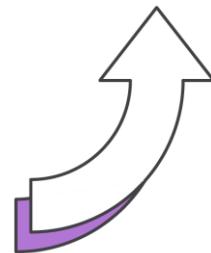
The Reliability pillar addresses the ability of a system to recover from infrastructure or service disruptions and dynamically acquire computing resources to meet demand. It also addresses the ability of a system to mitigate disruptions such as misconfigurations or transient network issues.

It can be difficult to ensure reliability in a traditional environment. Issues arise from single points of failure, lack of automation, and lack of elasticity. By applying the best practices outlined in the Reliability pillar, you can prevent many of these issues. It will help you and your customers to have a properly designed architecture with respect to high availability, fault tolerance, and overall redundancy.

For more information about reliability best practices, see the [Reliability Pillar whitepaper](#).

Performance Efficiency pillar

- Choose efficient resources and maintain their efficiency as demand changes
- Democratize advanced technologies
- Employ mechanical sympathy



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

16

When you consider performance, you want to maximize your performance by using computation resources efficiently. You also want to maintain that efficiency as the demand changes.

It is also important to democratize advanced technologies. In situations where technology is difficult to implement yourself, consider using a vendor. By implementing the technology for you, the vendor handles the complexity and the knowledge, freeing your team to focus on more value-added work.

Mechanical sympathy is when you use a tool or system with an understanding of how it operates best. Use the technology approach that aligns best to what you are trying to achieve. For example, consider data access patterns when you select database or storage approaches.

For more information about performance best practices, see the [Performance Efficiency Pillar whitepaper](#).

Cost Optimization pillar

- Measure efficiency
- Eliminate unneeded expense
- Consider using managed services



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

17

Cost optimization is an ongoing requirement of any good architectural design. The process is iterative, and it should be refined and improved throughout your production lifetime. Understanding how efficient your current architecture is in relation to your goals can remove unneeded expense. Consider using managed services because they operate at cloud scale, and they can offer a lower cost per transaction or service.

For more information, see the [Cost Optimization Pillar whitepaper](#).

Sustainability pillar

- Understand your impact
- Establish sustainability goals
- Maximize utilization
- Anticipate and adopt new, more efficient hardware and software offerings
- Reduce the downstream impact of your cloud workloads



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

18

The Sustainability pillar addresses the ability to build architectures that maximize efficiency and reduce waste.

Sustainability in the cloud is a continuous effort focused primarily on energy reduction and efficiency across all components of a workload by achieving the maximum benefit from the resources provisioned and minimizing the total resources required. This effort can range from the initial selection of an efficient programming language, adoption of modern algorithms, use of efficient data storage techniques, deploying to correctly sized and efficient compute infrastructure, and minimizing requirements for high-powered end-user hardware.

For more information about the design principles for sustainability in the cloud, see the *Sustainability Pillar: AWS Well-Architected Framework* whitepaper at <https://docs.aws.amazon.com/wellarchitected/latest/sustainability-pillar/sustainability-pillar.html>.

The AWS Well-Architected Tool



- Helps you review the state of your workloads and compares them to the latest AWS architectural best practices
- Gives you access to knowledge and best practices used by AWS architects, when you need it
- Delivers an action plan with step-by-step guidance on how to build better workloads for the cloud
- Provides a consistent process for you to review and measure your cloud architectures



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

19

If you would like help with designing a well-architected solution, you can use the AWS Well-Architected Tool. The AWS Well-Architected Tool is a self-service tool that provides you with on-demand access to current AWS best practices. These best practices can help you build secure, high-performing, resilient, and efficient application infrastructure on AWS.

The AWS Well-Architected Tool helps you review the state of your workloads and compare them to the latest AWS architectural best practices. It gives you access to knowledge and best practices used by AWS architects, when you need it.

This tool is available in the AWS Management Console. You define your workload and answer a series of questions in the areas of operational excellence, security, reliability, performance efficiency, and cost optimization. The AWS Well-Architected Tool then delivers an action plan with step-by-step guidance on how to improve your workload for the cloud.

The AWS Well-Architected Tool provides a consistent process for you to review and measure your cloud architectures. You can use the results that the tool provides to identify next steps for improvement, drive architectural decisions, and bring architecture considerations into your corporate governance process.

To learn more about the AWS Well-Architected Tool, see the [AWS Well-Architected Tool website](#).

Section 2 key takeaways



- The AWS Well-Architected Framework provides a **consistent approach** to evaluate cloud architectures and guidance to help implement designs
- The AWS Well-Architected Framework is organized into **six pillars**
- Each pillar documents a **set of foundational questions** that enable you to understand if a specific architecture aligns well with cloud best practices
- The **AWS Well-Architected Tool** helps you review the state of your workloads and compares them to the latest AWS architectural best practices

© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

20

Some key takeaways from this section of the module include:

- The AWS Well-Architected Framework provides a consistent approach to evaluate cloud architectures and guidance to help implement designs.
- The AWS Well-Architected Framework is organized into six pillars.
- Each pillar documents a set of foundational questions that enable you to understand if a specific architecture aligns well with cloud best practices.
- The AWS Well-Architected Tool helps you review the state of your workloads and compares them to the latest AWS architectural best practices.

Section 3: Best practices for building solutions on AWS

Module 2: Introducing Cloud Architecting



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

Introducing Section 3: Best practices for building solutions on AWS.

Design tradeoffs

- Evaluate tradeoffs so you can select an optimal approach
- Examples of tradeoffs include:
 - Trade consistency, durability, and space for time and latency to deliver higher performance
 - Prioritize speed to market of new features over cost
- Base design decisions on empirical data



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

22

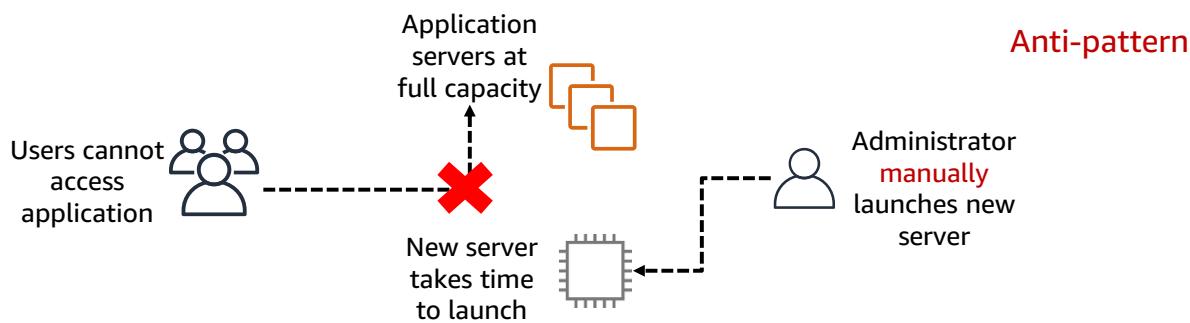
As you design a solution, think carefully about tradeoffs so that you can select an optimal approach. For example, you might trade consistency, durability, and space for time and latency to deliver higher performance. Or, you might prioritize speed to market over cost.

Tradeoffs can increase the cost and complexity of your architecture, so your design decisions should be based on empirical data. For example, you might need to perform load testing to ensure that a measurable benefit is obtained in performance. Or, you might need to perform benchmarking to achieve the most cost-optimal workload over time. When you evaluate performance-related improvements, you will also want to consider how your architecture design choices will impact customers and workload efficiencies.

In this section, you will learn about best practices for designing solutions on AWS. You will also learn about anti-patterns (or bad solution designs) to avoid.

1. Enable scalability (1 of 2)

Ensure that your architecture can handle changes in demand.



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

23

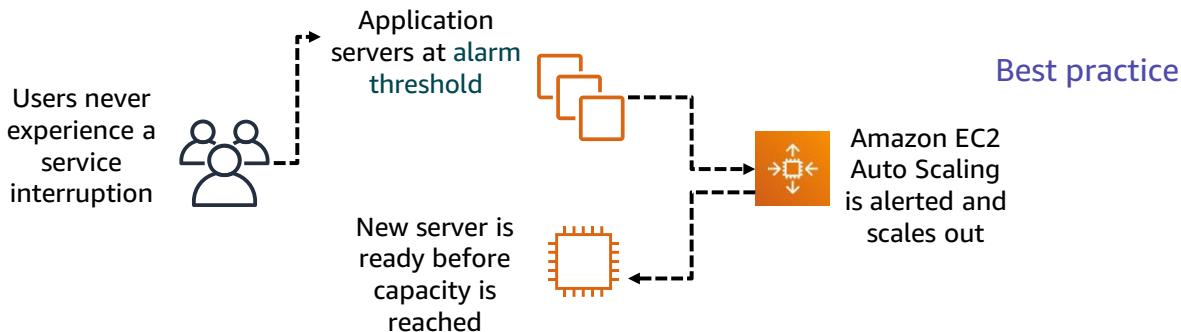
When you run your workloads on the AWS Cloud, you can scale your infrastructure quickly and proactively. Make sure that you implement scalability at every layer of your infrastructure.

To understand the importance of scalability, consider this anti-pattern, where scaling is done reactively and manually.

In this scenario, when application servers reach full capacity, users are prevented from accessing the application. Administrators then manually launch one or more new instances to manage the load. Unfortunately, it takes a few minutes for an instance to become available for use after it's launched. That increases the time that users can't access the application.

1. Enable scalability (2 of 2)

Ensure that your architecture can handle changes in demand.



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

24

By enabling scalability, you can improve your design to anticipate the need for more capacity and deliver it before it's too late.

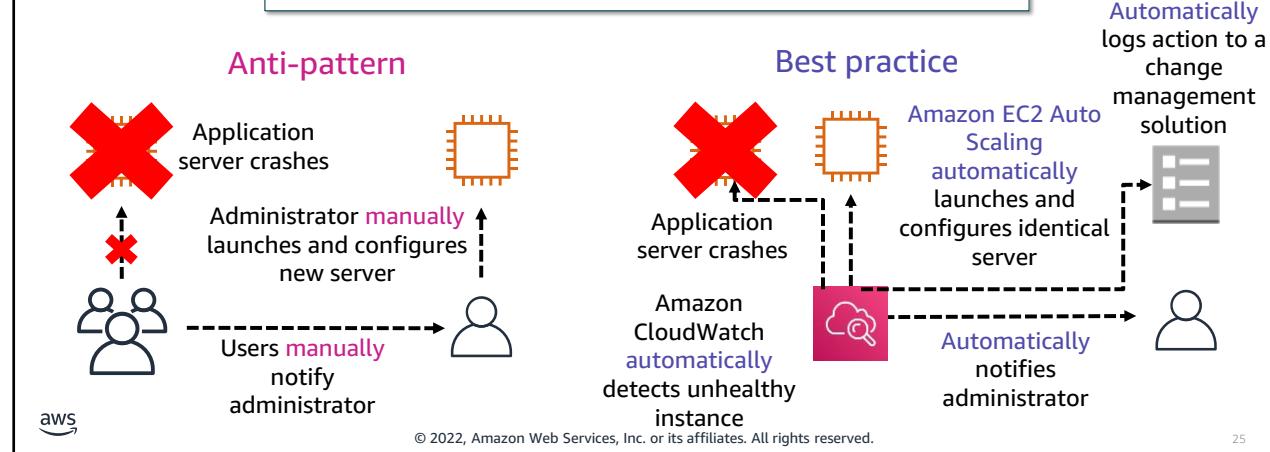
For example, you can use a monitoring solution like Amazon CloudWatch to detect whether the total load across your fleet of servers has reached a specified threshold. You can define this threshold to be *Stayed above 60% CPU utilization for longer than 5 minutes*, or anything related to the use of resources. With CloudWatch, you can also design custom metrics based on specific applications that can trigger the resource scaling that is required.

When an alarm is triggered, Amazon EC2 Auto Scaling immediately launches a new instance. That instance is then ready before capacity is reached, which provides a seamless experience for users.

Ideally, you should also design your system to *scale in* when demand drops off so that you're not running (and paying for) instances that you no longer need.

2. Automate your environment

Where possible, automate the provisioning, termination, and configuration of resources.



AWS offers built-in monitoring and automation tools at virtually every layer of your infrastructure. Take advantage of these tools to ensure that your infrastructure can respond quickly to changes.

You can use tools like CloudWatch and Amazon EC2 Auto Scaling to detect unhealthy resources and automate the launch of replacement resources. You can also be notified when resource allocations change.

3. Treat resources as disposable

Take advantage of the dynamically provisioned nature of cloud computing.

Anti-pattern

- Over time, different servers end up with different configurations
- Resources run when they're not needed
- Hardcoded IP addresses prevent flexibility
- It can be difficult or inconvenient to test new updates on hardware that's in use

Best practice

- Automate deployment of new resources with identical configurations
- Terminate resources that are not in use
- Switch to new IP addresses automatically
- Test updates on new resources, and then replace old resources with updated ones



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

26

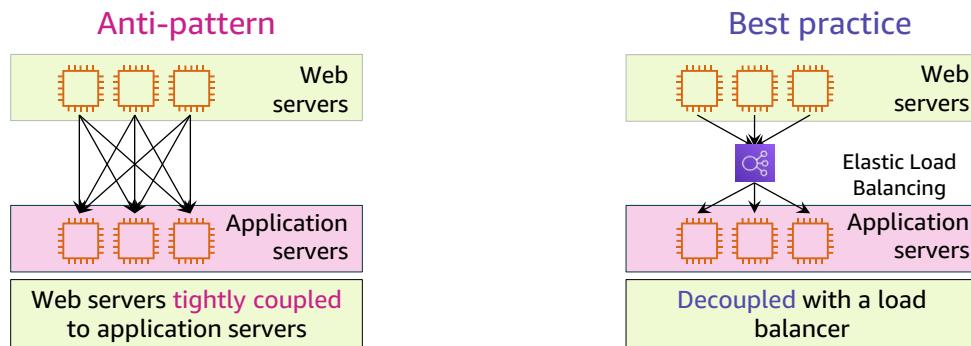
The best practice of treating resources as disposable refers to the idea of thinking about your infrastructure as software instead of hardware.

With hardware, it's easy to buy more specific components than you need so that you are prepared for spikes in usage. That's expensive and inflexible—it's harder to upgrade because of the sunk cost.

Instead, when you treat your resources as disposable, migrating between instances or other discrete resources is fairly straightforward. You can quickly respond to changes in capacity needs, upgrade applications, and manage the underlying software.

4. Use loosely coupled components

Design architectures with independent components.



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

27

Traditional infrastructures have chains of tightly integrated servers, each with a specific purpose. The problem is that when one of those components or layers goes down, the disruption to the system can be fatal. It also impedes scaling. If you add or remove servers at one layer, you must also connect every server on each connecting layer.

The example on the left illustrates a collection of web and application servers that are tightly coupled. If one application server goes down, an error will be caused because the web servers try and fail to connect to it.

With loose coupling, you use managed solutions as intermediaries between the layers of your system. With this design, the intermediary automatically handles both failures and the scaling of components or layers.

The example on the right shows a load balancer (in this case, an Elastic Load Balancing load balancer) that routes requests between the web servers and the application servers. If one application server goes down, the load balancer will automatically start directing all traffic to the two healthy servers.

Two primary solutions for decoupling your components are **load balancers** and **message queues**.

5. Design services, not servers

*Use the breadth of AWS services.
Don't limit your infrastructure to
servers.*

Anti-pattern

- Simple applications run on persistent servers
- Applications communicate directly with one another
- Static web assets are stored locally on instances
- Backend servers handle user authentication and user state storage

Best practice

- When appropriate, consider using containers or a serverless solution
- Message queues handle communication between applications
- Static web assets are stored externally, such as on Amazon Simple Storage Service (Amazon S3)
- User authentication and user state storage are handled by managed AWS services



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

28

The next best practice is to design services, not servers. Although Amazon Elastic Compute Cloud (Amazon EC2) offers tremendous flexibility for designing and setting up your solution, it shouldn't always be the first (or only) solution that you use for every need. In some cases, containers or a serverless solution might be more appropriate. Therefore, it's important to consider what your needs are and which solution is appropriate.

With AWS serverless solutions and managed services, you don't need to provision, configure, and manage an entire Amazon EC2 instance.

Managed solutions that have a lower profile and are more performant can replace server-based solutions at a lower cost. A few examples are AWS Lambda, Amazon Simple Queue Service (Amazon SQS), Amazon DynamoDB, Elastic Load Balancing, Amazon Simple Email Service (Amazon SES), and Amazon Cognito.

6. Choose the right database solution

*Match technology to the workload,
not the other way around.*

Things to consider:

- Read and write needs
- Total storage requirements
- Typical object size and nature of access to these objects
- Durability requirements
- Latency requirements
- Maximum concurrent users to support
- Nature of queries
- Required strength of integrity controls



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

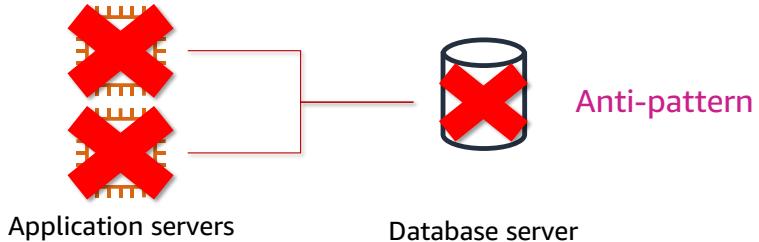
29

It is important that you choose the right database solution. In traditional data centers and on-premises environments, limits on available hardware and licenses can constrain your choice of a data store solution. AWS recommends that you choose a data store based on your needs for your application environment.

7. Avoid single points of failure (1 of 2)

*Assume everything fails.
Then, design backward.*

Where possible, use redundancy to prevent single points from bringing down an entire system.



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

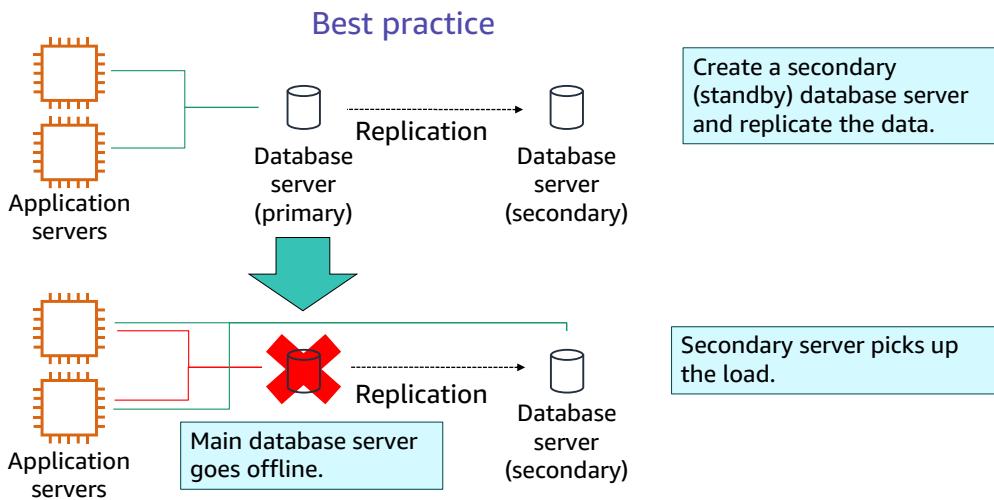
30

Where possible, eliminate single points of failure from your architecture. This doesn't mean that you must always duplicate every component. Depending on your downtime service-level agreements (SLAs), you can use automated solutions that only launch components when needed. You can also use a managed service, where AWS automatically replaces malfunctioning underlying hardware for you.

This simple system shows two application servers connected to a single database server. The database server represents a single point of failure and should be avoided. When it goes down, the application servers also go down.

Application servers should continue to function even if the underlying physical hardware fails, is removed, or replaced.

7. Avoid single points of failure (2 of 2)



A common way to avoid single points of failure is to create a secondary (standby) database server and replicate the data. This way, if the main database server goes offline, the secondary server can pick up the load.

In this example, when the main database goes offline, the application servers automatically send their requests to the secondary database. This example also exemplifies Best Practice #3: Treat resources as disposable, and design your applications to support changes in hardware.

8. Optimize for cost

Take advantage of the flexibility of AWS to increase your cost efficiency.

Things to consider:

- Are my resources the right size and type for the job?
- What metrics should I monitor?
- How do I make sure to turn off resources that are not in use?
- How often will I need to use this resource?
- Can I replace any of my servers with managed services?



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

32

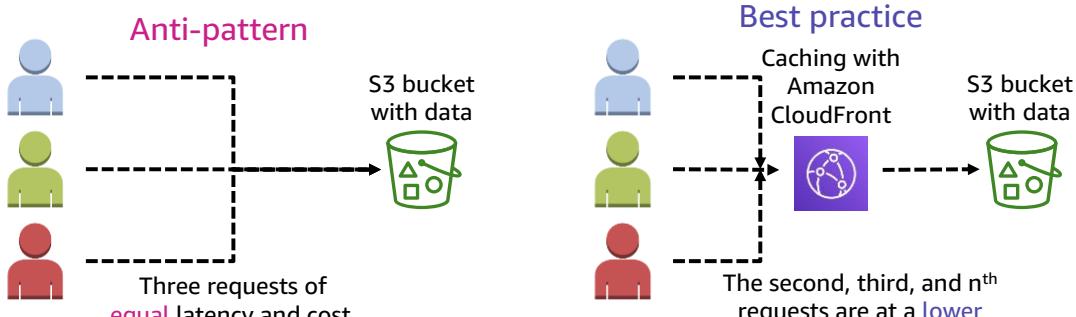
Cloud computing allows you to trade capital expense for variable expense. *Capital expenses (capex)* are funds that a company uses to acquire, upgrade, and maintain physical assets such as property, industrial buildings, or equipment. Under this model, you pay for the servers in the data center, whether they are active or not.

By contrast, AWS services use a *variable expense* cost model, which means that you pay only for the individual services you need, for as long as you use them. Within each service, you can optimize for cost. Many services offer different pricing tiers, models, or configurations.

Keep in mind that it can be very expensive to replicate an on-premises data center setup of servers running 24/7 in the cloud. Therefore, the best way to build your infrastructure, from a cost perspective, is to only provision the resources that you need and stop services when they are not being used.

9. Use caching

Caching minimizes redundant data retrieval operations, improving performance and cost.



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

33

Caching is a technique to make future requests faster and reduce network throughput by temporarily storing data in an intermediary location between the requester and the permanent storage.

In the anti-pattern example, no caching service is used. When anyone requests a file from one of the Amazon Simple Storage Service (Amazon S3) buckets, each request takes the same amount of time to complete, and each request costs the same.

In the best-practice-pattern example, the infrastructure uses Amazon CloudFront in front of Amazon S3 to provide caching. In this scenario, the initial request checks for the file in Amazon CloudFront. If it is not found, CloudFront requests the file from Amazon S3. CloudFront then stores a copy of the file at an edge location close to the user, and sends a copy to the user who made the request. Subsequent requests for the file are retrieved from the (now closer) edge location in CloudFront instead of Amazon S3.

This reduces latency *and* cost because, after the first request, you no longer pay for the file to be transferred out of Amazon S3.

10. Secure your entire infrastructure

Build security into every layer of your infrastructure.

Things to consider:

- Isolate parts of your infrastructure
- Encrypt data in transit and at rest
- Enforce access control granularly, using the principle of least privilege
- Use multi-factor authentication (MFA)
- Use managed services
- Log access of resources
- Automate your deployments to keep security consistent



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

34

Security isn't only about getting through the outer boundary of your infrastructure. It also involves ensuring that your individual environments and their components are secured from each other.

For example, in Amazon EC2, you can create security groups that allow you to determine which ports on your instances can send and receive traffic. Security groups can also determine where that traffic can come from or go to.

You can use security groups to reduce the probability that a security threat on one instance will spread to every other instance in your environment. You should take similar precautions with other services. Specific ways to implement this best practice are discussed throughout the course.

Section 3 key takeaways



- As you design solutions, evaluate tradeoffs and base your decisions on empirical data
- Follow these best practices when building solutions on AWS –
 - Enable scalability
 - Automate your environment
 - Treat resources as disposable
 - Use loosely-coupled components
 - Design services, not servers
 - Choose the right database solution
 - Avoid single points of failure
 - Optimize for cost
 - Use caching
 - Secure your entire infrastructure

© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

35

The key takeaways from this section of the module are:

- As you design solutions, evaluate tradeoffs and base your decisions on empirical data
- Follow these best practices when building solutions on AWS –
 - Enable scalability
 - Automate your environment
 - Treat resources as disposable
 - Use loosely-coupled components
 - Design services, not servers
 - Choose the right database solution
 - Avoid single points of failure
 - Optimize for cost
 - Use caching
 - Secure your entire infrastructure

Section 4: AWS global infrastructure

Module 2: Introducing Cloud Architecting



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

Introducing Section 4: AWS global infrastructure.

AWS Regions

- An **AWS Region** is a geographical area
- Each AWS Region consists of **two or more Availability Zones**
- Communication between Regions uses **AWS backbone network** infrastructure
- You enable and control **data replication** across Regions



Example: London Region



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

37

The AWS Cloud infrastructure is built around Regions. AWS has 22 Regions worldwide. An *AWS Region* is a physical geographical location with two or more *Availability Zones*. Availability Zones in turn consist of one or more *data centers*.

AWS Regions are connected to multiple Internet Service Providers (ISPs). Regions are also connected to a private global network backbone, which provides lower cost and more consistent cross-Region network latency when compared with the public internet.

AWS Regions that were introduced before March 20, 2019 are *enabled* by default. Regions that were introduced after March 20, 2019—such as Asia Pacific (Hong Kong) and Middle East (Bahrain)—are *disabled* by default. You must enable these Regions before you can use them. You can use the AWS Management Console to enable or disable a Region.

Some Regions have restricted access. An AWS (**China**) account provides access to the Beijing and Ningxia Regions only. To learn more about AWS in China, see the [AWS in China page](#). The isolated **AWS GovCloud (US)** Region is designed to allow US government agencies and customers to move sensitive workloads into the cloud by addressing their specific regulatory and compliance requirements.

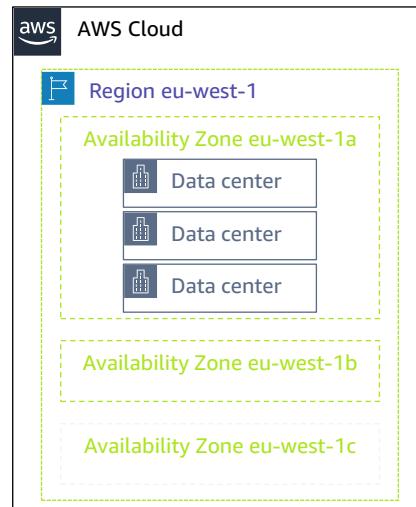
To achieve fault tolerance and stability, Regions are isolated from one another. Resources in one Region are not automatically replicated to other Regions. When you store data in a specific Region, it is not replicated outside that Region. It is your responsibility to replicate data across Regions, if your business needs require it. AWS provides information about the country and—where applicable—the state where each Region resides. You are responsible for selecting the Region where you should store data, based on your compliance and network latency requirements.

AWS products and services are available by Region, so you might not see all Regions available for a given service. For a list of AWS services offered by Region, see the [Region Table](#).

For more information about the AWS global cloud infrastructure, see the [Global Infrastructure website](#). For a current and interactive map of the AWS global infrastructure, see the [Interactive AWS Global Infrastructure map](#).

AWS Availability Zones

- Each Availability Zone is –
 - Made up of **one or more** data centers
 - Designed for **fault isolation**
 - Interconnected with other Availability Zones in a Region using high-speed **private** links
- For certain services, you can choose your Availability Zones
- AWS recommends replicating across Availability Zones for resiliency



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

38

Each AWS Region consists of two or more isolated locations that are known as *Availability Zones*. Each Availability Zone comprises one or more data centers, with some Availability Zones having as many as six data centers. However, no data center can be a part of two Availability Zones.

Each Availability Zone is designed as an independent failure zone. This means that Availability Zones are physically separated within a typical metropolitan Region. They are also located in lower-risk flood plains (specific flood-zone categorization varies by Region). In addition to having a discrete uninterruptible power supply and onsite backup generation facilities, they are each fed via different grids from independent utilities to further reduce single points of failure. Availability Zones are all redundantly connected to multiple tier-1 transit providers.

An Availability Zone is the most granular level of specification that you can make for certain services, such as Amazon EC2.

You are responsible for selecting the Availability Zones where your systems will reside. Systems can span multiple Availability Zones. You should design your systems to survive temporary or prolonged failure of an Availability Zone if a disaster occurs. Distributing applications across multiple Availability Zones enables them to remain resilient in most failure situations, including natural disasters or system failures.

AWS Local Zones

- Enable you to run **latency-sensitive** portions of applications closer to end users and resources in a specific geography
- Are an extension of an AWS Region where you can use AWS services in **geographic proximity to end users**
- Let you place AWS compute, storage, database, and other select services closer to large population, industry, and IT centers where no Region exists today
- Are managed and supported by AWS
- **Los Angeles (LA) AWS Local Zone** is available by invitation



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

39

AWS Local Zones are a new type of AWS infrastructure deployment that places AWS compute, storage, database, and other select services closer to large population, industry, and IT centers where no AWS Region exists today. With AWS Local Zones, you can run latency-sensitive portions of applications closer to end users and resources in a specific geography. You can use AWS Local Zones to deliver single-digit millisecond latency for use cases such as media and entertainment content creation, real-time gaming, reservoir simulations, electronic design automation, and machine learning.

Each AWS Local Zone location is an extension of an AWS Region. You can run latency-sensitive applications in an AWS Local Zone by using AWS services such as Amazon EC2, Amazon Virtual Private Cloud (Amazon VPC), Amazon Elastic Block Store (Amazon EBS), Amazon FSx, and Elastic Load Balancing in geographic proximity to end users. AWS Local Zones provide a high-bandwidth, secure connection between local workloads and workloads that run in the AWS Region. Thus, AWS Local Zones enable you to seamlessly connect back to your other workloads.

that are running in AWS and connect to the full range of in-Region services through the same APIs and toolsets.

AWS Local Zones are managed and supported by AWS, and provide you with all the elasticity, scalability, and security benefits of the cloud. With AWS Local Zones, you can build and deploy latency-sensitive applications closer to your end users by using a consistent set of AWS services. You can also scale up or scale down, and you pay only for the resources that you use.

The Los Angeles AWS Local Zone is generally available by invitation today, and you can expect more Local Zones to come.

To learn more about AWS Local Zones, see the [AWS Local Zones page](#).

AWS data centers

- **Data centers** are where the data resides and data processing occurs
- A data center typically has tens of thousands of servers
- All data centers are online and serving customers
- AWS custom network equipment –
 - Is sourced from multiple ODMs
 - Has a customized network protocol stack



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

40

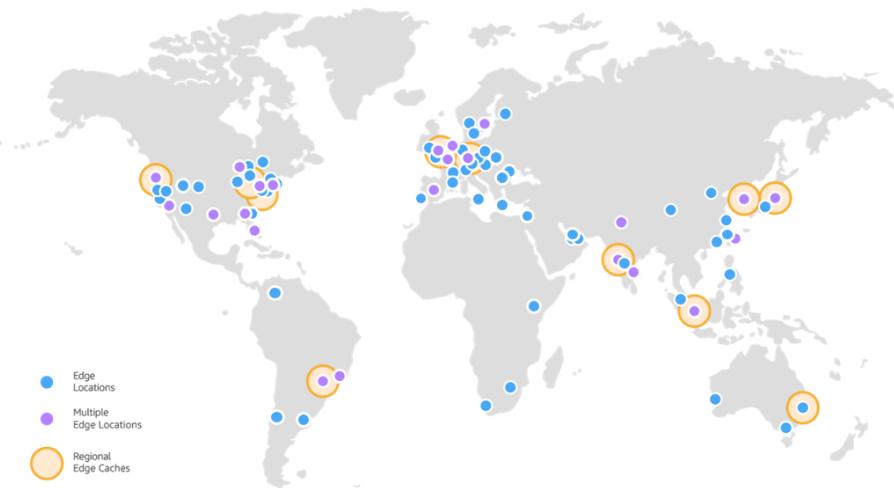
The foundation for the AWS infrastructure is the data centers. You do not specify a data center for the deployment of resources. However, a data center is the location where the actual data resides. Amazon operates state-of-the-art, highly available data centers. Though rare, failures that affect the availability of instances in the same location can occur. If you host all your instances in a single location that is affected by such a failure, none of your instances will be available.

All data centers are online and serving customers. In case of failure, automated processes move customer data traffic away from the affected area. Core applications are deployed in an N+1 configuration, so that in the event of a data center failure, there is sufficient capacity to enable traffic to be load balanced to the remaining sites.

AWS uses custom network equipment sourced from multiple original device manufacturers (ODMs). ODMs design and manufacture products based on specifications from a second company. The second company then rebrands the products for sale.

For more information about AWS data centers, see [Learn how we secure AWS data centers by design](#).

AWS Points of Presence



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

41

To deliver content to end users with lower latency, Amazon CloudFront uses a global network that includes over 200 Points of Presence that are comprised of Edge Locations and Regional Edge Caches.

Edge locations are located in North America, Europe, Asia, Australia, South America, the Middle East, Africa, and China. Edge locations support AWS services like Amazon Route 53 and Amazon CloudFront.

Regional edge caches are used by default with Amazon CloudFront. They are used when you have content that is not accessed frequently enough to remain in an edge location. Regional edge caches absorb this content and provide an alternative to fetching the content from the origin server.

For more information about the Amazon CloudFront infrastructure, see **Amazon CloudFront infrastructure** at https://aws.amazon.com/cloudfront/features/?whats-new-cloudfront.sort-by=item.additionalFields.postDateTime&whats-new-cloudfront.sort-order=desc#Amazon_CloudFront_Infrastructure

Section 4 key takeaways



- The AWS global infrastructure consists of **Regions, Availability Zones, and edge locations**
- Your choice of a Region is typically based on **compliance requirements** or to **reduce latency**
- Each **Availability Zone** is physically separate from other Availability Zones and has redundant power, networking, and connectivity
- **Edge locations** and **Regional edge caches** improve performance by caching content closer to users

© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

42

Some key takeaways from this section of the module include:

- The AWS global infrastructure consists of Regions and Availability Zones
- Your choice of a Region is typically based on compliance requirements or to reduce latency
- Each Availability Zone is physically separate from other Availability Zones and has redundant power, networking, and connectivity
- Edge locations and Regional edge caches improve performance by caching content closer to users

Module wrap-up

Module 2: Introducing Cloud Architecting



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

It's now time to review the module and wrap up with a knowledge check.

Module summary

In summary, in this module, you learned how to:

- Define cloud architecture
- Describe how to design and evaluate architectures using the AWS Well-Architected Framework
- Explain best practices for building solutions on AWS
- Describe how to make informed decisions on where to place AWS resources



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

45

In summary, in this module, you learned how to:

- Define cloud architecture
- Describe how to design and evaluate architectures using the AWS Well-Architected Framework
- Explain best practices for building solutions on AWS
- Describe how to make informed decisions on where to place AWS resources

Complete the knowledge check



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

46

It is now time to complete the knowledge check for this module.

Additional resources

- [AWS Global Infrastructure page](#)
- [Interactive AWS Global Infrastructure map](#)
- [AWS Well-Architected Framework whitepaper](#)
- [Security Pillar whitepaper](#)
- [Operational Excellence Pillar whitepaper](#)
- [Reliability Pillar whitepaper](#)
- [Performance Efficiency Pillar whitepaper](#)
- [Cost Optimization Pillar whitepaper](#)
- [Sustainability Pillar whitepaper](#)



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

47

If you want to learn more about the topics covered in this module, you might find the following additional resources helpful:

- [AWS Global Infrastructure page](#)
- [Interactive AWS Global Infrastructure map](#)
- [AWS Well-Architected Framework whitepaper](#)
- [Security Pillar whitepaper](#)
- [Operational Excellence Pillar whitepaper](#)
- [Reliability Pillar whitepaper](#)
- [Performance Efficiency Pillar whitepaper](#)
- [Cost Optimization Pillar whitepaper](#)
- [Sustainability Pillar whitepaper](#)

Thank you

Corrections, feedback, or other questions?
Contact us at <https://support.aws.amazon.com/#/contacts/aws-training>.
All trademarks are the property of their owners.



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

Thank you for completing this module.



**AWS Academy Cloud Architecting
Module 03 Student Guide
Version 2.0.8**

200-ACACAD-20-EN-SG

© 2022 Amazon Web Services, Inc. or its affiliates. All rights reserved.

This work may not be reproduced or redistributed, in whole or in part, without prior written permission from Amazon Web Services, Inc. Commercial copying, lending, or selling is prohibited.

All trademarks are the property of their owners.

Contents

Module 3: Adding a Storage Layer	4
----------------------------------	---



Module 3: Adding a Storage Layer

AWS Academy Cloud Architecting

© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

Welcome to Module 3: Adding a Storage Layer.

Module overview

Sections

1. The simplest architecture
2. Using Amazon S3
3. Storing data in Amazon S3
4. Moving data to and from Amazon S3
5. Choosing Regions for your architecture

Demonstrations

- Amazon S3 Versioning
- Amazon S3 Transfer Acceleration

Labs

- Guided Lab: Hosting a Static Website
- Challenge Lab: Creating a Static Website for the Café



Knowledge check



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

2

This module includes the following sections:

1. The simplest architecture
2. Using Amazon S3
3. Storing data in Amazon S3
4. Moving data to and from Amazon S3
5. Choosing Regions for your architecture

This module also includes:

- An educator-led demonstration that will show you how the Amazon S3 versioning feature works.
- An educator-led demonstration that will show you how to configure Amazon S3 Transfer Acceleration.
- A hands-on guided lab, where detailed step-by-step instructions explain how to create an Amazon S3 bucket and configure it to host a simple website.
- A hands-on challenge lab where you will deploy a static website to support the café scenario. You will be provided with limited guidance because the tasks that are involved closely mirror the guided lab activity that you completed earlier in the module.

Finally, you will be asked to complete a knowledge check that will test your understanding of key concepts covered in this module.

Module objectives

At the end of this module, you should be able to:

- Recognize the problems that Amazon Simple Storage Service (Amazon S3) can solve
- Describe how to store content efficiently using Amazon S3
- Recognize the various Amazon S3 storage classes and cost considerations
- Describe how to move data to and from Amazon S3
- Describe how to choose a Region
- Create a static website



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

3

At the end of this module, you should be able to:

- Recognize the problems that Amazon Simple Storage Service (Amazon S3) can solve
- Describe how to store content efficiently by using Amazon S3
- Recognize the various Amazon S3 storage classes and cost considerations
- Describe how to move data to and from Amazon S3
- Describe how to choose a Region
- Create a static website

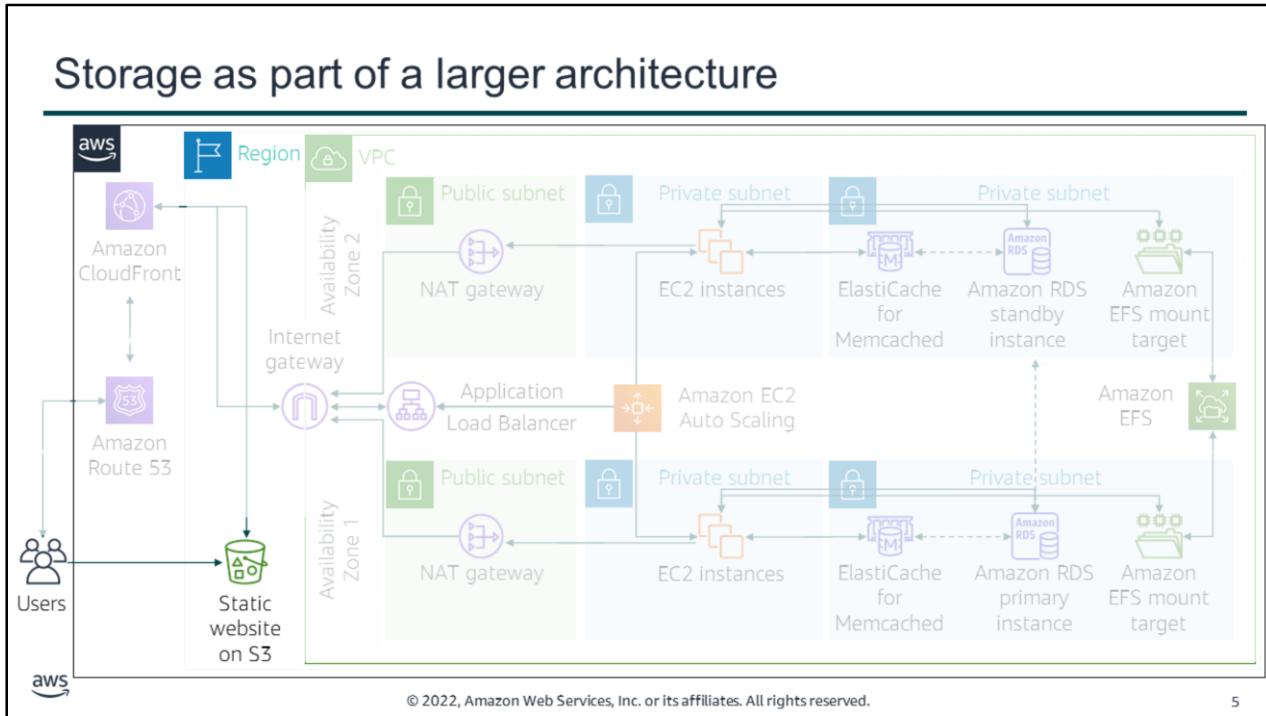
Section 1: The simplest architecture

Module 3: Adding a Storage Layer



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

Introducing Section 1: The simplest architecture.



As each module introduces new features, those parts of this larger diagram will be revealed.

In this module, you start with one of the simplest of architectures that can be implemented on AWS, which is creating a static website by hosting it entirely on Amazon S3. You will also learn about the various Amazon S3 storage options and some key considerations for when you choose a Region on AWS.

Café business requirement

The café has just started up. They want to establish a simple static website that provides customers with basic information about the café (including a menu, store hours, location, and more).



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

6

The café has a single location in a large city, where they sell desserts and coffee. The business is owned by Frank and Martha, a husband-and-wife team who work at the café. Their daughter, Sofía, and their other employee, Nikhil—who is a secondary school student—also work at the café.

The café currently doesn't have a marketing strategy. They mostly gain new customers when someone walks by, notices the café, and decides to give it a try. The café has a reputation for high-quality desserts and coffees, but their reputation is limited to people who have visited, or who have heard about them from their customers.

Sofía suggested that they should expand community awareness of what the cafe has to offer. Frank and Martha agreed. The café doesn't have a web presence yet, and they don't currently use any cloud computing services. However, that situation is about to change. The first challenge will be to create a basic website for the café.

You will learn more details about the business requirements—and how to use Amazon Web Services to meet those business requirements—throughout this module.

Section 2: Using Amazon S3

Module 3: Adding a Storage Layer

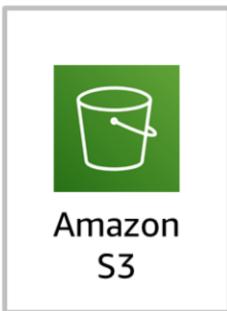


© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

Introducing Section 2: Using Amazon S3.

Amazon S3

An **object** storage service:



- It stores massive (unlimited) amounts of unstructured data
- Data files are stored as objects in a **bucket** that you define
- 5 TB is the maximum file size of a single object
- All objects have a REST-accessible globally unique URL (universal namespace)
- All objects have a **key**, **version ID**, **value**, **metadata**, and **subresources**



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

8

Amazon S3 is an *object storage service*. It enables you to store virtually unlimited amounts of data. Data files are stored as objects. You place objects in a bucket, which you define. Every bucket must have name that is globally unique across Regions. This means that the bucket name must be unique across all AWS customer accounts.

The objects you store can vary in size from 0 bytes to 5 TB. Though individual objects cannot be larger than 5 TB, you can store as much total data as you need.

Each object has five consistent characteristics.

First, it has a key, which is the name that you assign to an object. You use the object key to retrieve the object. In the AWS Management Console, you can create a directory inside a bucket, and upload an object to that directory. However, in reality, Amazon S3 does not know about directories, so the key value includes the full path relative to the bucket root.

Objects also include a version ID. In a bucket, a key and version ID uniquely identify an object. You will learn more about versioning later in this module.

The *value* of the object is the actual content that you store. It can be any sequence of bytes. Object values are immutable, which means that after you upload an object, you cannot modify the value. If you want to modify the object, you must make a change outside of Amazon S3 and then reupload the object.

Objects also include metadata, which is a set of name-value pairs you can use to store information about the object. You can assign metadata, which is referred to as *user-defined metadata*, to your objects in Amazon S3. Amazon S3 also assigns system-metadata to these objects, which it uses for managing objects.

Finally, Amazon S3 also uses subresources to store additional object-specific information.

Amazon S3 benefits



Durability

- It ensures data is not lost
- S3 Standard storage provides 11 9s (or 99.99999999%) of durability



Availability

- You can access your data when needed
- S3 Standard storage class is designed for four 9s (or 99.99%) availability



Scalability

- It offers virtually unlimited capacity
- Any single object of 5 TB or less



Security

- It offers fine-grained access control



Performance

- It is supported by many design patterns



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

9

Amazon S3 provides many features that make it an important component of many solutions built on AWS.

First, it provides *durability*, which describes the average annual expected loss of objects. 11 9s of durability means that every year, there is a 0.00000001 percent chance of losing an object. For example, if you store 10,000 objects on Amazon S3, you can expect to incur a loss of a single object once every 10,000,000 years on average. Amazon S3 redundantly stores your objects on multiple devices across multiple facilities in the Amazon S3 Region you designate. Amazon S3 is designed to sustain concurrent device failures by quickly detecting and repairing any lost redundancy. Amazon S3 also regularly verifies the integrity of your data by using checksums.

Amazon S3 also provide four 9s (or 99.99 percent) of *availability*. Availability refers to your ability to access your data quickly, when you want it. It also provides a virtually unlimited capacity to store your data, so it is *scalable*. Amazon S3 has robust *security* settings. It provides many ways to control access to the data that you store, and also enables you to encrypt your data.

Finally, Amazon S3 is *highly performant*, with a first-byte latency that is measured in milliseconds for most storage classes. For more information about [S3 performance design patterns](#), see the Amazon S3 Documentation. Common approaches include using caching for frequently accessed content; configurable retry and timeout logic for objects that receive significant request traffic in a short period of time; and horizontal scaling and request parallelization for high throughput across the network.

Amazon S3 common usage patterns



Amazon S3



What problems can you solve by using Amazon S3?
You will now consider some [use cases](#).



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

10

Now that you know about many Amazon S3 features, how can you use these features to address your needs?

In this section of the module, you will learn about four common use cases that use Amazon S3 as an essential part of a robust architectural solution.

Amazon S3 use case 1: Store and distribute web content and media

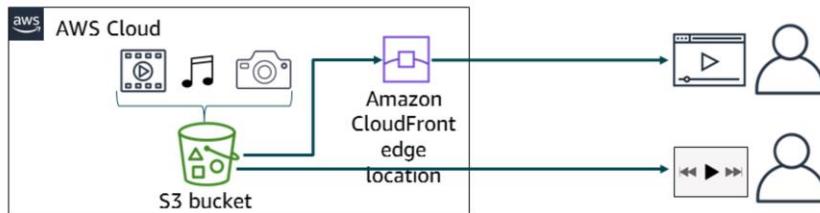
Build a redundant, scalable, and highly available infrastructure that hosts video, photo, or music uploads and downloads.



<https://<bucket-name>.s3.amazonaws.com>



<https://<bucket-name>.s3.amazonaws.com/video.mp4>



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

11

On common use scenario for Amazon S3 is to use it for *media hosting*. In this use case, Amazon S3 is used to store and distribute videos, photos, music files, and other media. This content can be delivered directly from Amazon S3 because each object in Amazon S3 has a unique HTTP URL.

Alternatively, Amazon S3 can serve as an origin store for a content delivery network (CDN), such as *Amazon CloudFront*. The elasticity of Amazon S3 makes it well-suited for hosting web content that needs bandwidth to address extreme demand spikes. Also, because you do not need to provision storage for Amazon S3, it works well for fast growing websites that host data-intensive, user-generated content, such as video and photo-sharing sites.

Securing Amazon S3 buckets and objects

- Newly created S3 buckets and objects are **private** and **protected** by default
- When use cases must share Amazon S3 data –
 - Manage and control the data access
 - Follow the **principle of least privilege**
- Tools and options for controlling access to Amazon S3 data –
 - [Block Public Access](#) feature: It is enabled on new buckets by default, simple to manage
 - [IAM policies](#): A good option when the user can authenticate using IAM
 - [Bucket policies](#): You can define access to a specific object or bucket
 - [Access control lists \(ACLs\)](#): A legacy access control mechanism
 - [S3 Access Points](#): You can configure access with names and permissions specific to each application
 - [Presigned URLs](#): You can grant time-limited access to others with temporary URLs
 - [AWS Trusted Advisor](#) bucket permission check: A free feature



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

12

By default, all S3 buckets are private and can be accessed only by users who are explicitly granted access. It is essential that you manage and control access to Amazon S3 data. AWS provides many tools and options for controlling access to your S3 buckets or objects, such as:

- Using Amazon S3 Block Public Access. These settings override any other policies or object permissions. Enable Block Public Access for all buckets that you don't want to be publicly accessible. This feature provides a straightforward method for avoiding unintended exposure of Amazon S3 data.
- Writing AWS Identity and Access Management (IAM) policies that specify the users or roles that can access specific buckets and objects.
- Writing bucket policies that define access to specific buckets or objects. This option is typically used when the user or system cannot authenticate by using IAM. Bucket policies can be configured to grant access across AWS accounts or to grant public or anonymous access to Amazon S3 data. If bucket policies are used, they should be written carefully and tested fully. You can specify a deny statement in a bucket policy to restrict access. Access will be restricted even if the users have permissions that are granted in an identity-based policy that is attached to the users.
- Creating S3 Access Points. Access points are unique hostnames that enforce distinct permissions and network controls for requests that are made through it. Customers with shared datasets can scale access for many applications by creating individualized access points with names and permissions that are customized for each application.
- Setting access control lists (ACLs) on your buckets and objects. ACLs are less commonly used (ACLs predate IAM). If you use ACLs, do not set access that is too open or permissive.
- AWS Trusted Advisor provides a bucket permission check feature. It is a useful tool for discovering if any of the buckets in your account have permissions that grant global access.

Three general approaches to configuring access

Configure the appropriate security settings for your use case on the bucket and objects.



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

13

Here are three different general approaches to configuring access to objects in an S3 bucket.

The scenario on the left shows the default security settings for Amazon S3. By default, all Amazon S3 buckets and the objects stored in them are private (*protected*). The only entities with access to a newly created, unmodified bucket are the account administrator and the AWS account root user. The resource owner can grant specific access permissions to others, but anyone not granted those permissions will not have access.

The scenario in the middle shows an occasion where S3 security settings have been disabled and anyone can publicly access the objects stored in the bucket.

Caution! Using an Amazon S3 bucket to host a static website is an example of setting up an AWS architecture quickly. However, for most Amazon S3 use cases, you would not want to grant public access to Amazon S3. Most use cases do not require public access. More often, you use Amazon S3 to store data that is used by an application that runs outside of Amazon S3, or to back up sensitive data. For these common use cases, public access to buckets that hold data should never be granted.

The scenario on the right shows a case where Amazon S3 was configured to provide *controlled access*. User A was granted access to the objects in the bucket, but User B was denied access. Controlled access scenarios are common. They can be configured by the bucket owner by using one or more of the tools or options for controlling access to Amazon S3 data this module discussed earlier.

Consider encrypting objects in Amazon S3

- **Encryption** encodes data with a **secret key**, which makes it unreadable
 - Only users who have the secret key can decode the data
 - Optionally, use AWS Key Management Service (AWS KMS) to manage secret keys



- **Server-side encryption**

- On the bucket, enable this feature by selecting the Default encryption option
- Amazon S3 encrypts objects before it saves the objects to disk, and decrypts the objects when you download them



- **Client-side encryption**

- Encrypt data on the client side and upload the encrypted data to Amazon S3
- In this case, you manage the encryption process



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

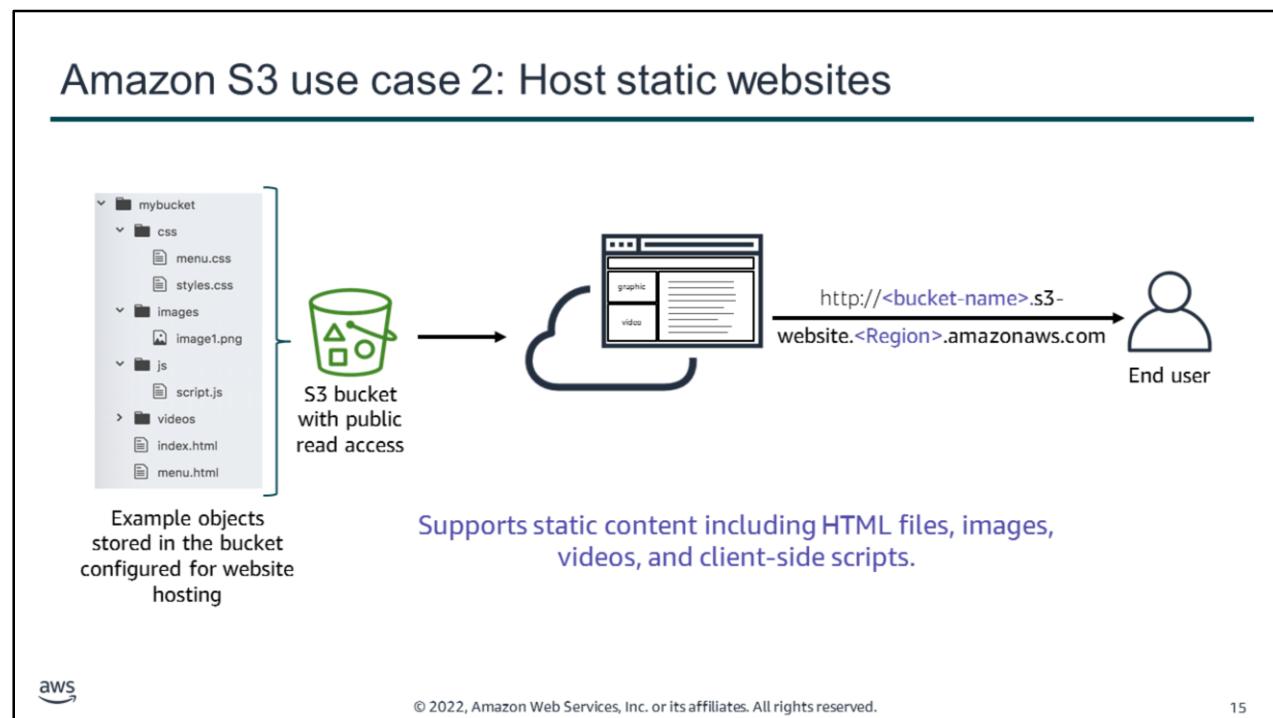
14

When your objective is to protect digital data, data encryption is an essential tool. Data encryption takes data that is legible and encodes. Encrypted data is unreadable to anyone who does not have access to the secret key that can be used to decode it. Thus, even if an attacker gains access to your data, they cannot make sense of it.

You have two primary options for encrypt data stored in Amazon S3.

When you set the Default encryption option on a bucket, it enables server-side encryption. With this feature, Amazon S3 encrypts your object before it saves the object to disk. And then Amazon S3 will decrypt it when you download the object.

Client-side encryption is the other option. When you use this approach, you encrypt the data on the client side before you upload it to Amazon S3. In this case, you manage the encryption process, the encryption keys, and related tools. Like server-side encryption, client-side encryption can reduce risk by encrypting the data with a key that is stored in a different mechanism than the mechanism that stores the data itself.



A second Amazon S3 use case is to use the service to host a static website. On a static website, individual webpages include static content. They might also contain client-side scripts.

By contrast, a *dynamic* website relies on server-side processing, which might involve database queries that run in response to server-side scripts, such as PHP, JSP, or ASP.NET. Amazon S3 does not support server-side scripting. However AWS offers other services that enable you to host dynamic websites.

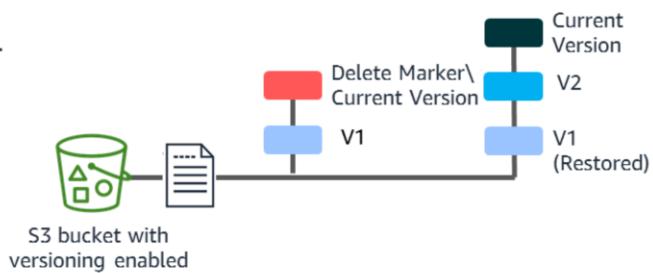
To host a *static* website, configure an S3 bucket for website hosting. Then, upload your website content to the bucket.

The example shows that the static site might consist of HTML files, images, videos, and client-side scripts in formats such as JavaScript.

With this approach, you do not need to run a virtual machine that hosts a web server. In fact, you do not need to run a server. However, you can still host a website. Amazon S3 provides a low-cost solution for web hosting that includes high performance, scalability, and availability.

Amazon S3 best practice: Versioning

- Protects against accidental overwrites and deletes with no performance penalty
- Generates a new version with every upload
- Enables easy retrieval of deleted objects or rollback to previous versions
- Three possible states of an S3 bucket –
 1. *Default: Versioning not enabled*
 2. Versioning-enabled
 3. Versioning-suspended



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

16

Amazon S3 provides customers with a highly secure and durable storage infrastructure. Versioning offers an additional level of protection. It provides a way to recover data if an application fails, or when customers accidentally overwrite or delete objects.

Versioning is a method of keeping multiple variants of an object in the same bucket. You can use versioning to preserve, retrieve, and restore every version of every object stored in an S3 bucket.

- If you delete an object, instead of removing it permanently, Amazon S3 inserts a delete marker, which becomes the current object version. You can always restore the previous version.
- Overwriting an object results in a new object version in the bucket. You can always restore the previous version.

Buckets can be in one of three states: unversioned (the default), versioning-enabled, or versioning-suspended. After you enable versioning for a bucket, you can never change it to an unversioned state. You can, however, suspend versioning on that bucket.

Demonstration: Amazon S3 Versioning



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

17

Now, the educator might choose to demonstrate Amazon S3 versioning using the AWS Management Console.

Support for Cross-Origin Resource Sharing (CORS)

The diagram shows an S3 bucket containing an HTML file and a JavaScript file. The HTML file contains links to 'graphic' and 'video' assets. A dashed arrow points from the JavaScript file to a 'CORS' section, which is highlighted with a purple border. This section contains an example of CORS configuration XML code.

```
<corsConfiguration>
<corsRule>
<allowedOrigin>http://www.example.com</allowedOrigin>
<allowedMethod>PUT</allowedMethod>
<allowedMethod>POST</allowedMethod>
<allowedMethod>DELETE</allowedMethod>
<allowedHeader>*</allowedHeader>
<maxAgeSeconds>3000</maxAgeSeconds>
<exposeHeader>x-amz-server-side-encryption</exposeHeader>
<exposeHeader>x-amz-request-id</exposeHeader>
<exposeHeader>x-amz-id-2</exposeHeader>
</corsRule>
</corsConfiguration>
```

© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

18

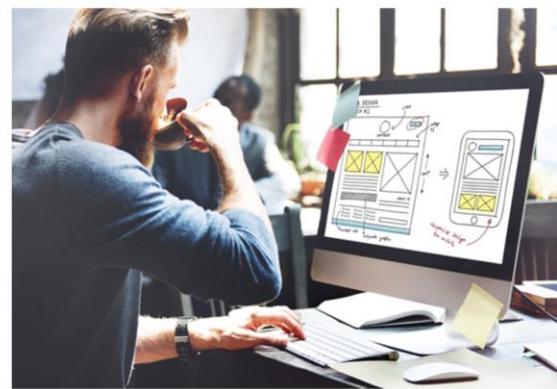
Cross-origin resource sharing (CORS) defines a way for client web applications that are loaded in one domain to interact with resources in a different domain. With CORS support, you can build rich client-side web applications with Amazon S3 and selectively allow cross-origin access to your Amazon S3 resources.

To configure your bucket to allow cross-origin requests, you create a CORS configuration. A CORS configuration is an XML document with rules that identify:

- The origins that you will allow to access your bucket.
- The operations (HTTP methods) that will support for each origin. In this example, PUT, POST, and DELETE requests are allowed from the `http://www.example.com` origin, which could be configured using Amazon Route 53 to be another S3 bucket.
- Other operation-specific information.

For more information about CORS, see the [Cross-Origin Resource Sharing \(CORS\)](#) AWS documentation.

Module 3 - Guided Lab: Hosting a Static Website



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

19

You will now complete Module 3 – Guided Lab: Hosting a Static Website.

Guided lab: Tasks

1. Create a bucket in Amazon S3
2. Upload content to your bucket
3. Enable access to the objects
4. Update the website

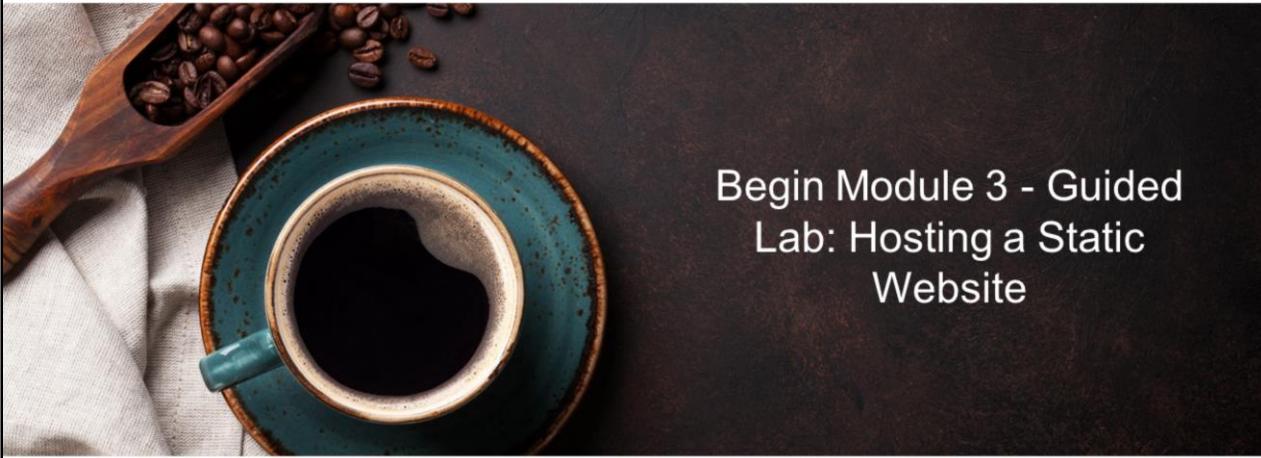


© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

20

In this guided lab, you will complete the following tasks:

1. Create a bucket in Amazon S3
2. Upload content to your bucket
3. Enable access to the objects
4. Update the website



A timer icon with the text "~ 20 minutes" indicating the duration of the lab.

Begin Module 3 - Guided
Lab: Hosting a Static
Website

aws

© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

21

It is now time to start the guided lab.

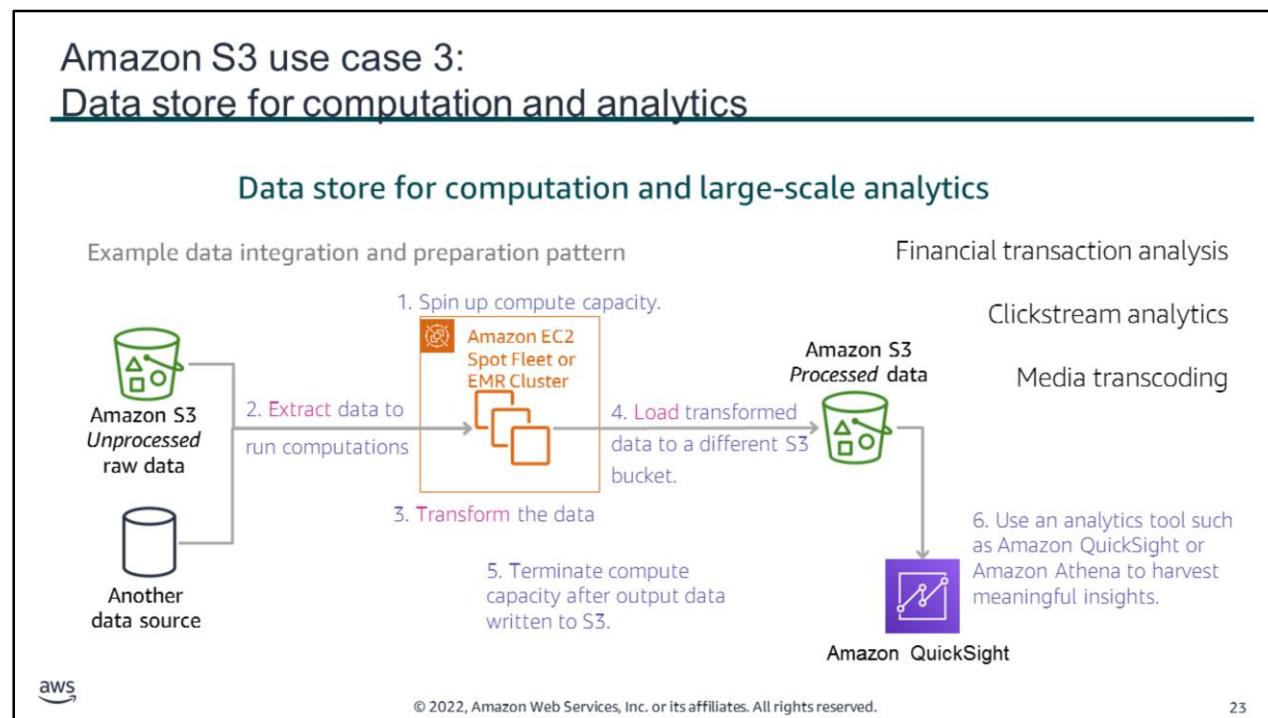
Guided lab debrief: Key takeaways



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

22

Your educator might choose to lead a conversation about the key takeaways from this guided lab after you have completed it.

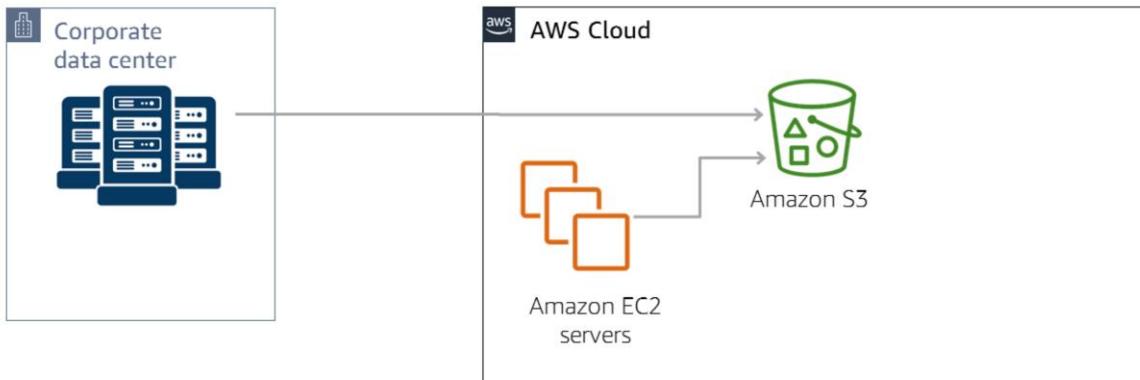


You can also use Amazon S3 as a data store for computation or large-scale analytics, such as financial transaction analysis, clickstream analytics, and media transcoding. Amazon S3 can support these workloads because of its horizontal scaling ability, which enables multiple concurrent transactions.

In the example here, an Amazon Elastic Compute Cloud (Amazon EC2) Spot Fleet is spun up when the bid price for Spot Instances is low, or when an Amazon EMR cluster is spun up. Regardless, after the compute capacity is available, *raw unprocessed* data is extracted from Amazon S3 and also from another data source. The data is run through compute algorithms that integrate and transform it. The resulting *processed* data is loaded into a different Amazon S3 bucket. Now that the data has been processed, the compute capacity is terminated to save on costs. Finally, an analytics tool, such as Amazon QuickSight, might be used to harvest meaningful insights from the processed data. This is just one example scenario of how Amazon S3 can play an essential role for data storage in a large-scale analytics solutions architecture.

Amazon S3 use case 4: Back up and archive critical data

Amazon S3 as a data backup solution



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

24

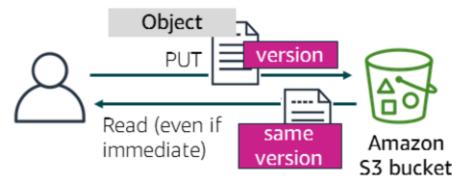
In the fourth and final use case discussed in this module, Amazon S3 is used as a data backup solution. Because of its highly durable and scalable nature, Amazon S3 works well as a data backup and archival tool.

In the scenario, data is backed up from an on-premises corporate data center, and also from a large number of Amazon EC2 servers. These servers run applications that generate data.

Additionally, you can move long-term data from Amazon S3 standard storage to Amazon Simple Storage Service Glacier. This process will be discussed in further detail later in this module. Another Amazon S3 option you can configure on your buckets—to achieve even higher levels of durability—is *cross-Region replication*. In cross-Region replication, objects that are uploaded to a bucket in one Region will be automatically copied to other S3 buckets in other Regions.

Amazon S3 data consistency model

- Amazon S3 is **strongly consistent** for all new and existing *objects* in all Regions
 - Provides **read-after-write consistency** for all **GET**, **LIST**, and **PUT** operations on objects in S3 buckets
 - The consistency model offers an advantage for big data workloads
 - Bucket configurations have an eventually consistent model



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

25

Many customers develop big data analytics applications that use Amazon S3 for object storage. These applications often require access to an object immediately after a write. Prior to December, 2020, Amazon S3 provided eventual consistency for overwrite PUTS and Deletes in all Regions. However, Amazon S3 is now strongly consistent for all new and existing S3 objects in all AWS Regions.

Amazon S3 achieves high availability by replicating data across multiple servers within AWS data centers. If a PUT request is successful, the data is safely stored. Any read (GET or LIST) that is initiated following a successful PUT response will return the data written by the PUT. This strong read-after-write consistency exists automatically for all applications, without changes to performance or availability.

Strong consistency simplifies the migration of on-premises analytics workloads by removing the need to make changes to support applications. It also removes the need for extra infrastructure, such as S3Guard, to provide strong consistency.

While objects are strongly consistent, Amazon S3 bucket configurations have an eventual consistency model. For example, if you delete a bucket and immediately list all buckets, the deleted bucket might still appear in the list. However, within a short period of time, if you run the list bucket command again, the deleted bucket will no longer appear in the list buckets results.

For more details, read the [Amazon S3 strong consistency](#) documentation.

Section 2 key takeaways



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

26

- Buckets must have a **globally unique name** and are defined at the Region level
- Buckets are **private** and protected by default
- Amazon S3 security can be configured with IAM policies, bucket policies, access control lists, S3 access points, and presigned URLs
- Amazon S3 is **strongly consistent** for all new and existing objects in all Regions
- **5 TB** is the maximum size of a single object
- Amazon S3 is often used as a data store for computation and analytics, and as a backup and archive service for critical data

Some key takeaways from this section of the module include:

- Buckets must have a **globally unique name** and are defined at the Region level
- Buckets are **private** and protected by default
- Amazon S3 security can be configured with IAM policies, bucket policies, access control lists, S3 access points, and presigned URLs
- Amazon S3 is **strongly consistent** for all new and existing objects in all Regions
- **5 TB** is the maximum size of a single object, but you can store a virtually unlimited number of objects
- Amazon S3 is often used as a data store for computation and analytics, and as a backup and archive service for critical data

Section 3: Storing data in Amazon S3

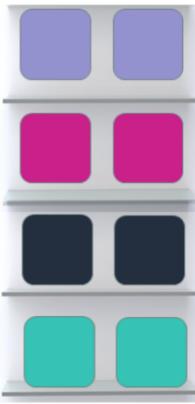
Module 3: Adding a Storage Layer



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

Introducing Section 3: Storing data in Amazon S3.

Amazon S3 and Amazon S3 Glacier storage classes



S3 Standard:
Frequently accessed data

S3 Standard IA:
Long-lived, infrequently accessed data

S3 One Zone IA:
Long-lived, infrequently accessed, non-critical data

Amazon S3 Glacier or Deep Archive:
Archiving rarely accessed data

Amazon S3 Intelligent Tiering

Automatically moves your objects between storage classes based on data access patterns.



For more information about Amazon S3 Storage Class details, see <https://aws.amazon.com/s3/storage-classes/>.

© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

28

Now that you have built a website using Amazon S3, here is a comparison of the different Amazon S3 storage classes and their characteristics.

S3 Standard offers high durability, availability, and performant object storage for frequently accessed data. Because it delivers low latency and high throughput, S3 Standard is appropriate for a wide variety of use cases, including cloud applications, dynamic websites, content distribution, mobile and gaming applications, and big data analytics. It provides durability across at least three Availability Zones.

S3 Standard-Infrequent Access (S3 Standard-IA) offers all the benefits of Amazon S3 Standard, but it runs on a different cost model to store infrequently accessed data, such as older digital images or older log files. There is a 30-day minimum storage fee applied to any data placed in it, and also a higher cost to retrieve data from S3 Standard-IA than from S3 Standard storage.

S3 One Zone-IA stores data in a single Availability Zone. It is ideal for customers who want a lower-cost option and who do not need the availability and resilience of S3 Standard or S3 Standard-IA. It's a good choice for storing secondary backup copies of on-premises data or easily re-creatable data. You can also use it as cost-effective storage for data that is replicated from another AWS Region.

S3 Intelligent-Tiering is designed to optimize costs by automatically moving data to the most cost-effective access tier, without performance impact or operational overhead. For a small monthly monitoring and automation fee per object, Amazon S3 monitors access patterns of the objects in S3 Intelligent-Tiering. It moves objects that have not been accessed for 30 consecutive days to the infrequent access tier. If an object in the infrequent access tier is accessed, it is automatically moved back to the frequent access tier. There are no retrieval fees when using S3 Intelligent-Tiering and no additional tiering fees when objects are moved between tiers.

Amazon S3 Glacier is a secure, durable, and low-cost storage class for data archiving. You can reliably store any amount of data at costs that are competitive with or cheaper than on-premises solutions. To keep costs low, but suitable for different needs, you have three options for retrieving data, with varying access times and cost:

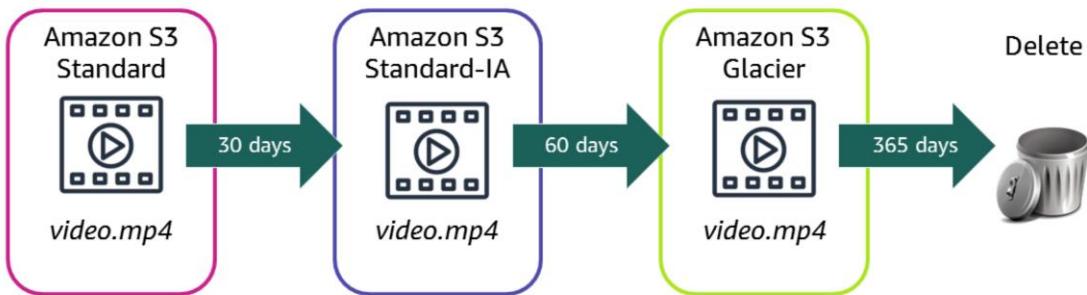
- Expedited retrievals are typically made available within 1–5 minutes
- Standard retrievals typically complete within 3–5 hours
- Bulk retrievals typically complete within 5–12 hours

Amazon S3 Glacier Deep Archive is the lowest-cost storage class for Amazon S3. It supports the long-term retention and digital preservation for data that might be accessed once or twice in a year. Data is stored across at least three geographically dispersed Availability Zones, protected by 11 9s (99.9999999 percent) of durability, and can be restored within 12 hours.

For more details about [Amazon S3 Storage Classes](#), see the AWS Documentation.

Amazon S3 lifecycle policies

Configure an **Amazon S3 Lifecycle Policy** to delete or move objects based on age.



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

29

You can configure the lifecycle of your objects to manage how they are stored throughout their lifecycle. A **lifecycle configuration** is a set of rules that define actions that Amazon S3 applies to a group of objects.

After an S3 lifecycle policy is set, *your data will automatically transfer to a different storage class* without any changes to your application.

By using lifecycle policies, you can cycle data at regular intervals among different Amazon S3 storage types. This cycling reduces your overall cost because you pay less for data as it becomes less important over time. In addition to being able to set lifecycle rules per object, you can also set lifecycle rules per bucket.

For more information about object lifecycle management, see the [Object Lifecycle Management](#) AWS documentation details.

Amazon S3 costs



Pay only for use, including:

GBs of objects stored (per month). Different pricing per *Region* and per *storage class*.

Transfer OUT to other Regions or the internet.

PUT, COPY, POST, LIST, GET, SELECT, lifecycle transition, data retrieval requests.

No charge for:

Data transfers IN from the internet to Amazon S3.

Transfers between S3 buckets or from Amazon S3 to any services in the same AWS Region.

Transfer OUT to Amazon CloudFront.

DELETE and CANCEL requests.



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

30

With Amazon S3, you pay only for what you use. There is no minimum fee. There are four cost components to consider when you decide which Amazon S3 storage class best fits your data profile—storage pricing, request and data retrieval pricing, data transfer and transfer acceleration pricing, and data management features pricing.

Details about Amazon S3 pricing can be found at [Amazon S3 Pricing](#).

Section 3 key takeaways



- Amazon S3 storage classes include –
 - S3 Standard
 - S3 Standard-IA
 - S3 One Zone-IA
 - S3 Intelligent-Tiering
 - S3 Glacier
 - S3 Glacier Deep Archive
- An Amazon S3 lifecycle policy can delete or move objects to less expensive storage classes based on age
- Transferring data in from the internet to Amazon S3 is free, but transferring out to other Regions or to the internet incurs a fee

© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

31

Some key takeaways from this section of the module include:

- Amazon S3 storage classes include – S3 standard, S3 standard-Infrequent Access, S3 One Zone-Infrequent Access, S3 intelligent-Tiering, S3 Glacier, and S3 Glacier Deep Archive
- An Amazon S3 lifecycle policy can delete or move objects to less expensive storage classes based on age
- Transferring data in from the internet to Amazon S3 is free, but transferring out to other Regions or to the internet incurs a fee

Section 4: Moving data to and from Amazon S3

Module 3: Adding a Storage Layer



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

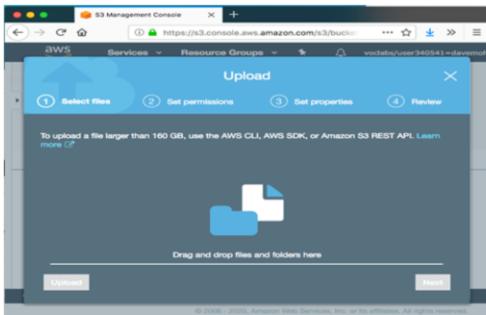
Introducing Section 4: Moving data to and from Amazon S3.

Moving objects to Amazon S3



AWS Management Console

Upload or download by using a browser.



AWS Command Line Interface

Upload or download from a terminal command prompt or in a call from a script.

- Example upload command:

```
$ aws s3 cp test.txt \  
s3://AWSDOC-EXAMPLE-BUCKET/test.txt
```



AWS Tools and SDKs

Move objects programmatically by using AWS tools or SDKs.



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

33

In the guided lab earlier in this module, you uploaded files to Amazon S3 by using the web browser interface provided by the AWS Management Console. It is the simplest way to move data in to or out of Amazon S3. It offers a wizard-based approach, including the option to drag and drop files that you want to copy into a bucket.

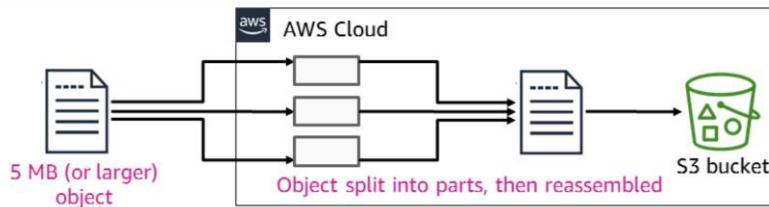
In this section of the module, you will learn about some additional options available for moving data into and out of Amazon S3.

Two of those options include using the AWS Command Line Interface (AWS CLI) or the AWS SDKs.

An example AWS CLI upload command is shown. In the command, you specify `aws` to invoke the AWS CLI, then you specify the service, which is `S3`. Next, you issue a `cp` (or copy) subcommand, followed by `test.txt` which is the local file (that exists on your computer) that should be copied. Finally, the `s3://AWSDOC-EXAMPLE-BUCKET/test.txt` parameter indicates the bucket where the file should be uploaded, and the key (`AWSDOC-EXAMPLE-BUCKET/test.txt`) where the object value (contents) should be stored.

The [S3 AWS CLI Command Reference](#) provides further details.

Multipart upload



- Files can be uploaded by using the Multipart Upload API
 - You can upload a single object as a set of parts
 - Each part is a contiguous portion of the object's data
 - After all parts of your object are uploaded, Amazon S3 assembles these parts and creates the object
- Typically only used for files larger than 100 MB
- Advantages –
 - Quick recovery from network issues: If transmission of any part fails, only need to retransmit that part
 - Ability to pause and resume object uploads
- aws • Improved throughput: Upload parts in parallel to improve throughput

© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

34

The Multipart Upload API enables you to consistently upload large objects in manageable parts.

Benefits include:

- Improved throughput – You can upload parts in parallel to improve throughput.
- Quick recovery from any network issues – Smaller part sizes minimize the impact of restarting a failed upload due to a network error.
- Ability to pause and resume object uploads – You can upload object parts over time. After you initiate a multipart upload, there is no expiration. You must explicitly complete or stop the multipart upload.
- Ability to begin an upload before you know the final object size – You can upload an object as you are creating it.
- Ability to upload large objects – By using the multipart upload API, you can upload large objects, up to 5 TB.

Note that files must be at least 5 MB in size to use the multipart upload feature.

Amazon S3 Transfer Acceleration

Standard S3 upload

versus

S3 Transfer Acceleration

- Accelerates Amazon S3 data transfers
- Uses optimized network protocols and the AWS edge infrastructure
- Typical speed improvement:
 - 50–500% for cross-country transfer of larger objects
 - Can go even higher under certain conditions
- Amazon S3 Transfer Acceleration Speed Comparison Tool*
 - Shows speed advantage gained (by Region)

* For more information, see <http://s3-accelerate-speedtest.s3-accelerate.amazonaws.com/en/accelerate-speed-comparsion.html>.

© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

35

Amazon S3 Transfer Acceleration enables fast and easy data transfer into an S3 bucket by taking advantage of Amazon CloudFront and AWS edge locations, which are globally distributed. This data is then routed to Amazon S3 over an optimized network path.

Use Transfer Acceleration when:

- You have customers all around the world who upload to a centralized bucket
- You transfer gigabytes or terabytes of data across continents on a regular basis
- You underutilize the available bandwidth when you upload files to Amazon S3 over the internet

Demonstration: S3 Transfer Acceleration



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

36

Now, the educator might choose to demonstrate the S3 Transfer Acceleration tool.

Moving large amounts of data into Amazon S3: AWS Snowball



AWS
Snowball



AWS Snowball

Petabyte-scale data transport

- Can transport multiple terabytes of data into or out of Amazon S3
 - Multiple devices can be used to transfer petabytes
- Addresses concerns of large data transfers (network costs, transfer times, security)
 - *Example:* To transfer 10 petabytes (10 million GB) over the internet with a 10 Gbps upload speed would take over 100 days
- To use –
 - Create a job in the AWS Management Console and a Snowball will be shipped to you.
 - Attach to your local network, then download and run the Snowball Client
 - Select the file directories to transfer (encrypted) to the device
 - Ship the device back and track the status



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

37

AWS Snowball is a petabyte-scale data transport option that doesn't require you to write any code or purchase any hardware to transfer your data. All you need to do is create a job in the AWS Management Console, and a Snowball appliance will be shipped to you. Attach the appliance to your local network and transfer files directly onto it. Then, ship it back and track the status of your shipment. When it arrives at the secure Amazon facility, the data will be transferred into your AWS account.

Moving large amounts of data into Amazon S3: AWS Snowmobile



AWS
Snowmobile



AWS Snowmobile
Exabyte-scale data transport

- A 45-foot-long (13.7 meters) shipping container, pulled by a semi-trailer truck
- Can transfer up to 100 PB per Snowmobile
- Offers multiple layers of security –
 - Dedicated security personnel
 - GPS tracking, alarm monitoring, 24/7 video surveillance
 - Optional escort security vehicle while in transit
 - Data encrypted with 256-bit encryption keys



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

38

AWS Snowmobile is an even larger data transfer option that operates in exabyte scale. An exabyte is 1 million terabytes or 1 billion gigabytes. It should only be used to move extremely large amounts of data into AWS. A Snowmobile is a 45-foot-long (13.7 meters) ruggedized shipping container that is pulled by a semi-trailer truck. You can transfer 100 PB per Snowmobile.

If you tried to transfer 100 petabytes of data over the internet, with an upload speed of 10 Gbps (assuming a TCP/IP overhead of 10%), it would take approximately 1018 days (almost three years) to finish uploading the data. That would not be practical. In such cases, using AWS Snowmobile to transfer the data would be a better option.

Snowmobile uses multiple layers of security designed to protect your data, including dedicated security personnel, GPS tracking, alarm monitoring, 24/7 video surveillance, and an optional escort security vehicle while in transit. All data is encrypted with 256-bit encryption keys managed through AWS Key Management Service (AWS KMS) and designed to ensure both security and full chain-of-custody of your data.

Section 4 key takeaways



- The [S3 multipart upload](#) option is a good option for files larger than 100 MB and in situations where network connectivity might be inconsistent
- [Amazon S3 Transfer Acceleration](#) uses edge locations and can significantly increase the speed of uploads
- [AWS Snowball](#) provides a way to transfer *petabytes* of data, and [AWS Snowmobile](#) provides a way to transfer *exabytes* of data to AWS

© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

39

Some key takeaways from this section of the module include:

- The S3 multipart upload option is a good option for files larger than 100 MB and in situations where network connectivity might be inconsistent
- Amazon S3 Transfer Acceleration uses edge locations, and can significantly increase the speed of uploads
- AWS Snowball provides a way to transfer petabytes of data, and AWS Snowmobile provides a way to transfer exabytes of data to AWS

Section 5: Choosing Regions for your architecture

Module 3: Adding a Storage Layer



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

Introducing Section 5: Choosing Regions for your architecture.

Choosing a Region: Compliance and latency considerations

- Data residency and regulatory compliance
 - Are there relevant Region **data privacy laws**?
 - Can customer data be stored **outside the country**?
 - Can you meet your **governance obligation**?
- Proximity of users to data
 - Small differences in **latency** can impact customer experience
 - Choose the Region closest to your users



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

41

There are many considerations when you decide what Region to host your data in.

First, you should consider *data privacy laws* and your regulatory compliance requirements. Data you store on AWS is subject to the laws of the country and locality where it is stored. In addition, some laws dictate that if you are operating your business in their jurisdiction, you cannot store that data anywhere else. Similarly, *compliance* standards (such as the U.S. Health Insurance Portability and Accountability Act, or HIPAA) have strict guidelines on how and where data can be stored.

Second, *proximity* is an important factor in choosing your Region, especially when *latency* is a critical factor. In most cases, the latency difference between using the closest Region and the farthest Region is relatively small, but even small differences in latency can impact customer experience. Customers expect responsive environments, and as time passes and technology becomes more and more powerful, those expectations also rise.

Choosing a Region: Service availability and cost considerations

• Service and feature availability

- Not all AWS services are available in all Regions
 - Consult the [AWS Region Table](#) for details
 - Services expand to new Regions regularly
- Can use some services cross-Region, but at increased latency



• Cost-effectiveness

- Costs vary by Region
- Some services like Amazon S3 have costs for transferring data out
- Consider the cost-effectiveness of replicating the entire environment in another Region



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

42

When you choose a Region, a third important consideration is the availability of AWS services and features. Though AWS strives to make services and features available everywhere, the complications that arise from having a global reach make it challenging to accomplish that goal. Instead of waiting until a service is available everywhere before launching it, services are released when they are ready. Service availability is then expanded as soon as possible.

A fourth consideration when you choose a Region is cost. Service costs can differ depending on which Region they are used in. For example, an Amazon EC2 instance in the us-east-1 Region might not cost the same as if it ran in the eu-west-1 Region. Typically, the difference in cost might not be enough to supersede the other three considerations. However, in cases where the latency, compliance, and service availability differences between Regions are minimal, you might be able to save by using the lower-cost Region for your environment.

Finally, in circumstances where your customers are in different areas of the world, consider optimizing their experience by replicating your environment in multiple Regions that are closer to them. Because you would then be distributing your load across multiple environments, your costs for components in each environment might go down even as you add more infrastructure. For example, adding a second application environment might allow you to cut your processing and storage capacity requirements in half in each environment. Because AWS is designed to enable that kind of flexibility, and because you only pay for what you use, you could scale your existing environment down as a way to mitigate the cost of adding another environment.

The downside to that approach is that you now have two environments to manage. Also, not all of your components will scale down enough to mitigate all the costs of the new components.

Additionally, you might need to maintain one single storage source of truth in one Region, such as a primary Amazon Relational Database Service (Amazon RDS) instance. Your secondary Region would need to communicate with the storage instance, which might increase latency and cost for those operations.

Module 3 – Challenge Lab: Creating a Static Website for the Café



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

43

You will now complete Module 3 – Challenge Lab: Creating a Static Website for the Café.

The business need: A simple website

Sofía mentioned to Nikhil that she wants a website that will showcase the café visually through images, and provide customers with business details.



Frank likes the website idea. He has been taking photos that can be used to highlight the café's menu items.



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

44

Sofía mentioned to Nikhil that she would like the café to have a website that will showcase the café visually through images. The website should also provide customers with business details, such as the location of the store, the business hours, and telephone number.

Nikhil is pleased to create the first website for the café. During this activity, you will take on the role of Nikhil and work on producing the results that everyone back at the café hopes you can deliver. Perhaps you can even exceed their expectations!

Challenge lab: Tasks

1. Extracting the files that you need for the lab
2. Creating an S3 bucket to host your static website
3. Uploading content to your S3 bucket
4. Creating a bucket policy to grant public read access
5. Enabling versioning on the S3 bucket
6. Setting lifecycle policies
7. Enable cross-Region replication



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

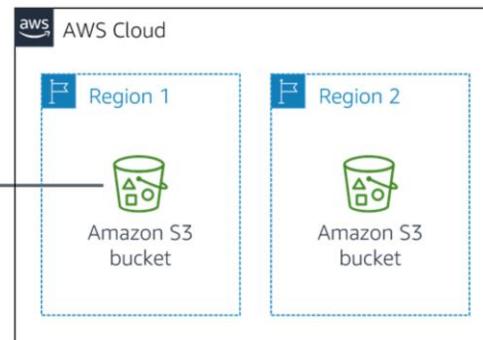
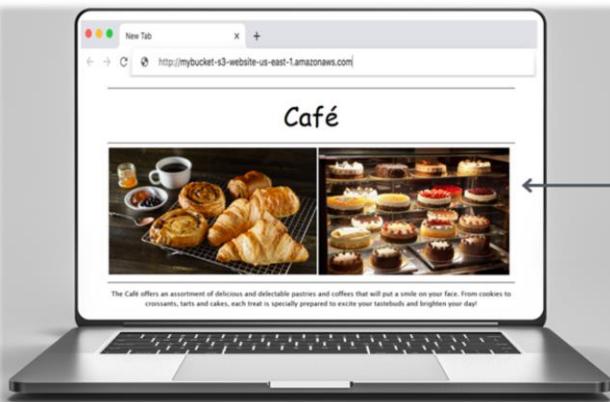
45

In this challenge lab, you will complete the following tasks:

1. Extracting the files that you need for the lab
2. Creating an S3 bucket to host your static website
3. Uploading content to your S3 bucket
4. Creating a bucket policy to grant public read access
5. Enabling versioning on the S3 bucket
6. Setting lifecycle policies
7. Enable cross-Region replication

Challenge lab: Final product

<http://<bucket-name>.s3-website-<region>.amazonaws.com>



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

46

In this challenge lab, you will create a static website for the café. The website will be hosted on Amazon S3. After the S3 bucket is created and properly configured for website hosting, a web browser should be able access the website directly by using the assigned Amazon S3 endpoint URL.

For accessibility: Architecture diagram showing two Regions with an S3 bucket in each Region. One S3 bucket points to the cafe's website. <http://<bucket-name>.s3-website-<region>.amazonaws.com>. **End of accessibility description.**



~ 60 minutes

Begin Module 3 – Challenge Lab: Creating a Static Website for the Café

aws

© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

47

It is now time to start the challenge lab.

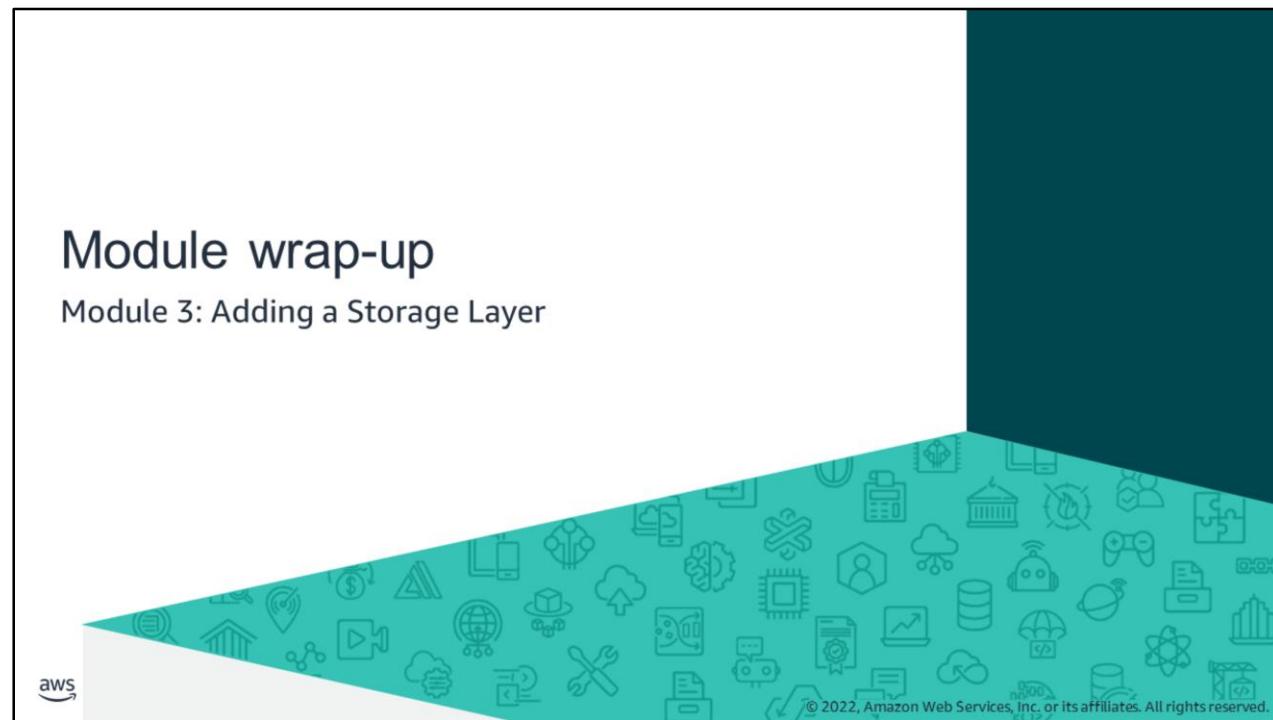
Challenge lab debrief: Key takeaways



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

48

Your educator might now choose to lead a conversation about the key takeaways from the challenge lab now after you have completed it.



It's now time to review the module and wrap up with a knowledge check and discussion of a practice certification exam question.

Module summary

In summary, in this module, you learned how to:

- Recognize the problems that Amazon Simple Storage Service (Amazon S3) can solve
- Describe how to store content efficiently using Amazon S3
- Recognize the various Amazon S3 storage classes and cost considerations
- Describe how to move data to and from Amazon S3
- Describe how to choose a Region
- Create a static website



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

50

In summary, in this module, you learned how to:

- Recognize the problems that Amazon Simple Storage Service (Amazon S3) can solve
- Describe how to store content efficiently using Amazon S3
- Recognize the various Amazon S3 storage classes and cost considerations
- Describe how to move data to and from Amazon S3
- Describe how to choose a Region
- Create a static website

Complete the knowledge check



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

51

It is now time to complete the knowledge check for this module.



Sample exam question

Company salespeople upload their sales figures daily. A Solutions Architect needs a durable storage solution for these documents that also protects against users accidentally deleting important documents.

Which action will protect against unintended user actions?

Choice	Response
A	Store data in an EBS volume and create snapshots once a week.
B	Store data in an S3 bucket and enable versioning.
C	Store data in two S3 buckets in different AWS Regions.
D	Store data on EC2 instance storage.

© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

52

Look at the answer choices and rule them out based on the keywords.

Sample exam question answer



Company salespeople upload their sales figures daily. A Solutions Architect needs a durable storage solution for these documents that also protects against users accidentally deleting important documents.

Which action will protect against unintended user actions?

The correct answer is B.

The keywords in the question are durable storage solution, protects against users accidentally deleting, and which action will protect.

© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

53

The following are the keywords to recognize: durable storage solution, protects against users accidentally deleting, and which action will protect.

The correct answer is C. If a versioned object is deleted, then it can still be recovered by retrieving the final version.

Incorrect answers:

Response A would lose any changes that were committed since the previous snapshot. Storing the data in two S3 buckets (response C) would provide slightly more protection, but a user could still delete the object from both buckets. EC2 instance storage (response D) is temporary storage and should never be used for data requiring durability.

Additional resources

- [Amazon S3 Developers Guide](#)
- [Amazon S3 FAQs](#)
- [Amazon S3 Common Use Scenarios](#)
- [AWS Storage Services Whitepaper](#)
- [Amazon S3 Storage Classes Comparison](#)
- [Amazon S3 Block Public Access](#)



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

54

If you want to learn more about the topics covered in this module, you might find the following additional resources helpful:

- [Amazon S3 Developers Guide](#)
- [Amazon S3 FAQs](#)
- [Amazon S3 Common Use Scenarios](#)
- [AWS Storage Services Whitepaper](#)
- [Amazon S3 Storage Classes Comparison](#)
- [Amazon S3 Block Public Access](#)



Thank you

Corrections, feedback, or other questions?
Contact us at <https://support.aws.amazon.com/#/contacts/aws-academy>.
All trademarks are the property of their owners.

© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

55

Thank you for completing this module.



**AWS Academy Cloud Architecting
Module 04 Student Guide
Version 2.0.8**

200-ACACAD-20-EN-SG

© 2022 Amazon Web Services, Inc. or its affiliates. All rights reserved.

This work may not be reproduced or redistributed, in whole or in part, without prior written permission from Amazon Web Services, Inc. Commercial copying, lending, or selling is prohibited.

All trademarks are the property of their owners.

Contents

Module 4: Adding a Compute Layer	4
----------------------------------	---



Module 4: Adding a Compute Layer

AWS Academy Cloud Architecting

© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

Welcome to Module 4: Adding a Compute Layer.

Module overview

Sections

1. Architectural need
2. Adding compute with Amazon EC2
3. Choosing an AMI to launch an Amazon EC2 instance
4. Selecting an Amazon EC2 instance type
5. Using user data to configure an Amazon EC2 instance
6. Adding storage to an Amazon EC2 instance
7. Amazon EC2 pricing options
8. Amazon EC2 considerations

Demonstrations

- Configuring an EC2 Instance with User Data
- Reviewing the Spot Instance History Page

Labs

- Guided Lab: Introducing Amazon EFS
- Challenge Lab: Creating a Dynamic Website for the Café



Knowledge check



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

2

This module contains the following sections:

1. Architectural need
2. Adding compute with Amazon EC2
3. Selecting an AMI to launch an EC2 instance
4. Selecting an EC2 instance type
5. Using user data to initialize an EC2 instance
6. Configuring storage for an EC2 instance
7. Amazon EC2 pricing options
8. Amazon EC2 considerations

This module also includes:

- A demonstration that will show you how to launch an Amazon Elastic Compute Cloud (Amazon EC2) instance with user data that installs a web server on the instance.
- A demonstration of the Spot Instance Pricing History page.
- A hands-on guided lab where you create a file storage system with Amazon Elastic Filesystem (Amazon EFS) and mount it on an EC2 instance.
- A hands-on challenge lab where you will launch an EC2 instance with a web application that uses a database for the Café use case.

Finally, you will be asked to complete a knowledge check that will test your understanding of key concepts covered in this module.

Module objectives

At the end of this module, you should be able to:

- Identify how Amazon Elastic Compute Cloud (Amazon EC2) can be used in an architecture
- Explain the value of using Amazon Machine Images (AMIs) to accelerate the creation and repeatability of infrastructure
- Differentiate between the EC2 instance types
- Recognize how to configure Amazon EC2 instances with user data
- Recognize storage solutions for Amazon EC2
- Describe EC2 pricing options
- Determine the placement group given an architectural consideration
- Launch an Amazon EC2 instance



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

3

At the end of this module, you should be able to:

- Identify how Amazon Elastic Compute Cloud (Amazon EC2) can be used in an architecture
- Explain the value of using Amazon Machine Images (AMIs) to accelerate the creation and repeatability of infrastructure
- Differentiate between the EC2 instance types
- Recognize how to launch Amazon EC2 instances with user data
- Recognize storage solutions for Amazon EC2
- Describe EC2 pricing options
- Determine the placement group given an architectural consideration
- Launch an Amazon EC2 instance

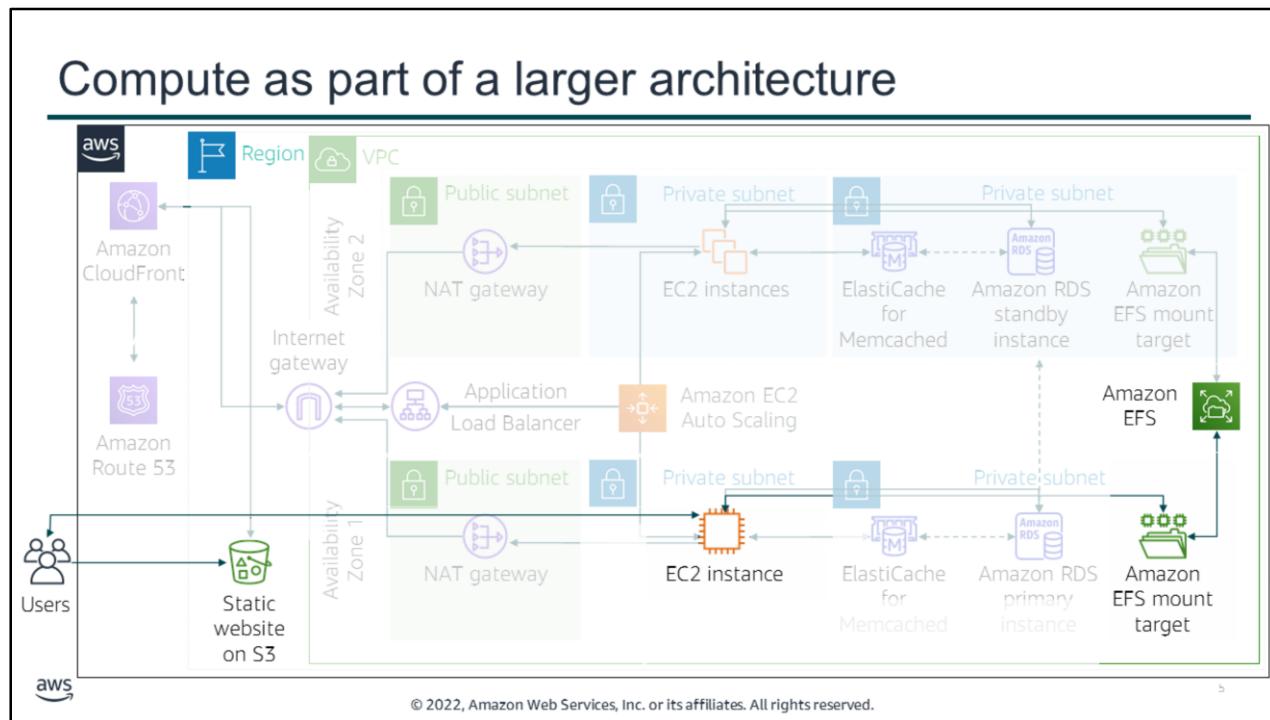
Section 1: Architectural need

Module 4: Adding a Compute Layer



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

Introducing Section 1: Architectural need.



As each module introduces new features, those parts of this larger diagram will be unveiled.

In this module, you will learn how to create an architecture for running a dynamic web application on AWS by using Amazon EC2. You also learn about some AWS storage services that can be used with Amazon EC2, including Amazon EFS.

Café business requirement

The café wants the website to display more than static content and to provide dynamic capabilities. They want to introduce online ordering for customers, and enable café staff to view submitted orders.



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

6

After the café launched the first version of their website, the café's customers told Sofía and Nikhil how nice the website looks. They liked that they could easily look up the business hours online, and browse through the available desserts before coming in to pick up food and drinks for their work colleagues. However, in addition to the praise, customers often asked Sofía and Nikhil whether they could place online orders that they could schedule for pickup.

Sofía, Nikhil, Frank, and Martha discussed the situation. They agreed that their business strategy and decisions should focus on delighting their customers and providing them with the best possible cafe experience. They want to improve their services so they can increase customer satisfaction, reduce customer wait times, and make ordering more convenient for customers who are in a hurry.

To meet these objectives, the café wants the website to display more than static content. They want to introduce online ordering for customers, and enable café staff to view submitted orders. Their current website architecture, where the website is hosted on Amazon S3, will not support the new business requirements.

Throughout this module, you will learn details about the capabilities of Amazon EC2, and how you could use it to successfully meet these new business requirements.

Section 2: Adding compute with Amazon EC2

Module 4: Adding a Compute Layer



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

Introducing Section 2: Adding compute with Amazon EC2.

AWS runtime compute choices

Virtual Machines (VMs)	Containers	Platform as a Service (PaaS)	Serverless	Specialized Solutions
 Amazon Elastic Compute Cloud (Amazon EC2)	 Amazon Elastic Container Service (Amazon ECS)	 AWS Elastic Beanstalk	 AWS Lambda	 AWS Outposts
 Amazon Lightsail			 AWS Fargate	 AWS Batch

Higher infrastructure control and customization Faster application deployment Fully managed services

Different compute services are available to meet the needs of different use cases.
This module will discuss Amazon EC2.

© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

AWS offers several compute options to meet different needs. As you design the architecture to support a given type of workload, it is important that you understand the available compute options. As the diagram shows, the key runtime compute choices can be grouped into four cloud compute model categories: *virtual machines (VMs)*; *containers*; *platform as a service*, which is also known as *PaaS*; and *serverless*. In addition, you can use *specialized solutions* to address specific compute use cases.

In the *virtual machines* category, AWS offers two core services. The first service is *Amazon Elastic Compute Cloud (Amazon EC2)*. It provides secure and resizable virtual servers in the cloud. The second service is *Amazon Lightsail*. It provides virtual private servers to run simple workloads in a cost-effective way.

In the *containers* category, AWS offers *Amazon Elastic Container Service (Amazon ECS)*. It enables you to run Docker container applications on AWS.

The *PaaS* category includes *AWS Elastic Beanstalk*. It is a solution that runs web applications and services that are developed in languages such as Java, .NET, PHP, Node.js, Python, Ruby, Go, and Docker.

The *serverless* category includes *AWS Lambda*, which is a serverless compute solution that runs Java, Go, PowerShell, Node.js, C#, Python, or Ruby code. This category also includes *AWS Fargate*, which provides a serverless compute platform for containers.

For *specialized solutions*, *AWS Outposts* provides a way to run AWS infrastructure and services on-premises, and *AWS Batch* is a service that runs batch jobs at any scale.

When you select an AWS compute runtime for your workload, consider that virtual machines and container-based services provide more control over your infrastructure and enable higher degrees of customization. PaaS and serverless services enable you to focus more on your application and less on infrastructure. They also enable quick deployment. The services in the specialized solutions category address specific types of workloads, or *hybrid cloud* and *batch*. These specialized services work well for these use cases because they are also fully managed by AWS. In this module, the focus will be on Amazon EC2.

Amazon EC2



Amazon Elastic Compute Cloud (Amazon EC2)

Amazon EC2 provides resizable compute capacity in the cloud.

- Provides virtual machines (servers)
- Provisions servers in minutes
- Can automatically scale capacity up or down as needed
- Enables you to pay only for the capacity that you use



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

9

Amazon Elastic Compute Cloud (Amazon EC2) enables computing in the cloud. You can use Amazon EC2 to provision virtual servers, and you can completely control the computing resources of those servers. You can obtain and start new server instances in minutes. You can quickly scale capacity both up and down as your computing requirements change. From a cost perspective, you pay only for the capacity that you use.

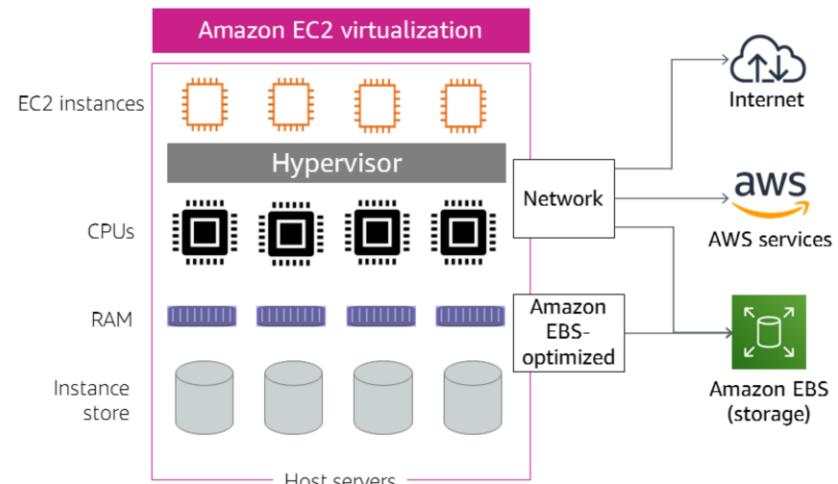
Why is it called *Elastic Compute Cloud*?

- *Elastic* because you can easily increase or decrease the number of servers you run to support an application automatically. You can also increase or decrease the size of existing servers
- *Compute* because most users run servers to host running applications or process data, which require compute resources. These resources include processing power (CPU) and memory (RAM)
- *Cloud* because the EC2 instances that you run are hosted in the cloud

EC2 instances

An EC2 instance is a **virtual machine** that runs on a physical host.

- You can choose different configurations of CPU and memory capacity
- Supports different storage options
 - Instance store
 - Amazon Elastic Block Store (Amazon EBS)
- Provides network connectivity



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

10

Amazon EC2 *instances* run as virtual machines on host computers that are located in AWS Availability Zones. Each virtual machine runs an operating system (OS), such as Amazon Linux or Microsoft Windows. You can install and run applications on the OS in each virtual machine. You can even run enterprise applications that span multiple virtual machines.

The virtual machines run on top of a *hypervisor* layer that is maintained by AWS. The hypervisor is the operating platform layer that provides an EC2 instance with access to the actual physical hardware resources that it needs to run, such as *processors*, *memory*, and *storage*.

Some EC2 instances use an *instance store*. The *instance store* is also known as *ephemeral storage*. It is storage that is physically attached to the host computer and provides temporary block-level storage to an instance.

Many EC2 instances use *Amazon Elastic Block Store (Amazon EBS)* for the boot disk and other storage needs. Amazon EBS provides persistent block storage volumes, which mean that the data will be persisted. For example, the data persists on that instance even when the EC2 instance is in a stopped state.

EBS-optimized instances provide faster access to an attached Amazon EBS volume by minimizing the I/O contention between the volume and other traffic from the instance.

EC2 instances can have *network* connectivity to other resources, such as other EC2 instances, AWS services, and the internet. You can configure the degree of network access to suit your needs and to balance accessibility needs with security requirements. Different instance types provide different levels of network performance.

Amazon EC2 use cases

Use Amazon EC2 when you need:

- Complete control of your computing resources, including *operating system* and *processor type*
- Options for optimizing your compute costs –
 - *On-Demand Instances, Reserved Instances, and Spot Instances*
 - *Savings Plans*
- Ability to run any type of workload, for example –
 - Simple websites
 - Enterprise applications
 - High performance computing (HPC) applications



Amazon
EC2



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

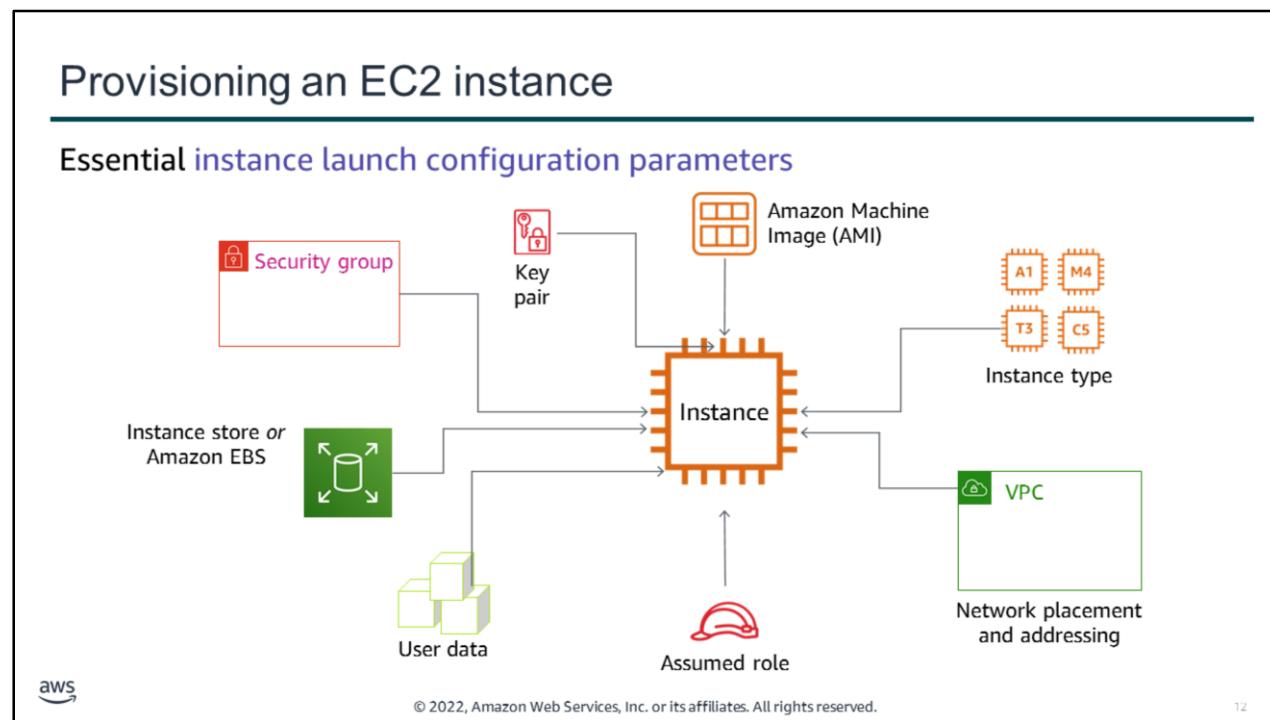
11

Amazon EC2 provides virtual machines where you can host the same kinds of applications that you might run on a traditional on-premises server. Common uses for EC2 instances include web servers, application servers, database servers, and media servers.

In particular, consider Amazon EC2 as a compute choice in situations where you need:

- Complete control of your computing resources – Amazon EC2 enables you to set up and configure everything about your instances, from your operating system to your applications. For example, you can use various operating systems that include *Microsoft Windows* and many variants of *Linux*. In addition, you can choose instances with either *x86* or *Advanced RISC Machine (ARM)* processor architecture.
- Options for optimizing your compute costs – Amazon EC2 offers several ways to pay for EC2 instances, including On-Demand Instances, Savings Plans, Reserved Instances, and Spot Instances. You can also pay for Dedicated Hosts, which provide you with EC2 instance capacity on physical servers that are dedicated for your use.
- A virtual server to run any type of workload.

You will learn more details about Amazon EC2 computing resource choices and pricing options in later sections of this module.



To provision an EC2 instance, you must make key decisions regarding its configuration details.

The main parameters that you must specify to launch a secure instance include:

- *Amazon Machine Image (AMI)* – An AMI defines the base software configuration for an instance and is used by Amazon EC2 to launch the instance.
- *Instance type* – An instance type defines a combination of CPU, memory, storage, and networking capacity that provides a certain level of compute capability to run an application.
- *Network placement and addressing* – When you launch an instance, you can specify the appropriate network placement and addressing to provide the network access and security that you want.
- *Assumed role* – Optionally, you can attach an AWS Identity and Access Management (IAM) role, which grants permissions for accessing AWS services to applications that run on the instance or users that are connected to the instance.
- *User data* – You can further initialize or customize instance configuration by specifying a user data script. This script automatically runs when the instance is launched.
- *Storage* – You must specify the type of storage that will be used to store the root or boot volume of the instance.
- *Security group* – You must also configure a new security group or use an existing one. The security group defines which ports network traffic is allowed on.
- *Key pair* – A key pair is typically also specified at launch. The key pair is used for Secure Shell (SSH) connections or for establishing Remote Desktop Protocol (RDP) access to the instance.

Section 2 key takeaways



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

13

- Amazon EC2 enables you to run Microsoft Windows and Linux virtual machines in the cloud.
- You can use an EC2 instance when you need complete control of your computing resources and want to run any type of workload.
- When you launch an EC2 instance, you must choose an AMI and an instance type. Launching an instance involves specifying configuration parameters, including network, security, storage, and user data settings.

Some key takeaways from this section of the module include:

- Amazon EC2 enables you to run Microsoft Windows and Linux virtual machines in the cloud.
- You can use an EC2 instance when you need complete control of your computing resources and want to run any type of workload.
- When you launch an EC2 instance, you must choose an AMI and an instance type. Launching an instance involves specifying configuration parameters, including network, security, storage, and user data settings.

Section 3: Choosing an AMI to launch an EC2 instance

Module 4: Adding a Compute Layer



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

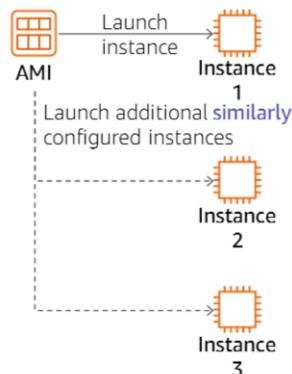
Introducing Section 3: Choosing an AMI to launch an EC2 instance.

Amazon Machine Image (AMI)

An **AMI** provides the information that is needed to launch an instance, including:

- A **template** for the root volume
 - Contains the guest operating system (OS) and perhaps other installed software
- **Launch permissions**
 - Control which AWS accounts can access the AMI
- **Block device mappings**
 - Specifies any storage volumes to attach to the instance

Create multiple instances from the same AMI



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

15

An AMI provides the information that is needed to launch an instance. You must specify a source AMI when you launch an instance. The AMI includes a *template* for the root volume of the instance, *launch permissions*, and *block device mappings*.

A root volume typically contains an operating system (OS) and everything that has been installed in that OS (applications, libraries, utilities, and others). Amazon EC2 copies the template to the root volume of the new instance and then starts it.

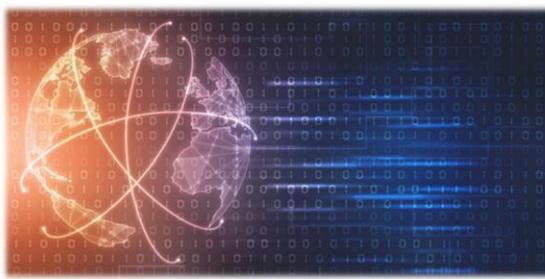
The *launch permissions* control which AWS accounts can use the AMI to launch instances. They also enable you to make the AMI available to the public.

The *block device mappings* specify additional storage volumes (if any) to attach to the instance when it is launched.

You can launch multiple instances from a single AMI when you need multiple instances that have the same configuration.

You can also use different AMIs to launch instances when you need instances with different configurations. For example, you might have one AMI to implement web server instances in your architecture, and another to implement application server instances.

AMI benefits



- **Repeatability**

- An AMI can be used repeatedly to launch instances with efficiency and precision

- **Reusability**

- Instances launched from the same AMI are identically configured

- **Recoverability**

- You can create an AMI from a configured instance as a restorable backup
- You can replace a failed instance by launching a new instance from the same AMI



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

16

The benefits of using an AMI include *repeatability*, *reusability*, and *recoverability*.

AMIs enable *repeatability* because an AMI packages the full configuration and content of an EC2 instance. As such, it can be used repeatedly to launch multiple instances with efficiency and precision.

AMIs promote *reusability* because instances that are launched from the same AMI are exact replicas of each other. This design makes it easier to build clusters of similar instances or recreate compute environments.

AMIs also facilitate *recoverability*. If an instance fails, you can replace it by launching a new instance from the same AMI that you used to launch the original instance. In addition, AMIs provide a way to back up a complete EC2 instance configuration, which you can use to launch a replacement instance if there is a failure.

Choosing an AMI

Choose an AMI based on:

- Region
- Operating system
 - Microsoft Windows or Linux
- Storage type of the root device
- Architecture
- Virtualization type



AMI sources:

- [Quick Start – Linux and Microsoft Windows AMIs that are provided by AWS.](#)
- [My AMIs – Any AMIs that you create.](#)
- [AWS Marketplace – Pre-configured templates from third parties.](#)
- [Community AMIs – AMIs shared by others. Use at your own risk.](#)



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

17

When you choose an AMI to launch an instance, your decision should be based on five key characteristics:

- *Region* – Each AMI exists in a specific Region. Therefore, you must select an AMI that is in the Region where you want the instance to run. You can copy an AMI from one Region to another as needed.
- *Operating system* – For an AMI that is provided by AWS, you can choose between Microsoft Windows or a variant of Linux.
- *Storage for the root device* – All AMIs are categorized as either *Amazon EBS-backed* or *instance store-backed*. The data on an instance store volume persists only during the lifetime of the instance, but the data on an EBS volume persists independently of the life of the instance.
- *Architecture* – This characteristic determines the type of processor architecture that best fits your workload. The choices are 32-bit or 64-bit, and either an x86 or Advanced RISC Machine (ARM) instruction set.
- *Virtualization type* – AMIs use one of two types of virtualization: paravirtual (PV) or Hardware Virtual Machine (HVM). The main differences between PV and HVM AMIs include how they boot and whether they can take advantage of special hardware extensions for better performance. For best performance, use an AMI with *HVM virtualization type*.

You can obtain an AMI from one of four sources:

- Quick Starts are AMIs that are built by AWS. They offer the choice of either *Microsoft Windows* or variants of the *Linux* operating system. The Linux options include: Amazon Linux, Ubuntu, Red Hat Enterprise Linux, SUSE Linux Enterprise Server, Fedora, Debian, CentOS, Gentoo Linux, Oracle Linux, and FreeBSD.
- AWS also enables you to create your own AMIs (**My AMIs**). You can create an AMI from an EC2 instance.

- You can also look for AMIs in the *AWS Marketplace*, which offers a digital catalog that lists thousands of software solutions. These solutions include AMIs from software vendors for specific use cases.
- *Community-built* AMIs are created by people from around the world, and they can offer solutions to many different types of problems. However, they are not vetted by AWS. Therefore, use them at your own risk. In particular, avoid using them in any production or corporate environment.

Instance store-backed versus Amazon EBS-backed AMI

Characteristic	Amazon EBS-Backed Instance	Instance Store-Backed Instance
Boot time for the instance	Boots faster	Takes longer to boot
Maximum size of root device	16 TiB	10 GiB
Ability to stop the instance	Can stop the instance	Can't stop the instance, only reboot or terminate it
Ability to change the instance type	Can change the instance type by stopping instance	Can't change the instance type because the instance can't be stopped
Instance charges	You are charged for instance usage, EBS volume usage, and storing your AMI as an EBS snapshot	You are charged for instance usage and storing your AMI in Amazon S3

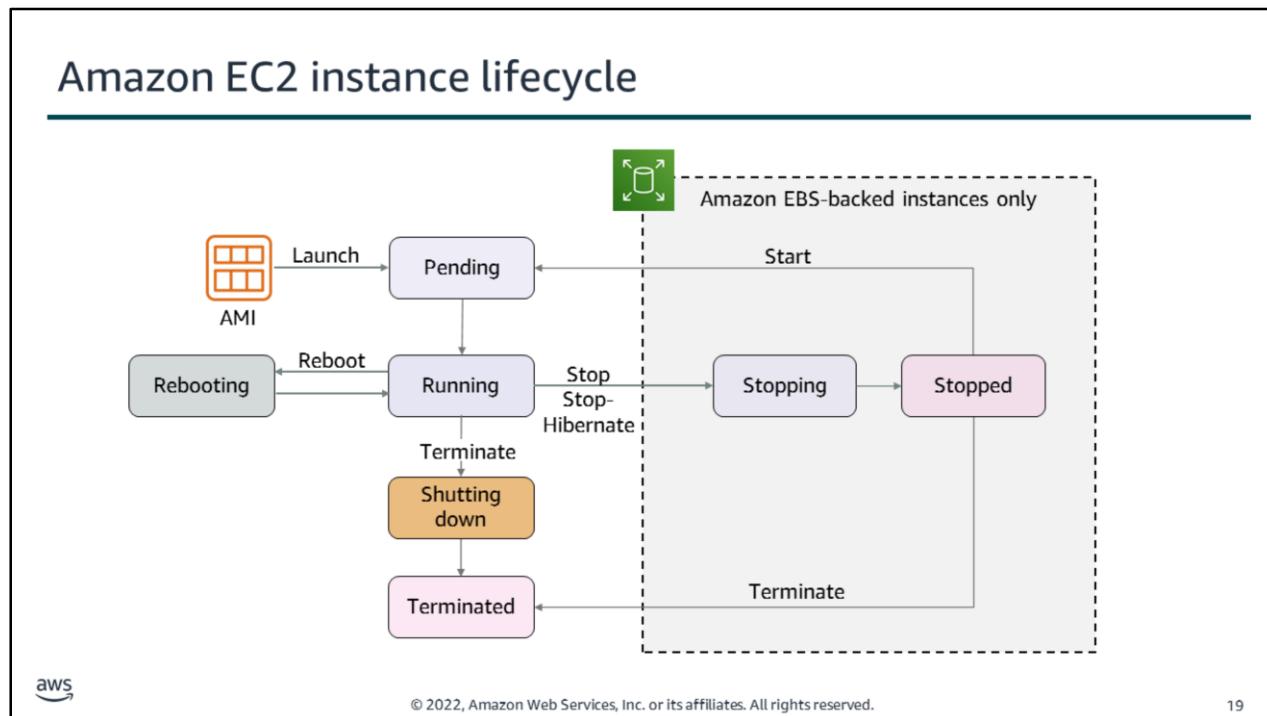


© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

18

There are important behavior and performance differences when you launch an instance by using an *instance store-backed* AMI versus an *Amazon EBS-backed* AMI. They are as follows:

- Amazon EC2 instances that use instance storage take longer to boot than Amazon EC2 instances that use Amazon EBS because all the image parts have to be retrieved from Amazon Simple Storage Service (Amazon S3).
- The maximum capacity of the root device of an instance that is backed by Amazon EBS is 16 tebibytes (TiB) and is greater than that of an instance store-backed instance, which is 10 gibibytes (GiB).
- You cannot stop an instance store-backed instance, only reboot or terminate it.
- You cannot change the instance type of an instance store-backed instance.
- The cost for an Amazon EBS-backed instance includes EBS storage charges. The cost of an instance store-backed instance includes Amazon S3 storage charges. Amazon S3 storage costs are usually cheaper.



This diagram shows the lifecycle of an instance and highlights the extra actions and states that an Amazon EBS-backed instance allows.

When an instance is first launched from an AMI, or when you start a stopped instance, it enters the *pending* state. This state indicates that the instance is being provisioned on a host computer and is booting. The instance type that is specified in the AMI, or for the original stopped instance, determines the hardware of the host computer for the new instance.

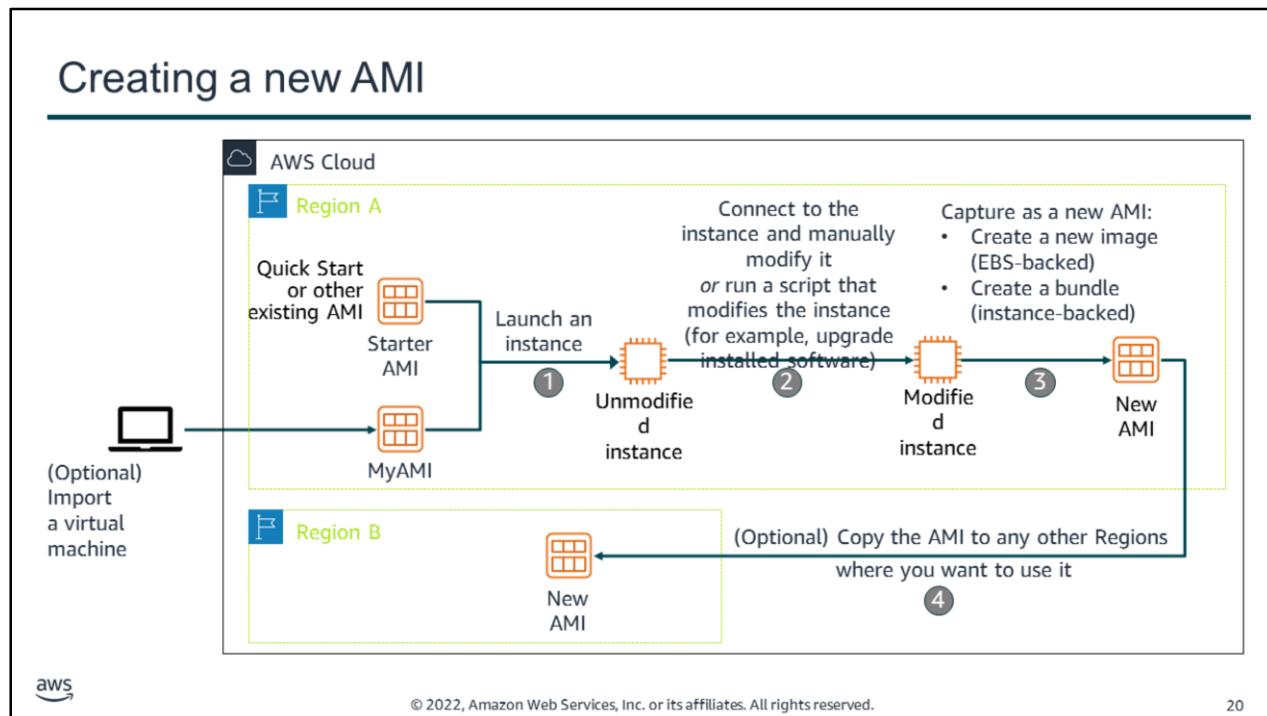
When the instance is fully booted and ready, it exits the *pending* state and enters the *running* state. At this point, you can connect to your running instance over the internet and start to use it.

As soon as an instance is in a *running* state, it can be rebooted by using the Amazon EC2 console, the AWS Command Line Interface (AWS CLI), or AWS SDKs. It then moves to a *rebooting* state. A rebooted instance stays on the same physical host, and it maintains the same public Domain Name System (DNS) name and public IP address. If the instance has *instance store* volumes, it retains the data on those volumes.

From the *running* state, you can also terminate an instance. Terminating an instance causes the instance to go into an intermediary *shutting down* state before it displays a *terminated* state. A terminated instance remains visible in the Amazon EC2 console for some time before the virtual machine is deleted. However, you can't connect to or recover a terminated instance.

Instances that are backed by Amazon EBS can also be stopped from a *running* state. They enter the *stopping* state before they attain the fully *stopped* state. A *stopped* instance does not incur the same cost as a *running* instance. Starting a *stopped* instance puts it back into the *pending* state, which moves the instance to a new host machine.

You can also hibernate an EBS-backed instance from the *running* state. Doing so causes the in-memory storage, private IP address, and Elastic IP address to be saved. They remain the same when you start the instance again, so you can pick up where you left off. In most cases, when you start a hibernated instance, it is moved to a new host computer. However, your instance might stay on the same host computer if the host computer has no problems.



You create an AMI from an EC2 instance. You start with a source AMI, which can be an AMI that already exists—such as a Quick Start AMI that is provided by AWS. Or, you can start with an AMI that you build from a virtual machine that you import. You then launch an EC2 instance from this AMI (step 1).

Regardless of the option you choose (step 1), you will have what the diagram refers to as *an unmodified instance*. From that instance, you might then create a *golden instance*. In other words, create a virtual machine that you configure with the specific OS and application settings that you want (step 2). Then, capture it as a new AMI (step 3).

For an EBS-backed AMI, you capture the new AMI by creating a *new image*, and AWS automatically registers it for you. For an instance-backed AMI, you capture the new AMI by using Amazon EC2 AMI tools to create a *bundle* for the instance root volume, and upload the bundle to an Amazon S3 bucket. You then must manually register the new AMI.

After an AMI is registered, the AMI can be used to launch new instances in the same AWS Region. You can now think of the new AMI as a new starter AMI. Optionally, you might want to also copy the AMI to other Regions (step 4), so that you can also launch new EC2 instances in those locations.

EC2 Image Builder



EC2 Image Builder automates the [creation](#), [management](#), and [deployment](#) of up-to-date and compliant [golden VM images](#).

- Provides a graphical interface to create image-building pipelines
- Creates and maintains [Amazon EC2 AMIs](#) and on-premises VM images
- Produces secure, validated, and up-to-date images
- Enforces version control



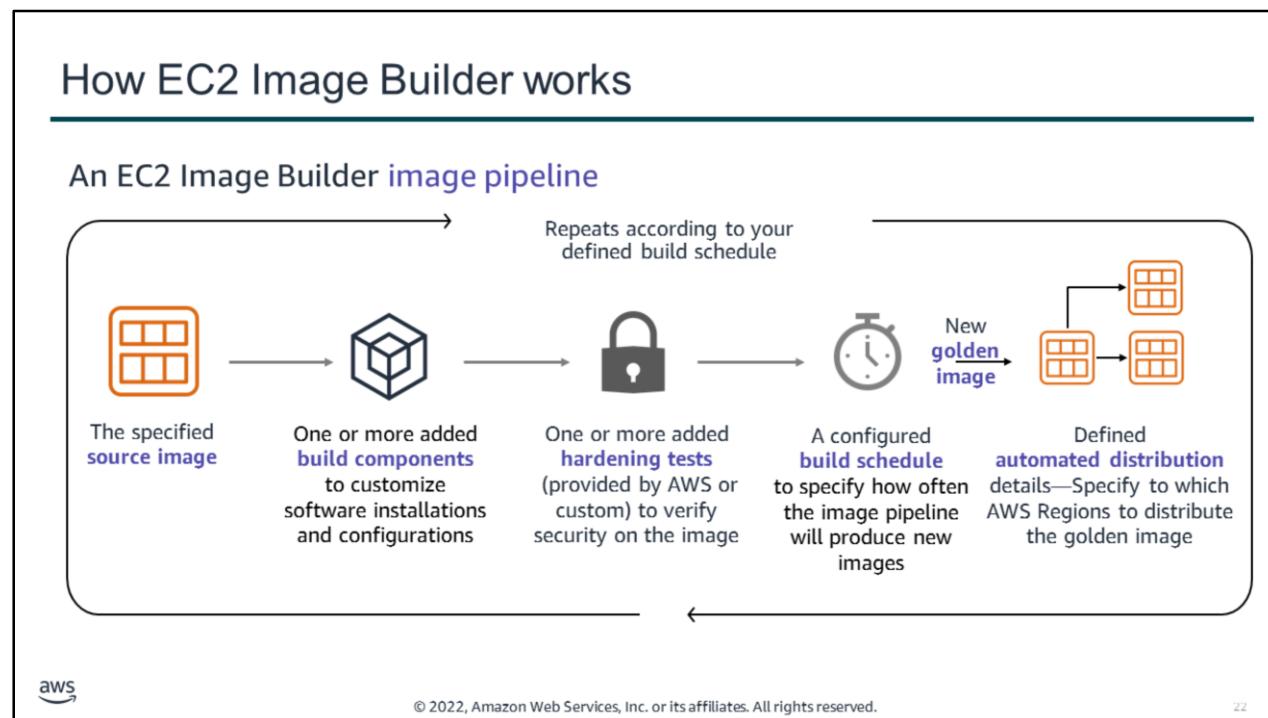
© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

21

Another way to create an AMI is to use EC2 Image Builder.

EC2 Image Builder is an AWS service that simplifies the creation, maintenance, validation, sharing, and deployment of Linux or Microsoft Windows images. It provides a simple graphical interface to produce AMIs for use on AWS and to generate VM images for use on premises.

One of the benefits of using EC2 Image Builder is that it enables you to create images with only the essential components, which can reduce your exposure to security vulnerabilities. Before you use your images in production, you can use EC2 Image Builder to validate your images with tests that are provided by AWS or your own tests. Finally, it provides version control for revision management.



You can create an *image pipeline* with EC2 Image Builder by using the AWS Management Console, AWS CLI, or application programming interfaces (APIs). When you use the AWS Management Console, the step-by-step wizard includes the following steps:

- Step 1: Identify a *source image* to build your new image from (for example, choose Amazon Linux 2 or Windows Server 2019).
- Step 2: Select *build components* software for installation or customization. You can choose from build components that are provided by Amazon (such as a component that installs Python 3), components that you have built, or components that have been shared with you.
- Step 3: Select *hardening tests* that should be run each time the image pipeline runs, so that the image passes security checks. It also enables you to catch incompatibilities that were introduced by OS updates before deployment to AWS Regions. You can run both tests that were provided by AWS and your own tests. An example of an AWS test is to test whether an AMI can run a sample application.
- Step 4: Specify a build schedule to identify how often this pipeline should run, and when.
- Step 5: Define whether you want the image to be distributed to selected Regions.

Image Builder outputs a server image that is in one of the following supported formats: AMI, VHDX, VMDK, and OVF. You can configure the image that you build to be patched on an ongoing basis. You can also monitor build progress and have Amazon EventBridge (formerly known as Amazon CloudWatch Events) notify you for troubleshooting and debugging.

Section 3 key takeaways



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

23

- An **AMI** provides the information that is needed to launch an EC2 instance
- For best performance, use an AMI with **HVM virtualization type**
- Only an instance launched from an Amazon EBS-backed AMI **can be stopped and started**
- An AMI is available in a **Region**

Some key takeaways from this section of the module include:

- An **AMI** provides the information that is needed to launch an EC2 instance
- For best performance, use an AMI with **HVM virtualization type**
- Only an instance launched from an Amazon EBS-backed AMI **can be stopped and started**
- An AMI is available in a **Region**

Section 4: Selecting an EC2 instance type

Module 4: Adding a Compute Layer

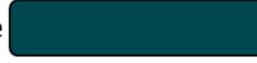


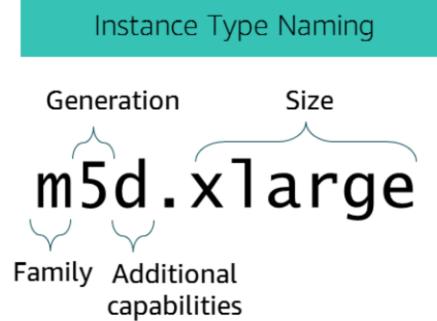
© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

Introducing Section 4: Selecting an EC2 instance type.

EC2 instance type

An [EC2 instance type](#) defines a configuration of CPU, memory, storage, and network performance characteristics that provides a given level of compute performance.

vCPU		4
Memory		16 GiB
Storage		1 x 150 NVMe SSD
Network performance		Up to 10 Gbps



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

25

Before you can launch an instance, you must select an instance type to use. An *EC2 instance type* defines a configuration of CPU, memory, storage, and network performance characteristics. This configuration provides a given level of compute performance. The instance type that you choose will depend on your workload's performance and cost requirements.

Instance type names follow a standard convention. An instance type name consists of multiple parts that describe the different characteristics of the instance type.

For example, in the *m5d.xlarge* instance type name, *m* is the *family name*. It is followed by a number, which is 5 in this case. It represents the *generation number* of that type. So, an *m5* instance is the fifth generation of the *m* family. In general, instances with a higher generation number are more powerful and provide a better value for the price than instances with a lower generation number.

Following the generation number, you find an optional part that indicates *additional capabilities* of the instance type. In the example, the *d* in the name indicates that the instance type uses a solid state drive (SSD) for the root EBS volume instead of a standard hard disk drive (HDD).

The next part of the name, which follows a period (.) separator, represents the *size* of the instance. The instance type size defines the performance specification of an instance across the CPU, memory, storage, and network performance categories. In the example, *xlarge* indicates that it is an extra-large instance. For the latest list of Amazon EC2 instance types, see the [Amazon EC2 Instance Types](#) documentation.

Suitability of instance types for workloads (1 of 2)

General purpose instance types

- Web or application servers
- Enterprise applications
- Gaming servers
- Caching fleets
- Analytics applications
- Development or test environments

Example instance types: 

Compute optimized instance types

- Batch processing
- Distributed analytics
- High performance computing (HPC)
- Ad server engines
- Multiplayer gaming
- Video encoding

Example instance types: 



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

26

You will now learn about the suitability of different instance types for different workloads.

Instance types can be categorized as general purpose, compute optimized, memory optimized, accelerated computing, or storage optimized. You will see each of the categories in this slide and the next slide.

General purpose instances provide a balance of compute, memory, and networking resources. They can be used for diverse workloads. These instances work well for applications that use these resources in equal proportions. Typical use cases for general purpose instances are: web or application servers, enterprise applications, gaming servers, caching fleets, analytics applications, and development or test environments.

Examples of general purpose instance types include *M5*, *T3*, and *A1* instances.

Compute optimized instances work well for compute-bound applications that benefit from high-performance processors. Instances that belong to this family are suited for workloads like batch processing, distributed analytics, high performance computing (HPC), ad server engines, multiplayer gaming, and video encoding.

Examples of compute optimized instance types include *C5* and *C5n* instances.

Suitability of instance types for workloads (2 of 2)

Memory optimized instance types

- In-memory caches
- High-performance databases
- Big data analytics

Example instance types: 

Accelerated computing instance types

- Machine learning, artificial intelligence (AI)
- HPC
- Graphics

Example instance types: 

Storage optimized instance types

- High-performance databases¹
- Real-time analytics¹
- Transactional workloads¹
- NoSQL databases¹
- Big data²
- Data warehouse²
- Log processing²

¹High I/O example instance type: 

²Dense Storage example instance types: 



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

27

Memory optimized instances are designed to deliver fast performance for workloads that process large datasets in memory. They are suited for applications like in-memory caches, high-performance databases, and big data analytics.

Examples of memory-optimized instance types include *R5*, *X1*, and *HMI* instances.

Accelerated computing instances use hardware accelerators (or co-processors) to perform functions like floating-point-number calculations, graphics processing, and data-pattern matching. They do so more efficiently than performing those functions in software that runs on CPUs. Accelerated computing instances are suited for *machine learning and artificial intelligence*, HPC, and graphics workloads.

Examples of accelerated computing instance types include *P3*, *G4*, and *F1* instances.

Storage optimized instances are designed for workloads that require high, sequential read/write access to very large datasets on local storage. They are optimized to deliver tens of thousands of low-latency, random IOPS. They are suited for applications like high-performance databases, NoSQL databases, real-time analytics, transactional workloads, big data, data warehouses, and log processing.

Examples of storage optimized instance types include *I3*, *D2*, and *H1* instances.

Choosing an instance type

- Choose the instance type that meets –
 - The [performance needs](#) of your application
 - Your [cost requirements](#)
- When you create a *new* instance –
 - In the EC2 console, use the [Instance Types](#) page to filter by characteristics that you choose
 - Recommendation: The latest generation in an instance family typically has a better price-to-performance ratio
- If you have an *already existing* instance –
 - You can get recommendations for optimizing the instance type by using the [AWS Compute Optimizer](#)
 - You can evaluate recommendations and modify the instance accordingly

With over 270 available instances types, how do you choose the correct one?



aws

© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

28

Given the combination of instance type categories, capabilities, and options that are associated with an instance type, it's no surprise that there are quite a few of them. As of March 2020, there are more than 270 instance types available. This wide selection reflects the deep and broad richness of AWS' cloud compute offerings.

The high number of instance types make it challenging to answer the question: How do you select the right instance type for your workload?

You want to choose an instance type that provides the level of *performance* that your application needs while ensuring the efficient utilization of your instance's resources to minimize cost.

If you are creating a new instance, use the *Instance Types* page in the Amazon EC2 console to help you search and compare the choices. This page displays all the available instance types in a Region and provides search and filtering capabilities based on attribute values. As a recommendation, when choosing an instance type in a family, select the latest generation instance type as it will typically have better price-to-performance ratio.

If you already have a running instance, you can use the *AWS Compute Optimizer* service to get recommendations on how to optimize the instance type. This service can analyze your instance's runtime behavior and make recommendations on how to optimize it. You can then

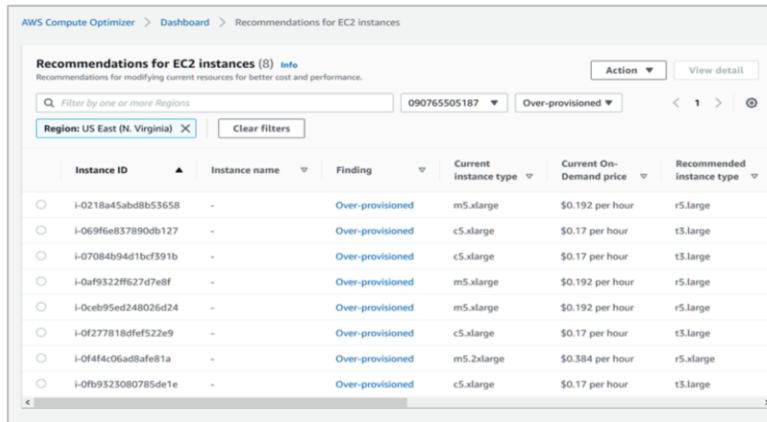
evaluate the recommendations and modify the instance accordingly.

AWS Compute Optimizer



AWS Compute Optimizer

- Recommends *optimal instance type, instance size, and Auto Scaling group configuration*
- Analyzes workload patterns and makes recommendations
- Classifies instance findings as Under-provisioned, Over-provisioned, Optimized, or None



The screenshot shows the AWS Compute Optimizer dashboard under the 'Recommendations for EC2 instances' section. It lists 8 recommendations for EC2 instances across the US East (N. Virginia) region. The columns include Instance ID, Instance name, Finding, Current instance type, Current On-Demand price, and Recommended instance type. Most instances are categorized as 'Over-provisioned'. The recommended instance types vary by instance.

Instance ID	Instance name	Finding	Current instance type	Current On-Demand price	Recommended instance type
i-0218a45abd8b53658	-	Over-provisioned	m5.xlarge	\$0.192 per hour	r5.large
i-069f6e837890db127	-	Over-provisioned	c5.xlarge	\$0.17 per hour	t3.large
i-070849b94d1bcf391b	-	Over-provisioned	c5.xlarge	\$0.17 per hour	t3.large
i-0af9322ff627d7e8f	-	Over-provisioned	m5.xlarge	\$0.192 per hour	r5.large
i-0ceb95ed248026d24	-	Over-provisioned	m5.xlarge	\$0.192 per hour	r5.large
i-0f277818dfe5f22e9	-	Over-provisioned	c5.xlarge	\$0.17 per hour	t3.large
i-0f4fc06d8afe81a	-	Over-provisioned	m5.2xlarge	\$0.384 per hour	r5.xlarge
i-0fb9323080785de1e	-	Over-provisioned	c5.xlarge	\$0.17 per hour	t3.large

aws

© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

29

AWS Compute Optimizer is a service that analyzes the configuration and utilization metrics of EC2 instances and Auto Scaling groups. It generates optimization recommendations to reduce the cost and improve the performance of your workloads. You can use these recommendations to decide whether to move to a new instance type.

Compute Optimizer uses Amazon Machine Learning (Amazon ML) to analyze your workloads. Currently, it generates EC2 instance type and size recommendations for M, C, R, T, and X instance families. When it is activated, Compute Optimizer will analyze your running AWS compute resources and start to deliver recommendations.

Compute Optimizer classifies its findings for EC2 instances as *Under-provisioned, Over-provisioned, Optimized, or None*. The *None* classification might occur if you have activated Compute Optimizer for less than 12 hours, when the instance has been running for less than 30 hours, or when the instance type is not supported by Compute Optimizer.

For more information, see the [AWS Compute Optimizer User Guide](#).

Section 4 key takeaways



- An [EC2 instance type](#) defines a configuration of CPU, memory, storage, and network performance characteristics
- As a recommendation, choose [new generation instance types in a family](#) because they generally have better price-to-performance ratios
- Use the [Instance Types](#) page in the Amazon EC2 console and [AWS Compute Optimizer](#) to find the right instance type for your workload



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

30

Some key takeaways from this section of the module include:

- An *EC2 instance type* defines a configuration of CPU, memory, storage, and network performance characteristics
- As a recommendation, choose *new generation instance types in a family* because they generally have better price-to-performance ratios
- Use the *Instance Types* page in the Amazon EC2 console and *AWS Compute Optimizer* to find the right instance type for your workload

Section 5: Using user data to configure an EC2 instance

Module 4: Adding a Compute Layer

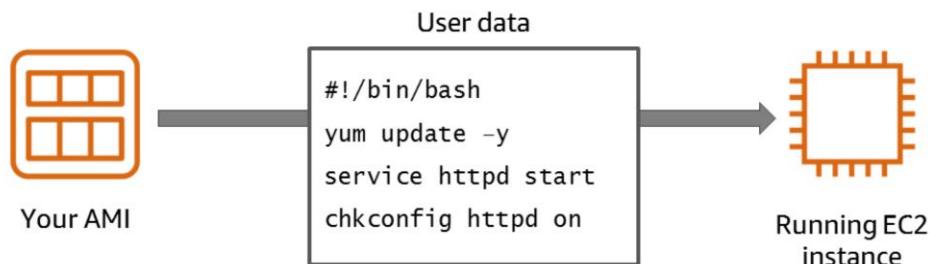


© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

Introducing Section 5: Using user data to initialize an EC2 instance.

EC2 instance user data

When you launch an EC2 instance, specify [user data](#) to run an initialization script (shell script or *cloud-init* directive).



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

32

When you launch an EC2 instance, you have the option of passing *user data* to it. User data enables you to provide a script that can be used to initialize it. For example, you can use user data to patch and update the software that is installed on the instance from the AMI, fetch and install software license keys, or install additional software.

User data is implemented as a script, which contains shell commands or *cloud-init* directives. It runs with root or Administrator privileges after the instance starts, but before it is accessible on the network. In the example, a user data script is provided to perform the following tasks when the instance is launched:

- Update all packages that are installed on the instance
- Start the installed Apache HTTP web server
- Configure the HTTP web server to automatically start when the instance boots

The *cloud-init* package is an open source application that was built by Canonical. It is used to bootstrap **Linux instances** in cloud-computing environments, such as Amazon EC2. Amazon Linux and many other Linux variants (such as Ubuntu) contain a version of *cloud-init*. The *cloud-init* package configures specific aspects of a new Amazon Linux instance when it is launched, even if you do not specifically add them to the user data scripts. Most notably, it configures the `.ssh/authorized_keys` file for the `ec2-user` so that you can log in with your own private key. However, you can also specify your own *cloud-init* user directives by adding them as user data directives. Details, including sample *cloud-init* directives, can be found in the [Running Commands on Your Linux Instance at Launch](#) documentation.

When the user data script runs, it produces messages in a log file located at `/var/log/cloud-init-output.log` on a Linux instance. For a Microsoft Windows instance, the log file is located at `C:\ProgramData\Amazon\EC2-Windows\Launch\Log\UserdataExecution.log`.

On **Microsoft Windows instances**, user data is processed by the EC2Config or EC2Launch tools, which include Windows PowerShell scripts. Windows 2016 and more recent Windows versions include EC2Launch. Older versions of Windows include EC2Config.

Retrieving instance metadata

Instance metadata is **information about your instance**.

- Is accessible from your instance at URL: `http://169.254.169.254/latest/meta-data/`
- Can be retrieved from a user data script

```
#!/bin/bash
yum update -y
hostname = $(curl -s http://169.254.169.254/latest/meta-data/public-hostname)
```

Metadata	Value
instance-id	i-1234567890abcdef0
mac	00-1B-63-84-45-E6
public-hostname	ec2-203-0-113-25.compute-1.amazonaws.com
public-ipv4	67.202.51.223
local-ipv4	10.251.50.12

© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

33

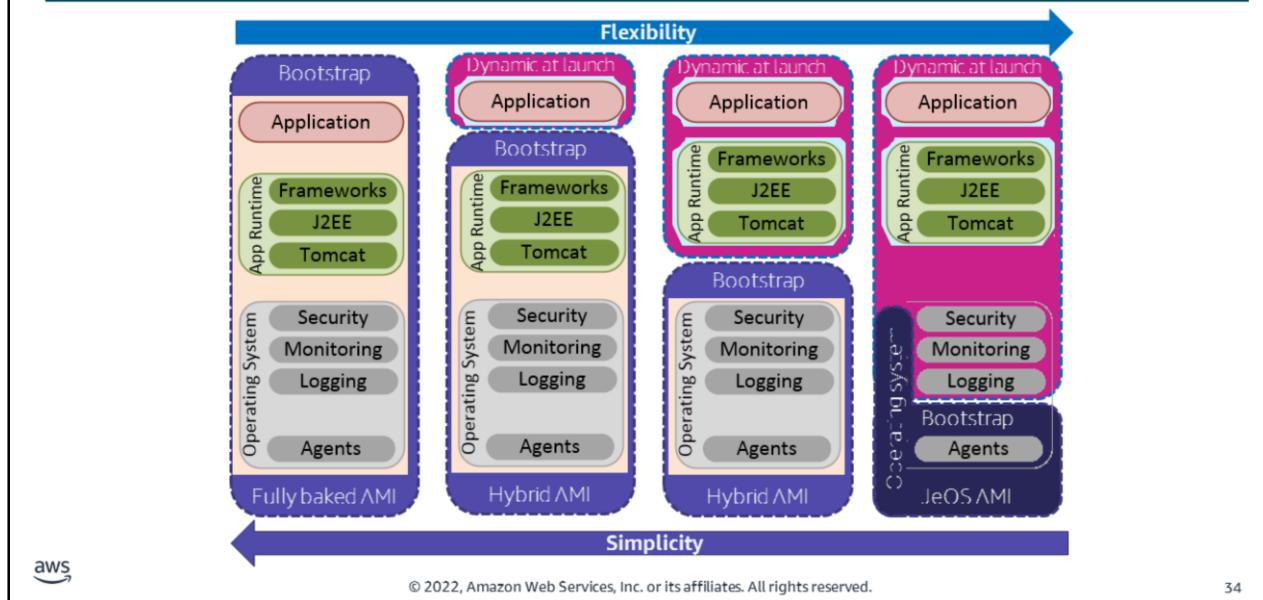
For user data to complete the launch of a new EC2 instance, it might need to look up information about the instance itself. For example, it might need to learn and share the public IP address, hostname, or media access control (MAC) address of the new instance to complete the launch. The Instance Metadata Service can provide that information.

Instance metadata is data about your instance. In many situations, you might need some information about the instance that you just launched. For example, in a user data script, you might need to know the instance's hostname or public IP address to configure a connection to an external resource. The Instance Metadata Service can provide that information.

Specifically, you can retrieve metadata information from your running instance by accessing the following URL: `http://169.254.169.254/latest/meta-data/`. The IP address `169.254.169.254` is a link-local address, and it is valid only from the instance.

Instance metadata provides much of the same information about the running instance that you can find in the AWS Management Console. For example, you can discover the public IP address, private IP address, public hostname, instance ID, security groups, Region, Availability Zone, and more. You can even access the user data specified at launch time for an instance by accessing the following URL:
`http://169.254.169.254/latest/user-data`.

Configuring an EC2 instance: AMI versus user data



When you launch an EC2 instance, you must make an important architectural decision. How much of the instance configuration should you pre-install in a base *AMI*, and how much should you dynamically build with *user data* at boot time?

At one end of the spectrum, you can build an AMI that contains all the configuration for the instance. This *fully baked AMI* includes everything to serve your workload, including the OS, the application runtime software, and the application itself. When you launch an instance with this AMI, it provisions a fully functional instance without any additional boot-time configuration.

At the other end of the spectrum, you can build an AMI that simply contains a minimal OS. This *Just Enough Operating System (JeOS) AMI* includes a configuration management agent that builds a fully functional system at instance launch. On first boot, the configuration agent downloads, installs, configures, and integrates all required software.

As an intermediate solution, you can build an AMI that contains a subset of the configuration that your workload needs. For example, the AMI could contain the OS and application runtime software only, or only the OS. This *hybrid AMI* then uses *user data* to complete the instance's configuration on first boot, based on the application's requirements.

Discovering the ideal approach typically results from considering the tradeoffs between simplicity and flexibility. Some of the factors that you should consider are:

- *Build times* – A fully baked AMI will lengthen the amount of time it takes to produce a build.
- *Boot times* – An AMI with an OS-only configuration will take a long time to boot when a new

instance is launched. Packaging prerequisites into a custom AMI shortens boot times.

- *Shelf life* – When you install more prerequisites on an AMI, you run a greater risk that your application will be vulnerable to a security risk if the underlying AMI is not frequently updated with security or application updates. Assess the risk that updates to your dependencies pose.

In summary, each approach provides tradeoffs:

- *Full AMI* – The applications and all dependencies are pre-installed, which shortens boot times but increases build times. Full AMIs typically have a shorter lifespan. Consider your rollback strategy.
- *Hybrid (partially configured) AMIs* – Only prerequisite software and utilities are pre-installed, which leads to a longer shelf life for the AMI. This approach provides a balance between boot speed and build time. Rollbacks become easier.
- *OS-only AMI* – This approach is fully configurable and upgradeable over time and shortens build times. However, it makes your EC2 instances slow to boot because all required installations and configurations must be run at boot time.

Many organizations decide on a hybrid approach, where some configurations are baked into a custom base AMI and other settings are configured dynamically at launch.

For more information, see [AWS AMI Design](#) on AWS Answers.

Demonstration: Configuring an EC2 Instance with User Data



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

35

Now, the educator might choose to demonstrate configuring an EC2 instance with user data.

Section 5 key takeaways



- User data enables you to [configure an EC2 instance when you launch it](#).
- Information about a running instance can be accessed in the instance through an [instance metadata URL](#).
- Baking configurations into an AMI [increases AMI build time, but decreases instance boot time](#). Configuring an instance by using user data [decreases AMI build time, but increases instance boot time](#).

© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

36

Some key takeaways from this section of the module include:

- *User data* enables you to pass configuration parameters or run an initialization script when you launch an instance.
- Information about a running instance can be accessed in the instance through an *instance metadata URL*.
- Baking configurations into an AMI increases AMI build time, but decreases instance boot time. Configuring an instance by using user data decreases AMI build time, but increases instance boot time.

Section 6: Adding storage to an Amazon EC2 instance

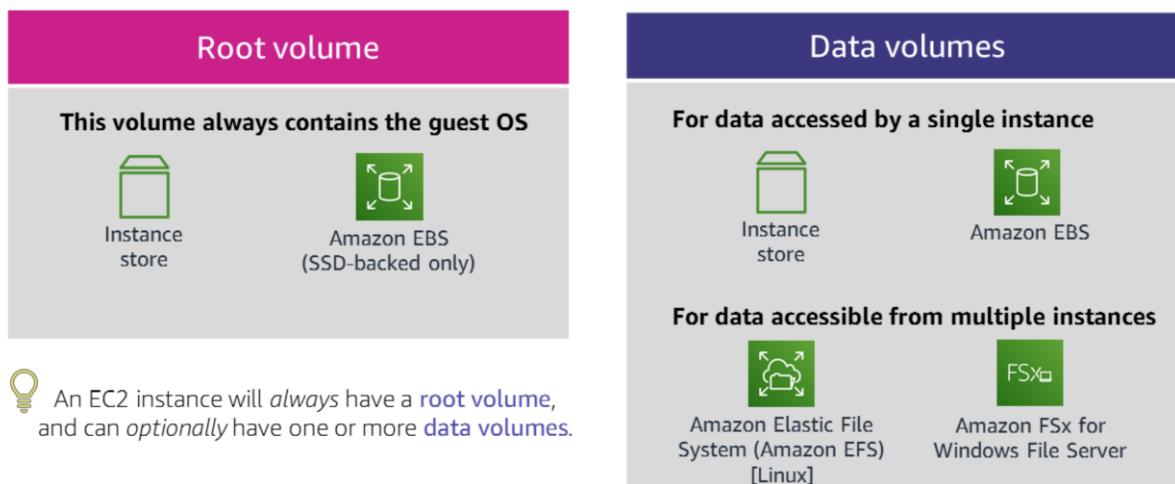
Module 4: Adding a Compute Layer



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

Introducing Section 6: Adding storage to an Amazon EC2 instance.

Amazon EC2 storage overview



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

38

EC2 instances have four main storage options:

- Instance store
- Amazon EBS
- Amazon Elastic File System (Amazon EFS)
- Amazon FSx for Windows File Server

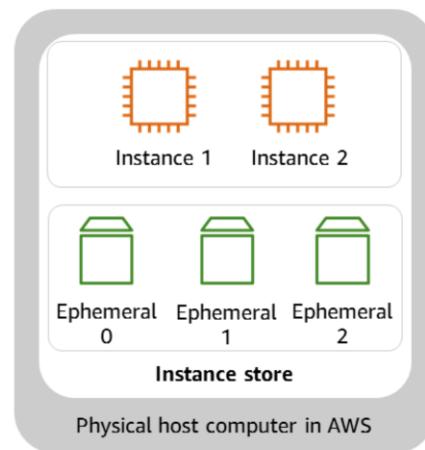
All four options can be used to store a data volume. However, only an instance store or an SSD-backed EBS volume can be used to store a root volume. In addition, an instance store or EBS volume must be used by a single instance at a time. In the case of an instance store volume, only the instance that the volume is added to can use it.

If you want to share a data volume among multiple instances at the same time, you can use Amazon EFS or Amazon FSx for Windows File Server. Amazon EFS provides a shared file system for Linux instances, but Amazon FSx provides a shared file system for Microsoft Windows instances.

In this section, you will review each of these options.

Instance store

- An instance store provides **non-persistent storage** to an instance –
 - The data is stored on the *same physical server* where the instance runs
- Characteristics –
 - Temporary block-level storage
 - Uses HDD or SSD
 - **Instance store data is lost when the instance is stopped or terminated**
- Example use cases –
 - Buffers
 - Cache
 - Scratch data



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

39

An *instance store* volume provides temporary block-level storage for your instance. This storage is on disks that are physically attached to the computer that hosts the running instance. An instance store works well for the temporary storage of information that changes frequently, such as buffers, caches, scratch data, and other temporary content.

An instance store consists of one or more volumes that are exposed as block devices. The volumes reside on either HDDs or SSDs. SSDs can be Serial ATA SSDs or Non-Volatile Memory Express (NVMe) SSDs. Instance stores that use NVMe have higher I/O performance. An instance store is dedicated to a particular instance, but the disk subsystem is shared among instances on a host computer.

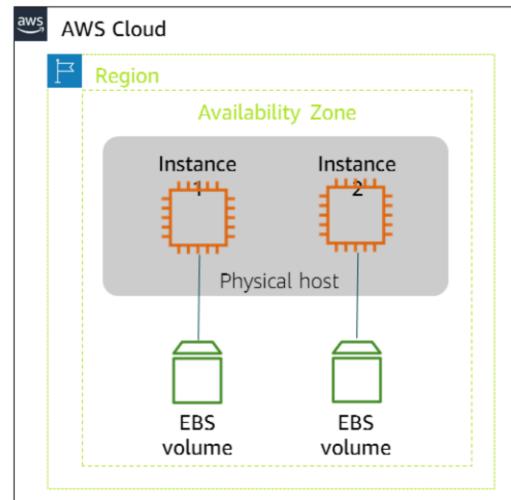
The instance type determines the available sizes for an instance store and the type of hardware that is used for the instance store volumes. Most instance types support instance stores, but not all.

The data in an instance store is preserved when the instance is rebooted, but not when it is terminated. Instance store-backed instances cannot be stopped. They can only be rebooted or terminated. If an instance reboots, data in the instance store persists. For more information, see

the [Amazon EC2 Instance Store](#) topic in the Amazon EC2 online documentation.

Amazon EBS

- Amazon EBS volumes provide **network-attached persistent storage** to an EC2 instance.
- Characteristics –
 - Is persistent block-level storage
 - Can attach to any instance in the same Availability Zone
 - Uses HDD or SSD
 - Can be encrypted
 - Supports snapshots that are persisted to S3
 - Data persists independently from the life of the instance
- Example use cases –
 - Stand-alone database
 - General application data storage



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

40

Amazon EBS provides block-level storage volumes, similar to an external hard drive, for use with Amazon EC2 instances. Amazon EBS volumes are highly available and reliable, and can be network-attached to running instances in the same Availability Zone.

Amazon EBS volumes persist independently from the life of the instance and can be encrypted. In addition, you can back up the data on an Amazon EBS volume to Amazon S3 by taking a point-in-time *snapshot*.

Amazon EBS volumes reside on either HDDs or SSDs (Serial ATA SSD or NVMe SSD).

Because they are mounted on the instance, EBS volumes can provide extremely low access latency. For this reason, EBS volumes can be used to run a database on an EC2 instance, for example.

Amazon EBS SSD-backed volume types

Amazon EBS SSD-backed volumes are suited for use cases where the performance focus is on IOPS.

	General Purpose SSD (gp2)	Provisioned IOPS SSD (io1)
Description	Balances price and performance for a wide variety of workloads	<ul style="list-style-type: none">Highest-performance SSD volumeGood for mission-critical, low-latency, or high-throughput workloads
Use Cases	<ul style="list-style-type: none">Recommended for most workloadsCan be a boot volume	<ul style="list-style-type: none">Critical business applications that require sustained IOPS performanceLarge database workloadsTransactional workloadsIt can be a boot volume



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

41

Amazon EBS provides volume types that differ in performance characteristics and price to tailor storage performance and cost to the needs of different applications. They fall into two categories: HDDs and SSDs.

SSD-backed volumes are optimized for transactional workloads that involve frequent read/write operations with small I/O size, where the dominant performance attribute is IOPS (Input/Output Operations Per Second).

The two types of Amazon SSD-backed volumes are:

- General Purpose SSD (gp2)* – This volume type is the default EBS volume type for EC2 instances. These volumes are suitable for a range of transactional workloads, including development or test environments, low-latency interactive applications, and boot volumes.
- Provisioned IOPS SSD (io1)* – This volume type is the highest-performance EBS storage option. It is designed for critical, I/O-intensive database and application workloads, and throughput-intensive database and data warehouse workloads.

Amazon EBS SSD-backed volumes are ideal for both IOPS-intensive and throughput-intensive workloads that require extremely low latency.

Amazon EBS HDD-backed volume types

Amazon EBS HDD-backed volumes work well when the focus is on throughput.

	Throughput Optimized HDD (st1)	Cold HDD (sc1)
Description	<ul style="list-style-type: none">Low-cost volume typeDesigned for frequently accessed, throughput-intensive workloads	<ul style="list-style-type: none">Lowest-cost HDD volumeDesigned for less frequently accessed workloads
Use Cases	<ul style="list-style-type: none">Streaming workloadsBig dataData warehousesLog processingIt cannot be a boot volume	<ul style="list-style-type: none">Throughput-oriented storage for large volumes of infrequently accessed dataUse cases where the lowest storage cost is importantIt cannot be a boot volume



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

42

HDD-backed volumes are optimized for large streaming workloads where throughput (measured in MiB/s) is a better performance measure than IOPS.

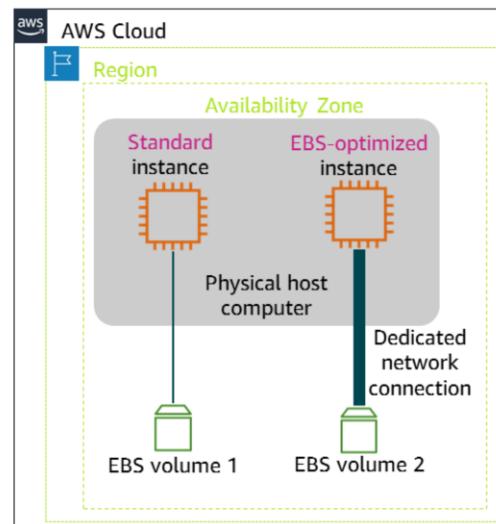
Amazon HDD-backed volumes come in two types:

- Throughput Optimized HDD (st1)* – This volume type is ideal for frequently accessed, throughput-intensive workloads with large datasets and large I/O sizes.
- Cold HDD (sc1)* – This volume type provides the lowest cost per GB of all EBS volume types. It works well for less-frequently accessed workloads with large, cold datasets.

For more information, see the [Amazon EBS Volume Types](#) topic in the Amazon EC2 User Guide.

Amazon EBS-optimized instances

- Certain EC2 instance types can be [EBS-optimized](#)
- Benefits –
 - Provides a [dedicated network connection](#) to attached EBS volumes
 - Increases I/O performance
 - Additional performance is achieved if using an Amazon EC2 [Nitro System-based](#) instance type
- Usage –
 - For EBS-optimized instance types, optimization is enabled by default
 - For other instance types that support it, optimization must be manually enabled



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

43

Certain EC2 instance types can be optimized so that I/O access to an EBS volume is increased. These instances are called *Amazon EBS-optimized instances*.

An EBS-optimized instance has a dedicated network connection between itself and an EBS volume. This optimization provides the best performance for accessing EBS volumes by minimizing contention between Amazon EBS I/O and other network traffic from the instance. Specifically, it provides a dedicated bandwidth to Amazon EBS, with options between 425 Mbps and 14,000 Mbps (depending on the instance type that you use). This dedicated bandwidth provides performance consistency. For example, for a *General Purpose SSD (gp2)* volume that is attached to an EBS-optimized instance, it can translate to a 10 percent improvement over its baseline performance.

If an EC2 instance type supports EBS optimization, it is automatically enabled when the instance type is categorized as *EBS-optimized*. Otherwise, you must manually enable the optimization when you launch the instance by setting its *Amazon EBS-optimized* attribute.

Furthermore, if an EBS-optimized instance type is built on the AWS Nitro system, it can benefit from additional performance increases. This system is a collection of AWS-built hardware and software components that enable high performance, high availability, high security, and bare metal capabilities to eliminate virtualization overhead.

For more information, see the [Amazon EBS–Optimized Instances](#) topic in the Amazon EC2 User Guide.

Shared file systems for EC2 instances

What if you have **multiple instances** that must use the **same storage**?

Amazon EBS: Attaches only to one instance



Amazon EBS

Amazon S3: Is an option, but is not ideal



Amazon S3

Amazon EFS and
Amazon FSx for
Windows File Server:
Both satisfy the
requirement



Amazon EFS
(Linux)



Amazon FSx for
Windows File
Server (Windows)



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

44

Amazon EC2 instance store and Amazon EBS provide good choices for storing both root and data volume for a single instance. However, what if you need to share data among multiple instances simultaneously? How do you handle an application that runs on multiple instances that must use the same file system?

Earlier in this section, you learned that an EBS volume must be attached to only one instance at a time. Therefore, it cannot solve the stated problem. Amazon S3 is another option, but it is an object store system, not a block store. Thus, changes in Amazon S3 overwrite entire files, not blocks of characters in files. For making high-throughput changes to files of various sizes, a file system works better than an object store system.

This requirement needs the performance and read/write consistency of a network file system. Amazon EFS and Amazon FSx for Windows File Server provide a shared file system for Linux instances and Windows instances. You will learn about these two services next.

Amazon EFS



Amazon EFS provides file system storage for **Linux-based** workloads.

- Fully managed elastic file system
- Scales automatically up or down as files are added and removed
- Petabytes of capacity
- Supports Network File System (NFS) protocols
 - Mount the file system to the EC2 instance
- Compatible with all Linux-based AMIs for Amazon EC2



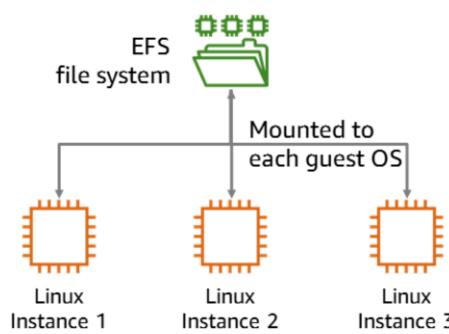
© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

45

Amazon EFS is a fully managed service that provides file system storage for Linux-based workloads. Files are accessed via a file system interface (using standard operating system file I/O APIs). They support full file system access semantics, such as strong consistency and file locking. Amazon EFS uses the Network File System (NFS) version 4.x protocol. It can be used with any AMIs that support this protocol.

Multiple EC2 instances can access an EFS file system at the same time. Thus, Amazon EFS can provide a common data source for workloads and applications that run on more than one Amazon EC2 instance. In addition, EFS file systems can automatically scale from gigabytes to petabytes of data without needing to provision storage.

Amazon EFS use cases



Common workloads and applications:

- Home directories
- File system for enterprise applications
- Application testing and development
- Database backups
- Web serving and content management
- Media workflows
- Big data analytics

Example command to mount the file system to each guest OS:

```
$ sudo mount -t nfs4 mount-target-DNS:/ ~/efs-mount-point
```



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

46

The main use cases for Amazon EFS include:

- *Home directories* – Provides storage for organizations that have many users who must access and share common datasets.
- *File system for enterprise applications* – Provides the scalability, elasticity, availability, and durability to be the file store for enterprise applications and for applications that are delivered as a service.
- *Application testing and development* – Provides a common storage repository that enables you to share code and other files in a secure and organized way.
- *Database backups* – Can be mounted with NFSv4 from database servers.
- *Web serving and content management* – Provides a durable, high-throughput file system for content management systems that store and serve information for a range of applications (such as websites, online publications, and archives).
- *Big data analytics* – Provides the scale and performance for big data applications that need high throughput to compute nodes, along with read-after-write consistency and low-latency file operations.
- *Media workflows* – Provides the strong data consistency model, high throughput, and shared-file access that can reduce the time it takes to perform video editing, studio production, broadcast processing, sound design, and rendering jobs. At the same time, it consolidates multiple local file repositories into a single location for all users.

For more information, see the [Amazon EFS User Guide](#).

Amazon FSx for Windows File Server

Provides fully managed shared file system storage for Microsoft Windows EC2 instances.



- Native Microsoft Windows compatibility
- New Technology File System (NTFS)
- Native Server Message Block (SMB) protocol version 2.0 to 3.1.1
- Distributed File System (DFS) Namespaces and DFS Replication
- Integrates with Microsoft Active Directory and supports Windows access control lists (ACLs)
- Backed by high-performance SSD storage



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

47

If you need file system-based storage for Microsoft Windows EC2 instances, Amazon FSx for Windows File Server provides fully managed Windows file servers, backed by a fully-native Windows file system. It is built on Microsoft Windows Server and has native support for Windows file system features and protocols including:

- *New Technology File System (NTFS)*, the default file system of Windows servers
- *Server Message Block (SMB)*, the open standard protocol for accessing file storage over a network
- *Distributed File System (DFS)*, a service that enables the organization of multiple distributed SMB file shares into a distributed file system
- *Microsoft Active Directory*, the Microsoft directory service that administers and organizes enterprise resources (including users, groups, and network resources)

Amazon FSx uses SSD storage to provide fast performance, high levels of throughput and IOPS, and low latencies. It also reduces the administrative overhead of setting up and provisioning file servers and storage volumes.

Amazon FSx for Windows File Server use cases

Amazon FSx for Windows File Server supports a **broad set of Microsoft Windows workloads**.

- Home directories
- Lift-and-shift application workloads
- Media and entertainment workflows
- Data analytics
- Web serving and content management
- Software development environments



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

48

Use cases for Amazon FSx for Windows File Server (FSx) include the following Microsoft Windows workloads:

- *Home directories* – Create a file system that hundreds of thousands of users can access, and establish permissions at the file or folder level.
- *Lift-and-shift applications* – It provides fully managed native Windows file shares with features, like Microsoft Active Directory integration and automatic backups.
- *Media and entertainment workflows* – Media workflows—like media transcoding, processing, and streaming—often depend on shared Windows file storage with consistent performance. FSx for Windows File Server provides you with a shared file system with the high throughput and low latencies that these Windows workloads need.
- *Data analytics* – By providing scalable file systems with high throughput and low latencies, FSx for Windows File Server enables you to run data-intensive analytics workloads.
- *Web serving and content management* – Multiple web or content servers typically need access to the same files. FSx for Windows File Server provides a file system that can be accessed across thousands of instances simultaneously.
- *Software development environments* – FSx for Windows File Server enables you to move to in-cloud development with no changes to your development workflows by providing shared file storage that your developers and their environments can access.

For more information, see the [Amazon FSx for Windows File Server User Guide](#).

Section 6 key takeaways



- Storage options for EC2 instances include **instance store**, Amazon EBS, Amazon EFS, and Amazon FSx for Windows File Server
- For a **root volume**, use instance store or SSD-backed Amazon EBS
- For a **data volume that serves only one instance**, use instance store or Amazon EBS storage
- For a **data volume that serves multiple Linux instances**, use Amazon EFS
- For a **data volume that serves multiple Microsoft Windows instances**, use Amazon FSx for Windows File Server

© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

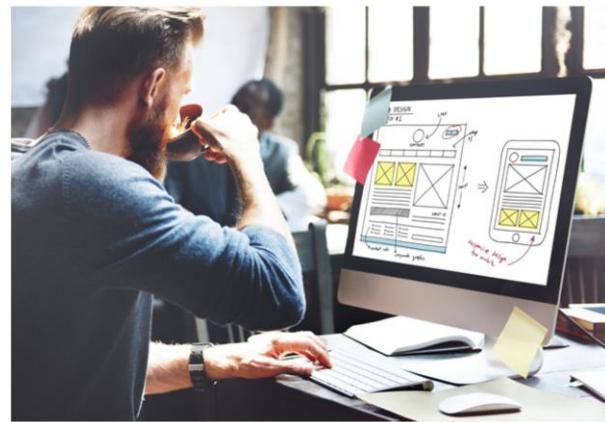
49

Some key takeaways from this section of the module include:

- Storage options for EC2 instances include instance store, Amazon EBS, Amazon EFS, and Amazon FSx for Windows File Server
- For a root volume, use instance store or SSD-backed Amazon EBS
- For a data volume that serves only one instance, use instance store or Amazon EBS storage
- For a data volume that serves multiple Linux instances, use Amazon EFS
- For a data volume that serves multiple Microsoft Windows instances, use Amazon FSx for Windows File Server

Module 4 – Guided Lab:

Introducing Amazon Elastic File System (Amazon EFS)



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

50

You will now complete Module 4 - Guided Lab: Introducing Amazon Elastic File System (Amazon EFS).

Guided lab: Tasks

1. Creating a security group to access your EFS file system
2. Creating an Amazon EFS file system
3. Connecting to your EC2 instance via SSH
4. Creating a new directory and mounting the EFS file system
5. Examining the performance behavior of your new EFS file system



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

51

In this guided lab, you will complete the following tasks:

1. Creating a security group to access your EFS file system
2. Creating an Amazon EFS file system
3. Connecting to your EC2 instance via SSH
4. Creating a new directory and mounting the EFS file system
5. Examining the performance behavior of your new EFS file system



A timer icon with the text "~ 20 minutes" indicating the duration of the lab.

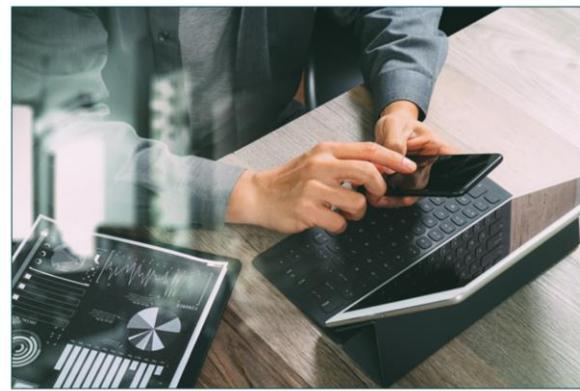
Begin Module 4 – Guided Lab: Introducing Amazon Elastic File System (Amazon EFS)

aws

© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

It is now time to start the guided lab.

Guided lab debrief: Key takeaways



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

53

Your educator might choose to lead a conversation about the key takeaways from the guided lab after you have completed it.

Section 7: Amazon EC2 pricing options

Module 4: Adding a Compute Layer



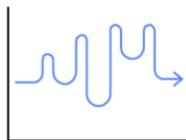
© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

Introducing Section 7: Amazon EC2 pricing options.

Amazon EC2 pricing options (1 of 2)

On-Demand Instances

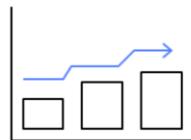
Pay for compute capacity by [the second or by the hour](#) with no long-term commitments.



Spiky workloads,
workload experimentation

Reserved Instances

Make a [1-year or 3-year commitment](#) and receive a significant discount off on-demand prices.



Committed and
steady-state workloads

Savings Plans

Same discounts as Reserved Instances with [more flexibility in exchange for a \\$/hour commitment](#).



All Amazon EC2,
AWS Fargate, and
AWS Lambda workloads



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

55

AWS offers five ways to pay for Amazon EC2 instances: *On-Demand Instances*, *Reserved Instances*, *Savings Plans*, *Spot Instances*, and *Dedicated Hosts*. The first three options are identified in this slide, and the last two are described in the next slide.

On-Demand Instances offer the most flexibility, with no long-term contract and low rates. They are a good choice for applications with short-term, spiky, or unpredictable workloads. They are also suited for developing and testing applications on Amazon EC2 for the first time. With On-Demand Instances, you pay for compute capacity by the hour or by the second, depending on which instances you run. *On-Demand Instances* have the lowest upfront cost and the most flexibility. They require no upfront commitments or long-term contracts.

Reserved Instances enable you to reserve computing capacity for a 1-year or 3-year term with lower hourly running costs. The discounted usage price is fixed for as long as you own the Reserved Instance. Reserved Instances are a good choice if you have predictable or steady-state compute needs (for example, an instance that you know you want to keep running most or all the time for months or years). If you expect consistent, heavy use, they can provide substantial savings compared to On-Demand Instances.

Savings Plans is a flexible pricing model that offers low prices on Amazon EC2, AWS Lambda, and AWS Fargate usage, in exchange for a commitment to a consistent amount of usage (measured in dollars per hour) for a 1-year or 3-year term. It offers lower prices on EC2 instance usage compared to On-Demand Instances, regardless of instance family, size, operating system, tenancy, or AWS Region. Each type of compute usage has an On-Demand Instance rate and a

Savings Plans price. You can get started with Savings Plans from AWS Cost Explorer in the console or by using the API or the AWS CLI.

Amazon EC2 pricing options (2 of 2)

Spot Instances

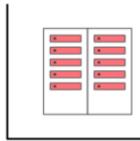
Spare Amazon EC2 capacity at [substantial savings](#) off On-Demand Instance prices.



Fault-tolerant, flexible, stateless workloads

Dedicated Hosts

[Physical server with Amazon EC2 instance capacity fully dedicated for your use.](#)



Workloads that require the use of your own software licenses or single tenancy to meet compliance requirements



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

5b

This slide lists the final two Amazon EC2 purchase options. They are *Spot Instances* and *Dedicated Hosts*.

Spot Instances enable you to bid on unused EC2 instances and request spare computing capacity at substantial savings. They are recommended for fault-tolerant, flexible (non-time-critical), stateless workloads. To use a Spot Instance, your workload must have the ability to be stopped and restarted: Amazon EC2 can interrupt it with a 2-minute notification to meet other capacity requirements. Your Spot Instance runs when capacity is available and the maximum price per hour for your request exceeds the Spot Instance price. Spot Instances are billed on 1-second increments, with a minimum of 60 seconds, if they are running the Amazon Linux or Ubuntu operating system. All other operating systems are billed in 1-hour increments, rounded up to nearest hour.

A variation of Spot Instances, called *Spot blocks*, enables you to run a workload continuously for a finite duration of 1–6 hours. Spot blocks are designed to not be interrupted and will run continuously for the duration you select, independent of the Spot Instance market price.

Dedicated Hosts are physical servers with instance capacity that is dedicated to your use. They are physically isolated at the level of the host hardware from instances that belong to other AWS accounts. Dedicated Hosts are a good choice when you have licensing restrictions for the software that you want to run on Amazon EC2, or when you have specific compliance or

regulatory requirements that preclude you from using other deployment options.

Demonstration: Reviewing the Spot Instance History Page



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

57

Now, the educator might choose to demonstrate the Spot Instance History Page in the Amazon EC2 console.

Amazon EC2 dedicated options

Amazon EC2 dedicated options provide EC2 instance capacity on physical servers that are dedicated for your use (single-tenant hardware).

Dedicated Instances	Dedicated Hosts
<ul style="list-style-type: none">• Per-instance billing• Automatic instance placement• Benefit – Isolates the hosts that run your instances	<ul style="list-style-type: none">• Per-host billing• Visibility of sockets, cores, and host ID• Affinity between a host and an instance• Targeted instance placement• Add capacity by using an allocation request• Benefit – Enables you to use your server-bound software licenses and address compliance requirements



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

58

You can use *Dedicated Hosts* and *Dedicated Instances* to launch EC2 instances on physical servers that are dedicated for your use. Both options enable you to isolate the hosts that run your instances from the hosts that run instances for other accounts. They also provide the same performance, security, and physical features.

Dedicated Hosts give you visibility and control over how instances are placed on a physical server. You can consistently deploy your instances to the same physical server over time. As a result, Dedicated Hosts enable you to use your existing server-bound software licenses, and to address corporate compliance and regulatory requirements.

Both options differ in the way they are billed. A Dedicated Instance is billed per instance, and its pricing has two components. The two components are an hourly per-instance usage fee and a dedicated per-Region fee that you pay once per hour, regardless of how many Dedicated Instances are running.

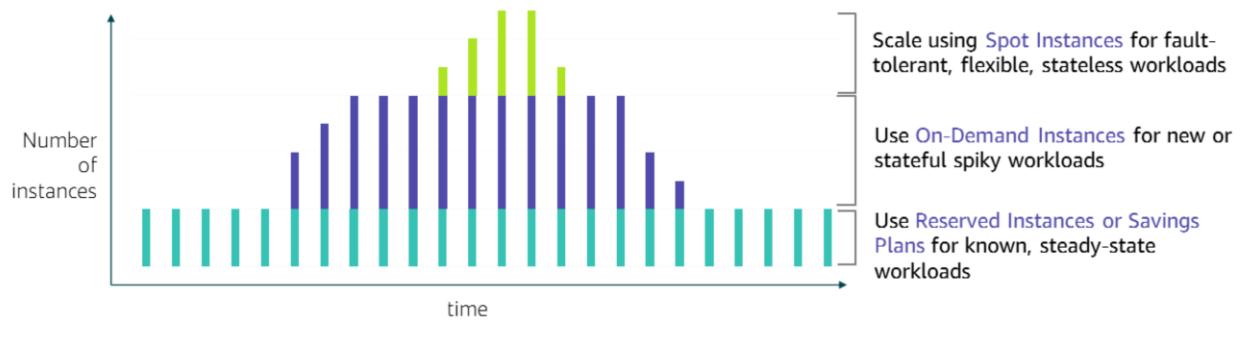
A Dedicated Host is billed per host. You can choose from three different pricing models to pay for Dedicated Hosts: On-Demand, Reservation Pricing, and Savings Plans.

To indicate that an instance should run as a Dedicated Instance or a Dedicated Host, change the instance's *tenancy* attribute to *dedicated* or *host*, respectively. For more information, see

[Amazon EC2 Dedicated Hosts.](#)

Amazon EC2 cost optimization guideline

To **optimize** the cost of Amazon EC2 instances, **combine** the available purchase options.



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

59

Each Amazon EC2 pricing model provides a different set of benefits. As a general guideline, the recommended way to optimize the cost of EC2 instances is to combine available purchase options.

Specifically, start by identifying the steady-state workloads in your application portfolio and run them on EC2 instances that use the Reserved Instance or Savings Plans pricing option.

Next, for your stateful spiky workloads, select EC2 instances that use the On-Demand Instance pricing plan.

Finally, use Spot Instances for your fault-tolerant, flexible, stateless workloads.

Section 7 key takeaways



- Amazon EC2 pricing models include On-Demand Instances, Reserved Instances, Savings Plans, Spot Instances, and Dedicated Hosts
- Per-second billing is available only for On-Demand Instances, Reserved Instances, and Spot Instances that run [Amazon Linux or Ubuntu](#)
- Use a combination of Reserved Instances, Savings Plans, On-Demand Instances, and Spot Instances to optimize Amazon EC2 compute costs

© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

60

Some key takeaways from this section of the module include:

- *Amazon EC2 pricing models* include On-Demand Instances, Reserved Instances, Savings Plans, Spot Instances, and Dedicated Hosts
- *Per-second billing* is available only for On-Demand Instances, Reserved Instances, and Spot Instances that run *Amazon Linux or Ubuntu*
- Use a *combination* of Reserved Instances, Savings Plans, On-Demand Instances, and Spot Instances to optimize Amazon EC2 compute costs

Section 8: Amazon EC2 considerations

Module 4: Adding a Compute Layer



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

Introducing Section 8: Amazon EC2 considerations.

Placement groups

Placement groups enable you to control where instances run in an Availability Zone.

- They influence where a group of interdependent instances run –
 - Increase network performance between them
 - Reduce correlated or simultaneous failure
- Placement strategies –
 - Cluster
 - Partition
 - Spread
- Limitations –
 - An instance can be launched in only one placement group at a time
 - Instances with a tenancy of *host* cannot be launched in a placement group



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

62

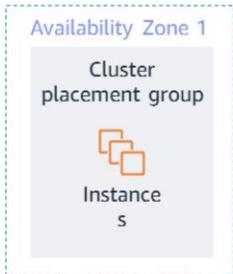
When you launch a new EC2 instance, the default behavior of Amazon EC2 is to minimize correlated failures by spreading out new instances across underlying hardware. You can use *placement groups* to influence the placement of a group of *interdependent* instances so they meet the needs of your workload.

Depending on the type of workload, you can create a placement group by using one of the following placement strategies:

- *Cluster* – Packs instances close together inside an Availability Zone. This strategy enables workloads to achieve low-latency network performance.
- *Partition* – Spreads your instances across logical partitions so that groups of instances in one partition do not share the underlying hardware with groups of instances in different partitions.
- *Spread* – Strictly places a small group of instances across distinct underlying hardware to reduce correlated failures.

Cluster placement group

Cluster placement groups provide **low-latency** and **high packet-per-second** network performance between instances in the same Availability Zone.



- Instances are placed in the same high-bisection bandwidth segment of the network
- Provides per-flow throughput limit of up to 10 Gbps for TCP/IP traffic
- Recommended for applications that benefit from low network latency, high network throughput, or both
- Best practice – Launch all instances in a single request



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

63

The cluster placement group is a logical grouping of instances in a single Availability Zone. This grouping provides low-latency and high packet-per-second network performance between instances.

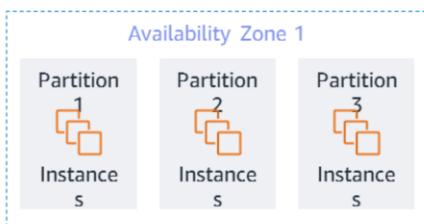
Instances in the same cluster placement group have a higher per-flow throughput limit of up to 10 Gbps for TCP/IP traffic. They are placed in the same high-bisection bandwidth segment of the network.

Cluster placement groups are recommended for applications that benefit from low network latency, high network throughput, or both. These applications include HPC applications, which require tightly coupled node-to-node communication. Cluster placement groups are also recommended when the majority of the network traffic is between the instances in the group.

AWS recommends that you launch all the instances that you need in a cluster grouping all at once in a single launch request. If you try to add more instances to the group later, you increase your chances of receiving an insufficient capacity error.

Partition placement group

A partition placement group spreads instances across logical partitions to reduce the likelihood of correlated hardware failure.



- Each partition has its own set of racks (network and power source)
- Each rack has its own network and power source
- Partitions can be in multiple Availability Zones
- They are recommended for large distributed and replicated workloads



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

64

Partition placement groups help reduce the likelihood of correlated hardware failures for your application. When you use partition placement groups, Amazon EC2 divides each group into logical segments that are called *partitions*. Amazon EC2 ensures that each partition in a placement group has its own set of racks. Each rack has its own network and power source. No two partitions in a placement group share the same racks, so you can isolate the impact of hardware failure in your application.

A partition placement group can have partitions in multiple Availability Zones in the same Region. A partition placement group can have a maximum of seven partitions per Availability Zone.

Partition placement groups also offer visibility into the partitions and allow topology-aware applications to use this information to make intelligent data replication decisions, which can increase data availability and durability. As a result, they are typically used to deploy large distributed and replicated workloads, such as Apache Hadoop, Apache HBase, and Apache Cassandra.

In the example, instances are placed into a partition placement group with three partitions in the same Availability Zone: *Partition 1*, *Partition 2*, and *Partition 3*. The instances in a partition

do not share racks with the instances in the other partitions, so you can contain the impact of a single hardware failure to only the associated partition.

Spread placement group

Spread placement groups place instances across distinct physical racks to **reduce correlated hardware failure**.



- Each rack has its own network and power source
- Group can span multiple Availability Zones
- They are recommended for applications that have a small number of critical instances that should be kept separate from each other



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

65

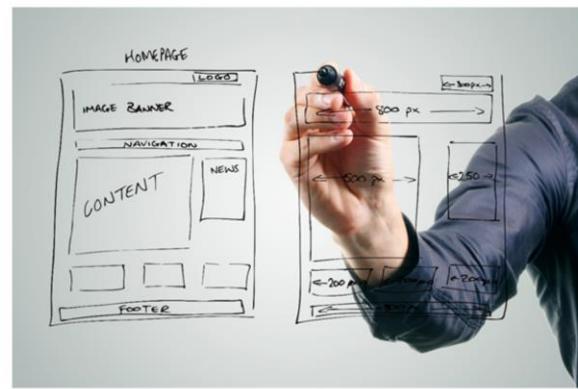
A *spread placement group* is a grouping of instances that are purposely positioned on distinct underlying hardware. This grouping reduces the risk of simultaneous failures that could occur if instances shared underlying hardware.

This type of group can span multiple Availability Zones, up to a maximum of seven instances per Availability Zone per group.

Spread placement groups are recommended for applications that have a small number of critical instances that should be kept separate from each other.

Module 4 – Challenge Lab:

Creating a Dynamic Website for the Café



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

66

You will now complete Module 4 – Challenge Lab: Creating a Dynamic Website for the Café.

The business need: Online ordering



- Customer liked the static website that the café introduced, but they now want to place orders online
- It will also be important to maintain an order history
- Amazon S3 worked well to host a static website, but that simple architecture will not meet this new business need
- The café staff also wants separate development and production environments



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

67

After the café launched the first version of their website, customers told the café staff how nice the website looks. However, in addition to the praise, customers often asked whether they could place online orders.

Sofía, Nikhil, Frank, and Martha discussed the situation. They agreed that their business strategy and decisions should focus on delighting their customers and providing them with the best possible café experience.

A business request for the café: Launching a dynamic website (Challenge #1)

The café wants to introduce online ordering for customers, and enable café staff to view submitted orders. Their current website architecture, where the website is hosted on Amazon S3, does not support the new business requirements.

In the first part of this lab, you will take on the role of Sofía, and use Amazon EC2 to create a dynamic website for the café.

Challenge Lab: Tasks

1. Analyzing the existing EC2 instance
2. Connecting to the IDE on the EC2 instance
3. Analyzing the LAMP stack environment and confirming that the web server is accessible
4. Installing the café application
5. Testing the web application
6. Creating an AMI and launching another EC2 instance
7. Verifying the new café instance



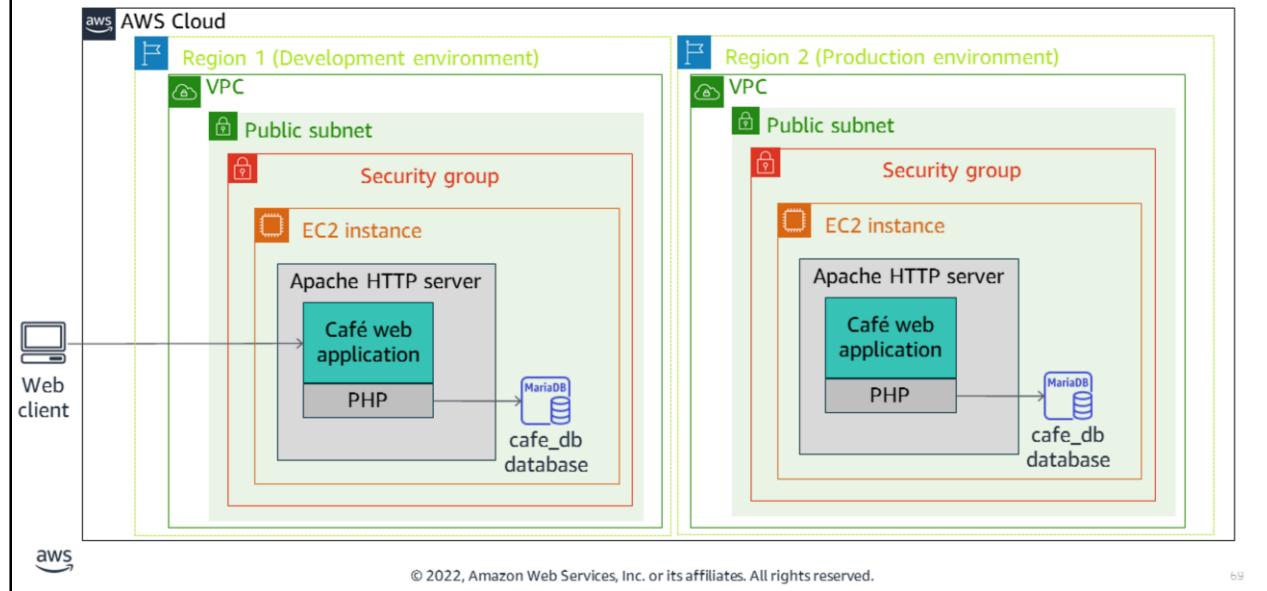
© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

68

In this challenge lab, you will complete the following tasks:

1. Analyzing the existing EC2 instance
2. Connecting to the IDE on the EC2 instance
3. Analyzing the LAMP stack environment and confirming that the web server is accessible
4. Installing the café application
5. Testing the web application
6. Creating an AMI and launching another EC2 instance
7. Verifying the new café instance

Challenge lab: Final product



The diagram summarizes what you will have built after you complete the lab. This challenge lab includes some sections where specific step-by-step instructions will not be provided. In these sections, you must figure out how to successfully complete the steps so that you deploy the version of the café website that supports online orders into both AWS Regions.



A timer icon with the text "~ 60 minutes" indicating the duration of the challenge lab.

Begin Module 4 – Challenge Lab: Creating a Dynamic Website for the Café

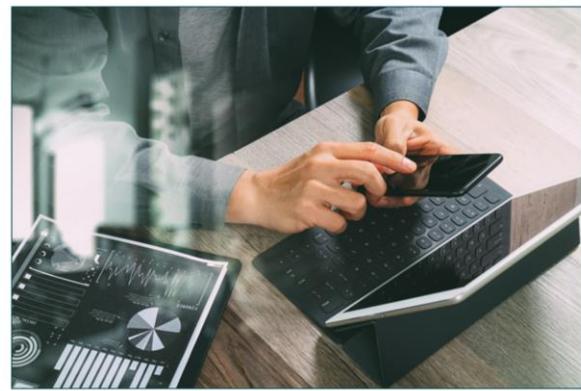
aws

© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

70

It is now time to start the challenge lab.

Challenge lab debrief: Key takeaways



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

71

Your educator might choose to lead a conversation about the key takeaways from this challenge lab after you have completed it.

Module wrap-up

Module 4: Adding a Compute Layer



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

It's now time to review the module and wrap up with a knowledge check and discussion of a practice certification exam question.

Module summary

In summary, in this module, you learned how to:

- Identify how Amazon Elastic Compute Cloud (Amazon EC2) can be used in an architecture
- Explain the value of using Amazon Machine Images (AMIs) to accelerate the creation and repeatability of infrastructure
- Differentiate between the EC2 instance types
- Recognize how to configure Amazon EC2 instances with user data
- Recognize storage solutions for Amazon EC2
- Describe EC2 pricing options
- Determine the placement group given an architectural consideration
- Launch an Amazon EC2 instance



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

73

In summary, in this module, you learned how to:

- Identify how Amazon Elastic Compute Cloud (Amazon EC2) can be used in an architecture
- Explain the value of using Amazon Machine Images (AMIs) to accelerate the creation and repeatability of infrastructure
- Differentiate between the EC2 instance types
- Recognize how to launch Amazon EC2 instances with user data
- Recognize storage solutions for Amazon EC2 (Amazon Elastic Block Store, or Amazon EBS, and Amazon Elastic File System, or Amazon EFS)
- Describe EC2 pricing options
- Determine the placement group given an architectural consideration
- Launch an Amazon EC2 instance

Complete the knowledge check



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

/4

It is now time to complete the knowledge check for this module.

Sample exam question



A Solutions Architect wants to design a solution to save costs for EC2 instances that do not need to run during a 2-week company shutdown. The applications running on the instances store data in instance memory (RAM) that must be present when the instances resume operation.

Which approach should the Solutions Architect recommend to shut down and resume the instances?

Choice	Response
A	Modify the application to store the data on instance store volumes. Reattach the volumes while restarting them.
B	Snapshot the instances before stopping them. Restore the snapshot after restarting the instances.
C	Run the applications on instances enabled for hibernation. Hibernate the instances before the shutdown.
D	Note the Availability Zone for each instance before stopping it. Restart the instances in the same Availability Zones after the shutdown.

© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

75

Look at the answer choices and rule them out based on the keywords that were previously highlighted.

Sample exam question answer



A Solutions Architect wants to design a solution to save costs for EC2 instances that do not need to run during a 2-week company shutdown. The applications running on the instances store data in instance memory (RAM) that must be present when the instances resume operation.

Which approach should the Solutions Architect recommend to shut down and resume the instances?

The correct answer is C.

The keywords in the question are “instance memory (RAM)”, “must be present when the instances resume operation”, and “shut down and resume”.

© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

76

The following are the keywords to recognize: **“instance memory (RAM)”, “must be present when the instances resume operation”, “shut down and resume”**

The correct answer is C. Hibernation saves the contents from the instance memory (RAM) to your Amazon EBS root volume, which persists after shutdown. When you start your instance, the Amazon EBS root volume is restored to its previous state, and the RAM contents are reloaded.

Additional resources

- [Amazon EC2 User Guide for Linux Instances](#)
- [Amazon EC2 User Guide for Windows Instances](#)
- [Amazon EC2 FAQs](#)
- [EC2 Image Builder User Guide](#)
- [EC2 Image Builder FAQs](#)
- [AWS Compute Optimizer User Guide](#)
- [AWS Compute Optimizer FAQs](#)
- [How AWS Pricing Works](#)



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

77

If you want to learn more about the topics covered in this module, you might find the following additional resources helpful:

- [Amazon EC2 User Guide for Linux Instances](#)
- [Amazon EC2 User Guide for Windows Instances](#)
- [Amazon EC2 FAQs](#)
- [EC2 Image Builder User Guide](#)
- [EC2 Image Builder FAQs](#)
- [AWS Compute Optimizer User Guide](#)
- [AWS Compute Optimizer FAQs](#)
- [How AWS Pricing Works](#)



Thank you

Corrections, feedback, or other questions?
Contact us at <https://support.aws.amazon.com/#/contacts/aws-training>.
All trademarks are the property of their owners.

© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

Thank you for completing this module.



**AWS Academy Cloud Architecting
Module 05 Student Guide
Version 2.0.8**

200-ACACAD-20-EN-SG

© 2022 Amazon Web Services, Inc. or its affiliates. All rights reserved.

This work may not be reproduced or redistributed, in whole or in part, without prior written permission from Amazon Web Services, Inc. Commercial copying, lending, or selling is prohibited.

All trademarks are the property of their owners.

Contents

Module 5: Creating the Database Layer	4
---------------------------------------	---



Module 5: Adding a Database Layer

AWS Academy Cloud Architecting

© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

Welcome to Module 5: Adding a Database Layer.

Module overview

Sections

1. Architectural need
2. Database layer considerations
3. Amazon RDS
4. Amazon DynamoDB
5. Database security controls
6. Migrating data into AWS databases

Demonstration

- Amazon RDS Automated Backup and Read Replicas

Labs

- Guided Lab: Creating an Amazon RDS Database
- Challenge Lab: Migrating a Database to Amazon RDS



Knowledge check



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

2

This module includes the following sections:

1. Architectural need
2. Database layer considerations
3. Amazon RDS
4. Amazon DynamoDB
5. Database security controls
6. Migrating data into AWS databases

This module also includes:

- A demonstration of Amazon Relational Database Service (Amazon RDS) automated backup and read replica configuration.
- A guided lab where you create an Amazon RDS database and connect to it by using a simple web application.
- A challenge lab that challenges you to migrate data from a database that runs on an Amazon Elastic Compute Cloud (Amazon EC2) instance to an Amazon RDS database.

Finally, you will be asked to complete a knowledge check to test your understanding of the key concepts that are covered in this module.

Module objectives

At the end of this module, you should be able to:

- Compare database types
- Differentiate between managed versus unmanaged services
- Explain when to use Amazon Relational Database Service (Amazon RDS)
- Explain when to use Amazon DynamoDB
- Describe available database security controls
- Describe how to migrate data into Amazon Web Services (AWS) databases
- Deploy a database server



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

3

At the end of this module, you should be able to:

- Compare database types
- Differentiate between managed versus unmanaged services
- Explain when to use Amazon Relational Database Service (Amazon RDS)
- Explain when to use Amazon DynamoDB
- Describe available database security controls
- Describe how to migrate data into Amazon Web Services (AWS) databases
- Deploy a database server

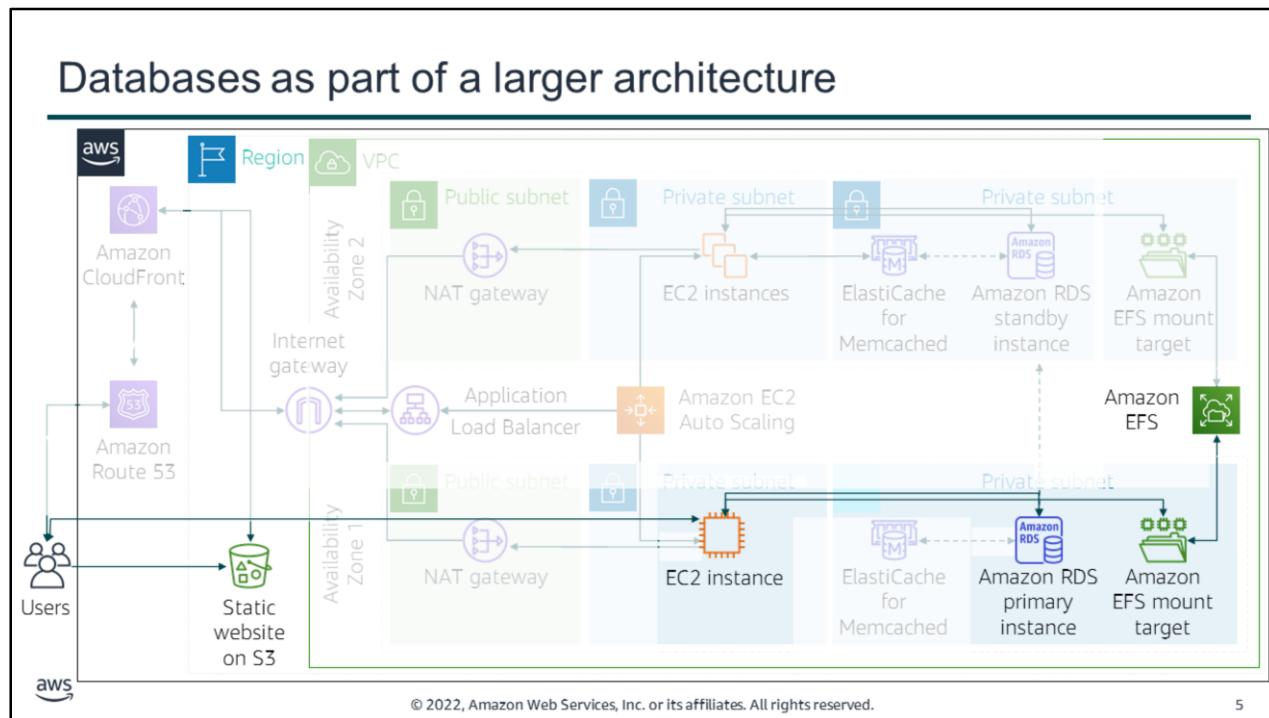
Section 1: Architectural need

Module 5: Adding a Database Layer



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

Introducing Section 1: Architectural need.



As each module introduces new features, those parts of this larger diagram will be unveiled.

In this module, you will learn how to create an architecture that uses AWS database services. Different relational and non-relational database service options—including Amazon RDS—are discussed. As you learn about the essential benefits of the various offerings, you will be able to make a more informed decision about the database service that best meets your particular architectural needs.

Café business requirement

The café needs a database solution that is easier to maintain, and that provides essential features such as durability, scalability, and high performance.



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

6

Ever since the cafe added the ability to place online orders to their website, the café staff have noticed that business has increased. In addition, they discovered that the order history that's stored in the database—which they installed on the same EC2 instance where the web server is running—provides valuable business information. Martha uses it for accounting purposes, and Frank looks at it occasionally to get an idea of how many of each dessert type he should bake.

However, Sofía has some concerns. The database needs to be upgraded and patched, and she doesn't always have time to do these tasks on a regular basis. Also, administering the database is a specialized skill, and training others to do it isn't something she wants to spend time on. Meanwhile, she is also concerned that the cafe isn't doing data backups as often as they should.

The café staff wants to reduce the labor costs associated with the technical learning investment that's needed to manage the database themselves. Thus, they decided that they need to use a managed database solution. Ideally, they will find one that provides essential features, such as durability, scalability, and high performance.

Throughout this module, you will learn details about the different database services that AWS provides, and the capabilities that are provided by these different services. With this understanding of the available options, you should be able to choose a database solution that can successfully meet these new business requirements.

Section 2: Database layer considerations

Module 5: Adding a Database Layer



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

Introducing Section 2: Database layer considerations.

Database considerations: Scalability



Scalability

How much throughput is needed?

Will the chosen solution be able to scale up later, if needed?

Total storage requirements

Object size and type

Durability



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

8

As an architect, you will often need to choose between different database types when you consider which type will best handle a particular workload. Before you choose a database, there are some key considerations that should inform your decision-making process.

First, consider the importance of *scalability*. With traditional on-premises databases, scaling up capacity can be a difficult task even for experienced database administrators. It can take hours, days, or weeks. The impact to database performance while it scales up can be unpredictable, and might require downtime. However, the importance of a properly scaled database cannot be overstated. If you underprovision your database, your applications might stop working. However, if you overprovision your database, you increase your upfront costs by procuring resources that you do not need, which violates the cost-optimization principle of the AWS Well-Architected Framework.

Ideally, you would choose a database solution that has the resources to handle the needed *throughput* at launch, and that can also be easily *scaled* up later if your throughput needs increase.

Another nice-to-have capability is the ability to scale down the provisioned database capacity if your throughput requirements or database load later decrease. This enables you to immediately realize cost savings by lowering provisioned capacity. An automated scaling solution could reduce costs and labor overhead.

Database considerations: Storage requirements



Scalability

How large does the database need to be?

Total storage requirements

Will it need to store GB, TB, or petabytes of data?



Object size and type



Durability



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

9

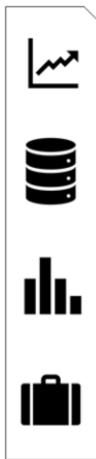
Second, when you must choose a database to handle a certain type of workload, consider the storage requirements of your workload.

How large must the database be to handle your data requirements? Does it need to store gigabytes? Terabytes? Petabytes?

Different database architectures support different maximum data capacities. Some database designs are ideal for traditional applications, while others are ideal for caching or session management. Still others are ideal for Internet of Things (IoT) or big data applications.

Understanding your total storage requirements is essential for choosing a database.

Database considerations: Object size and type



Scalability



Total storage requirements



Object size and type



Durability

Do you need to store simple data structures, large data objects, or both?



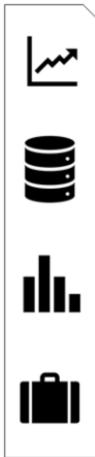
© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

10

Third, when you choose a database for your specific workloads, consider the size and the type of the objects you must store.

Do you need to store simple data structures, large data objects, or both?

Database considerations: Durability



Scalability

What level of **data durability, data availability, and recoverability** is required?

Total storage requirements

Do regulatory obligations apply?

Object size and type



Durability



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

11

Finally, as a fourth consideration for your database choice, think about the durability requirements for the data you will store.

Data durability refers to the assurance that your data will not be lost, and *data availability* refers to your ability to access your data when you want to. What level of data durability and data availability do you need? If the data you will store is absolutely critical to your business, you should choose a database solution that stores multiple redundant copies of your data across multiple geographically separated physical locations. This solution will usually result in an increased cost, so it is important to balance your business needs with cost considerations.

Another important consideration is to know whether data residency or *regulatory obligations* apply to your data. For example, are there regional data privacy laws that you must comply with? If so, choose a database solution that can support compliance.

Database types

Now that you reviewed key considerations, consider the two categories of database options available:

Relational

Traditional examples:

Microsoft SQL Server
Oracle Database
MySQL

Non-Relational

Traditional examples:

MongoDB
Cassandra
Redis



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

12

You can choose from many types of databases, many of which are purpose-built. However, database types typically fall into one of two broad categories: relational or non-relational.

Relational database systems are the most familiar type of databases to most people. Traditional examples include Microsoft SQL Server, Oracle Database, and MySQL.

Non-relational databases were developed more recently, but have been around for a few decades. They play an essential role in the modern computing landscape. Examples include MongoDB, Cassandra, and Redis.

Relational database type

Benefits:

- Ease of use
- Data integrity
- Reduced data storage
- Common language (structured query language, or SQL)

Relational is ideal when you:

- Need strict schema rules, ACID compliance, and data quality enforcement
- Do not need extreme read/write capacity
- Do not need extreme performance
 - An RDBMS can be the best, lowest-effort solution



Relational database management system (RDBMS)

© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

13

Relational databases are sometimes referred to as *relational database management system* (RDBMS). They are still the single most-popular and most-used database category in the world. In a 2019 [DB-Engines Ranking study](#), they were found to represent over 75 percent of the popularity score across all categories. The top four were Oracle, MySQL, Microsoft SQL Server, and PostgreSQL—which are all relational databases.

Relational databases maintain their popularity for many reasons. These reasons include their *ease of use*, data integrity controls, excellence at reducing overall data storage, and their support for structured query language (SQL). SQL is a popular language for querying or interacting with structured data stored in RDBMSs, and it is supported by most vendors.

If your use case lends itself well to *strict schema rules*, where the schema structure is well defined and does not need to change often, a relational database would be a good choice. Relational databases transactions are also *ACID*-compliant, which means that they ensure data integrity by providing transactions that are *atomic*, *consistent*, *isolated*, and *durable*. However, if your application needs extreme read/write capacity, a relational database might not be the right choice. Finally, extreme performance is a characteristic of relational databases when you compare them to other options. However, a relational database can be the best low-effort solution for many use cases.

Non-relational database type

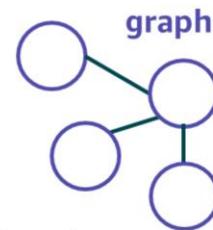
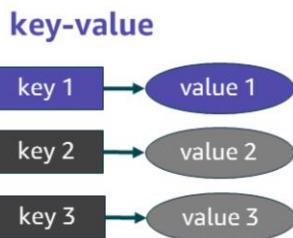
Benefits

- Flexibility
- Scalability
- High performance
- Highly functional APIs

Non-relational is ideal when:

- Database must scale **horizontally** to handle massive data volume
- Data does not lend itself well to traditional schemas
- Read/write rates exceed what can be economically supported through traditional RDBMS

Example models



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

14

Non-relational databases are sometimes called *NoSQL databases*. They are different from relational databases because they are purpose-built for specific data models and have flexible schemas for building modern applications. Non-relational databases are widely appreciated for their *flexibility, scalability, high performance, and highly functional APIs*.

They use a variety of data models, including key-value, graph, document, in-memory, and search. NoSQL schemas are dynamic. For example, a row does not need to contain data for each column. Also, a NoSQL database can *scale horizontally* by increasing the number of servers that it runs on.

Non-relational databases have *flexible schemas*, where each object can have a different structure. They can store structured data, such as relational database records; semistructured data, such as JavaScript Object Notation (JSON) documents; and unstructured data, such as photo files or email messages. Non-relational databases work well when the data might have structural inconsistencies.

Finally, non-relational databases are optimized for specific data models and access patterns that enable *higher performance* instead of trying to achieve the functionality of relational databases.

Amazon database options

More database options exist—these options are common examples

Relational databases



Amazon RDS



Amazon Redshift



Amazon Aurora

Non-relational databases



Amazon DynamoDB



Amazon ElastiCache



Amazon Neptune

Focus in this module



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

15

Now that you learned about the distinctions between relational and non-relational database, consider some example Amazon database offerings that fit into those two categories.

Among the more commonly used AWS *relational* database offerings are Amazon RDS, Amazon Redshift, and Amazon Aurora.

Similarly, some of the more commonly used *non-relational* databases are Amazon DynamoDB, Amazon ElastiCache, and Amazon Neptune.

In this module, you will explore both Amazon RDS and Amazon DynamoDB in more depth. For a full listing of database services that are available on AWS and their most common uses cases, see the [AWS Databases – Overview page](#).

Section 2 key takeaways



- When you choose a database, consider **scalability**, **storage requirements**, the **type and size of objects** to be stored, and **durability requirements**
- **Relational databases** have strict schema rules, provide data integrity, and support SQL
- **Non-relational databases** scale horizontally, provide higher scalability and flexibility, and work well for semistructured and unstructured data

© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

16

Some key takeaways from this section of the module include:

- When you choose a database, consider scalability, storage requirements, the type and size of objects to be stored, and durability requirements
- Relational databases have strict schema rules, provide data integrity, and support SQL
- Non-relational databases scale horizontally, provide higher scalability and flexibility, and work well for semistructured and unstructured data

Section 3: Amazon RDS

Module 5: Adding a Database Layer



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

Introducing Section 3: Amazon RDS.

Amazon RDS



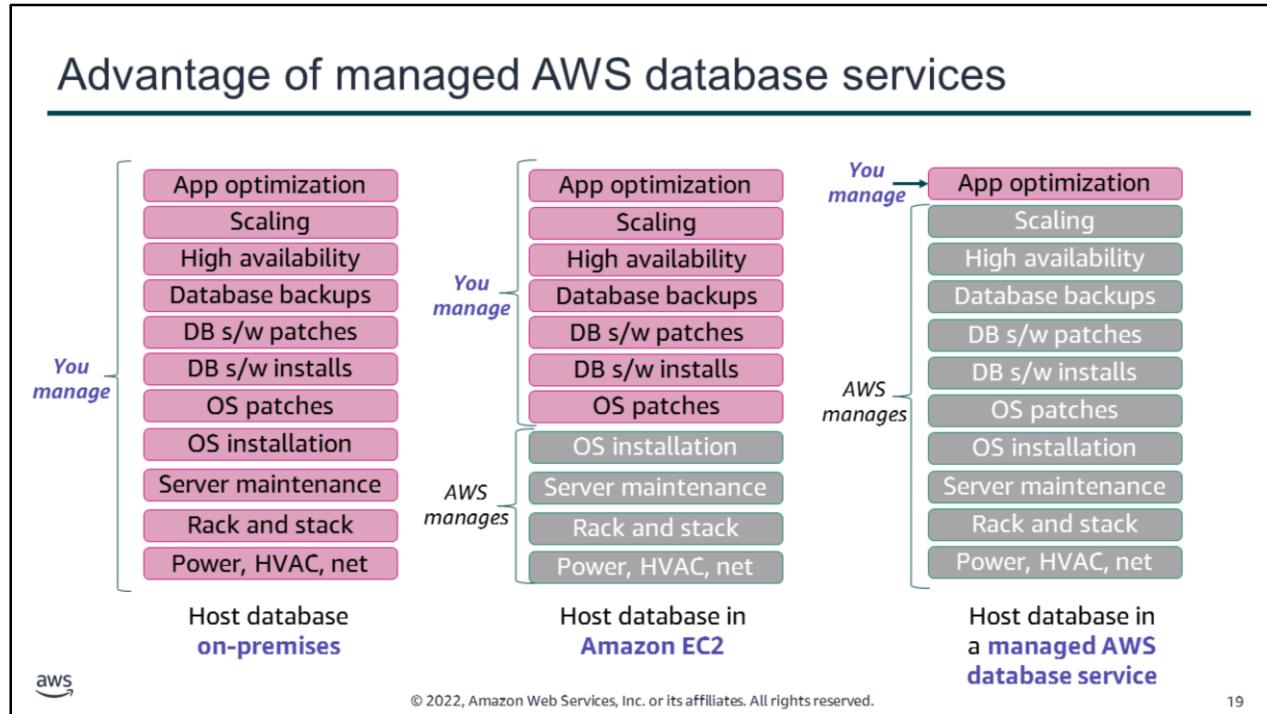
Amazon RDS is a **fully managed relational database service**.



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

18

Amazon RDS is a fully managed relational database service that creates and operates a relational database in the cloud. However, before you learn more details about Amazon RDS, you will first review the advantages of Amazon RDS as a *managed* database service.



You will now consider some of the advantages of using managed AWS database services.

In the use case on the left, you host the database in your *on-premises* data center. In this case, you are responsible for everything. You must power the racks of physical servers, maintain the operating systems of the servers where the databases run, and performing database software installations and patches. You must also perform backups, configure high availability and scaling solutions, and optimize the applications that use the database.

In the center use case, you install a *database on one or more Amazon EC2 instances*. In this case, AWS handles the maintenance of the physical data center environment and the OS is pre-installed on the EC2 instance that you launch. However, you are still responsible for every configuration layer above the OS installation, which means that you must manage many resources manually.

The *AWS managed database offering* is in the use case on the right. These solutions provide high availability, scalability, and database backups as built-in options that you can configure. AWS is responsible for handling common and repetitive database administration tasks. You are only responsible for optimizing your application, and making sure that the database layer works efficiently for your application.

Amazon RDS characteristics

**Access pattern**

Transactional

Light analytics

**Data size**

Low-TB range

**Performance**

Mid to high throughput

Low latency

**Business use cases**

Transactional

OLAP



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

20

Amazon Relational Database Service (Amazon RDS) is a fully managed relational database service.

Amazon RDS is typically used when the access pattern is transactional, or for light analytics.

As a relational database, the ideal data size ranges up to the low-terabyte range. As storage requirements grow, you can provision additional storage. The Amazon Aurora engine will automatically grow the size of your database volume as your database storage needs grow. It scales up to a maximum of 64 TB, or a maximum you define. The MySQL, MariaDB, Oracle, and PostgreSQL engines enable you to scale up to 32 TB of storage. Microsoft SQL Server supports up to 16 TB. Storage scaling is dynamic, with zero downtime.

In terms of *performance*, Amazon RDS offers two options. First, it offers the general-purpose solid state drive (SSD)-backed storage option. The SSD option delivers a consistent baseline of 3 IOPS per provisioned GB, and it can burst up to 3,000 IOPS above the baseline. This storage type is suitable for a range of database workloads. There is also a Provisioned IOPS SSD storage option, which works well for I/O-intensive transactional database workloads.

Amazon RDS: Uses and database types



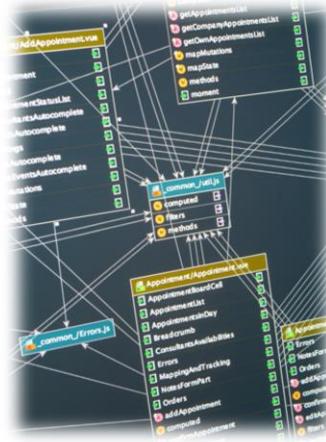
Amazon
RDS

Works well for applications that:

- Have more complex data
- Need to combine and join datasets
- Need enforced syntax rules

Six database types supported:

- Microsoft SQL Server
- Oracle
- MySQL
- PostgreSQL
- Aurora
- MariaDB



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

21

As a relational database offering, Amazon RDS is a good choice for applications that have complex, well-structured data. Amazon RDS is a good choice if your workloads must frequently combine and join datasets, and must have syntax rules that are strictly enforced. For example, Amazon RDS is frequently used to back traditional applications, enterprise resource planning (ERP) applications, customer relationship management (CRM) applications, and ecommerce applications.

Amazon RDS is available with six database engines to choose from, including Microsoft SQL Server, Oracle, MySQL, PostgreSQL, Amazon Aurora, and MariaDB.

Database instance sizing			
T family		M family	R family
Type	Burstable instances	General-purpose instances	Memory-optimized instances
Sizing	1 vCPU/1 GB RAM to 8 vCPU 32 GB RAM	2 vCPU/8 GB RAM to 96 vCPU 384 GB RAM	2 vCPU/16 GB RAM to 96 vCPU 768 GB RAM
Networking	Moderate performance	High performance	High performance
Ideal Workload	Smaller or variable	CPU-intensive	Query-intensive, high connection counts
Highlights	T3 can burst above baseline for extra charge	M5 offers up to 96 vCPU	R5 offers up to 96 vCPU 768 GiB RAM

 © 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved. 22

All Amazon RDS database types run on a serve. The exception is Aurora, which can run as a serverless option. Amazon RDS is available on several database instance types, which are optimized for different kinds of workloads.

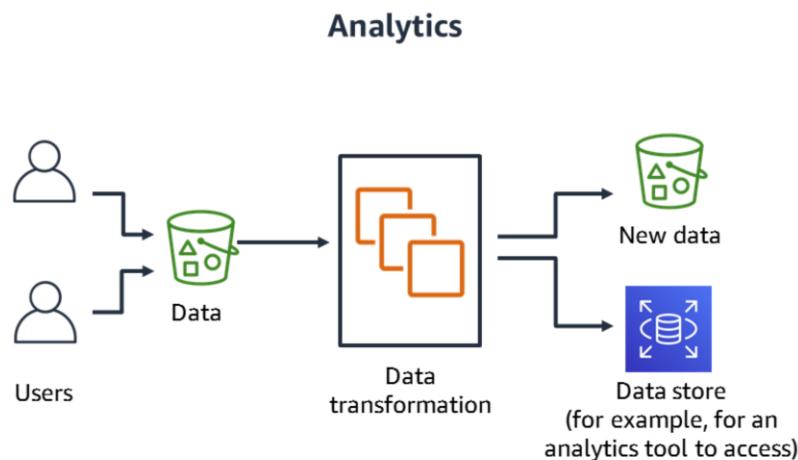
The T family of instance types provides a baseline level of CPU performance with the ability to burst CPU usage at any time for as long as needed. For example, T3 instances offer a balance of compute, memory, and network resources and work well for database workloads with moderate CPU usage that experience temporary spikes in use.

The M family of instances is another general-purpose option. However, the M family provides additional options for CPU-intensive workloads. M instances are a good choice for small and midsize databases for open source or enterprise applications.

Finally, the R family instances are optimized for memory-intensive database workloads.

Some database engines support additional database instance classes. For more information, see [Amazon RDS DB Instances](#), and specifically see the [Choosing the DB instance Class](#) AWS Documentation page for instance class details.

Amazon RDS: Example use case



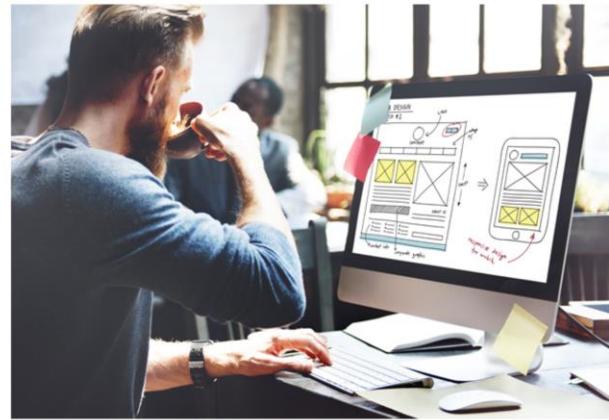
© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

23

Consider an example use case where Amazon RDS is used as part of a deployed solution to a business challenge.

In this example, a company wants to gain meaningful insight from some of their business data. Users upload the data to an Amazon Simple Storage Service (Amazon S3) bucket. Then, a cluster of Amazon EC2 instances is used to transform the data. This might occur as a batch process at a regular interval. The transformed data is then stored in a different S3 bucket. However, some of the transformed data is also inserted into an RDS instance, which is labeled *Data store*. This RDS instance can then be queried by using analytics software or an AWS analytics service that can make SQL queries to access the data for business insights.

Module 5 - Guided Lab: Creating an Amazon RDS Database



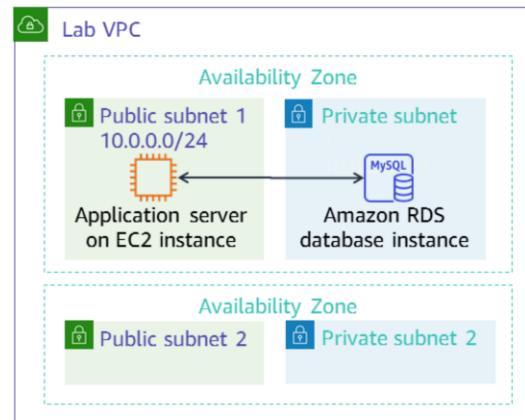
© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

24

You will now complete Module 5 – Guided Lab: Creating an Amazon RDS Database.

Guided lab: Tasks

1. Creating an Amazon RDS database
2. Configuring web application communication with a database instance

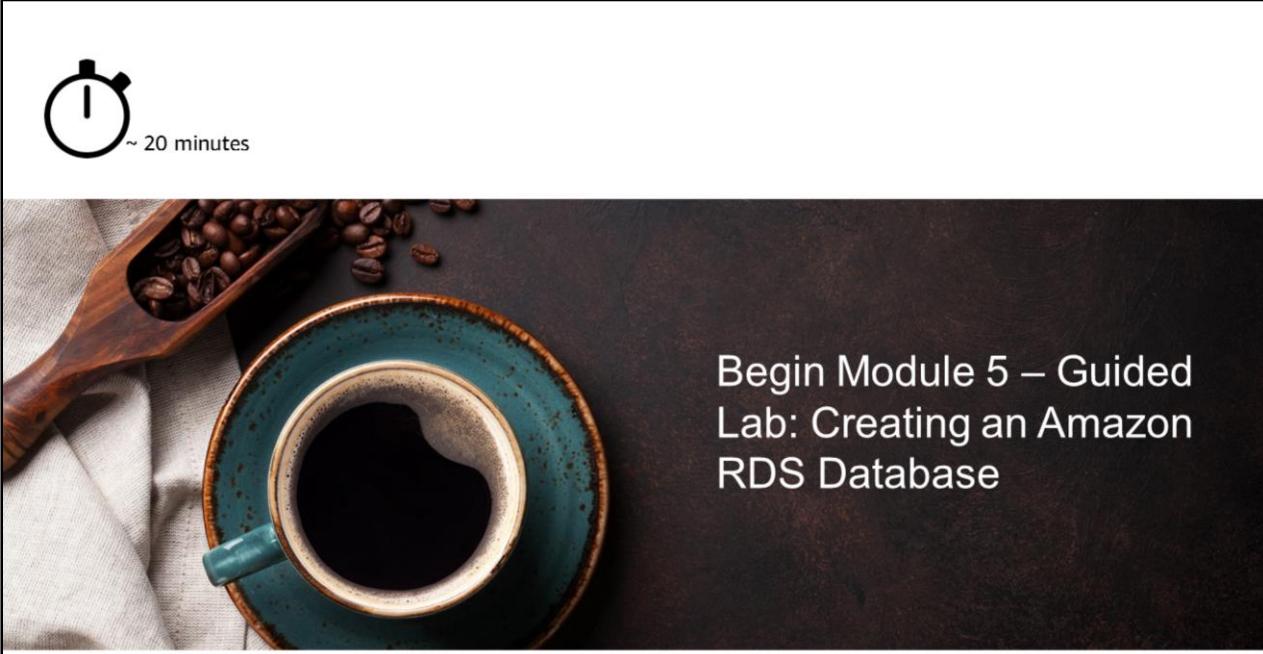


© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

25

In this guided lab, you will complete the following tasks:

1. Creating an Amazon RDS database
2. Configuring web application communication with a database instance



~ 20 minutes

Begin Module 5 – Guided Lab: Creating an Amazon RDS Database

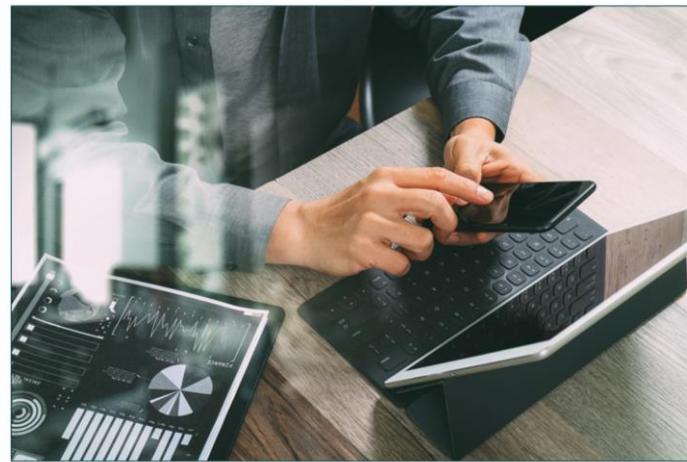
aws

© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

26

It is now time to start the guided lab.

Guided lab debrief: Key takeaways



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

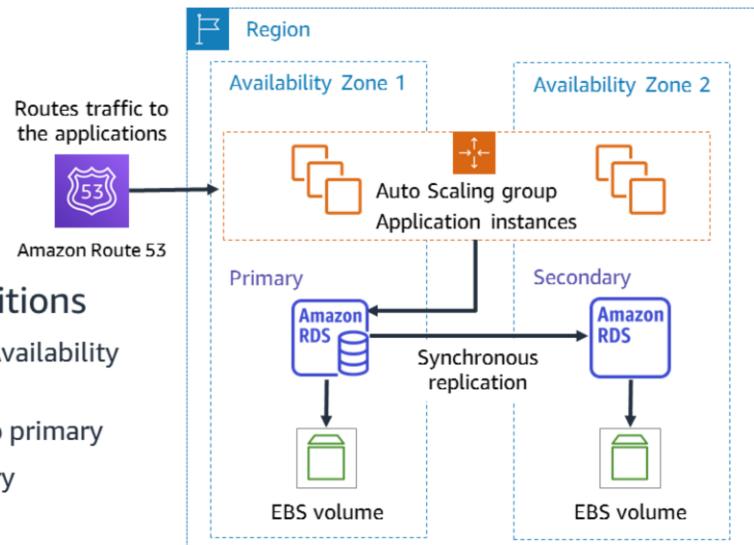
27

Your educator might choose to lead a conversation about the key takeaways from this guided lab after you have completed it.

Multi-AZ deployment for high availability

Benefits

- Enhanced durability
- Increased availability
- Fail over to standby occurs automatically



Automated failover conditions

- Loss of availability in primary Availability Zone
- Loss of network connectivity to primary
- Compute unit failure on primary
- Storage failure on primary



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

28

Amazon RDS provides high availability by implementing a Multi-AZ approach. Amazon RDS automatically provisions and maintains a synchronous standby instance in a different Availability Zone. The primary DB instance is synchronously replicated across Availability Zones to the standby instance 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. It helps protect your databases against DB instance failure and Availability Zone disruption. It also provides enhanced availability and durability for database instances, making it a good choice for production database workloads.

If there is an infrastructure failure, Amazon RDS performs an automatic failover to the standby instance. You can then resume database operations when the failover is complete. The endpoint for your DB instance remains the same after a failover, so your application can resume database operations without manual administrative intervention.

Failover conditions include the loss of availability in the primary Availability Zone, a loss of network connectivity to the primary database, compute unit failure on the primary instance, or storage failure on the primary instance.

Read replicas for performance

Benefits

- Enhanced performance
- Increased availability
- Designed for security

Supported by

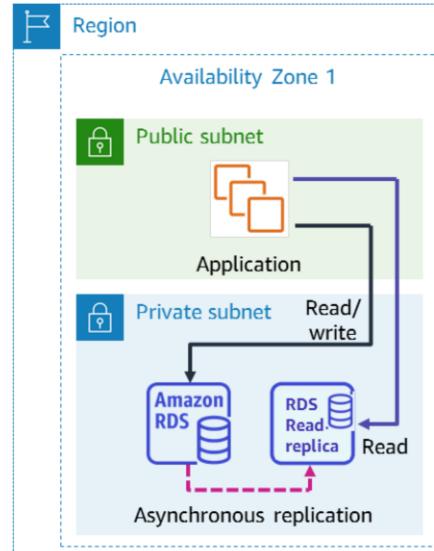
- MySQL
- MariaDB
- PostgreSQL
- Oracle

Limits

- Five read replicas per primary
- For strict read-after-write consistency, read from the primary



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

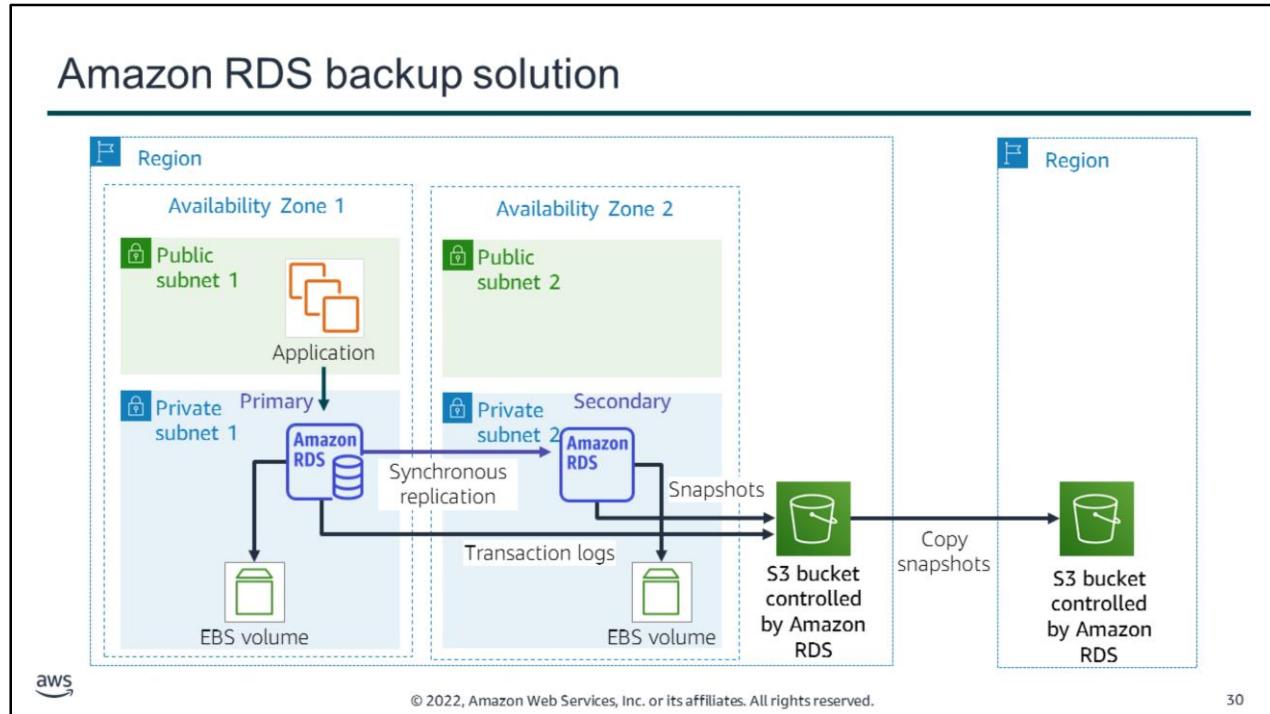


29

Amazon RDS enables you to use a source database instance to create a special type of database instance, which is called a read replica. Updates that are made to the source DB instance are asynchronously copied to the read replica.

Using read replicas enhances performance. For example, you can reduce the load on your source database instance by routing read queries from your applications to the read replica. Read replicas also increase availability. For read-heavy database workloads, you can elastically scale out beyond the capacity constraints of a single database instance. Read replicas can also be manually promoted to become standalone database instances, when needed.

Read replicas are available in Amazon RDS for MySQL, MariaDB, PostgreSQL, and Oracle. Each of these database engines enables up to a maximum of five read replicas per primary database. If you need strict read-after-write consistency (that is, what you read is always what you just wrote, even if you read immediately), then you should read from the main DB instance. Otherwise, you can spread the load, and read from a read replica.



Now, consider an example architecture for a backup solution that is based on Amazon RDS.

Here, snapshots and transaction logs from the RDS database are stored in an S3 bucket that is controlled by Amazon RDS. With Amazon RDS, you can also specify it to copy automated or manual DB to a second AWS Region. You can also copy the snapshots to separate AWS accounts.

For more information, see [Copying a Snapshot](#) in the AWS Documentation.

Demonstration: Amazon RDS Automated Backup and Read Replicas

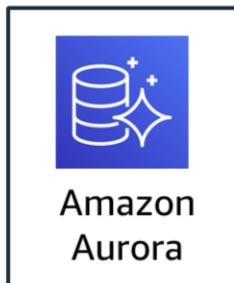


© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

31

Now, the educator might choose to demonstrate the configuration of automated backups and read replicas on Amazon RDS by using the AWS Management Console.

Amazon Aurora



Amazon Aurora is a fully managed, MySQL- and PostgreSQL-compatible, relational database engine.

- Used for online transactional processing (OLTP)
- Provides up to five times the throughput of MySQL*
- Provides up to three times the throughput of PostgreSQL*
- Replicates data six ways across three Availability Zones
- Requires little change to your existing application

* Benchmarking details are available for [MySQL](#) and [PostgreSQL](#).



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

32

Amazon Aurora is one of the database engine options for Amazon RDS.

Aurora is a MySQL- and PostgreSQL-compatible relational database that is built for the cloud. It combines the performance and availability of traditional enterprise databases with the simplicity and cost-effectiveness of open source databases.

It is up to five times faster than standard MySQL databases and three times faster than standard PostgreSQL databases. It provides the security, availability, and reliability of commercial databases, but at 1/10th the cost. Detailed instructions on this benchmark and how to replicate it yourself are provided in the [Amazon Aurora MySQL Performance Benchmarking Guide](#)[MySQL](#) and [Amazon Aurora PostgreSQL Performance Benchmarking Guide](#)[PostgreSQL](#).

Aurora features a distributed, fault-tolerant, self-healing storage system that automatically scales up to 64 TB per database instance. It delivers high performance and availability, with up to 15 low-latency read replicas, point-in-time recovery, continuous backup to Amazon S3, and replication across three Availability Zones.

Aurora is often used for online transaction processing (OLTP). OLTP systems must be able to handle a high volume of concurrent users, and be able to run insert and update requests. A common OLTP example is an order entry system. OLTP is often contrasted with online analytics processing (OLAP), which is characterized by a smaller volume of more complex queries.

Amazon Redshift

Amazon Redshift is a **data warehousing** service.



- Is used for online analytics processing (OLAP)
- Stores very large datasets
 - Store highly structured, frequently accessed data in Amazon Redshift
 - Can also store exabytes of structured, semistructured, and unstructured data in Amazon S3



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

33

Amazon Redshift is a different AWS relational database offering. It does not run on Amazon RDS.

Amazon Redshift provides a petabyte-scale data warehouse and data lake analytics. Amazon Redshift is typically used to store frequently accessed, highly structured data. Amazon Redshift can also access data directly in Amazon S3, so you can also maintain exabytes of structured, semistructured, and unstructured data in Amazon S3.

Amazon Redshift provides consistently fast performance, even with thousands of concurrent queries. It performs consistently whether you are querying data in the Amazon Redshift data warehouse, or querying data in your Amazon S3 data lake. You can query open file formats such as Parquet, JSON, Avro, CSV. You can also query Amazon S3 directly by using SQL.

Section 3 key takeaways



- Managed AWS database services handle administration tasks so you can focus on your applications
- Amazon RDS supports Microsoft SQL Server, Oracle, MySQL, PostgreSQL, Aurora, and MariaDB
- Amazon RDS Multi-AZ deployments provide high availability with automatic failover
- You can have up to five read replicas per primary database to improve Amazon RDS performance
- Amazon Aurora is a fully managed, MySQL- and PostgreSQL-compatible, relational database engine
- Amazon Redshift is a relational database offering for data warehousing

© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

34

Some key takeaways from this section of the module include:

- Managed AWS database services handle administration tasks so you can focus on your applications
- Amazon RDS supports Microsoft SQL Server, Oracle, MySQL, PostgreSQL, Aurora, and MariaDB
- Amazon RDS Multi-AZ deployments provide high availability with automatic failover
- You can have up to five read replicas per primary database to improve performance
- Amazon Aurora is a fully managed, MySQL- and PostgreSQL-compatible, relational database engine
- Amazon Redshift is a data warehousing relational database offering

Section 4: Amazon DynamoDB

Module 5: Adding a Database Layer



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

Introducing Section 4: Amazon DynamoDB.

Amazon DynamoDB



Amazon
DynamoDB

A fully managed **non-relational** key-value and document database service.



Performance
at any scale
Extreme horizontal
scaling capability



Serverless
Event-driven
programming
(serverless
computing)



Enterprise-ready
Encryption, access
controls, backups



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

36

Amazon DynamoDB is a fully managed non-relational NoSQL database service. It provides fast and predictable performance with seamless scalability.

Key benefits include:

- Performance at scale
 - It offers consistent, single-digit millisecond response times
 - You can build applications with virtually unlimited throughput
- Serverless
 - No servers to provision, patch, or manage
 - No software to install, maintain, or operate
- Enterprise-ready
 - It supports ACID transactions
 - It encrypts all data at rest by default
 - Multi-Region replication (global tables) is available
 - It provides fine-grained identity and access control
 - You can perform full backups of data with no performance impact

Amazon DynamoDB characteristics

Works well for applications that:



- Have simple high-volume data (high-TB range)
- Must scale quickly
- Don't need complex joins
- Require ultra-high throughput and low latency

Key features

- NoSQL tables
- Items can have differing attributes
- In-memory caching
- Support for peaks of more than 20 million requests per second



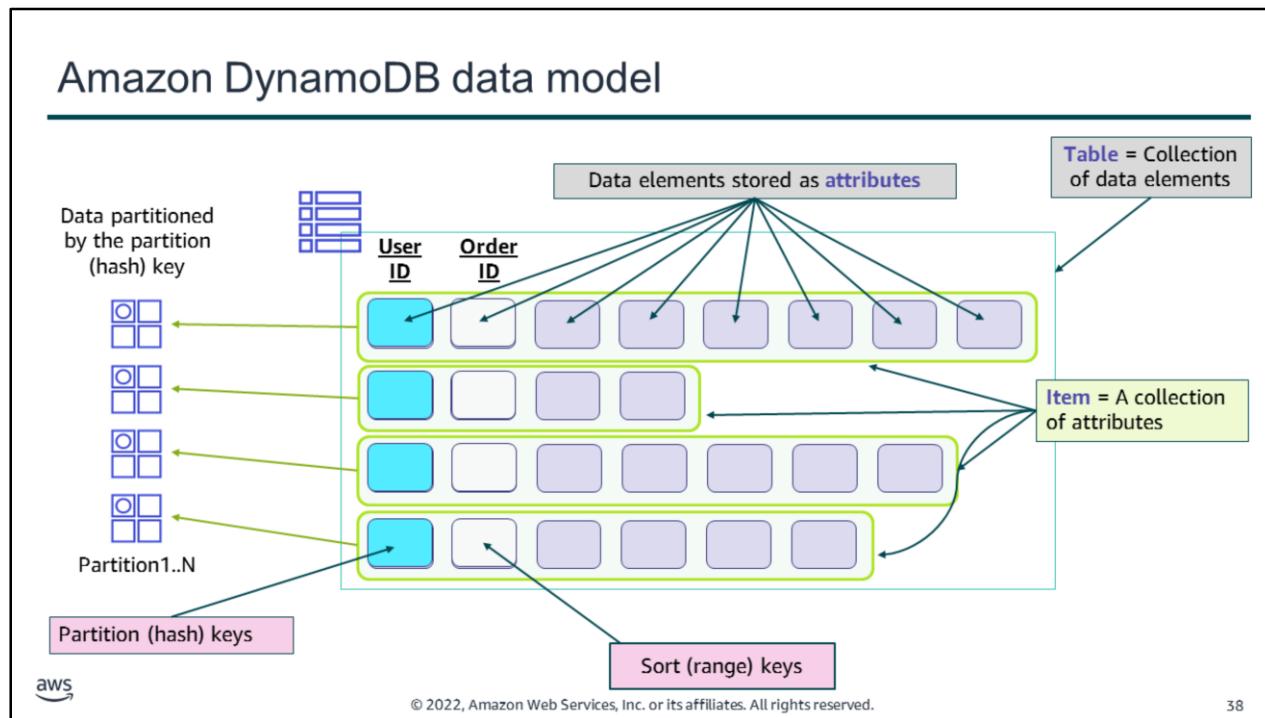
© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

37

DynamoDB works well for applications that handle a high volume of data and must scale quickly. It also provides ultra-high throughput and low latency, so it is a good option for gaming, adtech, mobile, and other applications that have these requirements. However, a relational database is likely a better choice if your workloads require complex joins.

DynamoDB tables do not have fixed schemas, and each item can have a different number of attributes.

Because DynamoDB provides in-memory caching and can handle more than 20 million requests per second, it is often used for applications that must maintain session state, for serverless web applications, and for microservices.



Now, consider the Amazon DynamoDB data model. Unlike relational databases—and because it is a NoSQL database—DynamoDB does not enforce strict schemas.

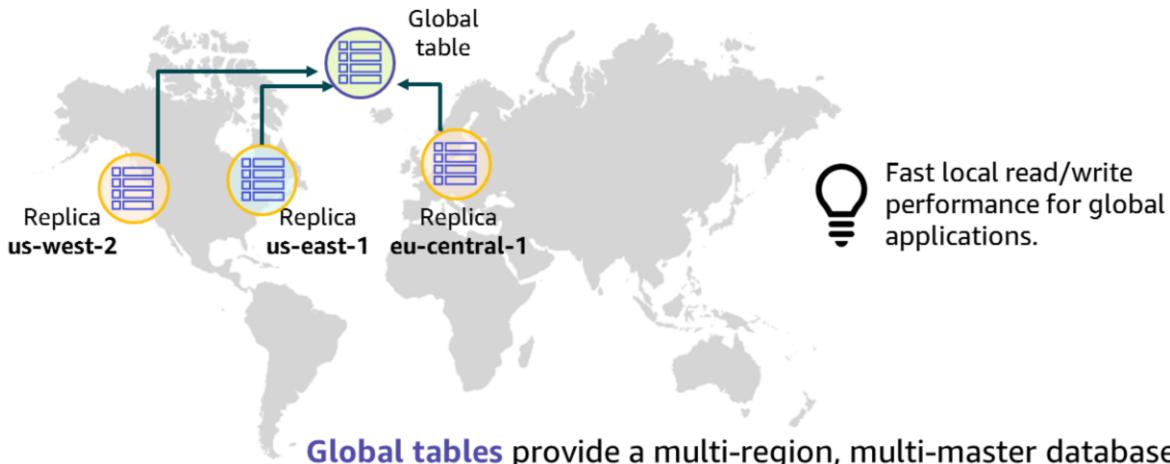
A DynamoDB table holds items. An item has a variable number of attributes. Each attribute consists of a key-value pair.

DynamoDB supports key-value GET/PUT operations that use a user-defined primary key (partition key, also known as a hash key). The primary key is the only required attribute for items in a table and it uniquely identifies each item. You specify the primary key when you create a table. In addition, DynamoDB provides flexible querying by allowing queries on non-primary key attributes using Global Secondary Indexes and Local Secondary Indexes.

Auto-partitioning occurs when the dataset size grows and as provisioned capacity increases.

Consider an example case where you create an Amazon DynamoDB table to hold order information. In the example, your table entries will always have a User ID and Order ID. However, some of the other details about an order—such as the user's email address—can only be collected for some orders, and not others. As additional items are added to the table, you can continue to collect new attributes. These attributes can include ones that you did not anticipate when you first created the table. This flexibility is what makes DynamoDB a good fit for rapidly changing unstructured, semistructured, and structured data.

Amazon DynamoDB global tables



Global tables provide a multi-region, multi-master database.



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

39

By default, Amazon DynamoDB replicates your data across multiple Availability Zones in a single Region. However, there might be occasions when you want to replicate your data across multiple Regions.

Amazon DynamoDB global tables provide a fully managed solution for deploying a multi-region, multi-master database. You do not need to build and maintain your own replication solution. When you create a global table, you specify the AWS Regions where you want the table to be available. DynamoDB performs all of the necessary tasks to create identical tables in these Regions. DynamoDB then propagates ongoing data changes to all the tables.

To illustrate one use case for a global table, suppose that you have a large customer base spread across three geographic areas—the US east coast, the US west coast, and western Europe. Customers must update their profile information while they use your application.

To handle this business requirement, you could create a global table that consists of your three Region-specific CustomerProfiles tables. DynamoDB would then automatically replicate data changes among those tables. Changes to CustomerProfiles data in one Region would seamlessly propagate to the other Regions. In addition, if one of the AWS Regions became temporarily unavailable, your customers could still access the same CustomerProfiles data in the other Regions.

Amazon DynamoDB use case 1

Leaderboards and Scoring

The diagram illustrates the interaction between three components: Players, Game servers, and Leaderboard. A Player icon points to two overlapping orange square icons representing Game servers. A double-headed arrow connects the Game servers to the Leaderboard icon, which is a blue square with a lightning bolt.

GameScores						
User Id	Game Title	Top Score	Top Score Date Time	Wins	Losses	...
"101"	"Galaxy Invaders"	5842	"2015-09-15:17:24:31"	21	72	...
"101"	"Meteor Blasters"	1000	"2015-10-22:23:18:01"	12	3	...
"101"	"Starship X"	24	"2015-08-31:13:14:21"	4	9	...
"102"	"Alien Adventure"	192	"2015-07-12:11:07:56"	32	192	...
"102"	"Galaxy Invaders"	0	"2015-09-18:07:33:42"	0	5	...
"103"	"Attack Ships"	3	"2015-10-19:01:13:24"	1	8	...
"103"	"Galaxy Invaders"	2317	"2015-09-11:06:53:00"	40	3	...
"103"	"Meteor Blasters"	723	"2015-10-19:01:13:24"	22	12	...
"103"	"Starship X"	42	"2015-07-11:06:53:00"	4	19	...
...

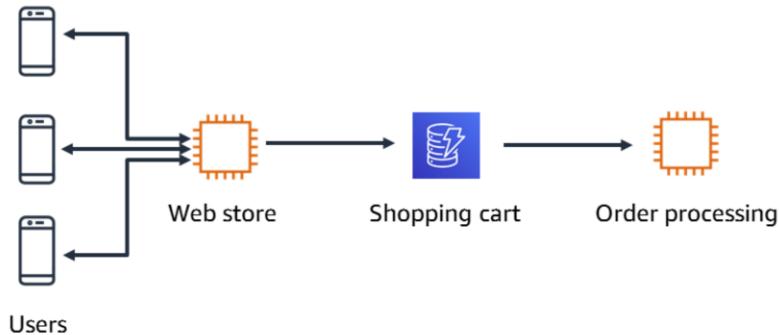
aws © 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved. 40

Gaming companies use Amazon DynamoDB in all parts of their game platforms, including game state, player data, session history, and leaderboards. You can scale reliably to millions of concurrent users and requests while you also ensure consistently low latency.

In the example use case, a game score leaderboard is maintained in Amazon DynamoDB. Players who are actively connected to the game servers might see some of this information presented in whatever way the gaming user interface (UI) presents it to them. The GameScores leaderboard is constantly updated when new top scores are achieved and each top scorer's win/loss record changes. As an example, consider Electronic Arts (EA), a large video game company with more than 300 million registered players around the world. For EA, high concurrency can mean more than 100,000 requests per second and millions of daily active users. See [DynamoDB Gaming Use Cases](#) for more details.

Amazon DynamoDB use case 2

Temporary Data (Online Cart)



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

41

Amazon DynamoDB was developed because customers needed a scalable and highly reliable key-value database to power essential Amazon ecommerce operations, such as the shopping cart.

In 2012, Amazon CTO Werner Vogels explained the business requirements of the Amazon shopping cart in a [blog post](#) when he announced the release of Amazon DynamoDB. "This non-relational, or NoSQL, database was targeted at use cases...such as the shopping cart....Any downtime or performance degradation...has an immediate financial impact and their fault-tolerance and performance requirements for their data systems are very strict. These services also require the ability to scale infrastructure incrementally to accommodate growth in request rates or dataset sizes. Another important requirement...was predictability."

The DynamoDB feature set continues to evolve. However, the core goals that inspired the development of the service remain relevant today to organizations that want to implement a processing solution for a web store shopping cart order.

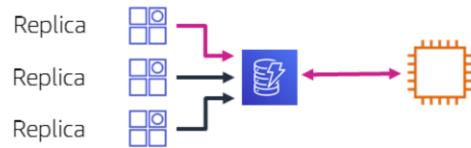
Amazon DynamoDB consistency options

Eventually consistent



The **default** setting. All copies of data usually reach consistency within **1 second**.

Strongly consistent



This feature is optional. Use for applications that require all reads to return a result that reflects all writes before the read.



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

42

Read consistency represents how and when the successful write or update of a data item is reflected in a subsequent read operation of that same item. Amazon DynamoDB exposes logic that enables you to specify the consistency characteristics you want for each read request in your application.

Eventually consistent reads

When you read data from a DynamoDB table, the response might not reflect the results of a recently completed write operation. The response might include some stale data. If you repeat your read request after a short time, the response should return the latest data.

Strongly consistent reads

When you request a strongly consistent read, DynamoDB returns a response with the most up-to-date data. The response reflects the updates from all previous write operations that were successful. A strongly consistent read might not be available if there is a network delay or outage.

DynamoDB uses eventually consistent reads, unless you specify otherwise. Read operations (such as GetItem, Query, and Scan) provide a ConsistentRead parameter. If you set this parameter to true, DynamoDB uses strongly consistent reads during the operation.

Class discussion: Which database should the café use?

Amazon
RDSAmazon
DynamoDB

© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

43

Now that you know the essential key features of both Amazon RDS and DynamoDB, consider which of these two database services would be a better match for the café's use case.

In an earlier module in the course, you completed a challenge lab where you deployed a MySQL database that runs on an EC2 instance. This database hosts a table that contains details about the café's menu. It also provides the data storage layer for the orders that customers place online.

If the café team decides that they want to migrate the database layer so they can use an Amazon managed database service, which of these two database options would be a better choice?

The educator will lead your class in a conversation about this topic. You are encouraged to participate in the dialog and explain the reasoning behind your answer.

Section 4 key takeaways



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

44

- Amazon DynamoDB is a fully managed non-relational **key-value** and document **NoSQL** database service.
- DynamoDB is **serverless**, provides extreme **horizontal scaling** and **low latency**.
- DynamoDB **global tables** ensure that data is replicated to multiple Regions.
- DynamoDB provides **eventual consistency** by default (in general, it is fully consistent for reads 1 second after the write). **Strong consistency** is also an option.

Some key takeaways from this section of the module include:

- Amazon DynamoDB is a fully managed non-relational key-value and document NoSQL database service.
- DynamoDB is serverless, provides extreme horizontal scaling and low latency.
- DynamoDB global tables ensure that data is replicated to multiple Regions.
- DynamoDB provides eventual consistency by default (in general, it is fully consistent for reads 1 second after the write). Strong consistency is also an option.

Section 5: Database security controls

Module 5: Adding a Database Layer



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

Introducing Section 5: Database security controls.

Securing Amazon RDS databases

Recommendations

- Run the RDS instance in a [virtual private cloud \(VPC\)](#)
 - Provides service isolation and IP firewall protection
- Use [AWS Identity and Access Management \(IAM\) policies](#) for authentication and access
 - Permissions determine who is allowed to manage Amazon RDS resources
- Use [security groups](#) to control what IP addresses or EC2 instances can connect to your databases
 - By default, network access is disabled
- Use [Secure Sockets Layer \(SSL\)](#) for encryption in transit
- Use Amazon RDS [encryption](#) on DB instances and snapshots to secure data at rest
- Use the [security features of your DB engine](#) to control who can log in to the databases on a DB instance
- Configure event notifications to alert you when important Amazon RDS events occur



Amazon
RDS



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

46

Security is a shared responsibility between you and AWS. AWS is responsible for security of the cloud, which means that AWS protects the infrastructure that runs Amazon RDS. Meanwhile, you are responsible for security in the cloud.

One security recommendation for Amazon RDS is to run your RDS instances in a virtual private cloud (VPC). A VPC enables you to place your instance in a private subnet, which secures it from public routes on the internet. The VPC also provides IP firewall protection and enables you to securely control the applicable network configuration.

In addition, we recommend that you:

- Use AWS Identity and Access Management (IAM) policies for authentication and to control access.
- Configure security groups to limit connections. Open the TCP port where the database is accessible. However, you limit where those connections can originate from.
- Use SSL connections to ensure that all communication to and from your database is secured.
- Use Amazon RDS encryption to encrypt your data and database snapshots.
- Use the security features of your DB engine, for example, to enforce password complexity.
- Enable event notifications on important events that can occur on your RDS instance. These events can include whether the instance was shut down, a backup was started, a failover occurred, the security group was changed, or your storage space is low.

Securing Amazon DynamoDB

Recommendations

- Use **IAM roles** to authenticate access
- Use **IAM policies**
 - To define fine-grain access permissions to use DynamoDB APIs
 - Define access at the table, item, or attribute level
 - Follow the **principle of granting least privilege**
- Configure **VPC endpoints**
 - Prevents connection traffic from traversing the open internet
 - VPC endpoint policies allow you to control and limit API access to a DynamoDB table
- Consider **client-side encryption**
 - Encrypt data as close as possible to its origin



Amazon
DynamoDB

Security provided by default

- **Encryption at rest** of all user data stored in tables, indexes, streams, and backups
- **Encryption in transit** – All communications to and from DynamoDB and other AWS resources use HTTPS



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

47

To secure Amazon DynamoDB, many of the same best practices that you should use to secure Amazon RDS also apply.

For example, use **IAM roles** to secure authentication, and **use IAM policies** to define access permissions.

If you only require access to DynamoDB from within a virtual private cloud (VPC), you should use a **VPC endpoint** to limit access from only the required VPC. Doing this prevents that traffic from traversing the open internet and being subject to that environment.

Also, If you store sensitive or confidential data in DynamoDB, you might want to encrypt that data as close as possible to its origin so that your data is protected throughout its lifecycle.

Note that DynamoDB provides certain security features by default. For example, DynamoDB protects user data stored at rest and also data in transit between on-premises clients and DynamoDB, and between DynamoDB and other AWS resources within the same AWS Region.

Read the [DynamoDB security best practices](#) documentation for additional details.

Section 6: Migrating data into AWS databases

Module 5: Adding a Database Layer



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

Introducing Section 6: Migrating data into AWS databases.

AWS Database Migration Service

- Use to migrate to and from most commercial and open source databases
- Migrate between databases on Amazon EC2, Amazon RDS, Amazon S3, and on-premises



AWS Database
Migration Service
(AWS DMS)

Example homogenous migration

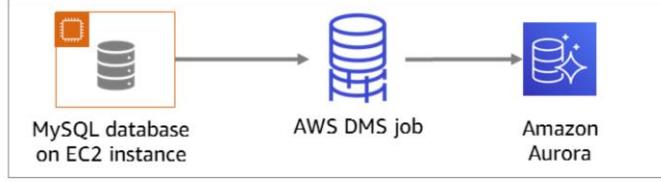


SOURCE
Oracle database
on-premises

AWS DMS job

TARGET
Amazon RDS for
Oracle database

Example heterogeneous migration



MySQL database
on EC2 instance

AWS DMS job

Amazon Aurora

aws

© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

49

You can use the AWS Database Migration Service (AWS DMS) to migrate or replicate existing databases to Amazon RDS. AWS DMS supports migration between the most widely used databases.

Supported *source databases* include Oracle, Microsoft SQL Server, MySQL, MariaDB, PostgreSQL, IBM Db2 LUW, SAP, MongoDB, and Amazon Aurora. *Target database* engines include Oracle, Microsoft SQL Server, PostgreSQL, MySQL, Amazon Redshift, SAP ASE, Amazon S3, and Amazon DynamoDB.

AWS DMS also supports both homogenous (same engine) migrations and heterogeneous (different engines) migrations.

The first example shown is a homogenous conversion, where an on-premises Oracle database is migrated to Amazon RDS for Oracle database.

The second example shows a heterogeneous migration, where a MySQL database that runs on an Amazon EC2 instance is migrated to Amazon Aurora.

AWS DMS key features

- Perform one-time migrations
- Or, accomplish continuous data replication
 - Example: Configure continuous data replication of an on-premises database to an RDS instance
- [AWS Schema Conversion Tool \(AWS SCT\)](#) supports changing the database engine between source and target
 - Typical migration major steps:
 1. Create a target database
 2. Migrate the database schema
 3. Set up the data replication process
 4. Initiate the data transfer, and confirm completion
 5. Switch production to the new database (for one-time migrations)



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

50

AWS DMS can be used to perform one-time migrations, but it can also be used to accomplish continuous data replication between two databases. For example, you could use it to configure continuous data replication of an on-premises database to an RDS instance.

When you want to perform a heterogeneous migration from one database engine to another, you might want to use the [AWS Schema Conversion Tool \(AWS SCT\)](#).

A typical database migration involves the following major steps:

1. Create a target database
2. Migrate the database schema
3. Set up the data replication process
4. Initiate the data transfer, and confirm completion
5. Switch production to the new database (for one-time migrations)

Using AWS Snowball Edge with AWS DMS

When migrating data is not practical:

- Database is too large
- Connection is too slow
- You have privacy and security concerns

Use [AWS Snowball Edge](#)

- Multi-terabyte transfer without using the internet



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

51

Larger data migrations can include many terabytes of information. This process can be difficult because of network bandwidth limits or the amount of data. AWS Database Migration Service (AWS DMS) can use [AWS Snowball Edge](#) and Amazon S3 to migrate large databases more quickly than other methods.

AWS Snowball is an AWS service that provides an Edge device that you use to transfer data to the cloud at faster-than-network speeds. An Edge device is an appliance that is owned by AWS. It provides large amounts of on-board storage. The Edge device also uses 256-bit encryption and an open standard Trusted Platform Module (TPM) to ensure both security and full chain of custody for your data.

When you use an Edge device, the data migration process includes the following stages:

- Use AWS SCT to extract the data locally and move it to the Edge device
- Ship the device back to AWS
- After AWS receives your shipment, the device automatically loads its data into an S3 bucket
- AWS DMS takes the files and migrates the data to the target data store

Module 5 - Challenge Lab: Migrating a Database to Amazon RDS



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

52

You will now complete Module 5 – Challenge Lab: Migrating a Database to Amazon RDS.

The business need: A managed database

The database that runs on the EC2 instance is becoming difficult for Sofía and Nikhil to maintain.



When Olivia visited the café recently, she told them about the features of Amazon RDS.

Sofía and Nikhil decided to migrate the café's database to Amazon RDS.



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

53

The administration of the café database is becoming difficult for Sofía and Nikhil. For example, they must work extra hours when the café is closed to do weekly database backups.

In addition, they recently struggled to install a required patch. Sofía and Nikhil almost did not complete it during the weekend afterhours window. Frank and Martha now realize that these maintenance tasks increase the business' labor costs because they must pay for the overtime hours that Sofía and Nikhil work.

Raquel, an AWS solutions architect, is a café customer and Sofía's friend. Sofía mentioned the topic of databases to Olivia during a recent conversation. Olivia suggested that they migrate the database to Amazon RDS.

This solution will reduce the burden of manually performing common database maintenance tasks, such as backups, patch installation, and upgrades. As a fully managed service, Amazon RDS performs these tasks automatically.

In this activity, you will take on the roles of Nikhil and Sofía, and work to migrate the café database to Amazon RDS.

Challenge lab: Tasks

1. Creating an RDS instance
2. Analyzing the existing café application deployment
3. Working with the database on the EC2 instance
4. Working with the RDS database
5. Importing the data into the RDS database instance
6. Connecting the café application to the new database



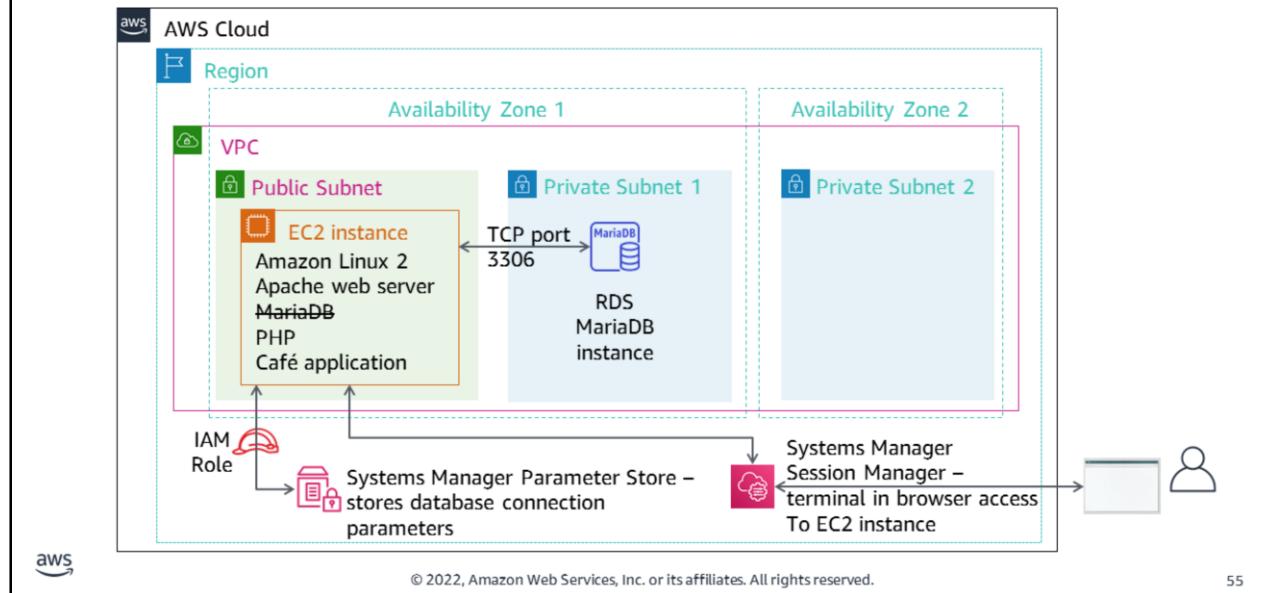
© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

54

In this challenge lab, you will complete the following tasks:

1. Creating an RDS instance
2. Analyzing the existing café application deployment
3. Working with the database on the EC2 instance
4. Working with the RDS database
5. Importing the data into the RDS database instance
6. Connecting the café application to the new database

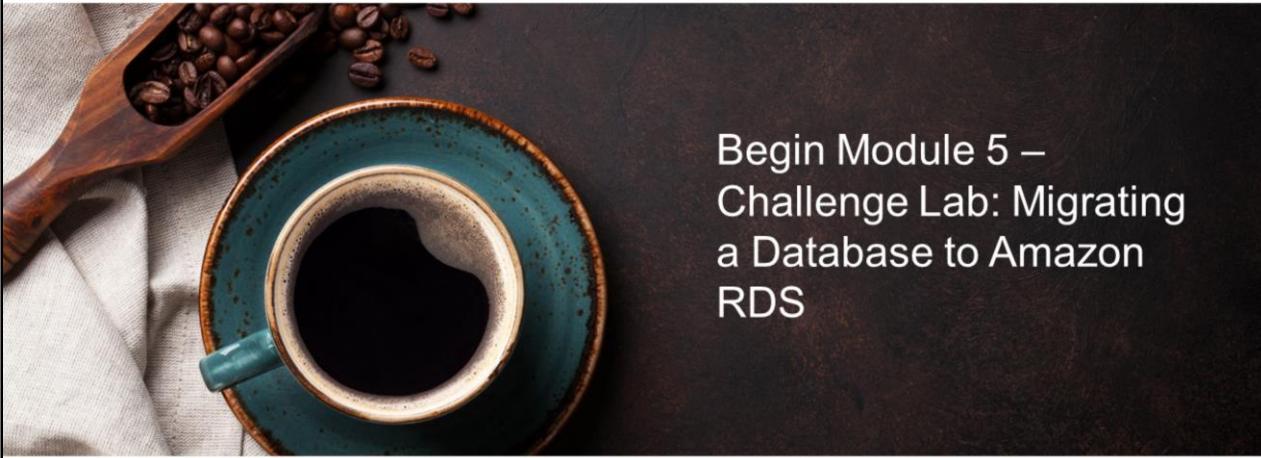
Challenge lab: Final product



The diagram summarizes what you will have accomplished after you complete the lab.

The café web application originally used the MariaDB that is installed on the EC2 instance. However, in the course of the lab, you create an RDS MariaDB instance, dump the data out of the database on the EC2 instance, and migrate the data into the RDS database.

To accomplish the migration, you connect to the EC2 instance by using the AWS Systems Manager Session Manager, which provides terminal access through a web browser. You use a MySQL client that is installed on the EC2 instance to connect to both databases, as needed, during the migration. Database credentials and database connection information are stored in the AWS Systems Manager Parameter Store. Thus, an IAM role is attached to the EC2 instance to allow the web application to read data out of the Parameter Store.



A timer icon with the text "~ 80 minutes" next to it.

Begin Module 5 –
Challenge Lab: Migrating
a Database to Amazon
RDS

aws

© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

56

It is now time to start the challenge lab.

Challenge lab debrief: Key takeaways



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

57

The educator might now choose to lead a conversation about the key takeaways from the challenge lab after you have completed it.

Module wrap-up

Module 5: Adding a Database Layer



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

It's now time to review the module and wrap up with a knowledge check and discussion of a practice certification exam question.

Module summary

In summary, in this module, you learned how to:

- Compare database types
- Differentiate between managed versus unmanaged services
- Explain when to use Amazon Relational Database Service (Amazon RDS)
- Explain when to use Amazon DynamoDB
- Describe available database security controls
- Describe how to migrate data into Amazon Web Services (AWS) databases
- Deploy a database server



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

59

In summary, in this module, you learned how to:

- Compare database types
- Differentiate between managed versus unmanaged services
- Explain when to use Amazon Relational Database Service (Amazon RDS)
- Explain when to use Amazon DynamoDB
- Describe available database security controls
- Describe how to migrate data into Amazon Web Services (AWS) databases
- Deploy a database server

Complete the knowledge check



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

60

It is now time to complete the knowledge check for this module.

Sample exam question



An application requires a highly available relational database with an initial storage capacity of 8 TB. The database will grow by 8 GB every day. To support expected traffic, at least eight read replicas will be required to handle database reads.

Which option will meet these requirements?

Choice	Response
A	DynamoDB
B	Amazon S3
C	Amazon Aurora
D	Amazon Redshift

© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

61

Look at the answer choices and rule them out based on the keywords that were previously highlighted.

Sample exam question answer



An application requires a highly available relational database with an initial storage capacity of 8 TB. The database will grow by 8 GB every day. To support expected traffic, at least eight read replicas will be required to handle database reads.

Which option will meet these requirements?

The correct answer is C.

The keywords in the question are highly available relational database, grow by 8 GB every day, and read replicas.

© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

62

The following are the keywords to recognize: **highly available relational database, grow by 8 GB every day, and read replicas.**

The correct answer is C. Amazon Aurora is a relational database that will automatically scale to accommodate data growth.

Incorrect answers:

- Answer A: DynamoDB is a NoSQL service, not a relational database.
- Answer B: Amazon S3 is object storage, not a relational database.
- Answer D: Amazon Redshift does not support read replicas and will not automatically scale.

Additional resources

- [AWS Databases – Resource page](#)
- [Amazon RDS Getting Started Guide](#)
- [Best Practices for Amazon RDS](#)
- [Amazon RDS FAQs](#)
- [Amazon DynamoDB Developer Guide](#)
- [Amazon DynamoDB FAQs](#)



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

63

If you want to learn more about the topics covered in this module, you might find the following additional resources helpful:

- [AWS Databases – Resource page](#)
- [Amazon RDS Getting Started Guide](#)
- [Best Practices for Amazon RDS](#)
- [Amazon RDS FAQs](#)
- [Amazon DynamoDB Developer Guide](#)
- [Amazon DynamoDB FAQs](#)



Thank you

Corrections, feedback, or other questions?
Contact us at <https://support.aws.amazon.com/#/contacts/aws-academy>.
All trademarks are the property of their owners.

© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

64

Thank you for completing this module.



**AWS Academy Cloud Architecting
Module 06 Student Guide
Version 2.0.8**

200-ACACAD-20-EN-SG

© 2022 Amazon Web Services, Inc. or its affiliates. All rights reserved.

This work may not be reproduced or redistributed, in whole or in part, without prior written permission from Amazon Web Services, Inc. Commercial copying, lending, or selling is prohibited.

All trademarks are the property of their owners.

Contents

[Module 6: Creating a Networking Environment](#)

4



Module 6: Creating a Networking Environment

AWS Academy Cloud Architecting

© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

Welcome to Module 6: Creating a Networking Environment.

Module overview

Sections

1. Architectural need
2. Creating an AWS networking environment
3. Connecting your AWS networking environment to the internet
4. Securing your AWS networking environment

Demonstration

- Creating a Virtual Private Cloud

Labs

- Guided Lab: Creating a Virtual Private Cloud
- Challenge Lab: Creating a VPC Networking Environment for the Café



Knowledge check



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

2

This module includes the following sections:

1. Architectural need
2. Creating an network environment
3. Connecting your network environment to the internet
4. Securing your network environment

The module also includes:

- A demonstration that will show you how manually create a virtual private cloud (VPC)
- A guided lab where you will create a VPC on your own
- A challenge lab where you will create a VPC, enable private resources to connect to the internet, and create a security layer to control traffic to and from private resources in your VPC.

Finally, you will be asked to complete a knowledge check that will test your understanding of key concepts covered in this module.

Module objectives

At the end of this module, you should be able to:

- Explain the foundational role of a virtual private cloud (VPC) in Amazon Web Services (AWS) Cloud networking
- Identify how to connect your AWS networking environment to the internet
- Describe how to isolate resources within your AWS networking environment
- Create a VPC with subnets, an internet gateway, route tables, and a security group



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

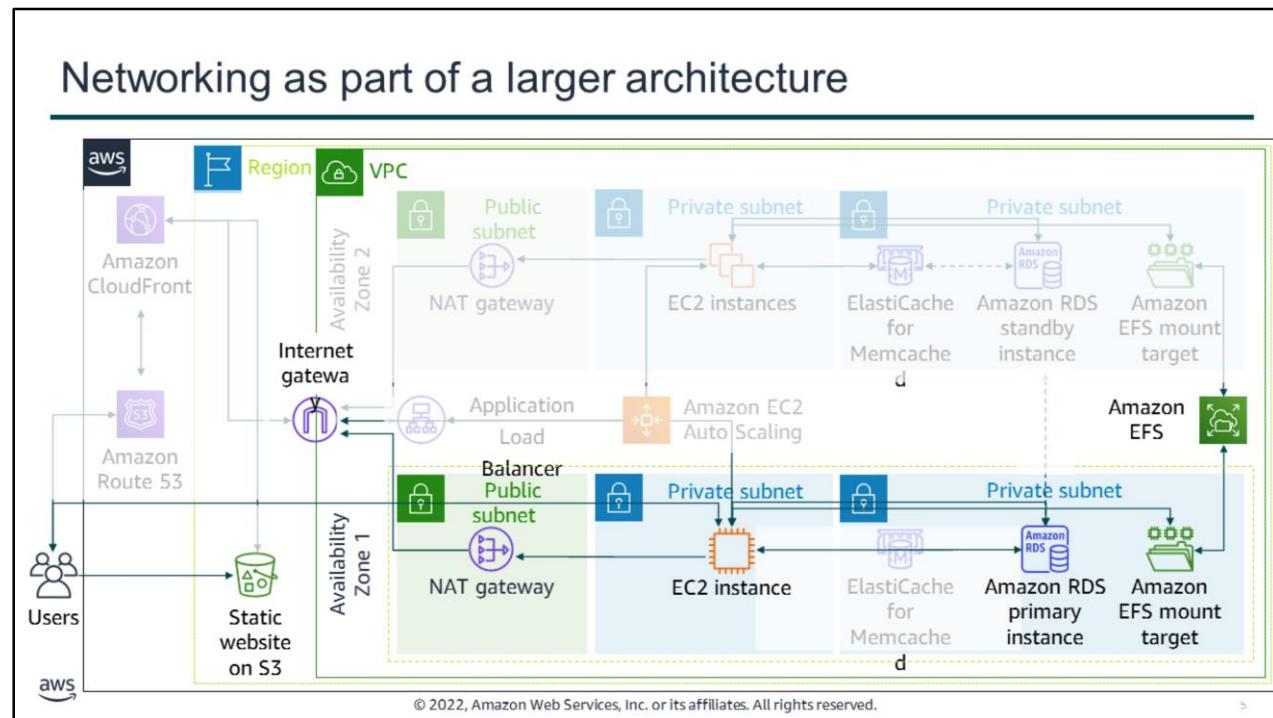
3

At the end of this module, you should be able to:

- Explain the foundational role of a virtual private cloud (VPC) in Amazon Web Services (AWS) Cloud networking
- Identify how to connect your AWS networking environment to the internet
- Describe how to isolate resources within your AWS networking environment
- Create a VPC with subnets, an internet gateway, route tables, and a security group



Introducing Section 1: Architectural need.



In this module, you will learn how to design a network on AWS and build a VPC with subnets. You will also learn how to connect instances in your public and private subnets to the internet.

Café business requirement

The café must deploy and manage AWS resources in a secure, isolated network environment.



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

6

The café's business has been steadily increasing. Sofía and Nikhil have become friends with a few of the café regulars, who are AWS consultants, and they have started to discuss the café's current architecture. Olivia, one of the regulars and an AWS Solutions Architect, identified a need for the café's online business to scale. Scaling will require additional servers to run the online ordering application, but the current subnet size is too small and can't support this growth. Therefore, they will need to rearchitect some aspects of the network that the application runs in.

On further review of the café's architecture, Olivia also noticed a vulnerability: the TCP port that's used to administer the application server is accessible to the internet. Sofía explained that she and Nikhil must be able to manage and maintain the server. Olivia advises them to set up a bastion host to reduce public access to the server, and to make it more secure.

Section 2: Creating an AWS networking environment

Module 6: Creating a Networking Environment



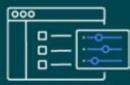
© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

Introducing Section 2: Creating an AWS networking environment.

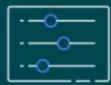
Amazon VPC

Provision a logically isolated section of the AWS Cloud where you can launch AWS resources in a virtual network that you define.

Bring your own network



IP Addresses



Subnets



Routing rules



Network configuration



Security rules



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

8

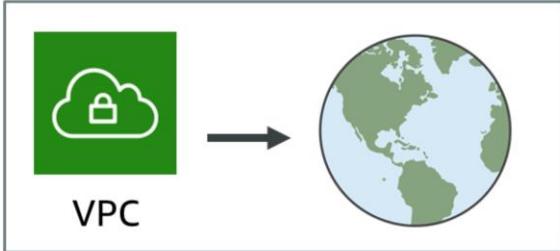
Amazon Virtual Private Cloud (Amazon VPC) is a service that enables you to provision a logically isolated section of the AWS Cloud (called a virtual private cloud, or VPC) where you can launch your AWS resources.

Amazon VPC gives you control over your virtual networking resources. For example, you can select your own IP address range, create subnets, and configure route tables and network gateways. You can use both IPv4 and IPv6 in your VPC for secure access to resources and applications.

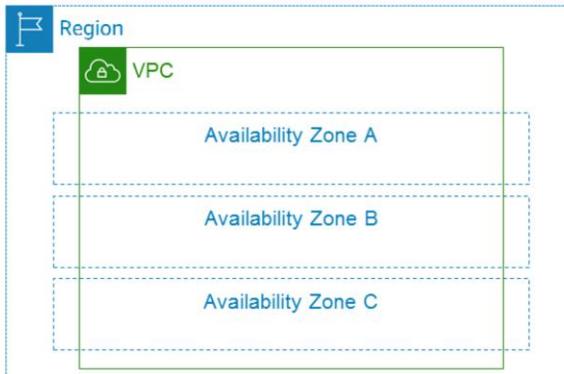
You can also customize the network configuration for your VPC. For example, you can create a public subnet for your web servers that can access the public internet. You can place your backend systems (such as databases or application servers) in a private subnet with no public internet access.

Finally, you can use multiple layers of security to help control access to Amazon Elastic Compute Cloud (Amazon EC2) instances in each subnet. These security layers include security groups and network access control lists (network ACLs).

VPC deployment



You can deploy a VPC in any AWS Region.



A VPC can host supported resources from any Availability Zone within its Region.



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

9

A VPC belongs to a single AWS Region. A VPC spans all the Availability Zones in a Region, so it can host supported resources from any Availability Zone within its Region.

Classless Inter-Domain Routing (CIDR)

0.0.0.0/0	= All IP addresses
10.22.33.44/32	= 10.22.33. 44
10.22.33.0/24	= 10.22.33.*
10.22.0.0/16	= 10.22.*.*

CIDR	Total IP addresses
/28	16
...	...
/20	4,096
/19	8,192
/18	16,384
/17	32,768
/16	65,536



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

10

When you create a VPC, you provide the set of private IP addresses that you want instances in your VPC to use. You specify this set of addresses as a Classless Inter-Domain Routing (CIDR) block—for example, 10.0.0.0/16. This is the primary CIDR block for your VPC. You can assign block sizes of between /28 (16 IP addresses) and /16 (65,536 IP addresses).

Amazon VPC supports IPv4 and IPv6 addressing and has different CIDR block size limits for each. By default, all VPCs and subnets must have IPv4 CIDR blocks—you can't change this behavior. You can optionally associate an IPv6 CIDR block with your VPC.

Your VPC can operate in dual-stack mode: your resources can communicate over IPv4, or IPv6, or both. IPv4 and IPv6 addresses are independent of each other, so you must configure routing and security in your VPC separately for IPv4 and IPv6.

Subnets: Dividing your VPC

- A **subnet** is a segment or partition of a VPC's IP address range where you can allocate a group of resources
- Subnets are **not isolation boundaries**
- Subnets are a **subset** of the VPC CIDR block
- Subnet CIDR blocks **cannot overlap**
- Each subnet resides entirely within one Availability Zone
- You can add one or more subnets in each Availability Zone or in a Local Zone
- AWS **reserves five IP addresses** in each subnet



Example: A VPC with **CIDR /22** includes 1,024 total IP addresses.



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

11

You can divide a VPC into one or more subnets. A subnet is a segment or partition of a VPC's IP address range where you can allocate a group of resources. It is important to remember that subnets are not isolation boundaries around your application. Instead, they are containers for routing policies, which you will learn about in the next section of this module.

When you create a subnet, you specify the CIDR block for the subnet, which is a subset of the VPC CIDR block. Subnet CIDR blocks cannot overlap.

Though each subnet must reside entirely within one Availability Zone and cannot span zones, each Availability Zone can have one or more subnets. You can optionally add subnets in a Local Zone. When you create a subnet in a Local Zone, the VPC is also extended to that Local Zone. For more information about how to extend your VPC resources to a Local Zone, see [Extending Your VPC Resources to AWS Local Zones](#) in the AWS Documentation.

Because VPC subnets are mapped to specific Availability Zones, subnet placement is one way to ensure that Amazon EC2 instances are properly distributed across multiple locations.

AWS reserves the first four IP addresses and the last IP address in each subnet CIDR block. For example, in a subnet with CIDR block 10.0.0.0/24, AWS reserves the following five IP addresses for:

- 10.0.0.0: Network address
- 10.0.0.1: VPC local router
- 10.0.0.2: Domain Name System (DNS) resolution
- 10.0.0.3: Future use
- 10.0.0.255: Network broadcast address

For more information about VPCs and subnets, see [VPCs and Subnets](#) in the AWS Documentation.

VPC design best practices

- Create one subnet per available Availability Zone for each group of hosts that have unique routing requirements.
- Divide your VPC network range evenly across all available Availability Zones in a Region.
- Do not allocate all network addresses at once. Instead, ensure that you reserve some address space for future use.
- Size your VPC CIDR and subnets to support significant growth for the expected workloads.
- Ensure that your VPC network range (CIDR block) does not overlap with your organization's other private network ranges.



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

12

When you configure any computer network, consider the following universal network design principles:

- Create one subnet per available Availability Zone for each group of hosts that have unique routing requirements.
- Divide your VPC network range evenly across all available Availability Zones in a Region.
- Do not allocate all network addresses at once. Instead, ensure that you reserve some address space for future use.
- Size your VPC CIDR and subnets to support significant growth for the expected workloads.
- Ensure that your VPC network range (CIDR block) does not overlap with your organization's other private network ranges.

For more information about designing and sizing individual VPCs, see [AWS Single VPC Design](#).

Single VPC deployment

There are limited use cases where deploying **one VPC** might be appropriate:

- Small, single applications managed by a small team
- High performance computing (HPC)
- Identity management

For **most** use cases, there are two primary patterns for organizing your infrastructure: multi-VPC and multi-account.



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

13

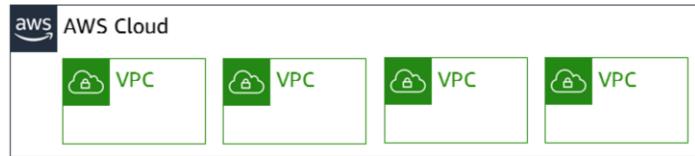
When you design and create your network environment, there are a limited number of use cases where a single VPC environment might be appropriate:

- Small, single applications managed by a small team
- High performance computing (HPC) environments (such as physics simulations)—a single VPC environment has lower latency than one spread across multiple VPCs
- Identity management environments—a single VPC might provide best security.

For most use cases, however, a multi-VPC environment is required. You can create multiple VPCs within the same Region or in different Regions. You can also create multiple VPCs in the same AWS account or in different AWS accounts.

Multiple VPCs

- Best suited for –
 - Single team or single organizations, such as managed service providers
 - Limited teams, which makes it easier to maintain standards and manage access
- Exception –
 - Governance and compliance standards might require greater workload isolation regardless of organizational complexity



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

14

Multiple VPCs are best suited for a single team or organization that maintains full control over the provisioning and management of all resources in each application environment. For example, consider a single team that develops a large ecommerce application. They might use this pattern when the developers have full access to the Development and Production environments. This pattern is also common with managed service providers (MSPs) that manage all resources in Test and Production environments.

To learn more about services and best practices for multiple-VPC deployments, see:

- [Single Region Multi-VPC Connectivity](#)
- [Multiple Region Multi-VPC Connectivity](#)

Multiple accounts

- Best suited for –
 - Large organizations and organizations with multiple IT teams
 - Medium-sized organizations that anticipate rapid growth
- Why?
 - It can be more challenging to manage access and standards in more complex organizations



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

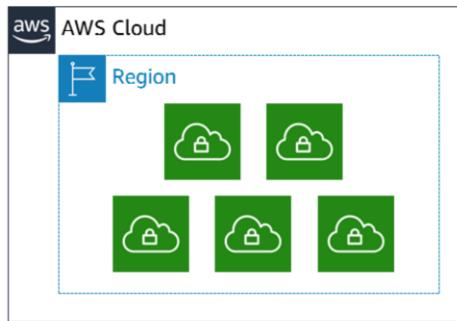
15

As mentioned, you can create multiple VPCs in the same AWS account or in different accounts.

Multiple-account patterns are best suited for enterprise customers or organizations that deploy applications managed across multiple teams. For example, consider an organization that supports two or more teams. They might use this pattern to support developers who have full access to Development environment resources, but limited or no access to the Production environment.

Amazon VPC quotas

Default quota: 5 VPCs per Region per account *



* The default quota is 5 VPCs per Region, but you can request a quota increase.



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

16

Be aware of Amazon VPC quotas. The default quota is 5 VPCs per Region. However, you can request an increase for this quota.

For more information about Amazon VPC service limits, see [Amazon VPC quotas](#) in the AWS Documentation.

Section 2 key takeaways



- Amazon VPC enables you to provision VPCs, which are logically isolated sections of the AWS Cloud where you can launch your AWS resources.
- A VPC belongs to only one Region and is divided into subnets.
- A subnet belongs to one Availability Zone or Local Zone. It is a subset of the VPC CIDR block.
- You can create multiple VPCs in the same Region or in different Regions, and in the same account or different accounts.
- Follow best practices when you design your VPC.

© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

17

Some key takeaways from this section of the module include:

- Amazon VPC enables you to provision VPCs, which are logically isolated sections of the AWS Cloud where you can launch your AWS resources.
- A VPC belongs to only one Region and is divided into subnets.
- A subnet belongs to one Availability Zone or Local Zone. It is a subset of the VPC CIDR block.
- You can create multiple VPCs within the same Region or in different Regions, and in the same account or different accounts.
- Follow these best practices when designing your VPC –
 - Create one subnet per Availability Zone for each group of hosts that have unique routing requirements.
 - Divide your VPC network range evenly across all available Availability Zones in a Region.
 - Do not allocate all network addresses at once. Instead, ensure that you reserve some address space for future use.
 - Size your VPC CIDR and subnets to support significant growth for the expected workloads.
 - Ensure that your VPC network range does not overlap with your organization's other private network ranges.

Section 3: Connecting your AWS networking environment to the internet

Module 6: Creating a Networking Environment



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

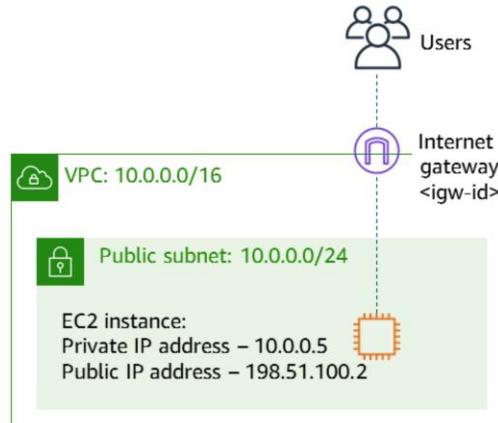
Introducing Section 3: Connecting your AWS networking environment to the internet.

Creating a public subnet



Internet gateways

- Allow communication between resources in your VPC and the internet
- Are horizontally scaled, redundant, and highly available by default
- Provide a target in your subnet route tables for internet-routable traffic



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

19

Now that you know how to design and create an isolated network environment for your workloads, you will want to connect it to the internet.

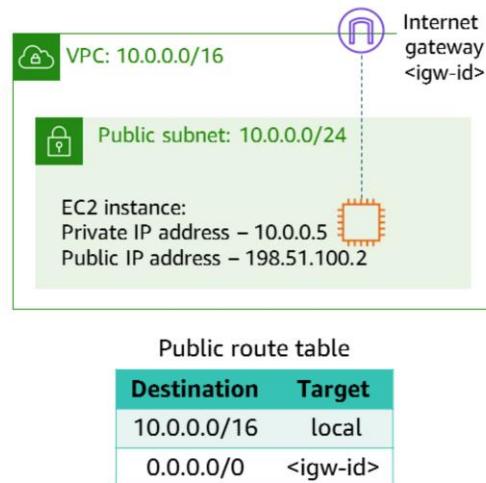
An *internet gateway* is a VPC component that allows communication between resources in your VPC and the internet. It is horizontally scaled, redundant, and highly available. An internet gateway supports IPv4 and IPv6 traffic. An internet gateway serves two purposes. First, it provides a target in your VPC route tables for internet-routable traffic. Second, the internet gateway performs network address translation (NAT) for instances that were assigned public IPv4 addresses.

To make a subnet *public*, you must first create an internet gateway and attach it to your VPC.

Directing traffic between VPC resources

- **Route tables** are required to direct traffic between VPC resources
- Each VPC has a **main (default)** route table
- All subnets **must** be associated with a route table
- You can create **custom** route tables

Best practice: Use custom route tables for each subnet.



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

20

Next, you must update the route table associated with the subnet you want to connect to the internet. A *route table* contains a set of rules, called *routes*. Routes are used to determine where network traffic is directed.

When you create a VPC, it automatically has a main route table. Initially, the main route table (and every route table in a VPC) contains only a single local route that enables communication for all the resources in the VPC. You can't modify the local route in a route table. When you launch an instance in the VPC, the local route automatically covers that instance. You don't need to add the new instance to a route table. You can create additional custom route tables for your VPC.

Each subnet in your VPC must be associated with a route table, which controls the routing for the subnet. If you don't explicitly associate a subnet with a particular route table, the subnet is implicitly associated with and uses the main route table. A subnet can be associated with only one route table at a time, but you can associate multiple subnets with the same route table.

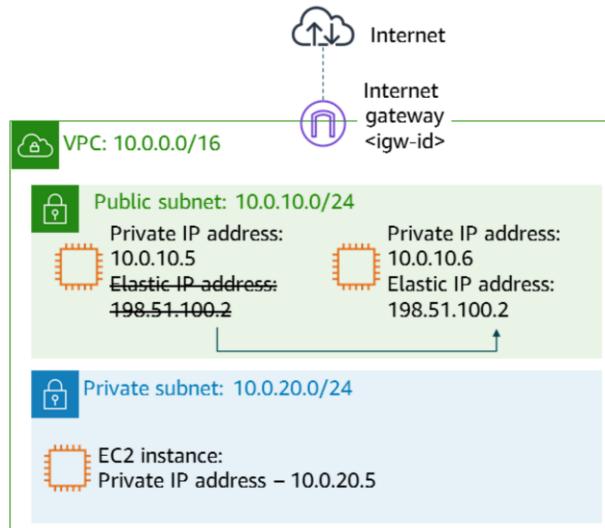
You can create custom route tables for each subnet to enable granular routing for destinations.

To send non-local traffic through the internet gateway to the internet, create a route with destination **0.0.0.0/0** and target **<igw-id>** in the route table associated with the subnet.

Remapping an IP address from one instance to another

➡ Elastic IP addresses

- Are static, public IPv4 addresses associated with your AWS account
- Can be associated with an instance or elastic network interface
- Can be remapped to another instance in your account
- Are useful for redundancy when load balancers are not an option



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

21

Next, you must make sure that your instances have either public IP or Elastic IP addresses.

An *Elastic IP address* is a static and public IPv4 address that is designed for dynamic cloud computing. You can associate an Elastic IP address with any instance or elastic network interface for any VPC in your account. With an Elastic IP address, you can mask the failure of an instance by rapidly remapping the address to another instance in your VPC. Associating the Elastic IP address with the network interface has an advantage over associating it directly with the instance. You can move all of the attributes of the network interface from one instance to another in a single step.

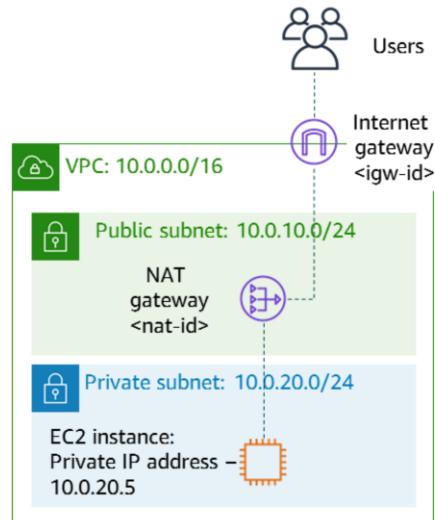
Connecting private subnets to the internet



NAT gateways

- Enable instances in a private subnet to initiate outbound traffic to the internet or other AWS services
- Prevent private instances from receiving inbound connection requests from the internet

Public route table	
Destination	Target
10.0.0.0/16	local
0.0.0.0/0	<igw-id>
Private route table	
Destination	Target
10.0.0.0/16	local
0.0.0.0/0	<nat-id>



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

22

To connect instances in your private subnet to the internet or other AWS services, you need a *network address translation (NAT) gateway*. A NAT gateway enables instances in a private subnet to connect to the internet or other AWS services, but prevents the internet from initiating a connection with those instances.

To create a NAT gateway, you must specify the public subnet where the NAT gateway should reside. You must also specify an Elastic IP address to associate with the NAT gateway. After you create a NAT gateway, you must update the route table that is associated with one or more of your private subnets to point internet-bound traffic to the NAT gateway. Thus, instances in your private subnets can communicate with the internet.

Subnet use case examples (1 of 2)



Data store instances



Batch-processing instances



Backend instances



Web application instances

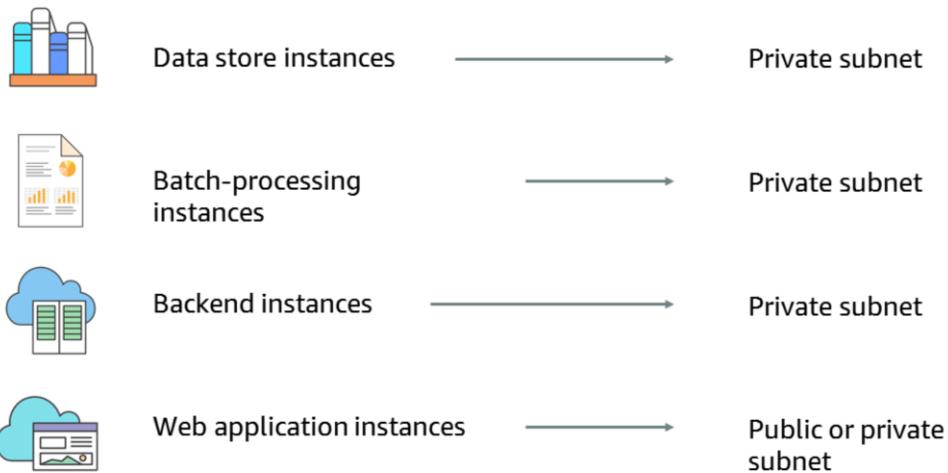


© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

25

Take a moment to think about whether the instances in these examples should be put into a public or private subnet.

Subnet use case examples (2 of 2)



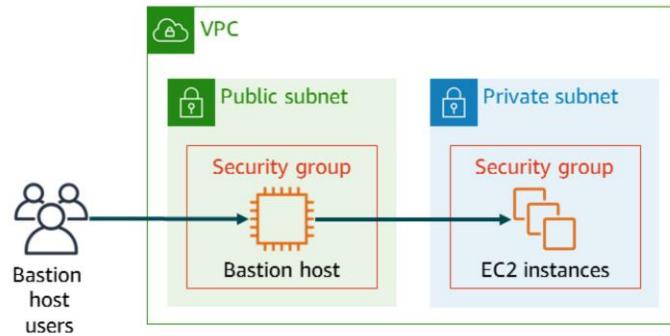
© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

24

Data store instances, batch-processing instances, and backend instances should be put into private subnets. You can put web-tier instances into a public subnet. However, AWS recommends that you put web-tier instances inside *private* subnets behind a load balancer that is placed in a public subnet. In some environments, you must attach web application instances to Elastic IP addresses directly (though you can also attach an Elastic IP address to a load balancer). In those cases, web application instances must be in a public subnet.

Bastion hosts

- A server whose purpose is to provide access to a private network from an external network
- Must minimize the chances of penetration



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

25

A *bastion host* is a server that provides access to a private network from an external network, such as the internet. You can use a bastion host to minimize the chances of penetration and potential attack on resources in your private network.

For example, suppose you want to allow connections from an external network to Linux instances in a private subnet of your VPC via Secure Shell, or SSH.

You can use a bastion host to mitigate the risk of allowing these external SSH connections to the instances in the private subnet. A bastion host typically runs on an EC2 instance in a public subnet of your VPC, as shown in this example. The Linux instances in the private subnet are in a security group that allows SSH access from the security group attached to the bastion host. Bastion host users connect to the bastion host so they can connect to the Linux instances.

Though you can adapt this architecture to meet your own requirements, the bastion host should be the only source of SSH traffic to your Linux instances.

For more information about this architecture, see the blog post [How to Record SSH Sessions Established Through a Bastion Host](#). To learn how to deploy a Linux bastion host in a VPC environment on AWS, complete this [Linux Bastion Hosts on AWS Quick Start](#).

Demonstration: Creating a Virtual Private Cloud



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

26

Now, your educator might choose to demonstrate how to use Amazon VPC to manually create a VPC with subnets, an internet gateway, and a route table.

Section 3 key takeaways



- An [internet gateway](#) allows communication between instances in your VPC and the internet.
- [Route tables](#) control traffic from your subnet or gateway.
- [Elastic IP addresses](#) are static, public IPv4 addresses that can be associated with an instance or elastic network interface. They can be remapped to another instance in your account.
- [NAT gateways](#) enable instances in the private subnet to initiate outbound traffic to the internet or other AWS services.
- A [bastion host](#) is a server whose purpose is to provide access to a private network from an external network, such as the internet.

© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

27

Some key takeaways from this section of the module include:

- An internet gateway allows communication between instances in your VPC and the internet.
- Route tables control traffic from your subnet or gateway.
- Elastic IP addresses are static, public IPv4 addresses that can be associated with an instance or elastic network interface. They can be remapped to another instance in your account.
- NAT gateways enable instances in the private subnet to initiate outbound traffic to the internet or other AWS services.
- A bastion host is a server whose purpose is to provide access to a private network from an external network, such as the internet.

Section 4: Securing your AWS networking environment

Module 6: Creating a Networking Environment

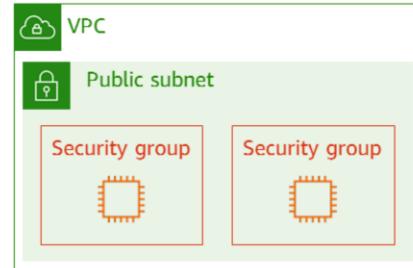


© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

Introducing Section 4: Securing your AWS networking environment.

Security groups

- Are **stateful firewalls** that control inbound and outbound traffic to AWS resources
- Act at the **level of the instance or network interface**



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

29

Now that you know how to design and deploy a network environment, and connect it to the internet, you must isolate your applications and workloads.

You can achieve isolation by deploying the EC2 instances that host your application or workload into a security group that is attached to your VPC.

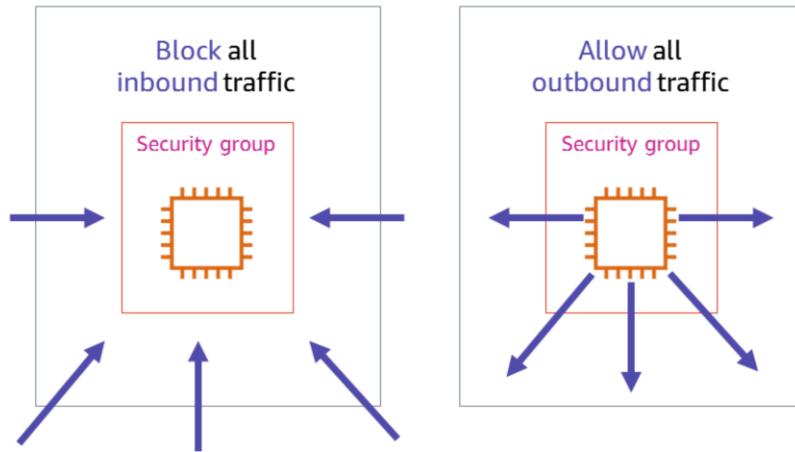
Security groups are stateful firewalls that act at the level of instance or network interface.

Stateful means that return traffic is allowed is automatically allowed, regardless of any rules. For example, say that you initiate an Internet Control Message Protocol (ICMP) *ping* command to your instance from your home computer. If your inbound security group rules allow ICMP traffic, information about the connection (including the port information) is tracked. Response traffic from the instance for the ping command is not tracked as a new request. Instead, it is tracked as an established connection. It is allowed to flow out of the instance, even if your outbound security group rules restrict outbound ICMP traffic.

Security group rules control inbound and outbound traffic to your AWS resources. You should tightly configure these rules to restrict traffic and allow access only as needed. Traffic can be restricted by any internet protocol, service port, and source or destination IP address (individual IP address or CIDR block).

Not all traffic flows are tracked. Consider a security group rule that permits transmission control protocol (TCP) or user datagram protocol (UDP) flows for all traffic (that is, 0.0.0.0/0). There is also a corresponding rule in the other direction that permits the response traffic. In this case, that flow of traffic is not tracked. The response traffic is therefore allowed to flow based on the inbound or outbound rule that permits the response traffic, and not on tracking information.

Default security groups

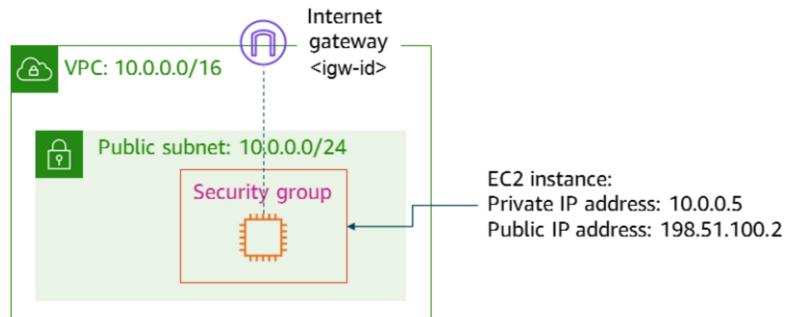


© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

30

When you create a security group, it has no inbound rules. This means that you must add inbound rules to the security group to allow inbound traffic that originates from another host to your instance. By default, a security group includes an outbound rule that *allows all outbound traffic*. You can remove the rule and add outbound rules that allow specific outbound traffic only. If your security group has no outbound rules, no outbound traffic that originates from your instance is allowed.

Custom security groups



Inbound				
Type	Protocol	Port Range	Source	Destination
HTTP	TCP	80	Anywhere	Allow web access

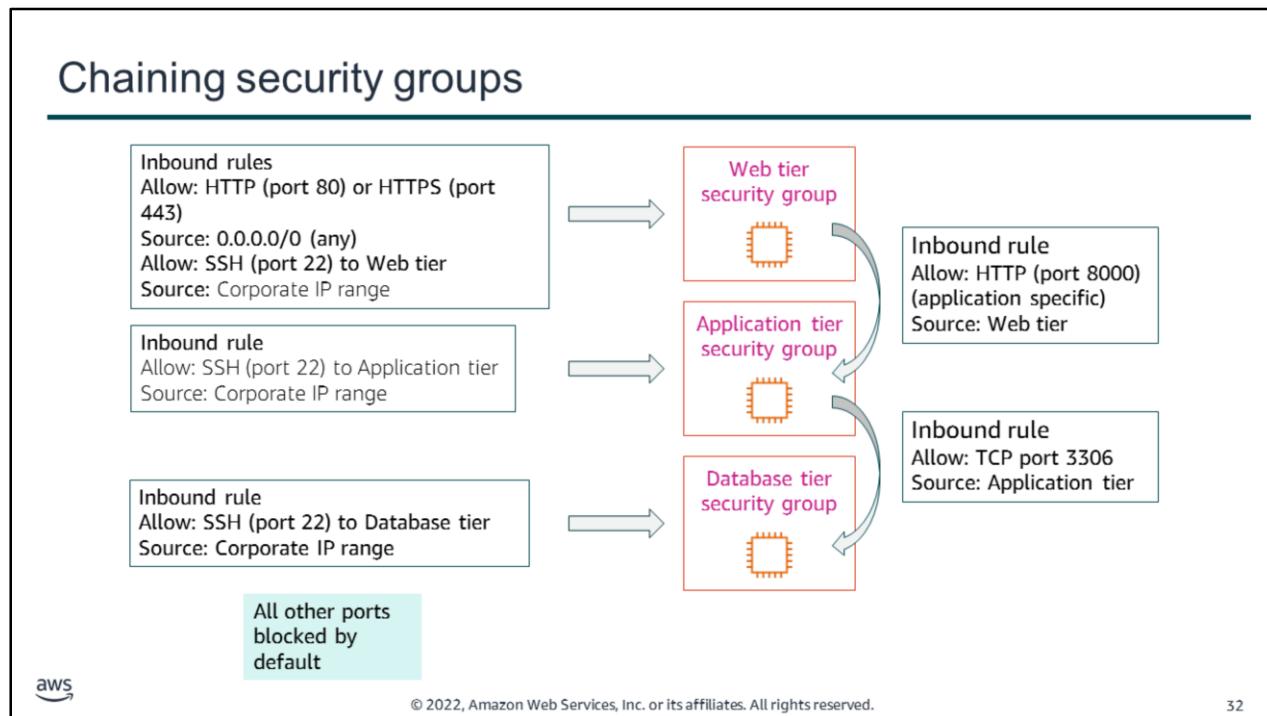


© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

31

When you create a custom security group, you can specify *allow* rules, but not *deny* rules. For example, when you create a public subnet for the instances that host your web application, the last step is to create a security group that allows HTTP traffic to those instances.

All rules are evaluated before the decision to allow traffic is made.

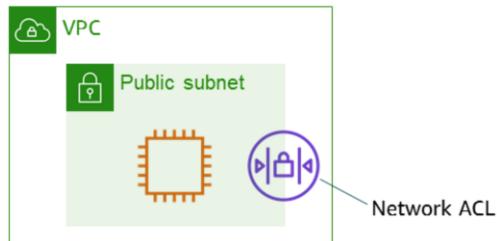


Most cloud organizations create security groups with inbound rules for each functional tier. This example shows a chain of security groups in a typical three-tier application. The inbound and outbound rules are set up so that traffic can only flow from the top tier to the bottom tier, and back up again. The security groups act as firewalls to prevent a security breach in one tier from automatically providing subnet-wide access of all resources to the compromised client.

Security groups can be configured to set different rules for different classes of instances. Consider this example of a traditional three-tier architecture for a web application. The group for the web servers would have port 80 (HTTP) 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. This mechanism enables the deployment of highly secure applications.

Network access control lists (network ACLs)

- Act at the **subnet level**
- Allow all inbound and outbound traffic by default
- Are **stateless firewalls** that require explicit rules for both inbound and outbound traffic



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

33

A *network access control list (network ACL)* is an optional layer of security for your VPC. It acts as a firewall for controlling traffic in and out of one or more subnets. To add another layer of security to your VPC, you can set up network ACLs with rules that are similar to your security groups.

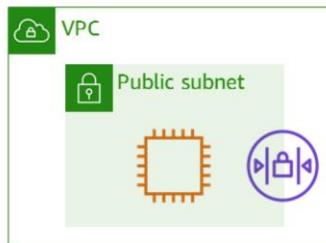
Each subnet in your VPC must be associated with a network ACL. If you don't explicitly associate a subnet with a network ACL, the subnet is automatically associated with the default network ACL. You can associate a network ACL with multiple subnets. However, a subnet can be associated with only one network ACL at a time. When you associate a network ACL with a subnet, the previous association is removed.

A network ACL has separate inbound and outbound rules, and each rule can either allow or deny traffic. Your VPC automatically comes with a modifiable default network ACL. By default, it allows all inbound and outbound IPv4 traffic and, if applicable, IPv6 traffic.

Network ACLs are *stateless*, which means that no information about a request is maintained after a request is processed. Return traffic must be explicitly allowed by rules.

Custom network ACLs

Recommended for
specific network security requirements only



Nacl-11223344

Inbound:
Rules # 100: SSH 172.31.1.2/32 ALLOW
Rules # *: ALL traffic 0.0.0.0/0 DENY

Outbound:
Rules # 100: Custom TCP 172.31.1.2/31
ALLOW
Rules # *: All traffic 0.0.0.0/0 DENY

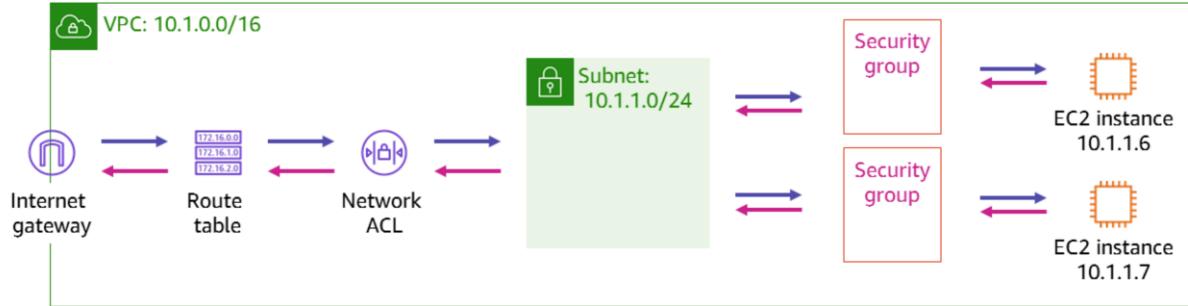


© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

54

You can create a *custom* network ACL and associate it with a subnet. By default, each custom network ACL denies all inbound and outbound traffic until you add rules.

Structure your infrastructure with multiple layers of defense



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

55

As a best practice, you should secure your infrastructure with multiple layers of defense. By running your infrastructure in a VPC, you can control which instances are exposed to the internet. You can define both security groups and network ACLs to further protect your infrastructure at the infrastructure and subnet levels, respectively. Additionally, you should secure your instances with a firewall at the operating system level, and follow other security best practices.

When you implement both network ACLs and security groups as a defense-in-depth way of controlling traffic, a mistake in the configuration of one of these controls will not expose the host to unwanted traffic.

Review: How to create a public subnet

To create a [public subnet](#) to allow communication between instances in your VPC and the internet, you must:



Attach an [internet gateway](#) to your VPC.

Destination	Target
10.0.0.0/16	local
0.0.0.0/0	<igw-id>

Point your instance subnet's [route table](#) to the internet gateway.



Make sure that your instances have [public IP](#) or [Elastic IP](#) addresses.



Security group

Make sure that your [security groups](#) and [network ACLs](#) allow relevant traffic to flow.



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

36

To review, to create a public subnet to allow communication between instances in your VPC and the internet, you must:

- Attach an internet gateway to your VPC
- Add a route to your subnet's route table that directs internet-bound traffic to the internet gateway
- Make sure that your instances have public IP or Elastic IP addresses
- Make sure that your security groups and network ACLs allow relevant traffic to flow

Section 4 key takeaways



- Security groups are **stateful** firewalls that act at the **instance level**
- Network ACLs are **stateless** firewalls that act at the **subnet level**
- When you set inbound and outbound rules to allow traffic to flow from the top tier to the bottom tier of your architecture, you can **chain security groups together** to isolate a security breach
- You should structure your infrastructure with **multiple layers of defense**

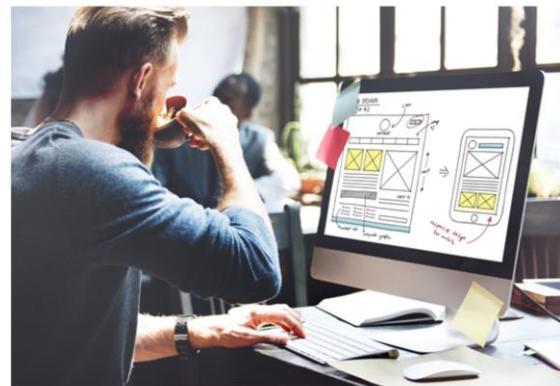
© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

37

Some key takeaways from this section of the module include:

- Security groups are stateful firewalls that act at the instance level
- Network ACLs are stateless firewalls that act at the subgroup level
- When you set inbound and outbound rules to allow traffic to flow from the top tier to the bottom tier of your architecture, you can chain security groups together to isolate a security breach
- You should structure your infrastructure with multiple layers of defense

Module 6 - Guided Lab: Creating a Virtual Private Cloud



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

38

You will now complete Module 6 – Guided Lab: Creating a Virtual Private Cloud.

Guided lab: Tasks

Use Amazon VPC to manually create a VPC with:

- Public and private subnets
- An internet gateway
- A route table with a route to direct internet-bound traffic to the internet gateway
- A security group for EC2 instances in the public subnet
- An application server to test the VPC



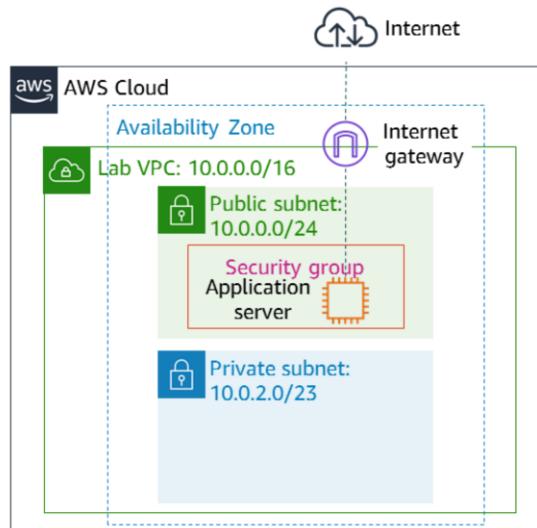
© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

39

In this lab, you will use Amazon VPC to manually create a VPC with the following components:

- Public and private subnets
- An internet gateway
- A route table with a route to direct internet-bound traffic to the internet gateway
- A security group for EC2 instances in the public subnet
- An application server to test the VPC

Guided lab: Final product



aws

© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

40

The diagram summarizes what you will have built after you complete the lab.



A timer icon with the text "~ 30 minutes" indicating the duration of the lab.

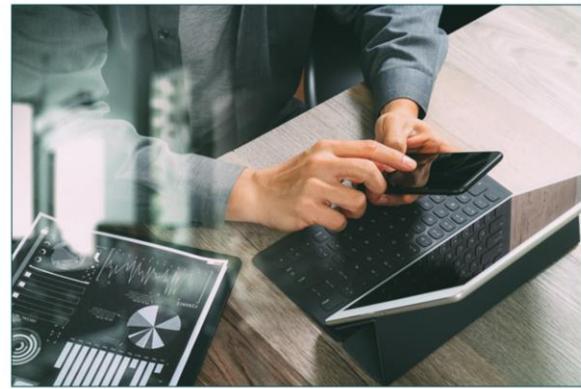
Begin Module 6 – Guided Lab: Creating a Virtual Private Cloud

aws

© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

It is now time to start the guided lab.

Guided lab debrief: Key takeaways



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

42

Your educator might choose to lead a conversation about the key takeaways from this guided lab after you have completed it.

Module 6 – Challenge Lab:

Creating a VPC Networking Environment for the Café



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

43

You will now complete Module 6 – Challenge Lab: Creating a VPC Networking Environment for the Café.

The business need: A secure networking environment

Sofía and Nikhil separated the café's database layer from the web application layer. They also moved the database resources from a public subnet to a private subnet.



Mateo advises them to enhance security by running the café's application server in a private subnet that is separate from the database instance.



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

44

Sofía and Nikhil have successfully created a two-tier architecture, in which they separated the café's database layer from the web application layer. They also moved their database resources from a public subnet to a private subnet.

Mateo advises them to enhance the security of their VPC by running the café's application server in a private subnet that is separate from the database instance.

Challenge lab: Tasks

1. Creating a public subnet
2. Creating a bastion host
3. Allocating an Elastic IP address for the bastion host
4. Testing the connection to the bastion host
5. Creating a private subnet
6. Creating a NAT gateway
7. Creating an EC2 instance in a private subnet
8. Configuring your SSH client for SSH passthrough
9. Testing the SSH connection from the bastion host
10. Creating a network ACL
11. Testing your custom network ACL



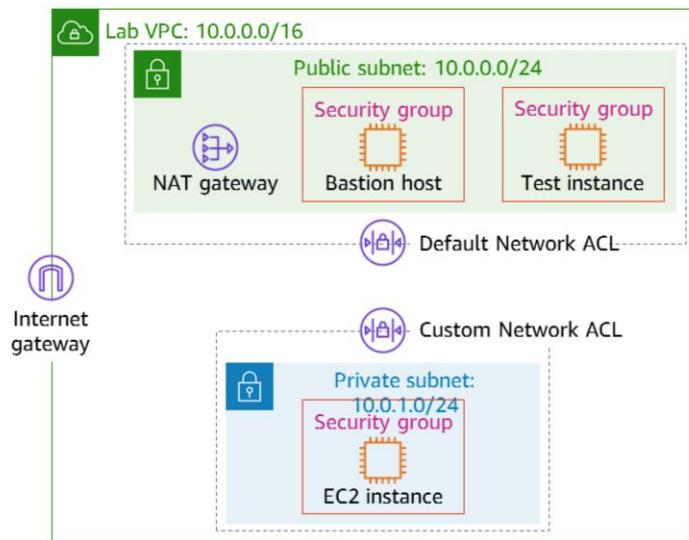
© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

45

In this challenge lab, you will complete the following tasks:

1. Creating a public subnet
2. Creating a bastion host
3. Allocating an Elastic IP address for the bastion host
4. Testing the connection to the bastion host
5. Creating a private subnet
6. Creating a NAT gateway
7. Creating an EC2 instance in a private subnet
8. Configuring your SSH client for SSH passthrough
9. Testing the SSH connection from the bastion host
10. Creating a network ACL
11. Testing your custom network ACL

Challenge lab: Final product



aws

© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

46

The diagram summarizes what you will have built after you complete the lab.



A timer icon with the text "~ 90 minutes" next to it.

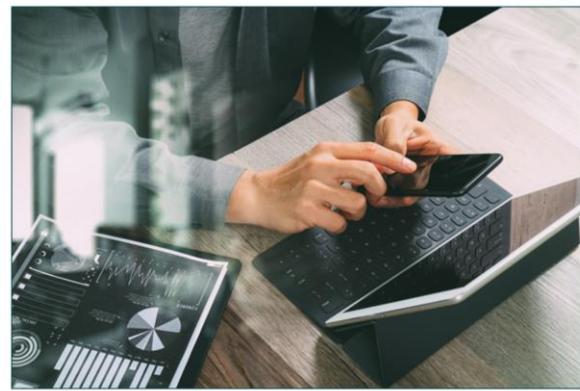
Begin Module 6 – Challenge Lab: Creating a VPC Networking Environment for the Café

aws

© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

It is now time to start the challenge lab.

Challenge lab debrief: Key takeaways



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

48

Your educator might choose to lead a conversation about the key takeaways from this challenge lab after you have completed it.

Module wrap-up

Module 6: Creating a Networking Environment



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

It's now time to review the module and wrap up with a knowledge check and discussion of a practice certification exam question.

Module summary

In summary, in this module, you learned how to:

- Explain the foundational role of VPC in AWS Cloud networking
- Identify how to connect your AWS networking environment to the internet
- Describe how to isolate resources within your AWS networking environment
- Create a VPC with subnets, an internet gateway, route tables, and a security group



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

50

In summary, in this module, you learned how to:

- Explain the foundational role of a VPC in AWS Cloud networking
- Identify how to connect your AWS networking environment to the internet
- Describe how to isolate resources within your AWS networking environment
- Create a VPC with subnets, an internet gateway, route tables, and a security group

Complete the knowledge check



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

51

It is now time to complete the knowledge check for this module.

Sample exam question



You have an application running on multiple Amazon Elastic Compute Cloud (Amazon EC2) instances in a single Availability Zone. The application calls a third-party application programming interface (API) via the internet.

How can you provide the third-party API with a single IP address to add to an access safelist?

Choice	Response
A	Assign an Elastic IP address to the instances.
B	Assign a public IP address to the instances.
C	Put the instances behind a NAT gateway.
D	Put the instances behind a Network Load Balancer.

© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

52

Look at the answer choices and rule them out based on the keywords.

Sample exam question answer



You have an application running on multiple Amazon Elastic Compute Cloud (Amazon EC2) instances in a single Availability Zone. The application calls a third-party application programming interface (API) via the internet.

How can you provide the third-party API with a single IP address to add to an access safelist?

The correct answer C.

The keywords in the question are “single IP address”.

© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

53

The following are the keywords to recognize: **“single IP address”**.

The correct answer is C: “Put the instances behind a NAT gateway.” Choices A and B can be eliminated because the application will end up having multiple public IP addresses, whereas a single IP address for safelisting is required. Choice D can also be eliminated because a Network Load Balancer will also have more than one IP address. Choice C is correct because a NAT gateway is accessible via a single IP address.

Additional resources

- [VPCs and Subnets](#)
- [One to Many: Evolving VPC Design](#)
- [AWS Single VPC Design](#)
- [AWS re:Invent 2018: Your Virtual Data Center: VPC Fundamentals and Connectivity Options](#)
- [AWS Networking Fundamentals](#)

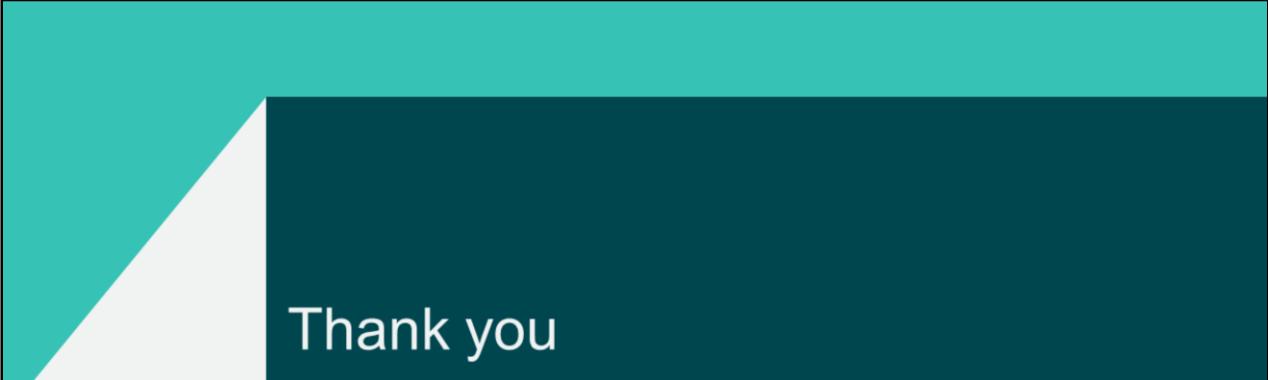


© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

54

If you want to learn more about the topics covered in this module, you might find the following additional resources helpful:

- [VPCs and Subnets](#)
- [One to Many: Evolving VPC Design](#)
- [AWS Single VPC Design](#)
- [AWS re:Invent 2018: Your Virtual Data Center: VPC Fundamentals and Connectivity Options](#)
- [AWS Networking Fundamentals](#)



Thank you

Corrections, feedback, or other questions?
Contact us at <https://support.aws.amazon.com/#/contacts/aws-academy>.
All trademarks are the property of their owners.



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

55

Thank you for completing this module.



**AWS Academy Cloud Architecting
Module 07 Student Guide
Version 2.0.8**

200-ACACAD-20-EN-SG

© 2022 Amazon Web Services, Inc. or its affiliates. All rights reserved.

This work may not be reproduced or redistributed, in whole or in part, without prior written permission from Amazon Web Services, Inc. Commercial copying, lending, or selling is prohibited.

All trademarks are the property of their owners.

Contents

Module 7: Connecting Networks	4
-------------------------------	---



Module 7: Connecting Networks

AWS Academy Cloud Architecting

© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

Welcome to Module 7: Connecting Networks.

Module overview

Sections

1. Architectural need
2. Connecting to your remote network with AWS Site-to-Site VPN
3. Connecting to your remote network with AWS Direct Connect
4. Connecting VPCs in AWS with VPC peering
5. Scaling your VPC network with AWS Transit Gateway
6. Connecting your VPC to supported AWS services



Activity

- AWS Transit Gateway

Lab

- Guided Lab: Creating a VPC Peering Connection



Knowledge check

© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

2

This module includes the following sections:

1. Architectural need
2. Connecting to your remote network with AWS Site-to-Site VPN
3. Connecting to your remote network with AWS Direct Connect
4. Connecting VPCs in AWS with VPC peering
5. Scaling your VPC network with AWS Transit Gateway
6. Connecting your VPC to supported AWS services

This module also includes:

- An activity in which you discuss how you would use AWS Transit Gateway to connect three virtual private clouds (VPCs)
- A guided lab where you will create a VPC peering connection

Finally, you will be asked to complete a knowledge check that will test your understanding of key concepts covered in this module.

Module objectives

At the end of this module, you should be able to:

- Describe how to connect an on-premises network to the Amazon Web Services (AWS) Cloud
- Describe how to connect VPCs in the AWS Cloud
- Connect VPCs in the AWS Cloud by using VPC peering
- Describe how to scale VPCs in the AWS Cloud
- Describe how to connect VPCs to supported AWS services



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

3

At the end of this module, you should be able to:

- Describe how to connect an on-premises network to the Amazon Web Services (AWS) Cloud
- Describe how to connect VPCs in the AWS Cloud
- Connect VPCs in the AWS Cloud by using VPC peering
- Describe how to scale VPCs in the AWS Cloud
- Describe how to connect VPCs to supported AWS services

Section 1: Architectural need

Module 7: Connecting Networks



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

Introducing Section 1: Architectural need.

Café business requirement

The workloads for the café are increasing in complexity. The architecture must support connectivity between multiple VPCs, and be highly available and fault tolerant.



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

5

The café has started a loyalty rewards program, where a customer gets a free beverage or dessert after they purchase 10 or more similar items. When customers order online, they must provide some personally identifiable information (PII), such as an email address and credit card number. The café can't store this information in the cloud because of compliance reasons. Thus, Sofía and Nikhil need a way to connect their on-premises database (which stores sensitive customer information) to their cloud system (which stores transactions data). They must then map the data between the two systems to offer the rewards that their customers have earned.

Additionally, for security purposes, Sofía tells Olivia that she wants to isolate the development environment in one VPC and the production environment in another VPC, but still have connectivity between the two. Olivia agrees that this is a good idea, and advises Sofía to design the networking environment so that it's highly available and fault-tolerant.

Section 2: Connecting to your remote network with AWS Site-to-Site VPN

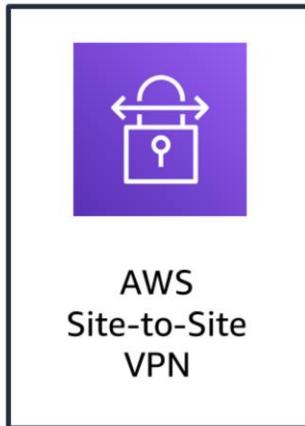
Module 7: Connecting Networks



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

Introducing Section 2: Connecting to your remote network with AWS Site-to-Site VPN.

AWS Site-to-Site VPN



AWS
Site-to-Site
VPN

AWS Site-to-Site is a highly available solution that enables you to securely [connect your on-premises network](#) or branch office site to your VPC.

- Uses internet protocol security (IPSec) communications to create encrypted virtual private network (VPN) tunnels
- Provides two encrypted tunnels per VPN connection
- Charged per VPN connection-hour



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

7

By default, instances that you launch into a virtual private cloud (VPC) on AWS cannot communicate with your on-premises network.

You can use AWS Site-to-Site Virtual Private Network (AWS Site-to-Site VPN) to securely connect your on-premises network or branch office site to your VPC. Each AWS Site-to-Site VPN connection uses internet protocol security (IPSec) communications to create encrypted VPN tunnels between two locations. A VPN tunnel is an encrypted link where data can pass from the customer network to or from AWS. The AWS side of the connection is the virtual private gateway. (Note that instead of a virtual private gateway, you can also create a Site-to-Site VPN connection as an attachment on a transit gateway. You will learn more about AWS Transit Gateway later in this module.) The on-premises side of the connection is the customer gateway.

AWS Site-to-Site VPN provides two VPN tunnels across multiple Availability Zones that you can use simultaneously for high availability. You can stream primary traffic through the first tunnel and use the second tunnel for redundancy. If one tunnel goes down, traffic will still get delivered to your VPC.

If you create a Site-to-Site VPN connection to your VPC, you are charged for each VPN connection-hour that your VPN connection is provisioned and available. For more information about pricing, see [AWS Site-to-Site VPN and Accelerated Site-to-Site VPN Connection Pricing](#).

Static and dynamic routing

Static routing

- Requires you to specify all routes (IP prefixes)
- Specify *static routing* if your customer gateway device **does not support** BGP

Dynamic routing

- Uses the Border Gateway Protocol (BGP) to advertise its routes to the virtual private gateway
- Specify *dynamic routing* if your customer gateway device **supports** BGP*

*We recommend that you use BGP-capable devices because the BGP protocol offers robust liveness detection checks.



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

8

When you create a Site-to-Site VPN connection, you must specify the type of routing that you plan to use and you must update the route table for your subnet.

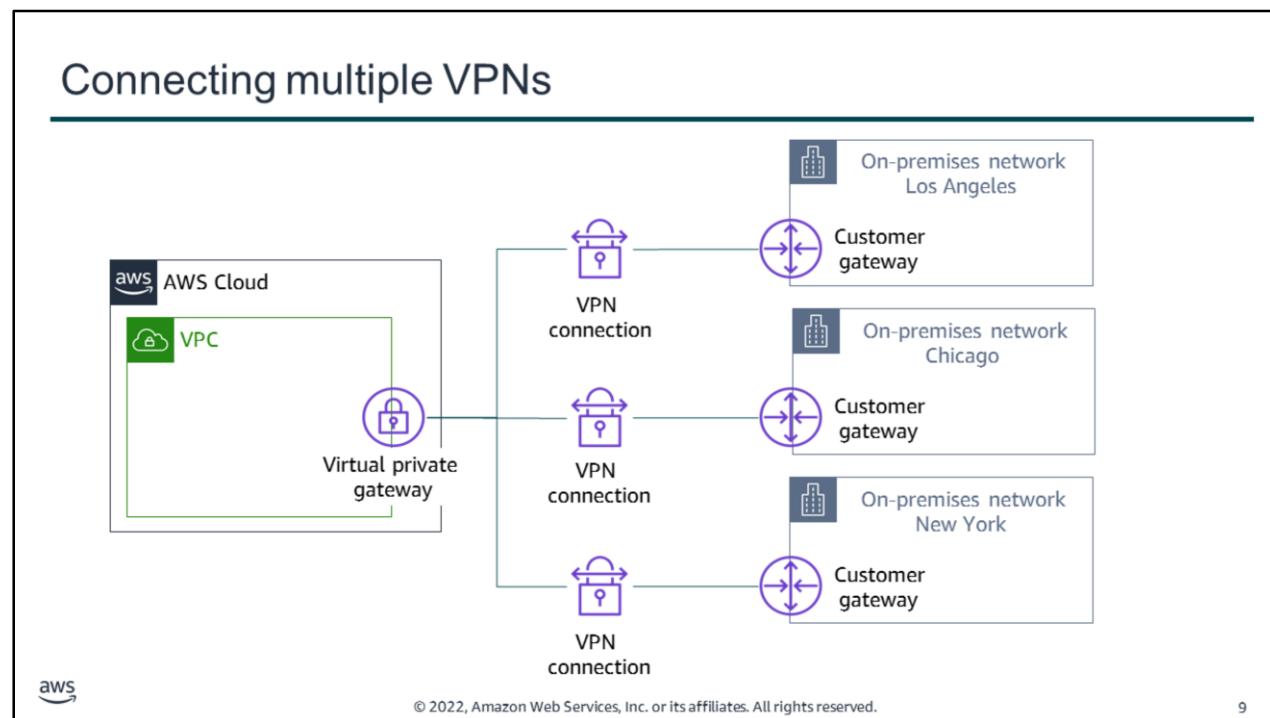
AWS Site-to-Site VPN supports two types of routing. The type of routing that you select depends on the make and model of your VPN devices:

- If your VPN device supports Border Gateway Protocol (BGP), specify dynamic routing when you configure your Site-to-Site VPN connection. Dynamic routing uses the BGP to advertise routes to the virtual private gateway. Dynamic routing supports a maximum of 100 propagated routes per route table. (For current limits, see [AWS Site-to-Site VPN limits](#).)
- If your VPN device does not support BGP, specify static routing. Static routing requires that you specify the routes (that is, IP prefixes) for your network that should be communicated to the virtual private gateway. Static routing supports 50 non-propagated routes per route table by default, up to a maximum of 1,000 non-propagated routes. (For current limits, see [AWS Site-to-Site VPN limits](#).)

We recommend that you use BGP-capable devices because the BGP protocol offers robust liveness detection checks that can assist failover to the second VPN tunnel if the first tunnel goes down. Devices that don't support BGP may also perform health checks to assist failover to the second tunnel when needed.

For a list of static and dynamic routing devices that have been tested with Amazon VPC, see [Customer Gateway Devices We've Tested](#) in the AWS Site-to-Site VPN Network Administrator Guide.

For more information about site-to-site VPN routing options, see [Static and Dynamic Routing Options](#).



To maintain high availability of your customer gateway, you can set up redundant customer gateway devices. If you have redundant customer gateway devices, each device advertises the same prefix (for example, 0.0.0.0/0) to the virtual private gateway. AWS uses BGP routing to determine the path for traffic. If one customer gateway device fails, the virtual private gateway directs all traffic to the working customer gateway device.

You can establish multiple VPN connections from multiple customer gateway devices to a single virtual private gateway using [AWS VPN CloudHub](#). This configuration can be used in different ways to implement redundancy and failover on your side of the VPN connection.

AWS VPN CloudHub operates on a hub-and-spoke model to enable multiple sites to access your VPC or to securely access each other. You can use it with or without a VPC. You configure each customer gateway device to advertise a site-specific prefix (such as 10.0.0.0/24, 10.0.1.0/24) to the virtual private gateway. The virtual private gateway routes traffic to the appropriate site and advertises the reachability of one site to all the other sites.

For more information about using AWS Site-to-Site VPN, see the following resources:

- [Site-to-Site VPN single and multiple connection examples](#)
- [Using redundant Site-to-Site VPN connections to provide failover](#)

Section 2 key takeaways



- AWS Site-to-Site VPN is a highly available solution that enables you to securely connect your on-premises network or branch office site to your VPC
- AWS Site-to-Site VPN supports both static and dynamic routing
- You can establish multiple VPN connections from multiple customer gateway devices to a single virtual private gateway

© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

10

Some key takeaways from this section of the module include:

- AWS Site-to-Site VPN is a highly available solution that enables you to securely connect your on-premises network or branch office site to your VPC
- AWS Site-to-Site VPN supports both static and dynamic routing
- You can establish multiple VPN connections from multiple customer gateway devices to a single virtual private gateway

Section 3: Connecting to your remote network with AWS Direct Connect

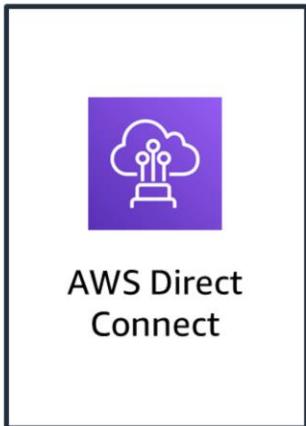
Module 7: Connecting Networks



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

Introducing Section 3: Connecting to your remote network with AWS Direct Connect.

AWS Direct Connect (DX)

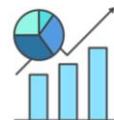


AWS Direct Connect

AWS Direct Connect (which is also known as DX) provides you with a **dedicated, private network connection capacity** of either 1 Gbps or 10 Gbps.



Reduces data transfer costs



Improves application performance with predictable metrics



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

12

As you learned, AWS Site-to-Site VPN is one option for connecting your on-premises network to the AWS global network. With this option, your data is transferred through encrypted tunnels over the public internet.

AWS Direct Connect (or DX) is another solution that goes beyond simple connectivity over the internet. DX uses open standard 802.1q virtual local area networks (VLANs) so you can establish a dedicated, private network connection from your premises to AWS. This private connection can reduce network costs, increase bandwidth throughput, and provide a more consistent network experience than internet-based connections.

Dedicated connections are available with 1-Gbps and 10-Gbps capacity.

DX use cases

- Hybrid environments
- Transferring large datasets
- Network performance predictability
- Security and compliance



AWS Direct Connect

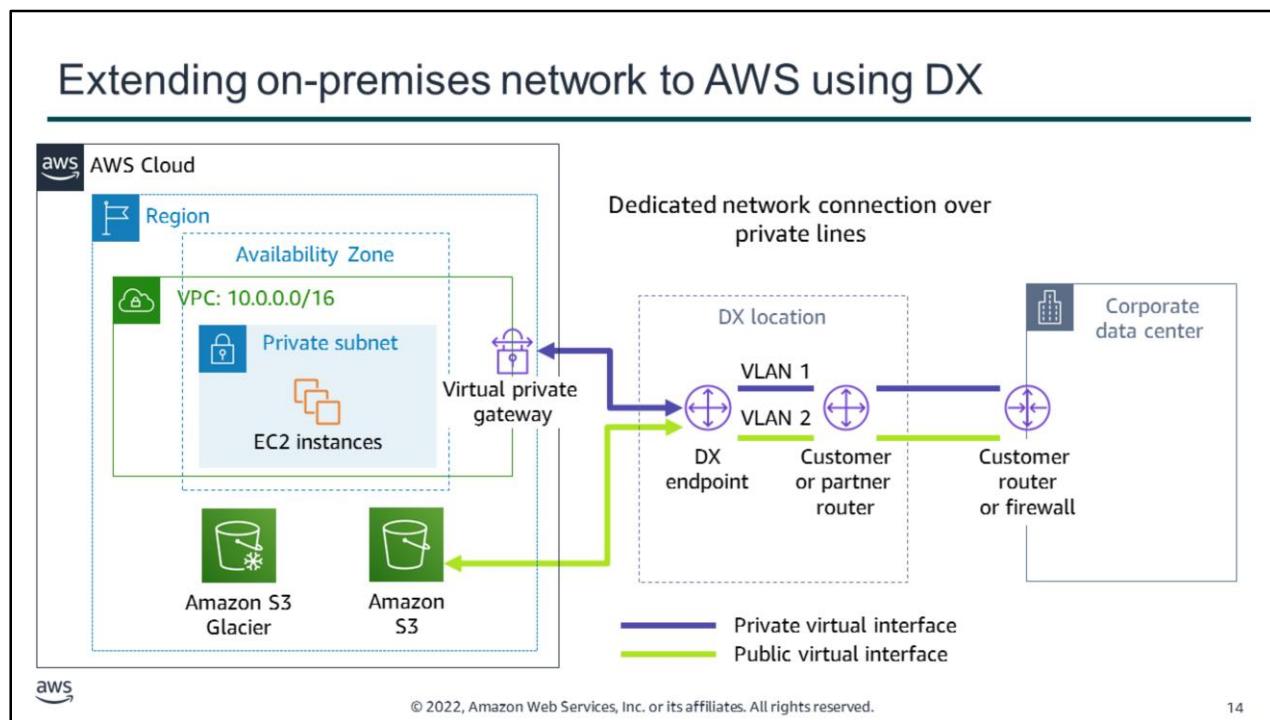


© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

13

DX is useful for several scenarios, for example:

- Hybrid environments – For applications that require access to existing data center equipment, such as an on-premises database, DX enables you to create a hybrid environment that allows you to take advantage of the elasticity and economic benefits of AWS.
- Transferring large datasets – For applications that operate on large datasets, such as high performance computing (HPC) applications, transferring large datasets over the internet between your data center and the AWS Cloud can be time consuming and expensive. For such applications, connecting to the AWS Cloud using DX is a good solution because:
 - Network transfers will not compete for internet bandwidth at your data center.
 - The high-bandwidth link reduces the potential for network congestion and degraded application performance.
 - By limiting the internet bandwidth used by your application, you can reduce network fees that you pay to your internet service provider (ISP) and avoid having to pay for increased internet bandwidth commitments or new contracts. In addition, all data that is transferred over DX is charged at the reduced DX data transfer rate instead of internet data transfer rates, which can reduce your network costs.
- Improved application performance – Applications that require predictable network performance can also benefit from DX. Examples include applications that operate on real-time data feeds, such as audio or video streams. In such cases, a dedicated network connection can provide more consistent network performance than standard internet connectivity.
- Security and compliance – Enterprise security or regulatory policies sometimes require that applications hosted on the AWS Cloud can be accessed through private network circuits only. DX is a natural solution to this requirement because traffic between your data center and your application flows through the dedicated private network connection.

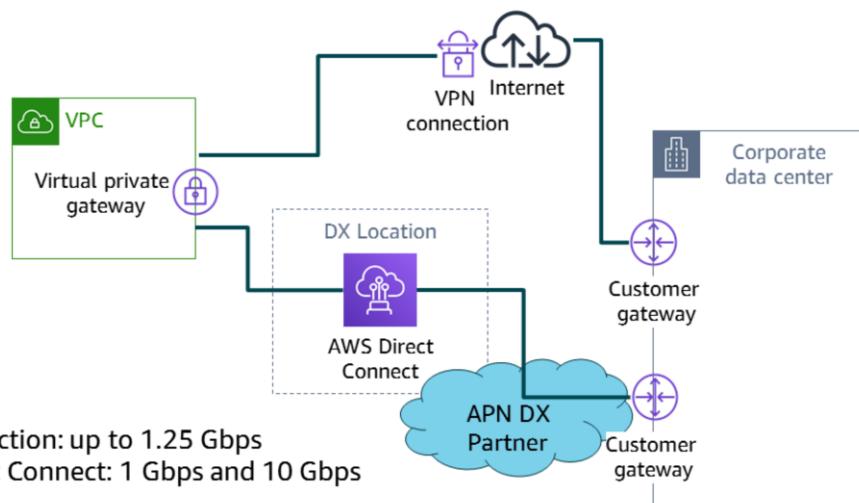


DX links your internal network to a DX location over a standard Ethernet fiber-optic cable. One end of the cable is connected to your router. The other end is connected to a DX router. With this connection, you can create virtual interfaces that enable direct access to AWS services. A public virtual interface enables access to public AWS services, such as Amazon Simple Storage Service (Amazon S3). A private virtual interface enables access to your VPC.

You can access any VPC or public AWS service in any Region (except China) from any supported [DX location](#). If you do not have equipment at a DX location, you can access DX with the assistance of a [DX AWS Partner Network \(APN\) Partner](#).

For information about DX, see [What is AWS Direct Connect?](#)

Enabling high availability: DX with backup VPN connection



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

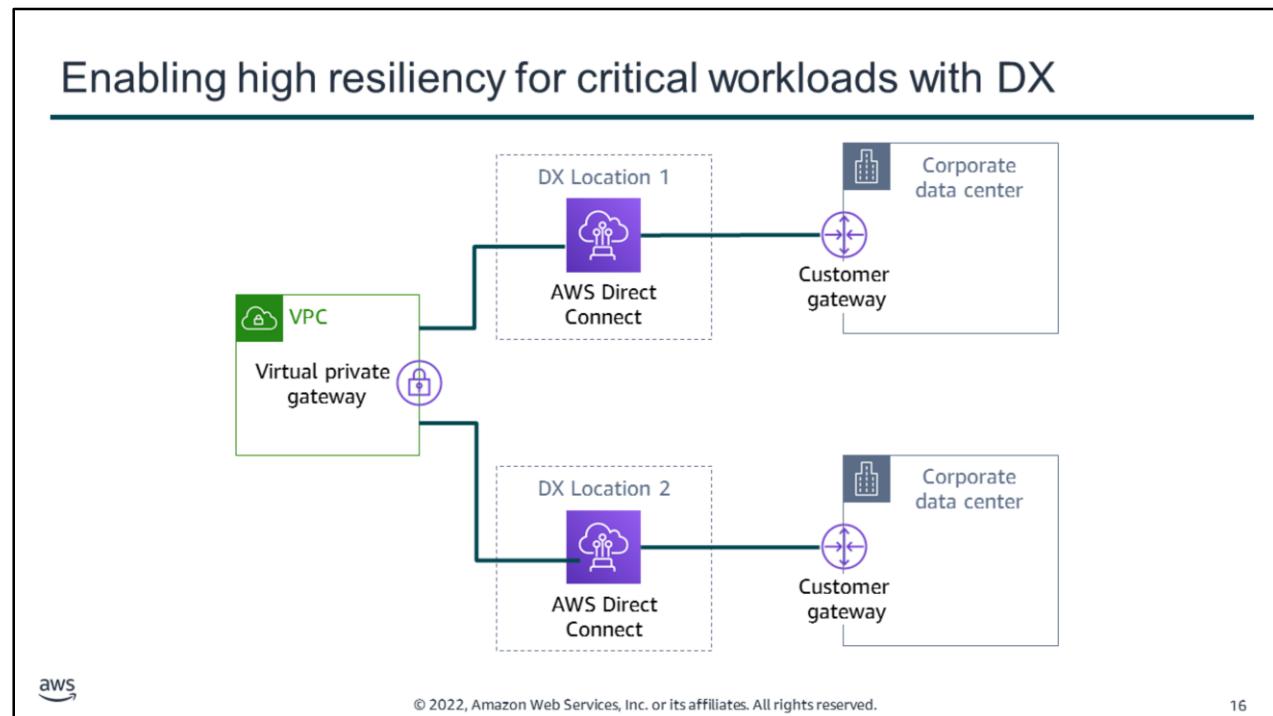
15

You can implement highly available connectivity between your data centers and your VPC by coupling one or more DX connections that you use for primary connectivity with a lower-cost backup VPN connection.

In this example, the configuration consists of two dynamically routed connections—one that uses DX and another that uses a VPN connection—from two different customer devices. AWS provides example router configurations to help you establish both DX and dynamically routed VPN connections. By default, AWS will always prefer to send traffic over your DX connection, so you do not need additional configuration specific to AWS to define primary and backup connections. However, you should configure DX and VPN-specific internal-route propagation to ensure that internal systems select the appropriate paths.

This approach enables you to choose the primary network path and network provider for your AWS traffic, with the option of using a different provider for a backup VPN connection. Choose network providers and DX locations that align with your organization's risk tolerance, financial expectations, and data center connectivity policies.

Finally, you can use multiple DX circuits and multiple VPN tunnels between separately deployed private IP address spaces. You can also use multiple DX locations for high availability. If you use multiple AWS Regions, you will also need multiple DX locations in at least two Regions. You might want to evaluate AWS Marketplace appliances that terminate VPNs.



Highly resilient, fault-tolerant network connections are key to a well-architected system. AWS recommends connecting from multiple data centers for physical location redundancy. When you design remote connections, consider using redundant hardware and telecommunications providers.

Additionally, it is a best practice to use dynamically routed, active/active connections for automatic load balancing and failover across redundant network connections. Provision sufficient network capacity to ensure that the failure of one network connection does not overwhelm and degrade redundant connections.

For critical production workloads that require high resiliency, AWS recommends that you have one connection at multiple locations. As shown in the architecture diagram, such a topology ensures resilience against connectivity failures due to a hardware failure or a complete location failure. You can use [Direct Connect Gateway](#) to access any AWS Region (except AWS Regions in China) from any DX location.

To learn more about additional topology guidelines to keep in mind when you connect to AWS, see [AWS Direct Connect Resiliency Recommendations](#).

Section 3 key takeaways



- AWS Direct Connect uses open standard 802.1q VLANs that enable you to establish a **dedicated, private network connection from your premises to AWS**
- You can access any VPC or public AWS service in any Region (except China) from any supported **DX location**
- You can **implement highly available connectivity between your data centers and your VPC** by coupling one or more DX connections that you use for primary connectivity with a lower-cost, backup VPN connection
- To implement a **highly resilient, fault-tolerant architecture**, connect to your AWS network from multiple data centers so you can have physical location redundancy

© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

17

Some key takeaways from this section of the module include:

- AWS Direct Connect uses open standard 802.1q VLANs to let you establish a dedicated, private network connection from your premises to AWS
- You can access any VPC or public AWS service in any Region (except China) from any supported DX location
- You can implement highly available connectivity between your data centers and your VPC by coupling one or more DX connections that you use for primary connectivity with a lower-cost, backup VPN connection
- To implement a highly resilient, fault-tolerant architecture, connect to your AWS network from multiple data centers so you can have physical location redundancy

Section 4: Connecting VPCs in AWS with VPC peering

Module 7: Connecting Networks

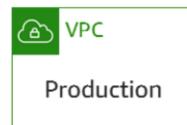
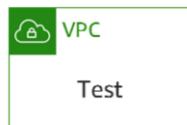
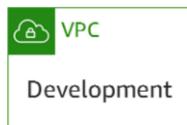


© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

Introducing Section 4: Connecting VPCs in AWS with VPC peering.

Connecting VPCs

- Isolating some of your workloads is generally a good practice
- However, you might need to transfer data between two or more VPCs



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

19

Isolating your workloads in individual VPCs is generally a good practice. For example, when your business or architecture becomes large enough, you might need to separate logical elements for security, for architectural purposes, or for simplicity. However, it can be highly desirable to have connectivity between your VPCs for situations when you need to transfer data between them.

VPC peering

- One-to-one networking connection between two VPCs
- No gateways, VPN connections, and separate network appliances needed
- Highly available connections
- No single point of failure or bandwidth bottleneck
- Traffic always stays on the global AWS backbone



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

20

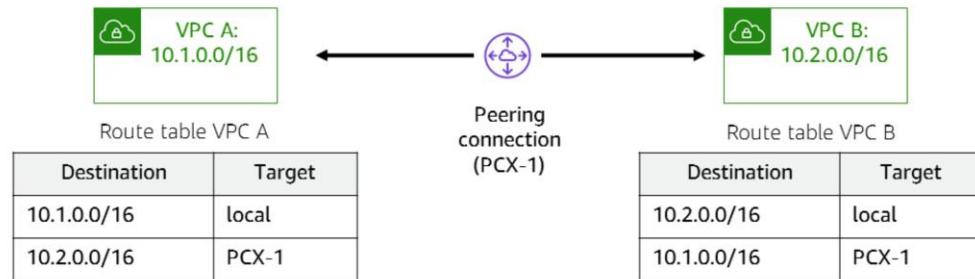
VPC peering is a one-to-one networking connection between two VPCs that enables you to route traffic between them privately. Instances in either VPC can communicate with each other like they are in the same network. You can create a VPC peering connection between your own VPCs, with a VPC in another AWS account, or with a VPC in a different AWS Region.

You can establish peering relationships between VPCs across different AWS Regions. Inter-Region VPC peering provides a simple and cost-effective way to share resources between Regions or replicate data for geographic redundancy. Data that is transferred across inter-Region VPC peering connections is charged at the standard inter-Region data transfer rates.

Inter-Region VPC peering enables VPC resources to communicate with each other using private IP addresses without requiring gateways, VPN connections, or separate network appliances. Some examples of VPC resources include Amazon Elastic Compute Cloud (Amazon EC2) instances, Amazon Relational Database Service (Amazon RDS) databases, and AWS Lambda functions that run in different Regions.

Traffic remains in the private IP address space. All inter-Region traffic is encrypted with no single point of failure or bandwidth bottleneck. Traffic always stays on the global AWS backbone. Traffic never traverses the public internet, which reduces threats, such as common exploits and distributed denial of service (DDoS) attacks.

Establishing VPC peering



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

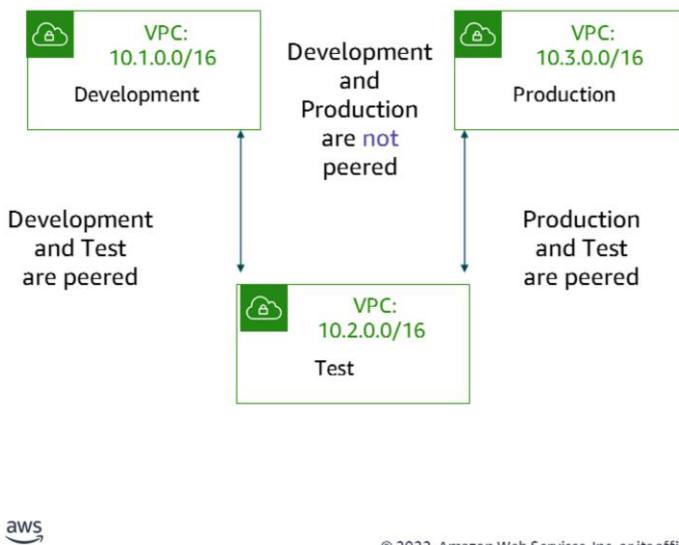
21

To establish a VPC peering connection, the owner of the requester VPC (or local VPC) sends a request to the owner of the VPC peer. To activate the connection, the owner of the VPC peer must accept the VPC peering connection request.

To enable the flow of the traffic between the VPC peers by using private IP addresses, you must add a route to one or more of your VPC's route tables. This route must point to the IP address range of the VPC peer. The owner of the VPC peer adds a route to one of their VPC's route tables that points to the IP address range of your VPC.

You might also need to update the security group rules that are associated with your instance so that traffic to and from the peer VPC is not restricted.

VPC peering connection restrictions



- Use **private IP addresses**
- Can be established between **different AWS accounts**
- Cannot have overlapping CIDR blocks
- Can have only one **peering resource** between any two VPCs
- Do not support **transitive peering relationships**

© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

22

There are some restrictions you should be aware of when you establish VPC peering connections:

- VPC peering connections use private IP addresses.
- VPC peering connections can be established between different AWS accounts. The CIDR block of the VPC peer cannot overlap with the CIDR block of the requester.
- You can have only one peering resource between any two VPCs.
- Transitive peering is not supported. For example, in the diagram, the Development and Test VPCs are peered, and the Production and Test VPCs are peered. However, this does not mean that the Production VPC is connected to the Development VPC. By default, VPC peering does not allow the Production VPC to connect to the Development VPC unless they are *explicitly established as peers*. Therefore, you control which VPCs can communicate with each other.

To learn more about VPC peering connection restrictions, see [VPC peering limitations](#).

Considerations for peering multiple VPCs

When you connect multiple VPCs, consider these [network design principles](#):

Only connect
essential VPCs



Make sure your
solution can scale



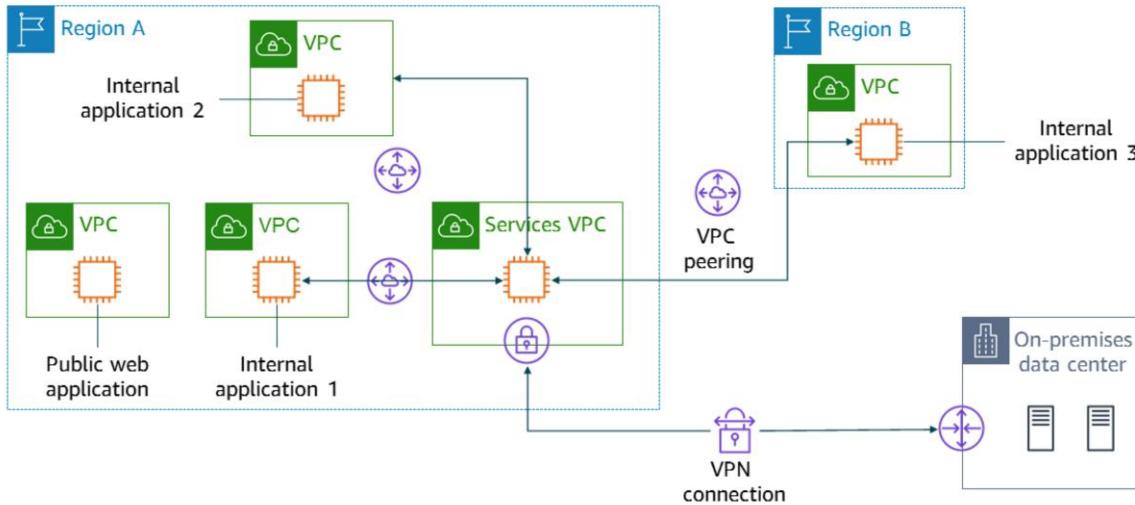
© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

23

When you connect multiple VPCs in a single AWS Region, consider these network design principles:

- Connect only those VPCs that truly must communicate with each other
- Make sure that the solution you choose can scale according to your current and future VPC connectivity needs

Example: VPC peering for shared resources



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

24

Here is an example of how you can use VPC peering for shared resources.

In this example, each department VPC in a company peers with a shared *Services VPC*. This VPC contains connections to Microsoft Active Directory, security scanning tools, monitoring and logging tools, and various other capabilities. It also provides a proxy through which the department VPCs can access some on-premises resources. VPC peering enables company applications that are in different VPCs to access the shared Services VPC but remain isolated from each other. In this example, also notice that a VPC peering connection was established between VPCs in different Regions.

Section 4 key takeaways



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

25

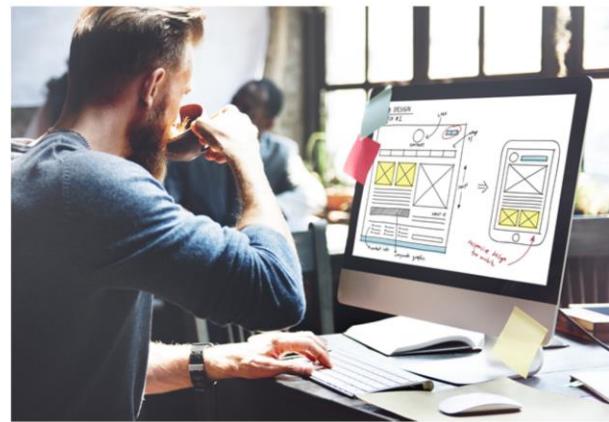
- VPC peering is a **one-to-one networking connection between two VPCs** that enables you to route traffic between them privately
- You can establish peering relationships between VPCs **across different AWS Regions**
- VPC peering connections –
 - Use private IP addresses
 - Can be established between different AWS accounts
 - Cannot have overlapping CIDR blocks
 - Can have only one peering resource between any two VPCs
 - Do not support transitive peering relationships

Some key takeaways from this section of the module include:

- VPC peering is a one-to-one networking connection between two VPCs that enables you to route traffic between them privately
- You can establish peering relationships between VPCs across different AWS Regions
- VPC peering connections –
 - Use private IP addresses
 - Can be established between different AWS accounts
 - Cannot have overlapping CIDR blocks
 - Can have only one peering resource between any two VPCs
 - Do not support transitive peering relationships

Module 7 – Guided Lab:

Creating a VPC Peering Connection



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

26

You will now complete Module 7 – Guided Lab: Creating a VPC Peering Connection.

Guided lab: Tasks

1. Create a peering connection between two VPCs
2. Configure route tables to send traffic to the peering connection
3. Test the peering connection



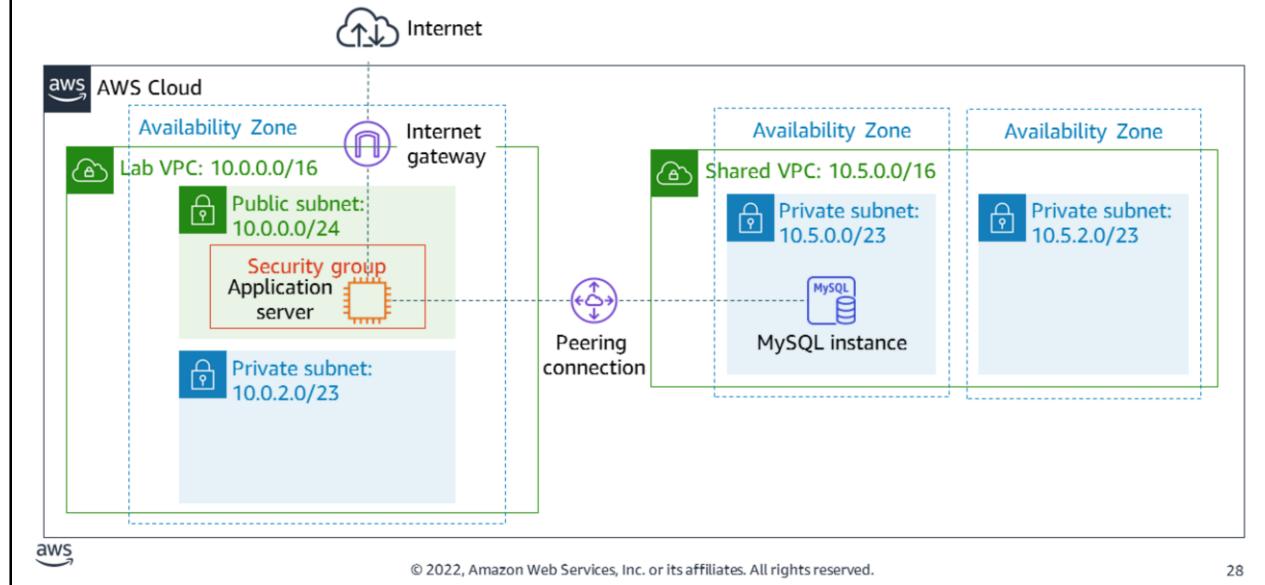
© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

27

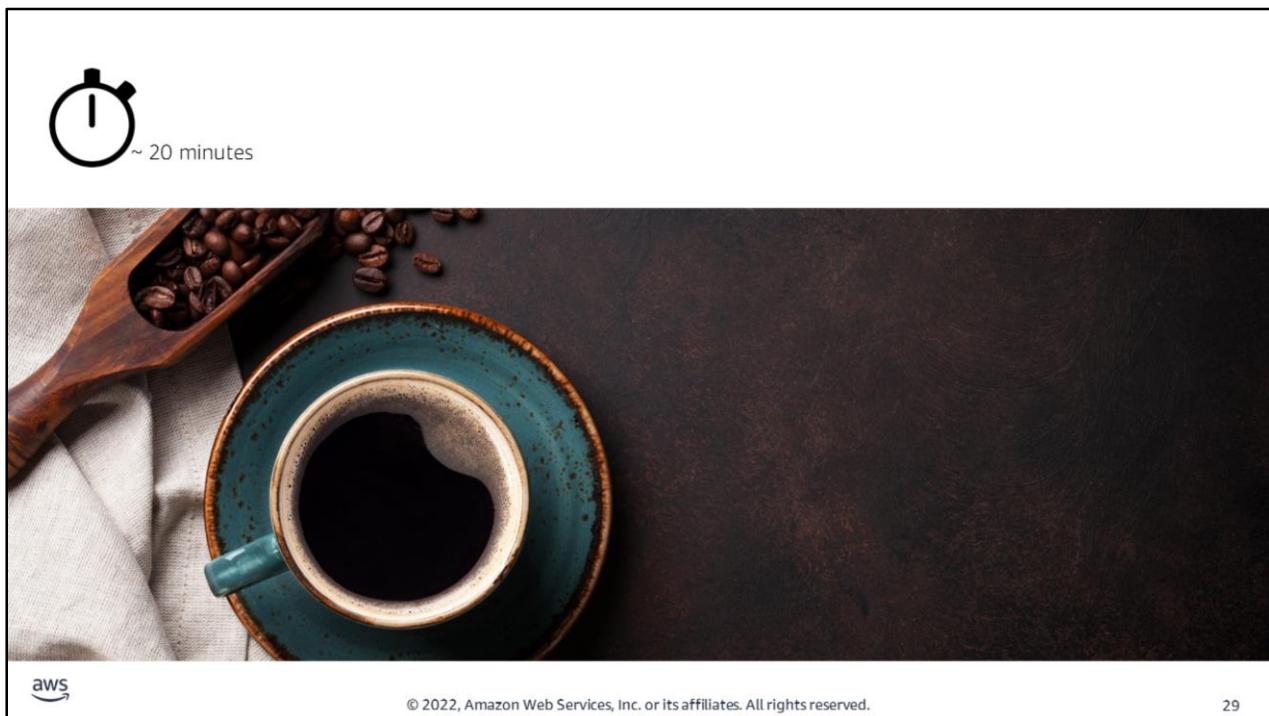
In this guided lab, you will complete the following tasks:

1. Create a peering connection between two VPCs
2. Configure route tables to send traffic to the peering connection
3. Test the peering connection

Guided lab: Final product



The diagram summarizes what you will have built after you complete the lab.



It is now time to start the guided lab.

Guided lab debrief: Key takeaways



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

30

Your educator might choose to lead a conversation about the key takeaways from this guided lab after you have completed it.

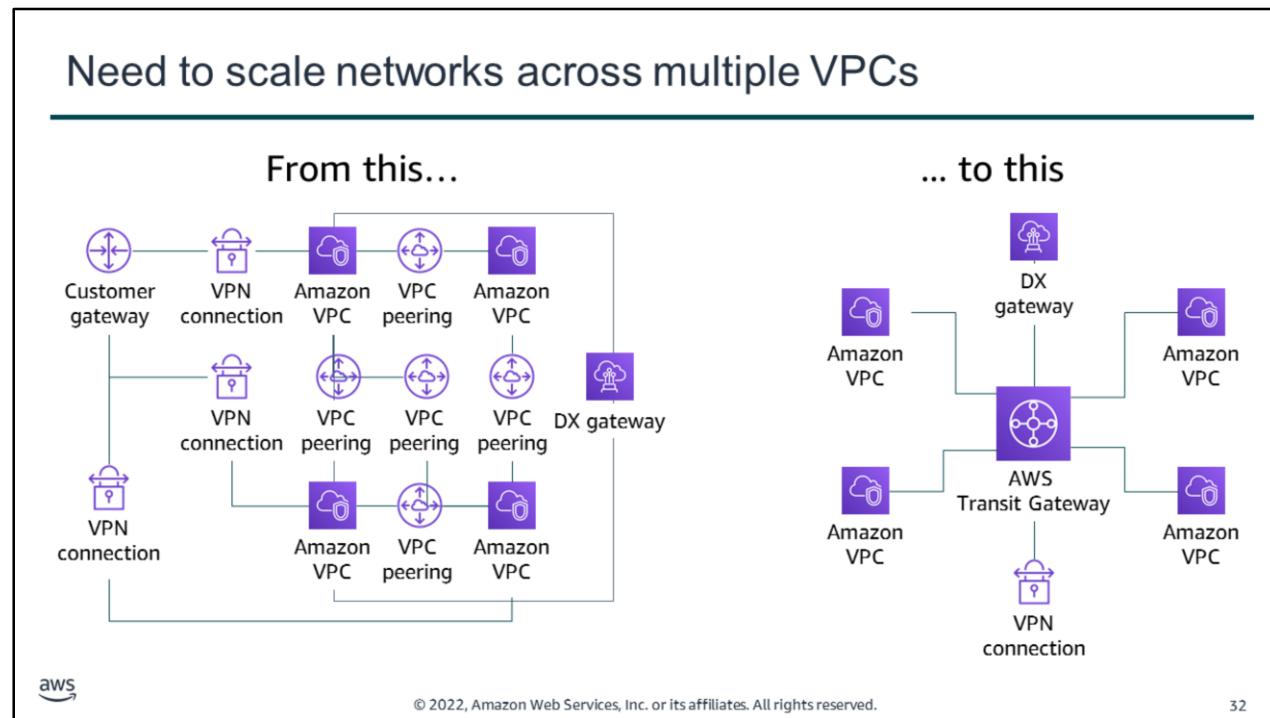
Section 5: Scaling your VPC network with AWS Transit Gateway

Module 7: Connecting Networks



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

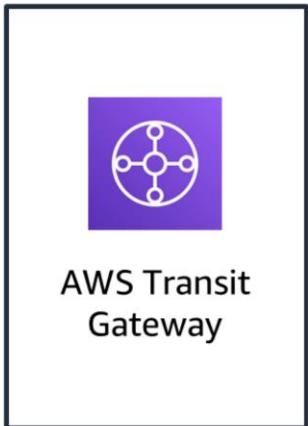
Introducing Section 5: Scaling your VPC network with AWS Transit Gateway.



As you grow the number of workloads that run on AWS, you must be able to scale your networks across multiple accounts and VPCs to keep up with the growth. You can use VPC peering to connect pairs of VPCs. However, it can be operationally costly and difficult to manage point-to-point connectivity across many VPCs if you cannot centrally manage the connectivity policies. For on-premises connectivity, you must attach your VPN to each individual VPC. This solution can be time-consuming to build and difficult to manage when the number of VPCs grows into the hundreds.

It's important to consider how large your environment might become over time, how well it will scale, and how you will organize your VPCs. To solve this problem, you can use AWS Transit Gateway to simplify your networking model.

AWS Transit Gateway



AWS Transit Gateway is a service that enables you to connect your VPCs and on-premises networks to a single gateway.

- Fully managed, highly available, flexible routing service
- Acts as a hub for all traffic to flow through between your networks
- Connects up to 5,000 VPCs and on-premises environments with a single gateway



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

33

AWS Transit Gateway is a service that enables you to connect your VPCs and on-premises networks to a single gateway (called a transit gateway). With AWS Transit Gateway, you only need to create and manage a single connection from the central gateway into each VPC, on-premises data center, or remote office across your network.

AWS Transit Gateway uses a hub-and-spoke model. This model significantly simplifies management and reduces operational costs because each network only needs to connect to the transit gateway and not to every other network. Any new VPC is connected to the transit gateway, and is then automatically available to every other network that is connected to the transit gateway. This ease of connectivity makes it easier to scale your network as you grow.

You can use AWS Transit Gateway to connect up to 5,000 VPCs and on-premises networks.

Connecting multiple VPCs

Scenario: We want to fully connect three VPCs.



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

34

To understand how to connect multiple VPCs by using AWS Transit Gateway, consider this scenario. You want to fully connect three VPCs in your network. In this scenario, you learn how to deploy AWS Transit Gateway and three VPCs with non-overlapping IP address space in a single Region. You then attach the transit gateway to these VPCs.

Step 1: Create a transit gateway

Scenario: We want to fully connect three VPCs.

The diagram illustrates the initial step of connecting multiple VPCs via a transit gateway. On the left, three separate VPC components are shown, each consisting of a green cloud icon and a box labeled 'VPC 1: 10.1.0.0/16', 'VPC 2: 10.2.0.0/16', and 'VPC 3: 10.3.0.0/16' respectively. These three boxes are connected by a vertical pink line. To the right of this pink line is a purple rectangular box labeled 'AWS Transit Gateway (tgw-xxx)'. A dashed line connects the bottom of the pink line to the bottom of the purple box, indicating they are part of the same network architecture. The entire diagram is contained within a white box with a thin black border.

© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

aws

35

The first step for connecting multiple VPCs is to create a transit gateway. A *transit gateway* is a transit hub that you can use to interconnect your VPCs and on-premises networks. It acts as a Regional virtual router for traffic that flows between your VPCs and VPN connections. A transit gateway scales elastically based on the volume of network traffic.

You can set up a transit gateway through the Amazon VPC dashboard. Various charges apply for using AWS Transit Gateway, so make sure that your architecture and budget can support using a transit gateway.

For more information, see [What Is a Transit Gateway?](#)

Step 2: Deploy elastic network interfaces

Scenario: We want to fully connect three VPCs.

The diagram illustrates the connection between three VPCs and an AWS Transit Gateway. On the left, three VPCs are shown: VPC 1 (10.1.0.0/16), VPC 2 (10.2.0.0/16), and VPC 3 (10.3.0.0/16). Each VPC has two blue circular icons representing ENIs. To the right, a purple icon represents the AWS Transit Gateway (tgw-xxx). A pink vertical rectangle surrounds the three VPCs. An arrow points from the 'Destination' column of the VPC 3 route table to the pink rectangle, indicating that traffic from VPC 3 is sent to the transit gateway.

VPC 3 route table

Destination	Target
10.3.0.0/16	local

AWS Transit Gateway (tgw-xxx)

© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

aws

36

AWS Transit Gateway connects to a VPC through elastic network interfaces (or ENIs), which are deployed into subnets.

You must ensure that every Availability Zone that is part of the VPC has an ENI that connects the VPC to the transit gateway. You can do this by selecting at least one subnet from each Availability Zone for the ENI.

Notice in this example that the route table for VPC 3 has a single destination route that is local for VPC 3 using the 10.3.0.0/16 network.

Step 3: Update the VPC route table

Scenario: We want to fully connect three VPCs.

VPC 3 route table

Destination	Target
10.3.0.0/16	local
10.0.0.0/8	tgw-xxx

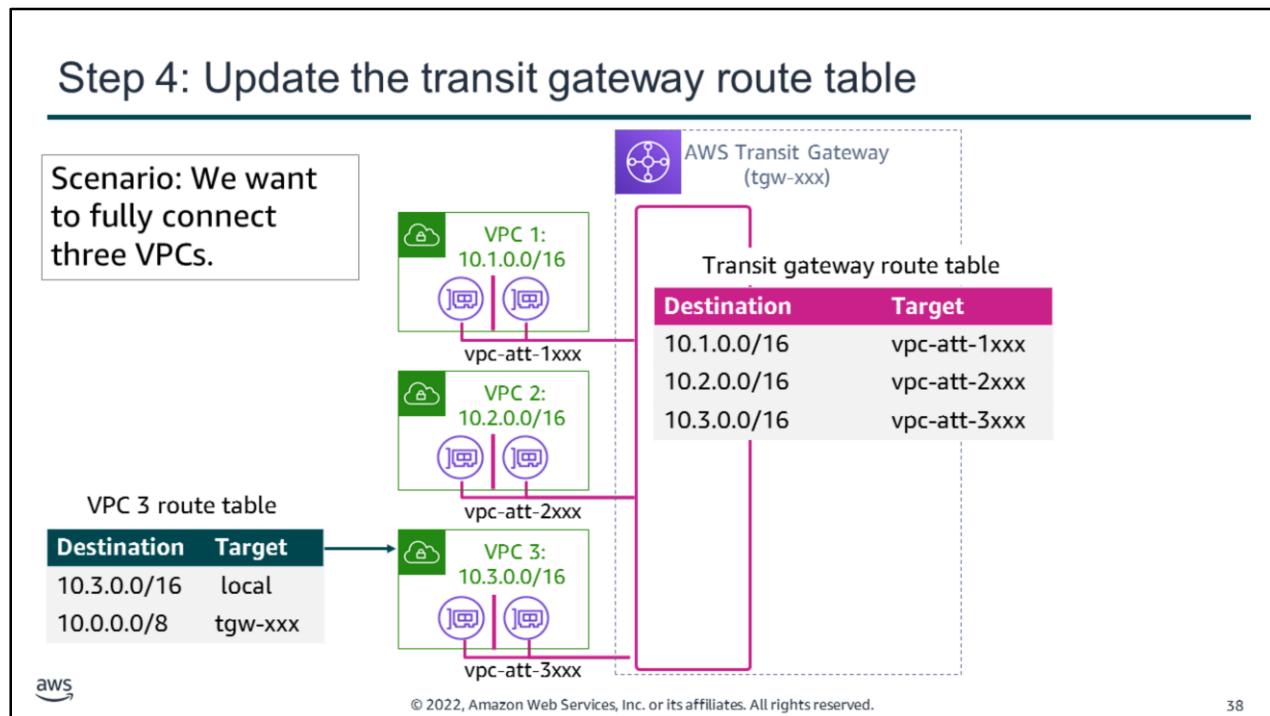
AWS Transit Gateway (tgw-xxx)

© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

37

After attaching the ENIs, the next step is to add a route in the VPC route table to send traffic that's destined for the other VPCs in the network to the transit gateway.

In this example, the second line of the VPC 3 route table shows that traffic destined for the 10.0.0.0/8 network is sent to the transit gateway. This route enables any traffic from VPC 3 going to either VPC 1 or VPC 2 to be sent to the transit gateway, because the CIDR block 10.0.0.0/8 includes the 10.X.0.0/16 CIDR blocks, which are used by the individual VPCs.

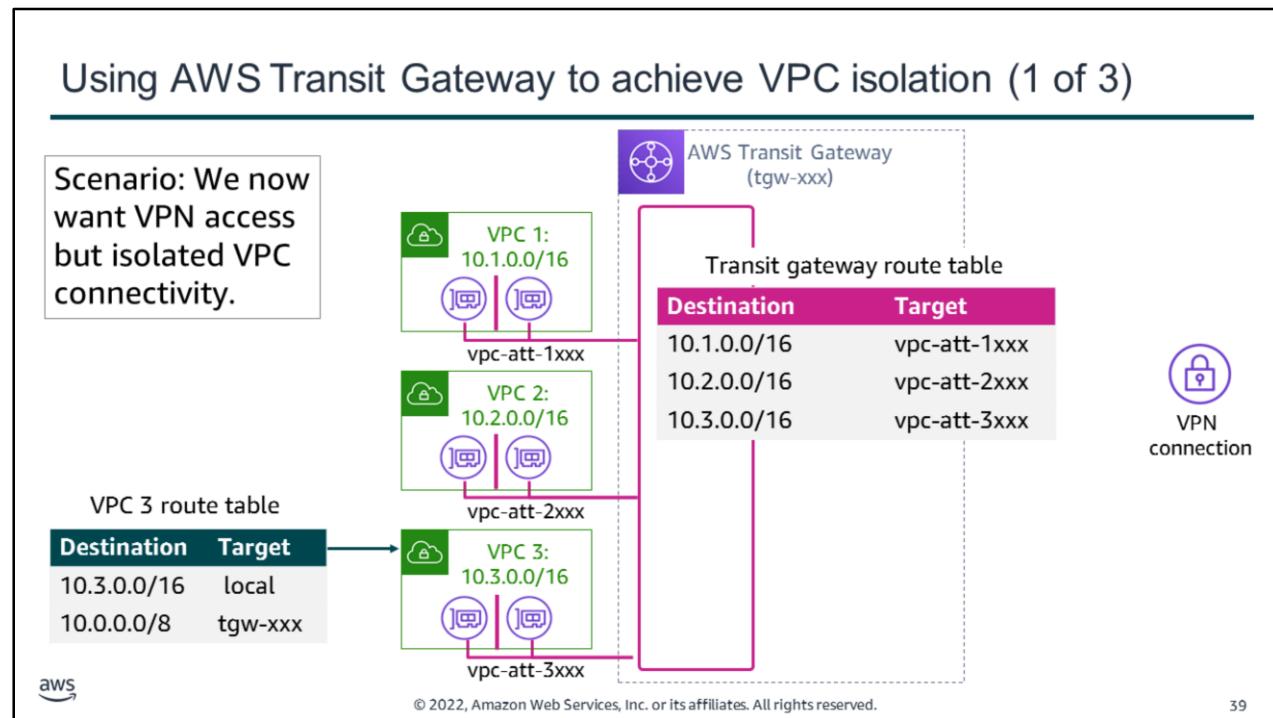


Next, you must configure the transit gateway route table to route traffic to the connected VPCs.

When you create a transit gateway, a default transit gateway route table is created. Each route in the transit gateway route table enables the transit gateway to send traffic destined for one of the VPCs to a corresponding attachment, which is a reference to the ENI that is attached to the VPC itself.

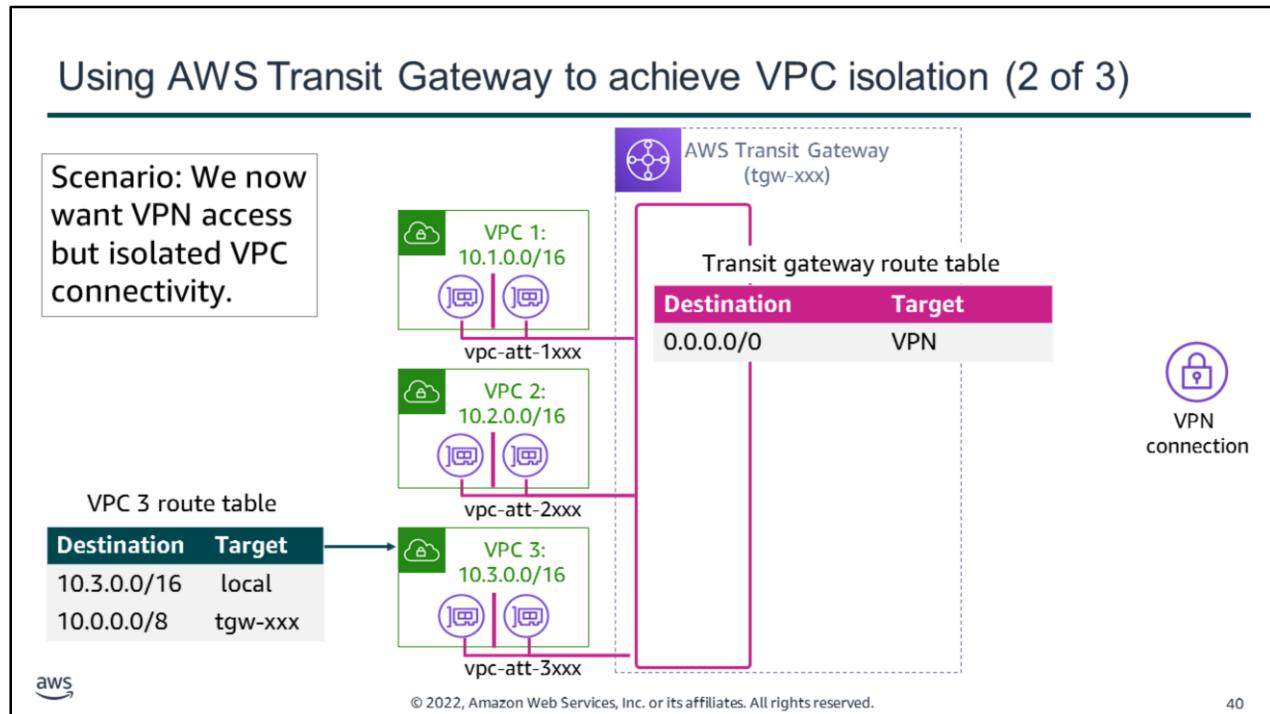
In this example, there is a route in the transit gateway route table that sends any traffic destined for the 10.1.0.0/16 network to vpc-att-1xxx, which is the attachment for VPC 1. Similarly, any traffic destined for the other VPC networks are sent to the corresponding attachments.

For more information about how to create connected environments using AWS Transit Gateway, see [Getting Started with Transit Gateways](#).



Though you can use AWS Transit Gateway to connect multiple VPCs, you can also use it to achieve isolation in your VPC environment. In this scenario, you want to connect your VPN source to your VPC environment. You also want to prevent your VPCs from directly connecting to each other, leaving the VPN to decide if traffic from one VPC has to be forwarded to another.

By setting up the route table appropriately for the transit gateway, you can prevent information sharing between the VPCs.

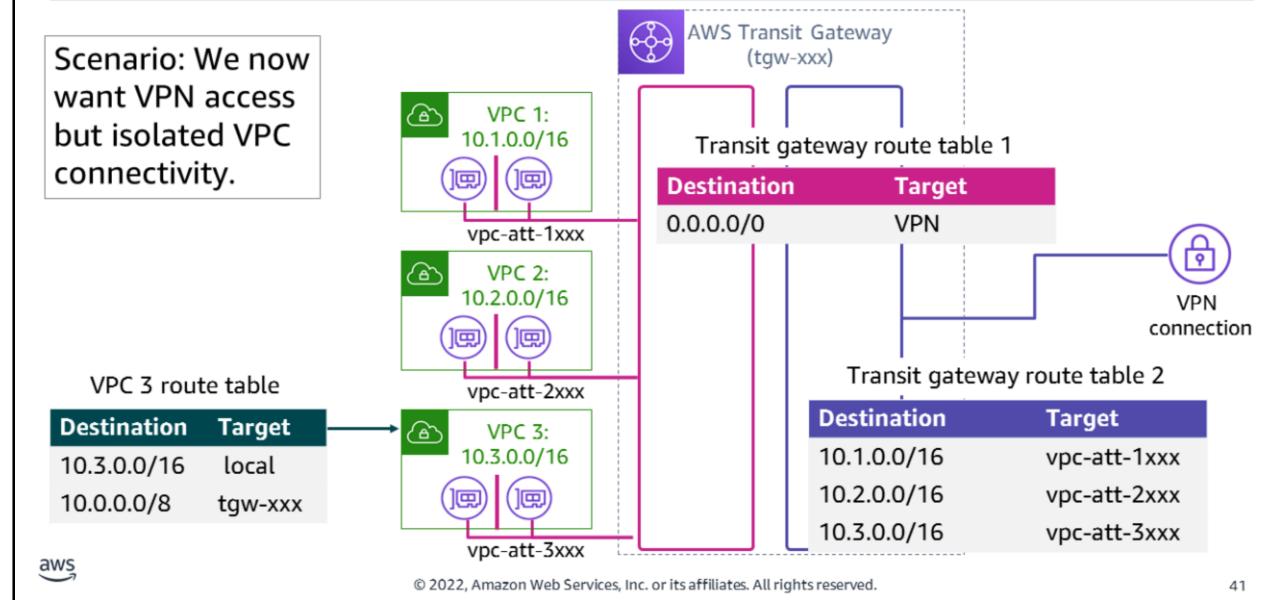


To implement this solution, update the route in the transit gateway route table to send all known traffic to the VPN connection.

In this example, when traffic for any of the VPCs in the 10.0.0.0/8 network is sent from VPC 3 to the transit gateway, the transit gateway will forward the traffic to the VPN as shown on the slide. The transit gateway will not send the traffic to any of the other VPCs because there are no routes pointing to any of the VPC attachments.

You now have isolated and secured VPN access to your VPC environment with no cross communication between the VPCs.

Using AWS Transit Gateway to achieve VPC isolation (3 of 3)



41

You can create multiple transit gateway route tables for specific interactions to direct traffic, as you see fit.

In this example, the second route table will direct inbound traffic from the VPN to one of the corresponding VPCs attached to the transit gateway.

Activity: AWS Transit Gateway



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

42

You will now complete the following activity: AWS Transit Gateway.

AWS Transit Gateway: Challenge

Scenario: How do you connect these five VPCs?

The diagram illustrates a network topology where five Virtual Private Clouds (VPCs) are interconnected via an AWS Transit Gateway (tgw-xxx). The VPCs are represented as green boxes with cloud icons, each containing a purple interface icon. The transit gateway is shown as a central purple square with a circular connection points. Pink lines represent the connections between the VPCs and the transit gateway.

VPC #	route table
Destination	Target
10.#.0.0/16	local
?	?

Transit gateway route table	
Destination	Target
?	?

© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

aws

In this activity, you have five VPCs that you want connect to each other through AWS Transit Gateway.

Answer the following questions:

- What routes are necessary to add to each of the VPC route tables to enable full connectivity?
- What routes are necessary to add to the transit gateway route table to enable full connectivity?

AWS Transit Gateway activity: Solution

Scenario: How do you connect these five VPCs?

VPC 3 route table

Destination	Target
10.3.0.0/16	local
10.0.0.0/8	tgw-xxx

Transit gateway route table

Destination	Target
10.1.0.0/16	vpc-att-1xxx
10.2.0.0/16	vpc-att-2xxx
10.3.0.0/16	vpc-att-3xxx
10.4.0.0/16	vpc-att-4xxx
10.5.0.0/16	vpc-att-5xxx

© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

aws

What routes must you add in each VPC route table to enable full connectivity?

- See the provided solution for the VPC 3 route table. You update the other VPC route tables in a similar way.

What routes must you add to the transit gateway route table to enable full connectivity?

- Add a route for each VPC attachment to direct traffic to each VPC.

Section 5 key takeaways



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

45

- AWS Transit Gateway enables you to connect your VPCs and on-premises networks to a **single gateway** (called a transit gateway)
- AWS Transit Gateway uses a **hub-and-spoke model** to simplify VPC management and reduce operational costs

Some key takeaways from this section of the module include:

- AWS Transit Gateway enables you to connect your VPCs and on-premises networks to a single gateway (called a transit gateway)
- AWS Transit Gateway uses a hub-and-spoke model to simplify VPC management and reduce operational costs

Section 6: Connecting your VPC to supported AWS services

Module 7: Connecting Networks



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

Introducing Section 6: Connecting your VPC to supported AWS services.

VPC endpoints

- Enable you to privately connect your VPC to supported AWS services and to VPC endpoint services that are powered by AWS PrivateLink
- Enable traffic between your VPC and the other service **without leaving the Amazon network**
- Do not require an internet gateway, VPN, network address translation (NAT) devices, or firewall proxies
- Are horizontally scaled, redundant, and highly available



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

47

A *VPC endpoint* enables you to privately connect your VPC to supported AWS services and to VPC endpoint services that are powered by AWS PrivateLink. VPC endpoint services that are powered by AWS PrivateLink include some AWS services, services hosted by other AWS customers and AWS Partner Network (APN) Partners in their own VPCs (which are referred to as *endpoint services*), and supported AWS Marketplace Partner services.

VPC endpoints do not require an internet gateway, NAT device, VPN connection, or DX connection. Instances in your VPC do not require public IP addresses to communicate with resources in the service. Traffic between your VPC and the other service does not leave the Amazon network.

Endpoints are virtual devices. They are horizontally scaled, redundant, and highly available VPC components. Endpoints allow communication between instances in your VPC and services without imposing availability risks or bandwidth constraints on your network traffic.

Two types of VPC endpoints

- **Interface endpoint** – An elastic network interface with a private IP address that serves as an entry point for traffic destined to a supported service
- Powered by [AWS PrivateLink](#)
- Examples –
 - Amazon CloudWatch
 - Amazon EC2 API
 - Elastic Load Balancing
- **Gateway endpoint** – A gateway that you specify as a target for a route in your route table for traffic destined to a supported AWS service
- Supported AWS services –
 - Amazon S3
 - Amazon DynamoDB



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

48

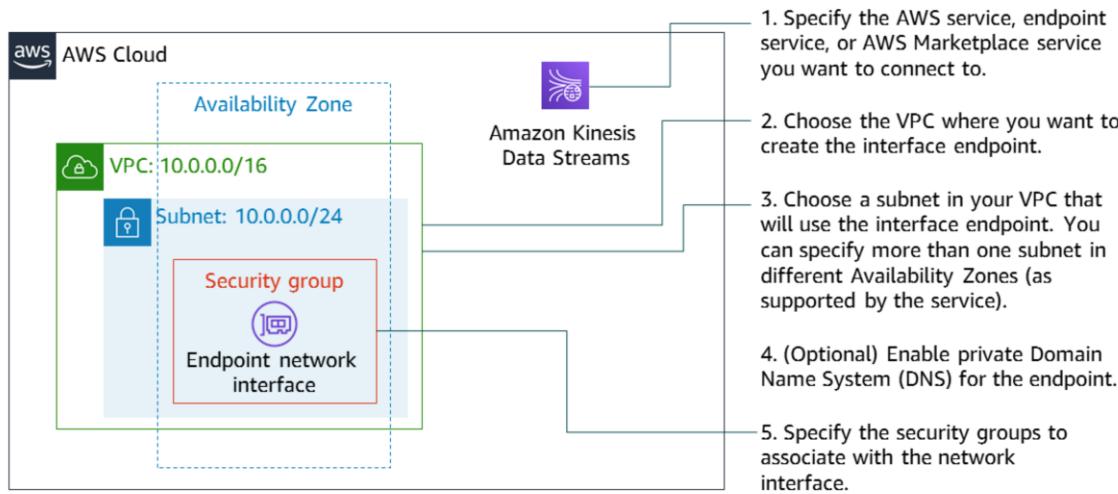
There are two types of VPC endpoints:

- An interface endpoint is an elastic network interface with a private IP address. This IP address serves as an entry point for traffic that is destined to a supported service. Interface endpoints enable you to connect to services that are powered by AWS PrivateLink. The owner of the service is the service provider. As the principal who creates the interface endpoint, you are the service consumer. For a complete list of services that are supported by interface endpoints, see [VPC Endpoints – Interface Endpoints](#).
- A gateway endpoint is a gateway that you specify as a target for a route in your route table. The route is for traffic that is destined to a supported AWS service. Amazon S3 and Amazon DynamoDB are supported by gateway endpoints.

There are no data processing or hourly charges for using gateway VPC endpoints. However, you will be billed for each hour that your VPC endpoint remains provisioned in each Availability Zone, regardless of the state of its association with the service. This hourly billing for your VPC endpoint will stop when you delete it. Hourly billing will also stop if the endpoint service owner rejects your VPC endpoint's attachment to their service. That service is subsequently deleted. For more information about interface endpoint pricing, see [AWS PrivateLink pricing](#).

To learn more about VPC endpoints, see [VPC Endpoints](#) in the AWS Documentation.

How to set up an interface endpoint



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

49

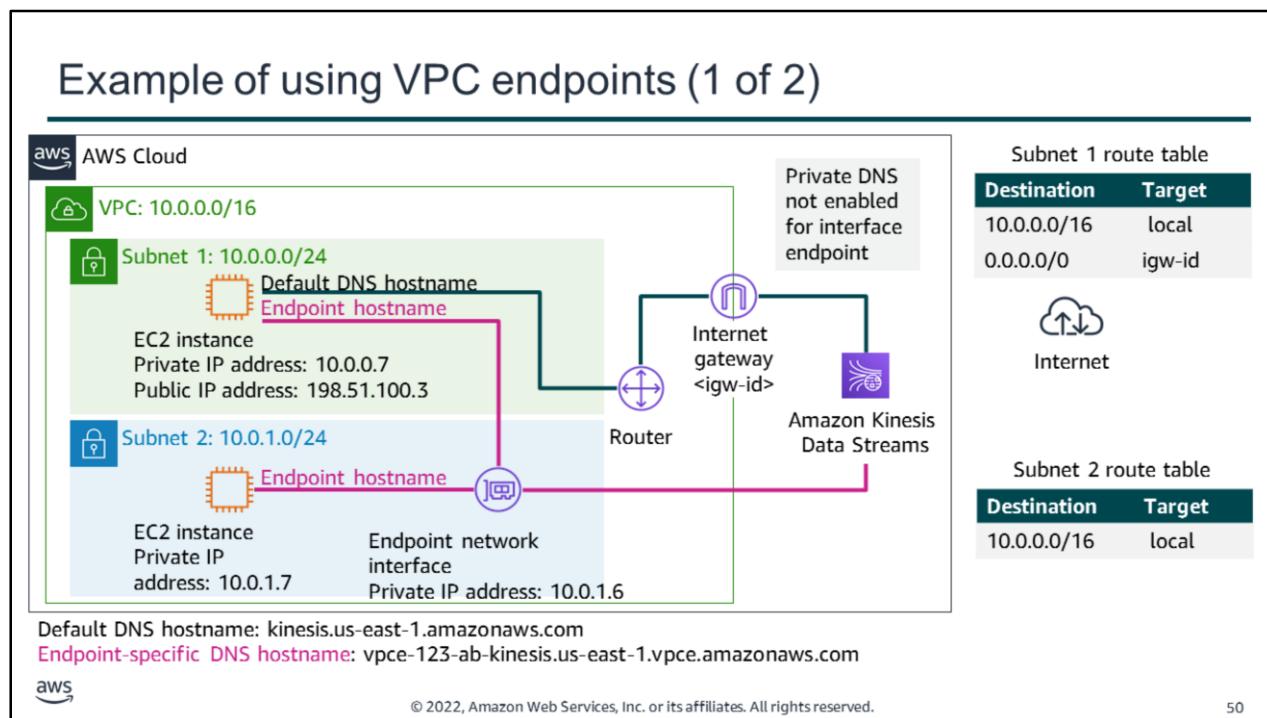
To set up an interface endpoint, follow these general steps from the Amazon VPC console:

- Specify the name of the AWS service, endpoint service, or AWS Marketplace service you want to connect to.
- Choose the VPC where you want to create the interface endpoint. You can specify more than one subnet in different Availability Zones (as supported by the service). Doing so helps ensure that your interface endpoint is resilient to Availability Zone failures. In that case, an endpoint network interface is created in each subnet that you specify.
- Choose a subnet in your VPC that will use the interface endpoint. When you create an interface endpoint for a service in your VPC, an endpoint network interface is created in the selected subnet. The endpoint network interface has a private IP address that serves as an entry point for traffic destined to the service.
- (Optional) Enable private Domain Name System (DNS) for the endpoint. Doing so enables you to make requests to the service by using its default DNS hostname (which is enabled by default for endpoints created for AWS services and AWS Marketplace Partner services).
- Specify the security groups to associate with the network interface. The security group rules control the traffic to the endpoint network interface from resources in your VPC. If you do not specify a security group, the default security group for the VPC is used.

Services cannot initiate requests to resources in your VPC through the endpoint. An endpoint only returns responses to traffic that are initiated from resources in your VPC.

For details about how to create interface endpoints, see:

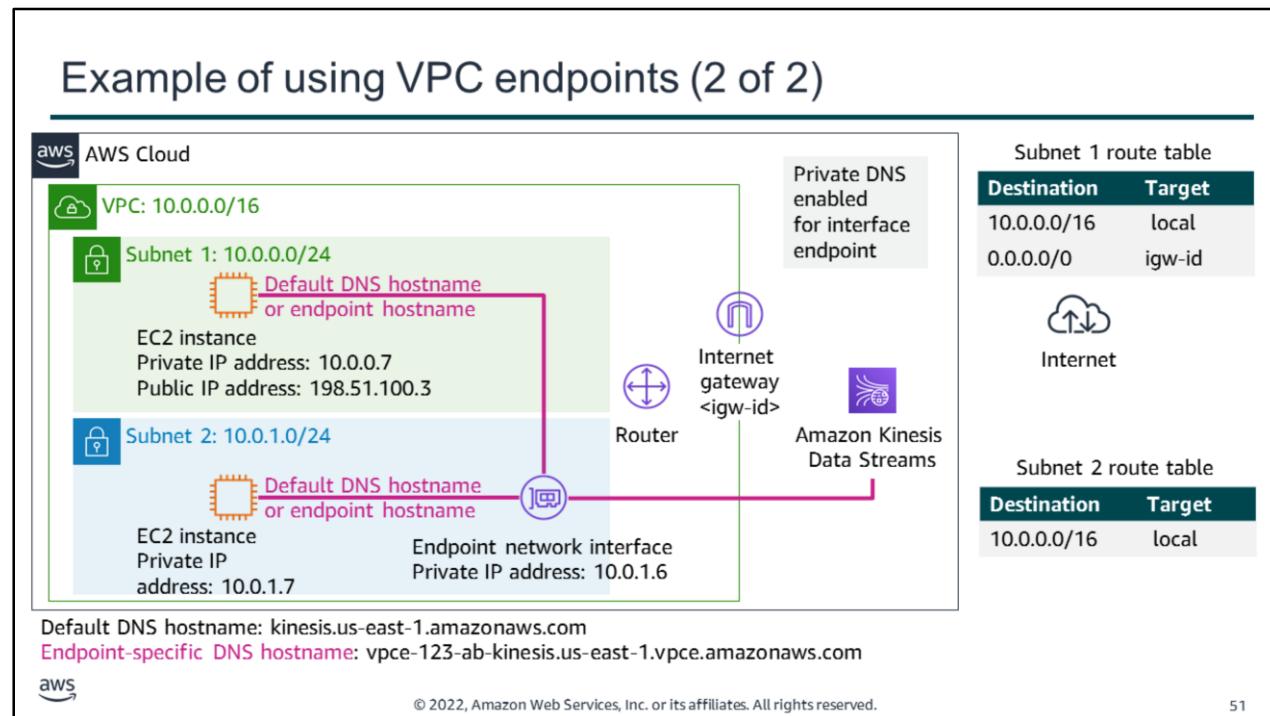
- [Creating an Interface Endpoint](#)
- [What is an Interface VPC Endpoint and How Can I Create Interface Endpoints for My VPC?](#)



50

When you create an interface endpoint, endpoint-specific DNS hostnames are generated that you can use to communicate with the service. For AWS services and AWS Marketplace Partner services, the private DNS option (enabled by default) associates a private hosted zone with your VPC. The hosted zone contains a record set for the default DNS name for the service (for example, `kinesis.us-east-1.amazonaws.com`) that resolves to the private IP addresses of the endpoint network interfaces in your VPC. This enables your application to make requests to the service using its default DNS hostname instead of the endpoint-specific DNS hostnames. This allows your existing applications to make requests to an AWS service through the interface endpoint without requiring any configuration changes.

In this example, there is an interface endpoint for Amazon Kinesis Data Streams and an endpoint network interface in subnet 2. Private DNS for the interface endpoint has not been enabled. Instances in either subnet can send requests to Amazon Kinesis Data Streams through the interface endpoint by using an endpoint-specific DNS hostname. Instances in subnet 1 can communicate with Amazon Kinesis Data Streams over the public IP address space in the AWS Region by using the service's default DNS hostname.



51

Here, private DNS for the endpoint has been enabled. Instances in either subnet can use either the default DNS hostname or the endpoint-specific DNS hostname to send requests to Amazon Kinesis Data Streams through the interface endpoint.

For more information about this example, see [Private DNS for interface endpoints](#).

Section 6 key takeaways



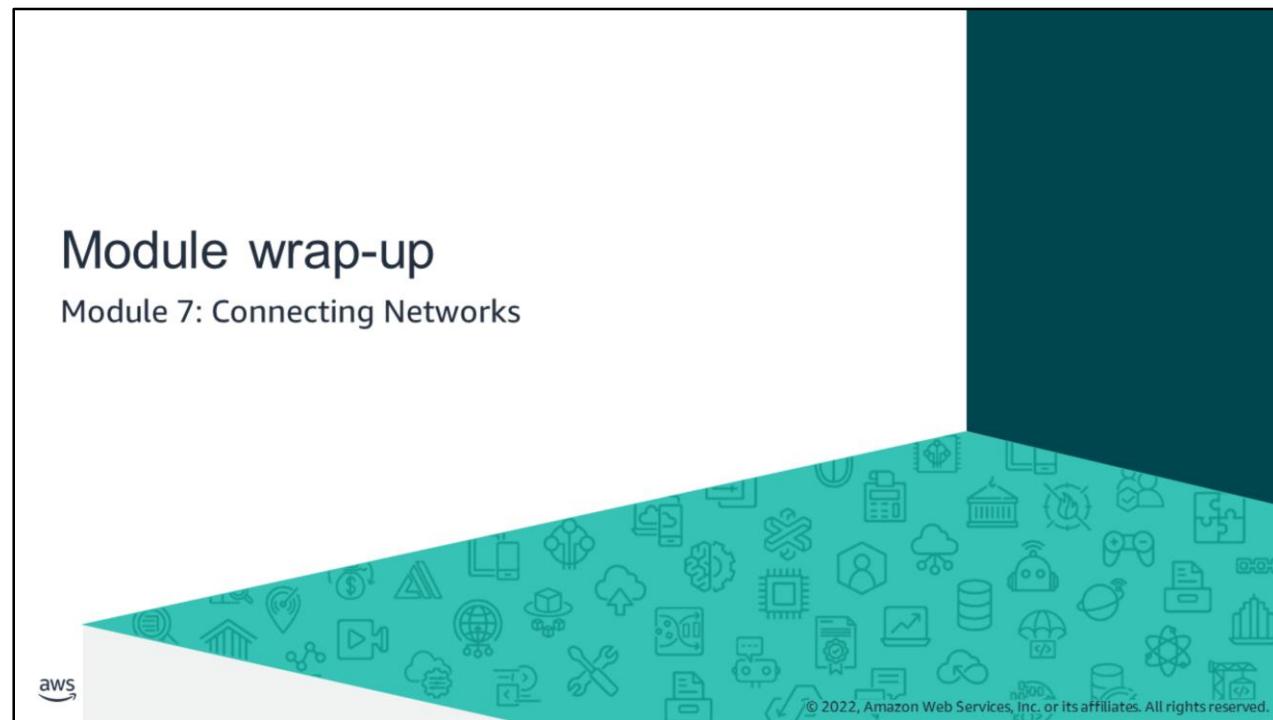
© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

52

- A **VPC endpoint** enables you to privately connect your VPC to supported AWS services and VPC endpoint services powered by AWS PrivateLink
- VPC endpoints **do not require** an internet gateway, NAT device, VPN connection, or AWS Direct Connect connection
- There are two types of VPC endpoints: **interface endpoints** and **gateway endpoints**

Some key takeaways from this section of the module include:

- A VPC endpoint enables you to privately connect your VPC to supported AWS services and VPC endpoint services powered by AWS PrivateLink
- VPC endpoints do not require an internet gateway, NAT device, VPN connection, or AWS Direct Connect connection
- There are two types of VPC endpoints: interface endpoints and gateway endpoints



It's now time to review the module and wrap up with a knowledge check and discussion of a practice certification exam question.

Module summary

In summary, in this module, you learned how to:

- Describe how to connect an on-premises network to the AWS Cloud
- Describe how to connect VPCs in the AWS Cloud
- Connect VPCs in the AWS Cloud by using VPC peering
- Describe how to scale VPCs in the AWS Cloud
- Describe how to connect VPCs to supported AWS services



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

54

In summary, in this module, you learned how to:

- Describe how to connect an on-premises network to the AWS Cloud
- Describe how to connect VPCs in the AWS Cloud
- Connect VPCs in the AWS Cloud using VPC peering
- Describe how to scale VPCs in the AWS Cloud
- Describe how to connect VPCs to supported AWS services

Complete the knowledge check



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

55

It is now time to complete the knowledge check for this module.



Sample exam question

An application running on Amazon Elastic Compute Cloud (Amazon EC2) instances processes sensitive information stored on Amazon Simple Storage Service (Amazon S3). The information is accessed over the internet. The security team is concerned that the internet connectivity to Amazon S3 is a security risk.

Which solution will resolve the security concern?

Choice	Response
A	Access the data through an internet gateway.
B	Access the data through a VPN connection.
C	Access the data through a NAT gateway.
D	Access the data through a VPC endpoint for Amazon S3.

© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

56

Look at the answer choices and rule them out based on the keywords.

Sample exam question answer



An application running on Amazon Elastic Compute Cloud (Amazon EC2) instances processes sensitive information stored on Amazon Simple Storage Service (Amazon S3). The information is accessed over the internet. The security team is concerned that the internet connectivity to Amazon S3 is a security risk.

Which solution will resolve the security concern?

The correct answer is D.

The keywords in the question are internet connectivity to Amazon S3 is a security risk.

© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

57

The following are the keywords to recognize: internet connectivity to Amazon S3 is a security risk.

The correct answer is D: “Access the data through a VPC endpoint for Amazon S3.”

Choice A (“Access the data through an internet gateway”) can be eliminated because making the data stored in Amazon S3 public would be a security risk. Choice B (“Access the data through a VPN connection”) can also be eliminated because you cannot connect to Amazon S3 by VPN. While choice C (“Access the data through a NAT gateway”) is not wrong, you can pick only one correct answer. Choice D is more appropriate because there is no additional cost or limit to performance.

Additional resources

- AWS re:Invent 2018 video: [AWS VPN Solutions](#)
- AWS Knowledge Center video: [How do I create a VPN with Amazon VPC?](#)
- [How do I configure a VPN over AWS Direct Connect?](#)
- AWS re:Invent 2019 video: [From one to many: Evolving Amazon VPC design](#)
- [Building a Scalable and Secure Multi-VPC AWS Network Infrastructure](#) whitepaper
- AWS Knowledge Center video: [What is AWS Peering?](#)
- AWS re:Invent 2019 video: [AWS Transit Gateway reference architectures for many VPCs](#)
- AWS Knowledge Center video: [What is an Interface VPC Endpoint and How Can I Create Interface Endpoints for my VPC?](#)



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

58

If you want to learn more about the topics covered in this module, you might find the following additional resources helpful:

- AWS re:Invent 2018 video: [AWS VPN Solutions](#)
- AWS Knowledge Center video: [How do I create a VPN with Amazon VPC?](#)
- [How do I configure a VPN over AWS Direct Connect?](#)
- AWS re:Invent 2019 video: [From one to many: Evolving Amazon VPC design](#)
- [Building a Scalable and Secure Multi-VPC AWS Network Infrastructure](#) whitepaper
- AWS Knowledge Center video: [What is AWS Peering?](#)
- AWS re:Invent 2019 video: [AWS Transit Gateway reference architectures for many VPCs](#)
- AWS Knowledge Center video: [What is an Interface VPC Endpoint and How Can I](#)



Thank you

Corrections, feedback, or other questions?
Contact us at <https://support.aws.amazon.com/#/contacts/aws-academy>.
All trademarks are the property of their owners.

© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

59

Thank you for completing this module.



training and
certification

AWS Academy Cloud Architecting
Module 08 Student Guide

Version 2.0.11

200-ACACAD-20-EN-SG

© 2023, Amazon Web Services, Inc. or its affiliates. All rights reserved.

This work may not be reproduced or redistributed, in whole or in part,
without prior written permission from Amazon Web Services, Inc.
Commercial copying, lending, or selling is prohibited.

All trademarks are the property of their owners.

Contents

Module 8: Securing User and Application Access

4



Module 8: Securing User and Application Access

AWS Academy Cloud Architecting

© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

Welcome to Module 8: Securing User and Application Access.

Module overview

Sections

1. Architectural need
2. Account users and IAM
3. Organizing users
4. Federating users
5. Multiple accounts

Demonstration

- EC2 Instance Profile

Activity

- **Examining IAM policies**

Lab

- Challenge Lab: Controlling AWS Account Access by Using IAM



Knowledge check



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

2

This module contains the following sections:

1. Architectural need
2. Account users and IAM
3. Organizing users
4. Federating users
5. Multiple accounts

This module also includes:

- A demonstration that will show you a commonly used feature. An IAM role that grants access to other services from Amazon Web Services (AWS) is attached to an Amazon Elastic Compute Cloud (Amazon EC2) instance
- An activity that challenges you to analyze AWS Identity and Access Management (IAM) policy documents to determine which actions the policies allow or deny
- A challenge lab where you use IAM to configure users, groups, and access policies that are appropriate for the café use case

Finally, you will be asked to complete a knowledge check that will test your understanding of key concepts covered in this module.

Module objectives

At the end of this module, you should be able to:

- Explain the purpose of AWS Identity and Access Management (IAM) users, groups, and roles
- Describe how to allow user federation within an architecture to increase security
- Recognize how AWS Organizations service control policies (SCPs) increase security within an architecture
- Describe how to manage multiple AWS accounts
- Configure IAM users



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

3

At the end of this module, you should be able to:

- Explain the purpose of AWS Identity and Access Management (IAM) users, groups, and roles
- Describe how to allow user federation within an architecture to increase security
- Recognize how AWS Organizations service control policies (SCPs) increase security within an architecture
- Describe how to manage multiple AWS accounts
- Configure IAM users

Section 1: Architectural need

Module 8: Securing User and Application Access



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

Introducing Section 1: Architectural need

Café business requirement

The café needs to define what level of access users and systems should have across cloud resources and then put these access controls into place across the AWS account.



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

5

The café must define what level of access users and systems should have across their cloud resources. They must then put these access controls into place across their AWS account.

The café is large enough now that team members who build, maintain, or access applications on AWS are specializing into roles (such as developer or database administrator). Up until now, they haven't made an effort to clearly define what level of access each user should have based on their roles and responsibilities.

Throughout this module, you will learn about IAM, which provides the features that you need to meet these new business requirements.

Section 2: Account users and IAM

Module 8: Securing User and Application Access

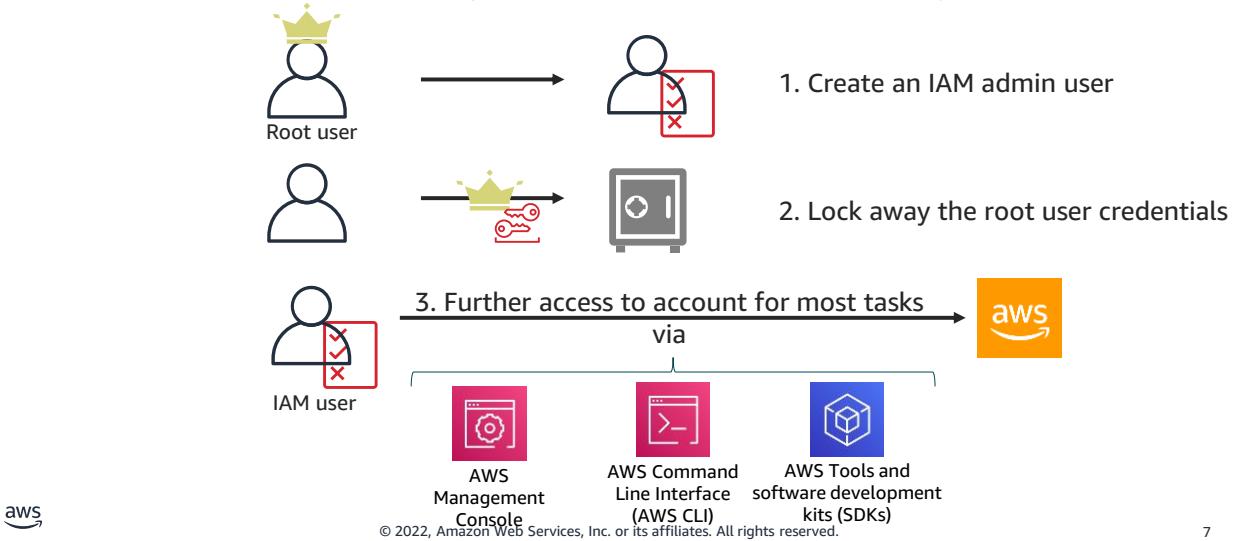


© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

Introducing Section 2: Account users and IAM.

Secure the root account

The account root user has a large amount of power. Recommended security steps:



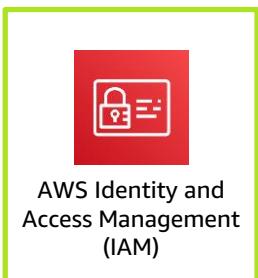
7

When you first create an AWS account, you begin with a *root user*. This user can log in to the AWS Management Console with the email address that was used to create the account.

The AWS account root user has full access to all resources in the account, including billing information, personal data in the user profile, and all resources that were created in any AWS services in the account. You cannot control the privileges of the AWS account root user credentials.

AWS strongly recommends that you not use root user credentials for day-to-day interactions with AWS. Instead, create one or more IAM users. Keep the root user credentials in a secure location. For most ongoing account access and management tasks, you can use IAM user credentials.

AWS Identity and Access Management (IAM)



- Securely control individual and group access to your AWS resources
- Integrates with other AWS services
- Federated identity management
- Granular permissions
- Support for multi-factor authentication



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

8

AWS Identity and Access Management is also known as IAM. It is a service that allows you configure fine-grained access control to AWS resources. IAM enables security best practices by allowing you to grant unique security credentials to users and groups. These credentials specify which AWS service application programming interfaces (APIs) and resources they can access. IAM is secure by default. Users have no access to AWS resources until permissions are explicitly granted.

IAM is integrated into most AWS services. You can define access controls from one place in the AWS Management Console, and they will take effect throughout your AWS environment.

You can use IAM to grant your employees and applications access to the AWS Management Console and to AWS service APIs by using your existing identity systems. AWS supports federation from corporate systems like Microsoft Active Directory and standards-based identity providers. IAM also supports multi-factor authentication (MFA). If MFA is enabled and an IAM user attempts to log in, they will be prompted for an authentication code. The authentication code is delivered to an AWS MFA device. The MFA device can be a hardware MFA device. It can also be a virtual MFA device that the user accesses through an application that runs on the user's smartphone, such as Google Authenticator.

You can create accounts that have privileges similar to the AWS account root user. However, it is better to create administrative accounts that grant only the account permissions that are needed.

Follow the principle of least privilege. For example, ask yourself if your database administrator (DBA) should be able to provision EC2 instances. If the answer is no, then provision accounts accordingly.

IAM components: Review



IAM user

Defined in your AWS account. Use credentials to authenticate programmatically or via the AWS Management Console.



IAM group

A collection of IAM users that are granted identical authorization.



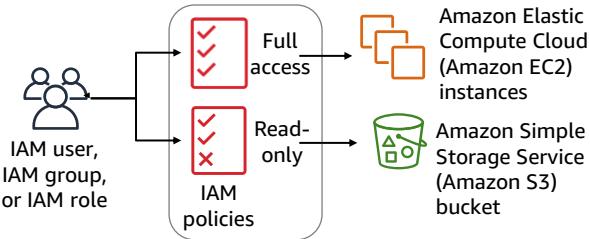
IAM policy

Defines which resources can be accessed and the level of access to each resource.



IAM role

Mechanism to grant temporary access for making AWS service requests. *Assumable* by a person, application, or service.



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

9

To understand how to use IAM to secure your AWS account, it is important to understand the role and function of each of the four IAM components.

An *IAM user* is a person or application that is defined in an AWS account, and that must make API calls to AWS products. Each user must have a unique name (with no spaces in the name) within the AWS account, and a set of security credentials that is not shared with other users. These credentials are different from the AWS account root user security credentials. Each user is defined in one and only one AWS account.

An *IAM group* is a collection of IAM users. You can use IAM groups to simplify how you specify and manage permissions for multiple users.

An *IAM policy* is a document that defines permissions to determine what users can and cannot do in the AWS account.

An *IAM role* is a tool for granting temporary access to specific AWS resources in an AWS account.

IAM permissions

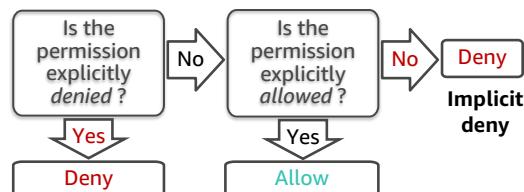


IAM policy

Permissions are specified in an [IAM policy](#):

- A document formatted in JavaScript Object Notation (JSON)
- It defines which resources and operations are allowed
- Best practice – follow the [principle of least privilege](#)
- Two types of policies –
 - [Identity-based](#): Attach to an IAM principal
 - [Resource-based](#): Attach to an AWS resource

How IAM determines permissions at the time of request:



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

10

In IAM, permissions are defined in IAM policy documents. Policies enable you to fine-tune privileges that are granted to principals. Example principals are IAM users, IAM roles, or other AWS services.

When IAM determines whether a permission is allowed, IAM first checks for the existence of any applicable *explicit denial policy*. If no explicit denial exists, it then checks for any applicable *explicit allow policy*. If an explicit deny or an explicit allow policy does not exist, IAM reverts to the default and denies access. This process is referred to as an *implicit deny*. The user will be permitted to take the action only if the requested action is *not* explicitly denied and *is* explicitly allowed.

When you develop IAM policies, it can be difficult to determine whether access to a resource will be granted to an IAM entity. The [IAM Policy Simulator](#) is a useful tool for testing and troubleshooting IAM policies.

Policies are stored as JavaScript Object Notation (JSON) documents. They are attached to principals as *identity-based policies*, or to resources as *resource-based policies*.

Identity-based versus resource-based policies



Identity-based policies

- Attached to a user, group, or role
- Types of policies
 - AWS managed
 - Customer managed
 - Inline

Resource-based policies

- Attached to AWS resources
 - Example: Attach to an Amazon S3 bucket
- Always an inline policy



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

11

Identity-based policies are permission policies that you can attach to a principal (or identity), such as an IAM user, role, or group. These policies *control what actions that identity can perform, on which resources, and under what conditions*.

Identity-based policies can be further categorized as AWS managed, customer managed, or inline. *AWS managed policies* are created and managed by AWS, and you can attach them to multiple users, groups, and roles in your AWS account. If you are new to using policies, we recommend that you start by using AWS managed policies. *Customer managed policies* are policies that you create and manage in your AWS account. Customer managed policies provide more precise control over your policies than AWS managed policies. You can create and edit an IAM policy in the visual editor or by creating the JSON policy document directly. *Inline policies* are policies that you create and manage, and that are embedded directly into a single user, group, or role.

Resource-based policies are JSON policy documents that you attach to a resource, such as an Amazon Simple Storage Service (Amazon S3) bucket. These policies control *which actions a specified principal can perform on that resource and under what conditions*. Resource-based policies are inline policies, and there are no managed resource-based policies.

IAM policy document structure

```
{  
    "version": "2012-10-17",  
    "Statement": [{  
        "Effect": "effect",  
        "Action": "action",  
        "Resource": "arn",  
        "Condition": {  
            "condition": {  
                "key": "value"  
            }  
        }  
    }]  
}
```

- **Effect:** Effect can be either *Allow* or *Deny*
- **Action:** Type of access that is allowed or denied
 - "Action": "s3:GetObject"
- **Resource:** Resources that the action will act on
 - "Resource": "arn:aws:sqs:us-west-2:123456789012:queue1"
- **Condition:** Conditions that must be met for the rule to apply
 - "Condition" : {
 "StringEquals" : {
 "aws:username" : "johndoe"
 }
}



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

12

IAM policies are stored in AWS as JSON documents. Identity-based policies are policy documents that you attach to a user or role. Resource-based policies are policy documents that you attach to a resource. A policy document includes one or more individual statements. Each statement includes information about a single permission. If a policy includes multiple statements, AWS applies a logical OR across the statements when it evaluates them.

The following are common elements found in an IAM policy document:

- **Version** – Specify the version of the policy language that you want to use. As a best practice, use the latest 2012-10-17 version.
- **Statement** – Use this main policy element as a container for the following elements. You can include more than one statement in a policy.
- **Effect** – Use Allow or Deny to indicate whether the policy allows or denies access.
- **Principal** – If you create a resource-based policy, you must indicate the account, user, role, or federated user that you would like to allow or deny access to. If you are creating an IAM permissions policy to attach to a user or role, you cannot include this element. The principal is implied as that user or role.
- **Action** – Include a list of actions that the policy allows or denies.
- **Resource** – If you create an IAM permissions policy, you must specify a list of resources to which the actions apply. If you create a resource-based policy, this element is optional.
- **Condition (Optional)** – Specify the circumstances where the policy grants permissions.

ARNs and wildcards

- Resources are identified by using Amazon Resource Name ([ARN](#)) format
 - Syntax – `arn:partition:service:region:account:resource`
 - Example – "Resource": "arn:aws:iam::123456789012:user/mmajor"
- You can use a [wildcard](#) (*) to give access to all actions for a specific AWS service
 - Examples –
 - "Action": "s3:*
 - "Action": "iam:*AccessKey*"



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

13

For identity-based (IAM permissions) policies, you must specify a list of resources that the actions apply to. The *Resource* element specifies the object or objects that the statement covers. Statements must include either a *Resource* or a *NotResource* element.

Most resources have a friendly name (for example, a user named *Bob* or a group named *Developers*). However, the permissions policy language requires you to specify the resource or resources using the following *Amazon Resource Name (ARN)* format.

Each service has its own set of resources. Although you always use an ARN to specify a resource, the details of the ARN for a resource depend on the service and the resource. For information about how to specify a resource, refer to the documentation for the service whose resources you are writing a statement for.

You can also use wildcards in IAM policy documents, such as in ARNs or in Actions. You can use the wildcard character (*). An asterisk (*) represents any combination of zero or more characters. For example, an "Action" value of "s3:*" applies to all S3 actions. You can also use wildcards (*) as part of the action name. For example, the "Action" value of "iam:*AccessKey*" applies to all IAM actions that include the string *AccessKey*, including *CreateAccessKey*, *DeleteAccessKey*, *ListAccessKeys*, and *UpdateAccessKey*.

IAM policy example

```
{
  "version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": ["DynamoDB:*", "s3:*"],
      "Resource": [
        "arn:aws:dynamodb:region:account-number-without-hyphens:table/table-name",
        "arn:aws:s3:::bucket-name",
        "arn:aws:s3:::bucket-name/*"
      ],
      "NotResource": [
        "arn:aws:dynamodb:region:account-number-without-hyphens:table/table-name",
        "arn:aws:s3:::bucket-name",
        "arn:aws:s3:::bucket-name/*"
      ]
    }
  ]
}
```

Explicit allow gives users access to a specific DynamoDB table and...

...Amazon S3 buckets.

Explicit deny ensures that the users cannot use any other AWS actions or resources other than that table and those buckets.

An explicit deny statement takes precedence over an allow statement.

© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

14

As mentioned previously, IAM policy documents are written in JSON.

This example IAM policy grants user access only to the following resources:

- The Amazon DynamoDB table whose name is represented by *table-name*.
- The AWS account's S3 bucket, whose name is represented by *bucket-name* and all the objects that it contains.

The IAM policy also includes an explicit deny ("Effect":"Deny") element. The *NotResource* element helps to ensure that users cannot use any other DynamoDB or S3 actions or resources, except the actions and resources that are specified in the policy. This is the case even if permissions have been granted in another policy. An explicit deny statement takes precedence over an allow statement.

Activity: Examining IAM policies



Photo by Pixabay from Pexels.



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

15

In this educator-led activity, you will be presented with example IAM policies. For each policy, you will be asked questions about whether the policy allows or denies particular actions. The educator will lead you in a discussion of each question and reveal the correct answers one at a time.

Activity: IAM policy analysis (1 of 3)

Consider this IAM policy, then answer the questions.

```
{  
    "version": "2012-10-17",  
    "Statement": [  
        {"Effect": "Allow",  
         "Action": [  
             "iam:Get*",  
             "iam>List*"  
         ],  
         "Resource": "*"  
     }  
}
```

1. Which AWS service does this policy grant you access to?
2. Does it allow you to create an IAM user, group, policy, or role?
3. Go to <https://docs.aws.amazon.com/IAM/latest/UserGuide/> and in the left navigation expand *Reference > Policy Reference > Actions, Resources, and Condition Keys*. Choose *Identity And Access Management*. Scroll to the Actions Defined by Identity And Access Management list.

Name at least three specific actions that the iam:Get* action allows.



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

16

Look at the example IAM policy document. The educator will now ask you a series of questions to assess whether you understand what actions this policy will allow and deny.

Activity: IAM policy analysis (1 of 3) - Answers

Consider this IAM policy, then answer the questions.

```
{  
    "version": "2012-10-17",  
    "Statement": [  
        {  
            "Effect": "Allow",  
            "Action": [  
                "iam:Get*",  
                "iam>List*"  
            ],  
            "Resource": "*"  
        }  
    ]  
}
```

1. Which AWS service does this policy grant you access to?
 - ANSWER: The IAM service.
2. Does it allow you to create an IAM user, group, policy, or role?
 - ANSWER: No. The access is limited to *get* and *list* requests. It effectively grants read-only permissions.
3. Go to <https://docs.aws.amazon.com/IAM/latest/UserGuide/> and in the left navigation expand *Reference > Policy Reference > Actions, Resources, and Condition Keys*. Choose *Identity And Access Management*. Scroll to the Actions Defined by Identity And Access Management list.
Name at least three specific actions that the iam:Get* action allows.
 - ANSWER: iam:Get* allows many specific actions, including GetGroup, GetPolicy, GetRole, and others.



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

17

The answers are revealed.

Activity: IAM policy analysis (2 of 3)

Consider this IAM policy, then answer the questions.

```
{  
    "Version": "2012-10-17",  
    "Statement": [  
        {  
            "Effect": "Allow",  
            "Action": ["ec2:TerminateInstances"],  
            "Resource": ["*"]  
        },  
        {  
            "Effect": "Deny",  
            "Action": ["ec2:TerminateInstances"],  
            "Condition": {  
                "NotIpAddress": {  
                    "aws:SourceIp": [  
                        "192.0.2.0/24",  
                        "203.0.113.0/24"  
                    ]  
                }  
            },  
            "Resource": ["*"]  
        }  
    ]  
}
```



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

1. Does the policy allow you to terminate any EC2 instance at any time without conditions?
2. Are you allowed to make the terminate instance call from anywhere?
3. Can you terminate instances if you make the call from a server that has an assigned IP address of 192.0.2.243?

18

Analyze the second IAM policy file example. The first part shows Effect: Allow and Action ec2:TerminateInstance for resource. The second part shows effect Deny for action ec2:TerminateInstances with condition NotIpAddress aws:SourceIp 192.0.2.0/24 and 203.0.113.0/24 for resource. The educator will again ask you a series of questions to assess whether you understand what actions this policy will allow and deny.

Activity: IAM policy analysis (2 of 3) - Answers

Consider this IAM policy, then answer the questions as they are presented.

```
{  
    "Version": "2012-10-17",  
    "Statement": [  
        {  
            "Effect": "Allow",  
            "Action": ["ec2:TerminateInstances"],  
            "Resource": ["*"]  
        },  
        {  
            "Effect": "Deny",  
            "Action": ["ec2:TerminateInstances"],  
            "Condition": {  
                "NotIpAddress": {  
                    "aws:SourceIp": [  
                        "192.0.2.0/24",  
                        "203.0.113.0/24"  
                    ]  
                }  
            },  
            "Resource": ["*"]  
        }  
    ]  
}
```



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

1. Does the policy allow you to terminate any EC2 instance at any time without conditions?
 - ANSWER: No. The first statement object allows it. However, the second statement object applies a condition.
2. Are you allowed to make the terminate instance call from anywhere?
 - ANSWER: No. You can only make the request from one of the two IP address ranges that are specified in aws:SourceIp.
3. Can you terminate instances if you make the call from a server that has an assigned IP address of 192.0.2.243?
 - ANSWER: Yes, because the 192.0.2.0/24 Classless Inter-Domain Routing (CIDR) IP address range includes IP addresses 192.0.2.0 through 192.0.2.255. A resource like the [CIDR to IP Range tool](#) can be used to calculate the range of a CIDR block.

19

The answers are revealed.

For accessibility: Example policy document in JSON format. Shows a statement section with two parts. Part one shows Effect:Allow and Action EC2:TerminateInstance for resource *. Part 2 shows effect Deny for action EC2:TerminateInstances with condition NotIpAddress aws:SourceIp 192.0.2.0/24 and 203.0.113.0/24 for resource *. **End of accessibility description.**

Activity: IAM Policy analysis (3 of 3)

Consider this IAM policy, then answer the questions.

```
{  
    "version": "2012-10-17",  
    "Statement": [{  
        "Condition": {  
            "StringNotEquals": {  
                "ec2:InstanceType": [  
                    "t2.micro",  
                    "t2.small"  
                ]  
            }  
        },  
        "Resource": "arn:aws:ec2:*:*:instance/*",  
        "Action": [  
            "ec2:RunInstances",  
            "ec2:StartInstances"  
        ],  
        "Effect": "Deny"  
    }]  
}
```



1. What actions does the policy allow?
2. Say that the policy included an additional statement object, like this example:

```
{  
    "Effect": "Allow",  
    "Action": "ec2:*",  
    "Resource": "*"
```
3. If the policy included both the statement on the left and the statement in question 2, could you terminate an m3.xlarge instance that existed in the account?

How would the policy restrict the access granted to you by this additional statement?

© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

20

Observe the third and last IAM policy document example. The educator will again ask you a series of questions to assess whether you understand what actions this policy will allow and deny.

Activity: IAM Policy analysis (3 of 3)

Consider this IAM policy, then answer the questions.

```
{  
    "version": "2012-10-17",  
    "Statement": [{  
        "Condition": {  
            "StringNotEquals": {  
                "ec2:InstanceType": [  
                    "t2.micro",  
                    "t2.small"  
                ]  
            }  
        },  
        "Resource": "arn:aws:ec2:*:*:instance/*",  
        "Action": [  
            "ec2:RunInstances",  
            "ec2:StartInstances"  
        ],  
        "Effect": "Deny"  
    }]  
}
```



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

1. What actions does the policy allow?
 - ANSWER: It does not allow you to do anything (the effect is to Deny).
2. Say that the policy included an additional statement object, like this example:

```
{  
    "Effect": "Allow",  
    "Action": "ec2:*",  
    "Resource": "*"  
}
```

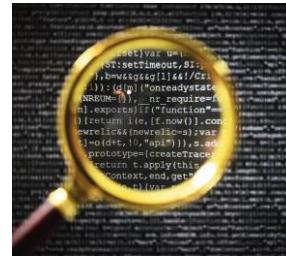
How would the policy restrict the access granted to you by this additional statement?
 - ANSWER: You would have full Amazon EC2 service access. However you would only be allowed to launch or start EC2 instances of instance type t2.micro or t2.small.
3. If the policy included both the statement on the left and the statement in question 2, could you terminate an m3.xlarge instance that existed in the account?
 - ANSWER: Yes.

21

The answers are revealed.

AWS CloudTrail

- Logs and monitors user activity
- Provides event history of AWS account
 - Actions taken through the AWS Management Console, SDKs, AWS CLI
 - Increases visibility into your user and resource activity
 - 90-day event history provided by default, at no cost
- Identify
 - Who accessed your account
 - When and from where
 - What action they took on an AWS service
- Helpful tool to
 - Perform security analysis
 - Discover which calls were blocked (for example, by IAM policies)



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

22

AWS CloudTrail is a service that enables governance, compliance, and auditing of your AWS account. With CloudTrail, you can continuously monitor and retain account activity that is related to actions across your AWS infrastructure. It provides an event history of account activity, including actions taken through the AWS Management Console, AWS SDKs, and command line tools. This event history simplifies security analysis, resource change tracking, and troubleshooting.

You can discover and troubleshoot security and operational issues by capturing a comprehensive history of changes that occurred in your AWS account within a specified period of time. You can identify which users and accounts called AWS, the source IP address that the calls were made from, and when the calls occurred. CloudTrail enables you to track and automatically respond to account activity that threatens the security of your AWS resources.

With Amazon EventBridge (formerly known as Amazon CloudWatch Events) integration, you can define workflows that run when it detects events that can result in security vulnerabilities. For example, you can create a workflow to add a specific policy to an S3 bucket when CloudTrail logs an API call that makes that bucket public.

CloudTrail records important information about each action, including who made the request, the services used, the actions performed, parameters for the actions, and the response elements that were returned by the AWS service. The service also helps organizations meet the compliance

and auditing requirements that they must adhere to.

Section 2 key takeaways



- Avoid using the **account root user** for common tasks. Instead, create and use IAM user credentials.
- **Permissions** for accessing AWS account resources are defined in one or more IAM policy documents.
 - Attach IAM policies to IAM users, groups, or roles.
- When IAM determines permissions, an explicit **Deny** will always override any **Allow** statement.
- It is a best practice to follow the **principle of least privilege** when you grant access.

© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

23

Some key takeaways from this section of the module include:

- Avoid using the account root user for common tasks. Instead, create and use IAM user credentials.
- Permissions for accessing AWS account resources are defined in one or more IAM policy documents.
 - Attach IAM policies to IAM users, groups, or roles.
- When IAM determines permissions, an explicit Deny will always override any Allow statement.
- It is a best practice to follow the principle of least privilege when you grant access.

Section 3: Organizing users

Module 8: Securing User and Application Access



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

Introducing Section 3: Organizing users.

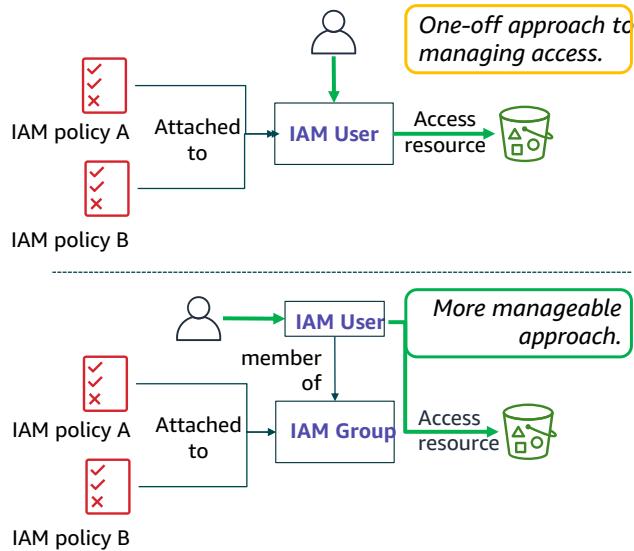
IAM groups

Use IAM groups to grant the same access rights to multiple users.

- All users in the group inherit the permissions assigned to the group
- Makes it easier to manage access across multiple users

Tip: Combine approaches for fine-grained individual access

- Add the user to a group to apply standard access based on job function
- Optionally attach an additional policy to the user for needed exceptions



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

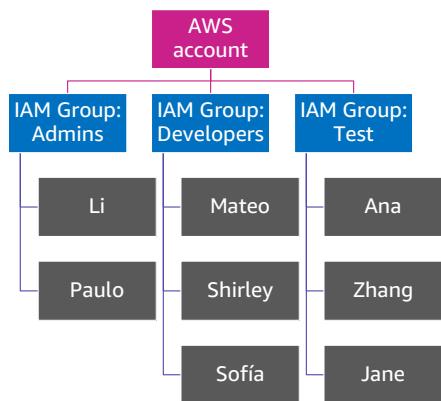
25

An IAM group is a collection of IAM users. Groups are a convenience that makes it easier to manage permissions for a collection of users, instead of managing permissions for each individual user.

Manage group membership as a simple list:

- Add users to a group or remove them from a group.
- A user can belong to multiple groups.
- Groups cannot belong to other groups.
- Groups can be granted permissions by using access control policies.
- Groups do not have security credentials and cannot access web services directly. They exist solely to make it easier to manage user permissions.

Example IAM groups



Tip: Create groups that reflect job functions

- If a new developer is hired, add them to the *Developer* group
 - Immediately inherit the same access granted to other developers
- If Ana takes on the new role of developer –
 - Remove her from the *Test* group
 - Add her to the *Developer* group
- Users can belong to more than one group
 - However the most restrictive policy will then apply



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

26

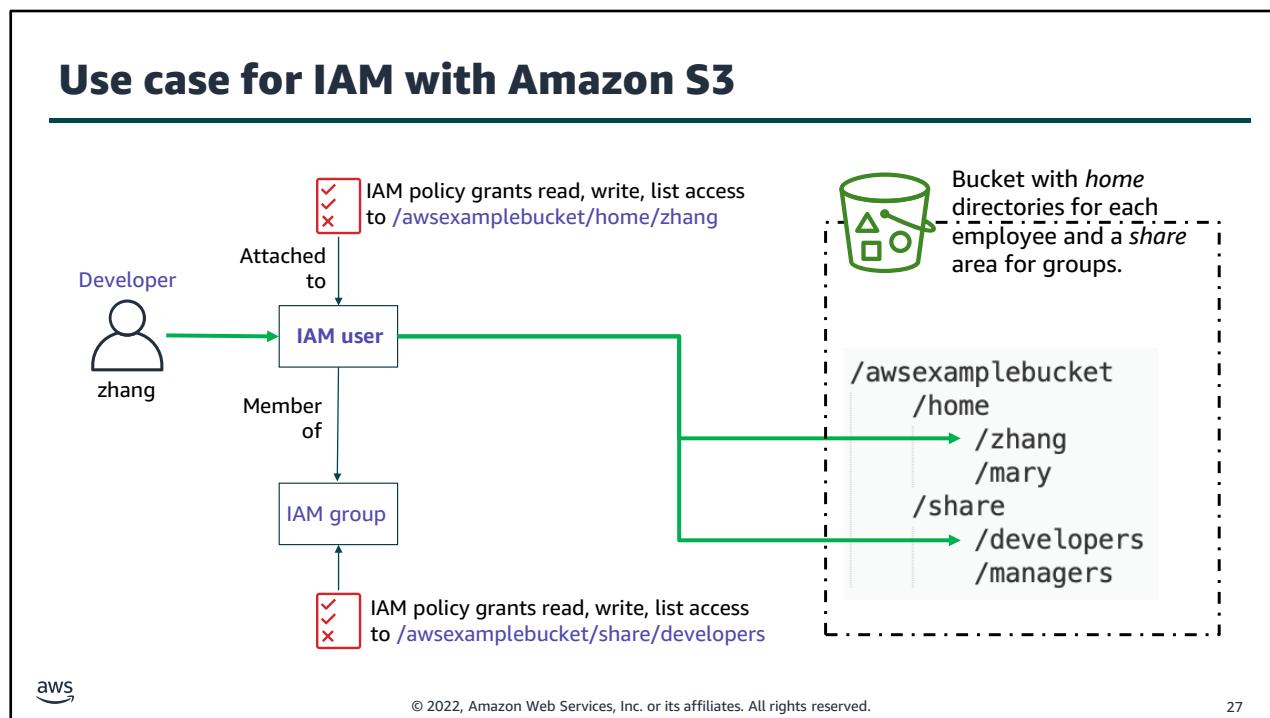
Typically, you will want to create groups that reflect job functions. For example, you could create one group for administrators, another group for developers, and yet another group for the team that performs testing functions.

Then, you attach one or more policy files to each group and add users to the groups. Users have the access rights that are assigned to the group or groups that they are in, because of their group membership.

If a new developer is hired, you can add them to the existing developer group. They will get the same access that the other developers already have.

If a person, such as Ana (shown in the example) takes on a new role in the organization, you can remove her from the *Test* group and add her to the *Developers* group. Or, if Ana will perform both functions, you can leave her in the *Test* group and add her to the *Developers* group.

If you discover that developers need access to some additional resource in the account, you can update or add a policy to the *Developers* group. All members of the group will gain that additional level of access. Groups make it easier to maintain consistent access rights across teams.



This example demonstrates how IAM permissions might be configured on an S3 bucket.

The `awsexamplebucket` has two main directories. The `home` directory has subdirectories for each user, where they can store individual work. The `share` directory has subdirectories where different teams can store content.

If a new team member, `zhang`, joins the organization as a developer, you can take three actions to grant them the proper access.

First, add `zhang` to the IAM group for developers. Notice that this group has an IAM policy attached to it that grants access to `/awsexamp1ebucket/share/develop1ers`.

Next, create the `/awsexamp1ebucket/home/zhang` directory in Amazon S3.

Finally, attach an IAM policy that grants access to the `/awsexamp1ebucket/home/zhang` directory directly to the `zhang` IAM user. `Zhang's` access will include both the rights that were granted from the group and also the rights that were directly attached to the IAM user principal.

Section 4: Federating users

Module 8: Securing User and Application Access

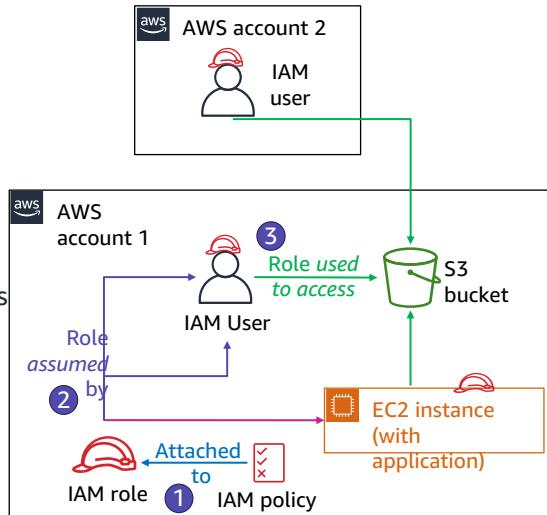


© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

Introducing Section 4: Federating users.

IAM roles

- **IAM role** characteristics
 - Provides *temporary* security credentials
 - Is not uniquely associated with one person
 - Is *assumable* by a **person, application, or service**
 - Is often used to delegate access
- Use cases
 - Provide AWS resources with access to AWS services
 - Provide access to externally authenticated users
 - Provide access to third parties
 - Switch roles to access resources in –
 - Your AWS account
 - Any other AWS account (cross-account access)



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

29

An IAM role enables you to define a set of permissions to access the resources that a user or service needs. However, the permissions are not attached to an IAM user or group. Instead, the permissions are attached to a role, and the role is assumed by the user or the service.

When a user assumes a role, the user's prior permissions are temporarily forgotten. AWS returns temporary security credentials that the user or application can then use to make programmatic requests to AWS.

By using IAM roles, you don't need to share long-term security credentials for each entity that requires access to a resource, such as creating an IAM user.

For a service like Amazon EC2, applications or AWS services can programmatically assume a role at runtime.

The principal that assumes the role could also be an IAM user, group, or role from another AWS account, including accounts that are not owned by you.

By creating a role for external account access, you don't need to manage user names and passwords for third parties. If you no longer want someone or some system to have access, you can modify or delete the role. Thus, you don't need to create and manage accounts for people outside of your organization.

Demonstration: EC2 Instance Profile



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

30

Now, the educator might choose to demonstrate how to attach an IAM role to an EC2 instance. This role grants AWS resource access to an application.

Grant permissions to assume a role

- For an IAM user, application, or service to assume a role, you must grant permissions to switch to the role



- AWS Security Token Service (AWS STS)
 - Web service that enables you to request temporary, limited-privilege credentials
 - Credentials can be used by IAM users or for users that you authenticate (federated users)
- Example policy – Allows an IAM user to assume a role

```
{  
    "Version": "2012-10-17",  
    "Statement": {  
        "Effect": "Allow",  
        "Action": "sts:AssumeRole",  
        "Resource": "arn:aws:iam::123456789012:role/Test*"  
    }  
}
```



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

32

AWS Security Token Service is also known as *AWS STS*. It is a web service that enables an IAM user, federated user, or application to assume an IAM role that they want.

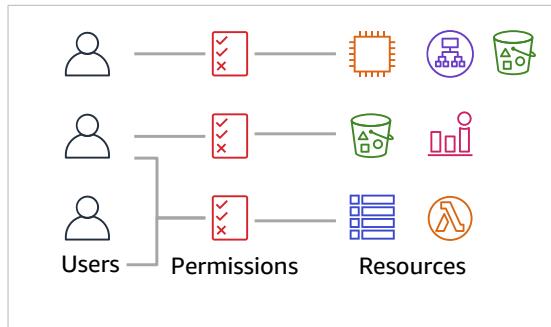
When the AssumeRole operation of the AWS STS API is successfully invoked, the web service returns the temporary, limited-privilege credentials that were requested by the IAM user or the user that was authenticated through federation. Typically, the AssumeRole operation is used for cross-account access or for federation.

The example policy allows an IAM user to assume any role that is defined in AWS account number 123456789012, as long as the role name starts with *Test*.

Role-based access control (RBAC)

Traditional approach to access control:

- Grant users specific permissions based on job function (such as database administrator)
- Create a distinct IAM role for each permission combination
- Update permissions by adding access for each new resource (it can become time-consuming to keep updating policies)



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

33

You will now consider two different approaches to access control: role-based access control (RBAC) and attribute-based access control (ABAC). You will first learn about RBAC.

RBAC has been used historically on-premises and in the cloud. With this model, you grant users explicit access to a set of permissions. Say that you have database administrators, network administrators, and developers. If you have one or more network administrators who are also developers, you would not create a new policy to grant those permissions. Instead, you add those users to both roles.

This approach is familiar and has many advantages. However, the person who maintains the permissions in this model might find that they must constantly update the permissions files to add access to certain roles each time a new resource is created. For example, they must update a policy with an ARN each time someone creates a new resource and wants to allow users access to it.

Best practice: Tagging

- A tag consists of a name and (optionally) a value
 - Can be applied to [resources](#) across your AWS accounts
 - Tag keys and values are returned by many different API operations
- Define *custom tags*
- Multiple practical uses
 - Billing, filtered views, access control, etc.
- Example tags applied to an EC2 instance:
 - Name = web server
 - Project = unicorn
 - Stack = dev

The screenshot shows the 'Add user' interface in the AWS IAM console. Under the 'Add tags (optional)' section, there are two entries: 'CostCenter' with value '1234' and 'EmailID' with value 'john@example.com'. At the bottom right, there are buttons for 'Cancel', 'Previous', and 'Next: Review'.



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

34

Before you consider the second approach to permissions controls, you should understand the tagging feature in AWS.

AWS enables customers to assign metadata to their AWS resources and identities in the form of *tags*. Each tag is a simple label that consists of a customer-defined key and an optional value. Tags can make it easier to manage, search for, and filter resources.

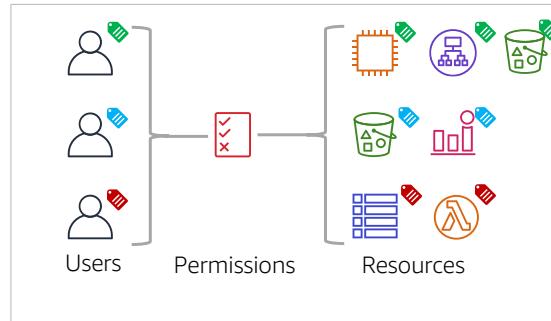
Tags have many practical uses. For example, you can create *technical tags* to identify that a resource is a web server, part of a specific project, part of a specific environment (test, development, or production), among others. You can also create *business tags* to identify the department or cost center that should be billed for this resource or the project that this resource is a part of. Finally, you can also set *security tags*, such as an identifier for the specific data-confidentiality level that a resource supports.

You can create up to 50 tags per resource. For each resource, each tag key must be unique, and each tag key can have only one value. Tag keys and values are case-sensitive.

You can also add tags to IAM users and IAM roles. Tags are an important part of the second access-control method that you will learn about next.

Attribute-based access control (ABAC)

- Highly scalable approach to access control
 - Attributes are a key or a key-value pair, such as a tag
 - Example attributes –
 - Team = Developers
 - Project = Unicorn
- Permissions (policy) rules are easier to maintain with ABAC than with RBAC
- Benefits
 - Permissions automatically apply, based on attributes
 - Granular permissions are possible *without* a permissions update for every new user or resource
 - Fully auditable



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

35

Now that you know about the tagging feature, you will learn about the second approach to access control: attribute-based access control (ABAC).

ABAC enables you to use attributes to create general permissions rules that scale with your organization.

In this model, IAM users have attributes that you created and applied, such as one or more tags.

Resources also have attributes, like matching tags, that you also applied to the resources.

With the RBAC approach, writing permissions is relatively straightforward. The policy checks to see if an attribute that is applied to the IAM user is also applied to the resource that they want to access. When you create new IAM users and new account resources, you apply the correct tags to the users and to the resources.

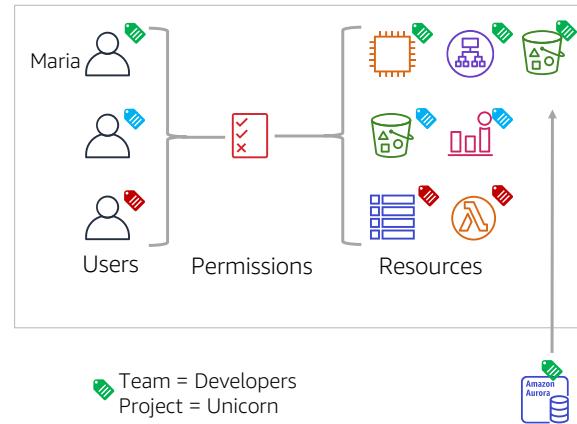
With the ABAC approach, you can grant developers access to their project resources, but you do not need to specify resources in the policy file.

You can imagine how scalable the ABAC approach to access management can be. You do not need to modify your permissions settings. Permissions apply automatically when resources or users are created with the correct tags.

Applying ABAC to your organization

How to apply ABAC to your organization:

1. Set access control attributes on identities
2. Require attributes for new resources
3. Configure permissions based on attributes
4. Test
 - a) Create new resources
 - b) Verify that permissions automatically apply



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

36

To apply ABAC to your organization, the first step is to create identities, such as IAM users or IAM roles. These identities must have the attributes that will be used for access control purposes. For example, you can apply the *Team = Developers* and *Project = Unicorn* tags to the *Maria* user.

Next, require attributes for new resources. You should create policies that enforce rule. For example, you could require that a *Project* attribute and a *Team* attribute are applied to any resource when it is created.

Third, configure access permissions based on the attributes. For example, say that an IAM user has the *Project = Unicorn* and *Team = Developers* tags. If that user tries to access a resource that has matching values for the same two tags, then the policy will allow the access. Otherwise, the policy will deny access.

Fourth, test your configuration. For example, you could try to create an Amazon Aurora database instance without the required tags. The attempt should fail. Try creating the database instance again with the required tags. This time, you should be able to create the resource successfully. Finally, you could try to access the database instance as the *Maria* user. Your access should succeed. However, your access should be denied if you try to access the database instance as a different user who does not have the matching tags.

Externally authenticated users

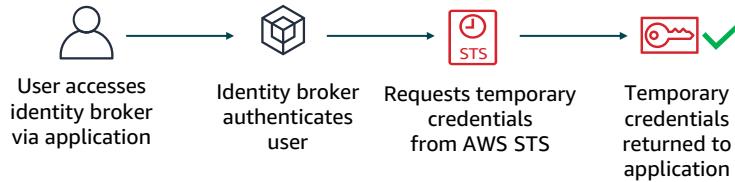
Identity federation

- User authentication completed by a system that is external to the AWS account
 - Example: corporate directory
- It provides a way to allow access through existing identities, without creating IAM users

Identity federation options

- AWS STS
 - Public identity service providers (IdPs)
 - Custom identity broker application
- Security Assertion Markup Language (SAML)
- Amazon Cognito

IdP authentication overview



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

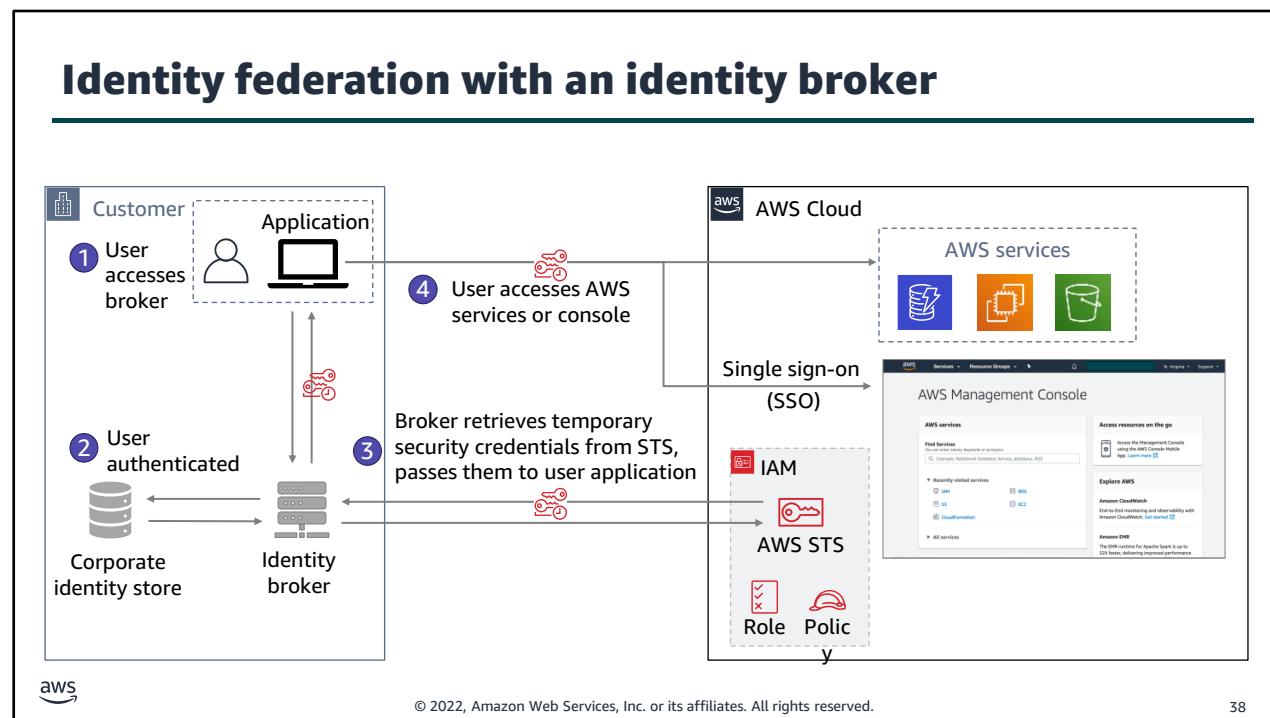
37

You will now learn about a new topic: externally authenticated users.

IAM supports identity federation for delegated access to the AWS Management Console or AWS APIs. With identity federation, external identities are granted secure access to resources in your AWS account *without* needing to create IAM users.

The graphic shows the four primary steps that occur when you use an *identity provider (IdP)* to create temporary credentials for a user or application.

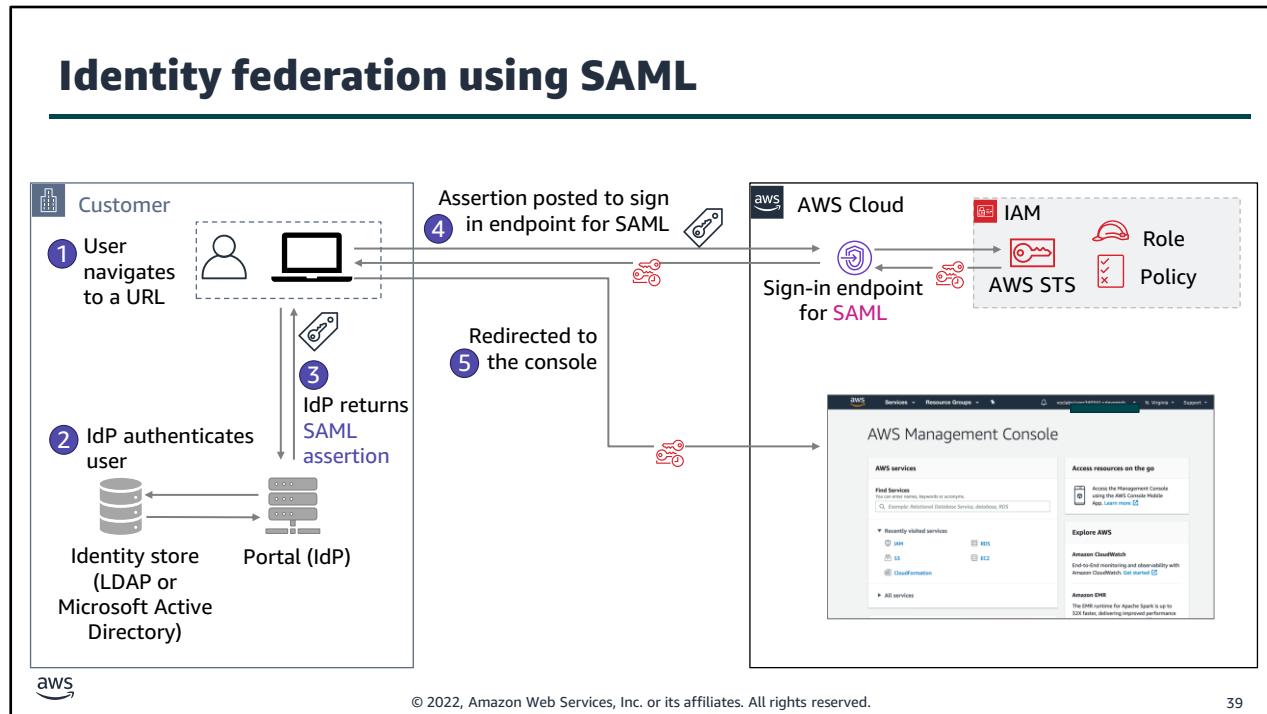
Identity federation can be accomplished in one of three ways. The first way is to use a corporate IdP (such as Microsoft Active Directory) or a custom identity broker application. Each option uses AWS STS. The second approach is to create an integration that uses Security Assertion Markup Language (SAML). The third approach is to use a web identity provider, such as Amazon Cognito. The next few slides discuss each of these three approaches.



You will now learn how to accomplish identity federation by using an identity broker.

The process includes these steps:

1. A user accesses an application. The user enters their user ID and password, and submits them.
2. The identity broker receives the authentication request. It then communicates with the corporate identity store, which might be Microsoft Active Directory or a Lightweight Directory Access Protocol (LDAP) server.
3. If the authentication request is successful, the identity broker makes a request to AWS STS. The request is to retrieve temporary AWS security credentials for the user application.
4. The user application receives the temporary AWS security credentials and redirects the user to the AWS Management Console. The user did not need to sign directly into AWS with a different set of credentials. This process is an example of a single-sign on (SSO) implementation. The user application could also use these same temporary AWS security credentials to access AWS services if the IAM policy document allows it.



You will now learn about the second option for accomplishing identity federation. This approach uses the *SAML* open standard for exchanging authentication and authorization data between IdPs and service providers.

The process involves these steps:

1. A user in your organization navigates to an internal portal in your network. The portal also functions as the IdP that handles the SAML trust between your organization and AWS.
2. The IdP authenticates the user's identity against the identity store, which might be an LDAP server or Microsoft Active Directory.
3. The portal receives the authentication response as a *SAML assertion* from the IdP.
4. The client posts the SAML assertion to the AWS sign-in endpoint for SAML. The endpoint communicates with AWS STS, and it invokes the AssumeRoleWithSAML operation to request temporary security credentials and construct a sign-in URL.
5. The client receives the temporary AWS security credentials. The client is redirected to the AWS Management Console and is authenticated with the temporary AWS security credentials.

Amazon Cognito

Amazon Cognito is a fully managed service.



Amazon
Cognito

- It provides [authentication, authorization, and user management](#) for web and mobile applications
- Amazon Cognito provides web identity federation
 - They can be used as the identity broker that supports IdPs that are compatible with [OpenID Connect \(OIDC\)](#)
- Federated identities
 - Users sign in with social identity providers (Amazon, Facebook, Google) or with SAML
- User pools
 - You can maintain a directory with user profiles authentication tokens



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

40

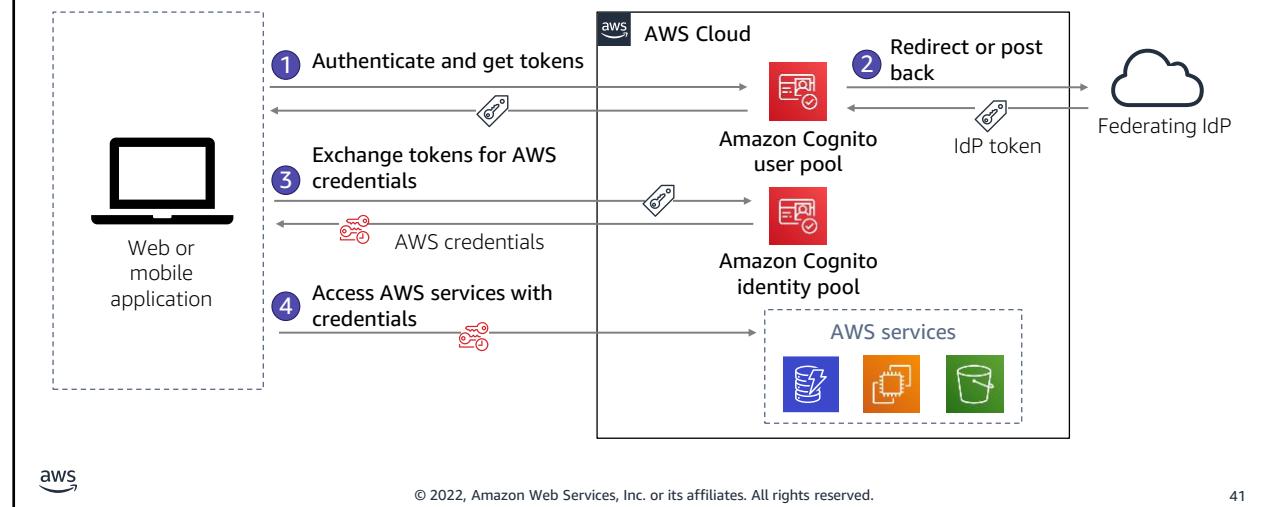
The third and final identity federation option is using Amazon Cognito. *Amazon Cognito* is a fully managed service that provides authentication, authorization, and user management for web and mobile applications. Users can sign in directly with a user name and password or through a third party, such as Facebook, Amazon, or Google.

The two main components of Amazon Cognito are *user pools* and *identity pools*.

A *user pool* is a user directory in Amazon Cognito. With a user pool, users can sign in to a web or mobile application through Amazon Cognito. They can also federate through a third-party IdP. All members of the user pool have a directory profile that can be accessed through an SDK.

Identity pools enable the creation of unique identities and permissions assignment for users. With an identity pool, users can obtain temporary AWS credentials to access AWS services or resources. Identity pools can communicate with Amazon Cognito user pools' social sign-in with Facebook, Google, and Login with Amazon; and OpenID Connect (OIDC) providers.

Amazon Cognito example



In this scenario, the goal is to authenticate a user using Amazon Cognito, and then grant that user access to another AWS service.

- In the first step, the app user signs in through an Amazon Cognito user pool and, after successfully authenticating, receives user pool tokens.
- Next, the app exchanges the user pool tokens for AWS credentials through an Amazon Cognito identity pool.
- Finally, the app user uses those AWS credentials to access other AWS services.

Section 4 key takeaways



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

42

- IAM roles provide temporary security credentials assumable by a person, application, or service
- The AWS Security Token Service (AWS STS) enables you to request temporary AWS credentials
- With identity federation, user authentication is external to the AWS account
 - Accomplished by using AWS STS, SAML, or Amazon Cognito

Some key takeaways from this section of the module include:

- IAM Roles provide temporary security credentials assumable by a person, application, or service.
- The AWS Security Token Service (STS) allows you to request temporary AWS credentials.
- With identify federation, user authentication occurs external to the AWS account.
 - Accomplished using STS, SAML, or Amazon Cognito.

Section 5: Multiple accounts

Module 8: Securing User and Application Access



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

Introducing Section 5: Multiple accounts.

One account or multiple accounts?

Two architectural patterns

- Most organizations choose to create multiple accounts

Advantages of multiple accounts

- Isolate business units or departments
- Isolate development, test, and production environments
- Isolate auditing data, recovery data
- Separate accounts for regulated workloads
- Easier to trigger cost alerts for each business unit's consumption

Multiple VPCs in a single account
architectural pattern



Multiple accounts, a VPC in each account
architectural pattern



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

44

When you use AWS to support the different teams and departments in an organization, you can choose between two general architectural patterns to isolate and separate the resources that each team uses.

The first pattern is to define multiple virtual private clouds (VPCs) in a single AWS account. If you prefer centralized information security management with minimum overhead, you could choose to use a single AWS account.

The second pattern is to create multiple AWS accounts and define a VPC in each account. In practice, large and small organizations tend to create multiple accounts for their organizations. For example, they might create individual accounts for various business units. They could also create separate accounts for their development, test, and production resources.

When customers use separate AWS accounts (usually with consolidated billing) for development and production resources, it enables them to cleanly separate different types of resources. It can also provide some security benefits.

Alternatively, if your business maintains separate environments for production, development, and testing, you could configure three AWS accounts and have one account for each environment. Also, if you have multiple autonomous departments, you could also create separate AWS accounts for each autonomous part of the organization.

When you use multiple accounts, a more efficient strategy is to create a single AWS account for common project resources. Common resources might include Domain Name System (DNS) services, Microsoft Active Directory, and content management systems (CMSs). You could also separate accounts for the autonomous projects or departments. This strategy enables you to assign permissions and policies under each department or project account, and grant access to resources across accounts.

Challenges for managing multiple accounts

- Security management across accounts
 - IAM policy replication
- Creating new accounts
 - Involves many manual processes
- Billing consolidation
- Centralized governance is needed to ensure consistency



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

45

Although most organizations choose to use multiple AWS accounts, that choice comes with some challenges.

First, you must determine how to effectively manage security across all your accounts. If you replicate the IAM policies that you defined across all accounts to ensure consistency, it could involve custom automation, manual effort, or both.

Also, you might be constantly asked to create more accounts. It takes time to manually create these accounts. It also might be difficult to track all the accounts and the purpose of each account.

It can also be a challenge to determine which cost center in the organization should be billed for which resources in which accounts. And finally, you might also want to achieve the centralized governance that is needed to ensure consistency.

Manage multiple accounts with AWS Organizations



Centrally manage and enforce policies across multiple AWS accounts.

- **Group-based** account management
- **Policy-based access** to AWS services
- **Automated account creation** and management
- Consolidated billing
- API-based



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

46

AWS offers a service that is designed to address these management challenges.

AWS Organizations is a managed service for account management. An organization is an entity that you create to consolidate, centrally view, and manage all your AWS accounts. You determine the functionality of an organization through the feature set that you enable.

Organizations helps you manage policies for multiple AWS accounts. You can use the service to create groups of accounts. You can then attach policies to a group so that the correct policies are applied across the accounts.

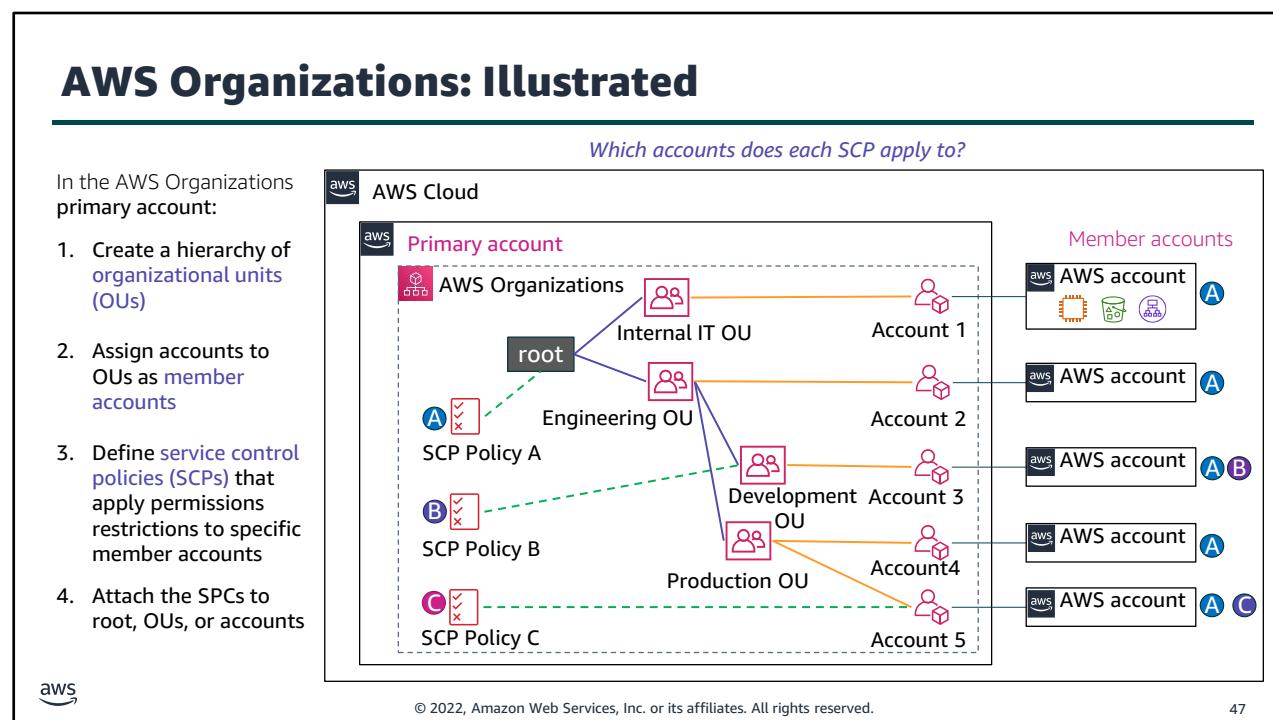
You can create groups of AWS accounts, and then apply different policies to each group.

The Organizations APIs can create new accounts programmatically and add them to a group. The policies that are attached to the group are automatically applied to the new account.

You can also set up a single payment method for all the AWS accounts in your organization through consolidated billing. With consolidated billing, you can see a combined view of charges that are incurred by all your accounts.

Finally, you can manage the use of AWS services at the API level. For example, you can apply a policy to a group of accounts that will only allow IAM users in those accounts to read data from

S3 buckets.



Here is an example AWS organization. It is defined inside a regular AWS account that is referred to on the slide as the *primary account* because the AWS organization is defined in it.

When you create an *organization* in the primary account, the organization automatically creates a parent container that is called *root*. Under each root in the organization, you can then define *organizational units*, which are also known as *OUs*. Each OU is a container for *member accounts*. An OU can also contain other OUs, and those OUs can contain more accounts. This feature enables you to create a tree-like hierarchy. You can think of the root and OUs as branches that reach out and end in accounts, which are like the leaves of the tree.

To configure access controls across accounts, you then define *service control policies (SCPs)*. Attach each policy to the appropriate place in the hierarchy of OUs and accounts. The policy flows out away from the root and it affects all OUs and accounts beneath it. Therefore, if you apply an SCP to the root (like *SCP Policy A* in the example), it will apply to all OUs and accounts in the organization. You can attach an SCP to the root, to any OU, or to an individual account.

Remember that like IAM policies, SCPs will only grant access if it is both explicitly allowed and is not explicitly denied by any other SCP or IAM policy that applies to the user. For example, say that SCP Policy A, which is applied to the root of the organization, sets more restrictions on a particular service or set of resources than SCP Policy C. Then, users in Account 5 are subject to the more restrictive permissions set by Policy A. Similarly, if any IAM policies at the individual

account level explicitly deny any actions for the user, these IAM policies override any permissions in the SCPs that are granted to the account.

Example uses of SCPs

- Characteristics of service control policies (SCPs)
 - They enable you to control which services are accessible to IAM users in member accounts
 - SCPs cannot be overridden by the local administrator
 - IAM policies that are defined in individual accounts still apply

- Example uses of SCPs

- Create a policy that *blocks* service access or specific actions
 - Example: Deny users from disabling AWS CloudTrail in all member accounts
- Create a policy that *allows* full access to specific services
 - Example: Allow full access to Amazon EC2 and CloudWatch
- Create a policy that *enforces the tagging* of resources



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

48

Service control policies (SCPs) enable you to control which services are accessible to IAM users in member accounts. Say that you have specific policies that you want to apply across multiple accounts. It is easier to define these policies in an SCP than to replicate these permissions settings into IAM policy documents in each account.

SCPs should be used with IAM policies that are defined in each individual account. You can think of the SCPs as providing general boundaries around the services and general permissions that users should be allowed or denied access to. Then, you can use IAM policies to set more granular access controls that are specific to individual accounts.

You can author SCPs that block (or deny) access to certain services. You can also define SCPs that allow access to certain services. Finally, you might decide to create an SCP that enforces the tagging of resources. By doing so, your tagging strategy for access control or cost allocation can remain effective when new resources are created in your accounts.

Section 5 key takeaways



- You can use [multiple AWS accounts](#) to isolate business units, development and test environments, regulated workloads, and auditing data
- [AWS Organizations](#) enables you to configure automated account creation and consolidated billing
- You can configure access controls across accounts by using [service control policies \(SCPs\)](#)

© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

49

Some key takeaways from this section of the module include:

- You can use multiple AWS accounts to isolate business units, development and test environments, regulated workloads, and auditing data
- AWS Organizations allows you to configure automated account creation, consolidated billing
- You can configure access controls across accounts by using service control policies (SCPs)

Module 8 – Challenge Lab: Controlling AWS Account Access by Using IAM



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

50

You will now complete the Module 8 – Challenge Lab: Controlling AWS Account Access by Using IAM.

The business need: User access control



The café must define what level of access users should have across cloud resources. They must then put these access controls into place across the AWS account.

When Mateo visited the café recently, he told **Sofía about the features of the IAM service**. She plans to use IAM to accomplish her objective.



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

51

After speaking with Mateo about the café's AWS infrastructure, Sofía realized that she must address some basic security concerns about the way that the café staff has been using the AWS account.

The café is now large enough that team members who build, maintain, or access applications on AWS are specializing into roles (such as developer or database administrator). Up to now, they haven't made an effort to clearly define what level of access each user should have based on their roles and responsibilities.

Challenge lab: Tasks

1. Configuring an IAM group with policies and an IAM user
2. Logging in as Nikhil and testing access
3. Configuring IAM for database administrator user access
4. Logging in as the database administrator and resolving the database connectivity issue
5. Using the IAM Policy Simulator and creating a custom IAM policy with the visual editor

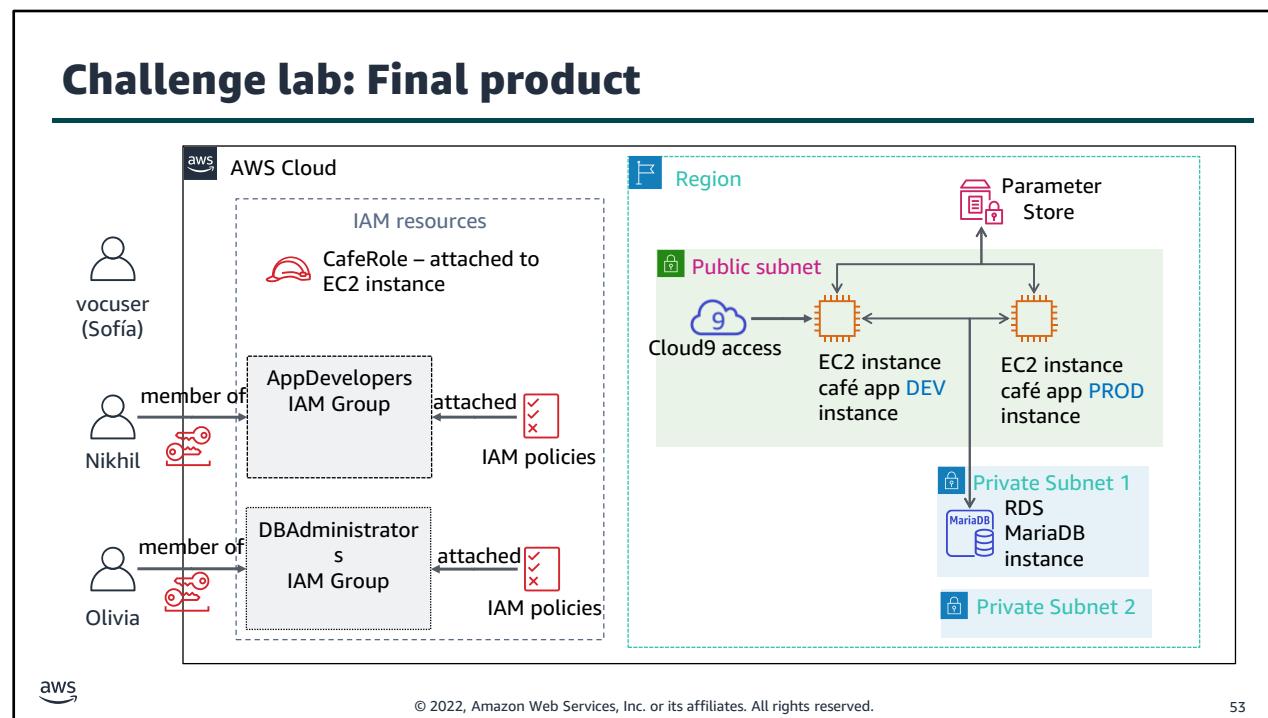


© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

52

In this challenge lab, you will complete the following tasks:

1. Configuring an IAM group with policies and an IAM user
2. Logging in as Nikhil and testing access
3. Configuring IAM for database administrator user access
4. Logging in as the database administrator and resolving the database connectivity issue
5. Using the IAM Policy Simulator and creating a custom IAM policy with the visual editor



The diagram summarizes what you will have built after you complete the lab.



~ 80 minutes



Begin Module 8 – Challenge Lab: Controlling AWS Account Access by Using IAM

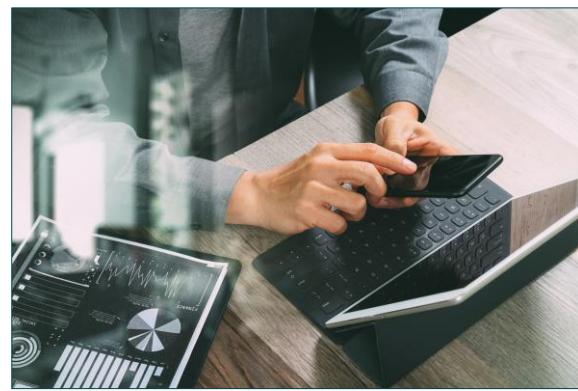


© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

54

It is now time to start the challenge lab.

Challenge lab debrief: Key takeaways



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

55

The educator might now choose to lead a conversation about the key takeaways from the challenge lab after you have completed it.

Module wrap-up

Module 8: Securing User and Application Access



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

It is now time to review the module and wrap up with a knowledge check and discussion of a practice certification exam question.

Module summary

In summary, in this module, you learned how to:

- Explain the purpose of AWS Identity and Access Management (IAM) users, groups, and roles
- Describe how to allow user federation within an architecture to increase security
- Recognize how AWS Organizations service control policies (SCPs) increase security within an architecture
- Describe how to manage multiple AWS accounts
- Configure IAM users



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

57

In summary, in this module, you learned how to:

- Explain the purpose of AWS Identity and Access Management (IAM) users, groups, and roles
- Describe how to allow user federation within an architecture to increase security
- Recognize how AWS Organizations service control policies (SCPs) increase security within an architecture
- Describe how to manage multiple AWS accounts
- Configure IAM users

Complete the knowledge check



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

58

It is now time to complete the knowledge check for this module.



Sample exam question

A company is storing an access key (access key ID and secret access key) in a text file on a custom AMI. The company uses the access key to access DynamoDB tables from instances created from the AMI. The security team has mandated a more secure solution.

Which solution will meet the security team's mandate?

Choice	Response
A	Put the access key in an S3 bucket, and retrieve the access key on boot from the instance.
B	Pass the access key to the instances through instance user data.
C	Obtain the access key from a key server launched in a private subnet.
D	Create an IAM role with permissions to access the table, and launch all instances with the new role.

© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

59

Look at the answer choices and rule them out based on the keywords.

Sample exam question answer



A company is storing an access key (access key ID and secret access key) in a text file on a custom AMI. The company uses the access key to access DynamoDB tables from instances created from the AMI. The security team has mandated a more secure solution.

Which solution will meet the security team's mandate?

The correct answer is D.

The keywords in the question are "storing an access key", "DynamoDB tables from instances", "custom AMI", and "most secure solution".

© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

60

The following are the keywords to recognize: **"storing an access key"**, **"DynamoDB tables from instances"**, **"custom AMI"**, and **"most secure solution"**.

The correct answer is D. IAM roles for EC2 instances allow applications that run on the instance to access AWS resources without needing to create and store any access keys. Any solution that involves the creation of an access key then introduces the complexity of managing that secret.

Additional resources

- [AWS Well-Architected Framework – Security Pillar](#)
- [IAM FAQs](#)
- [Creating IAM policies video](#)
- [Identity at different layers video](#)
- [Identity Providers and Federation](#)



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

61

If you want to learn more about the topics covered in this module, you might find the following additional resources helpful:

- [AWS Well-Architected Framework – Security Pillar](#)
- [IAM FAQs](#)
- [Creating IAM policies video](#)
- [Identity at different layers video](#)
- [Identity Providers and Federation](#)

Thank you

Corrections, feedback, or other questions?
Contact us at <https://support.aws.amazon.com/#/contacts/aws-academy>.
All trademarks are the property of their owners.



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

62

Thank you for completing this module.



**AWS Academy Cloud Architecting
Module 09 Student Guide
Version 2.0.8**

200-ACACAD-20-EN-SG

© 2022 Amazon Web Services, Inc. or its affiliates. All rights reserved.

This work may not be reproduced or redistributed, in whole or in part, without prior written permission from Amazon Web Services, Inc. Commercial copying, lending, or selling is prohibited.

All trademarks are the property of their owners.

Contents

Module 9: Implementing Elasticity, High Availability, and Monitoring

4



Module 9: Implementing Elasticity, High Availability, and Monitoring

AWS Academy Cloud Architecting

© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

Welcome to Module 9: Implementing Elasticity, High Availability, and Monitoring.

Module overview

Sections

1. Architectural need
2. Scaling your compute resources
3. Scaling your databases
4. Designing an environment that's highly available
5. Monitoring

Demonstrations

- Creating Scaling Policies for Amazon EC2 Auto Scaling
- Creating a Highly Available Web Application
- Amazon Route 53

Labs

- Guided Lab: Creating a Highly Available Environment
- Challenge Lab: Creating a Scalable and Highly Available Environment for the Café



Knowledge check



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

2

This module includes the following sections:

1. Architectural need
2. Scaling your compute resources
3. Scaling your databases
4. Designing an environment that's highly available
5. Monitoring

This module also includes:

- A demonstration on how to create target tracking and step scaling policies for Amazon EC2 Auto Scaling
- A demonstration on how to deploy a highly available web application with an Application Load Balance
- A demonstration on Amazon Route 53
- A guided lab where you will create a highly available environment
- A challenge lab where you will create a scalable and highly available environment for the café

Finally, you will be asked to complete a knowledge check that will test your understanding of key concepts covered in this module.

Module objectives

At the end of this module, you should be able to:

- Use Amazon EC2 Auto Scaling within an architecture to promote elasticity
- Explain how to scale your database resources
- Deploy an Application Load Balancer to create a highly available environment
- Use Amazon Route 53 for Domain Name System (DNS) failover
- Create a highly available environment
- Design architectures that use Amazon CloudWatch to monitor resources and react accordingly



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

3

At the end of this module, you should be able to:

- Use Amazon EC2 Auto Scaling within an architecture to promote elasticity
- Explain how to scale your database resources
- Deploy an Application Load Balancer to create a highly available environment
- Use Amazon Route 53 for Domain Name System (DNS) failover
- Create a highly available environment
- Design architectures that use Amazon CloudWatch to monitor resources and react accordingly

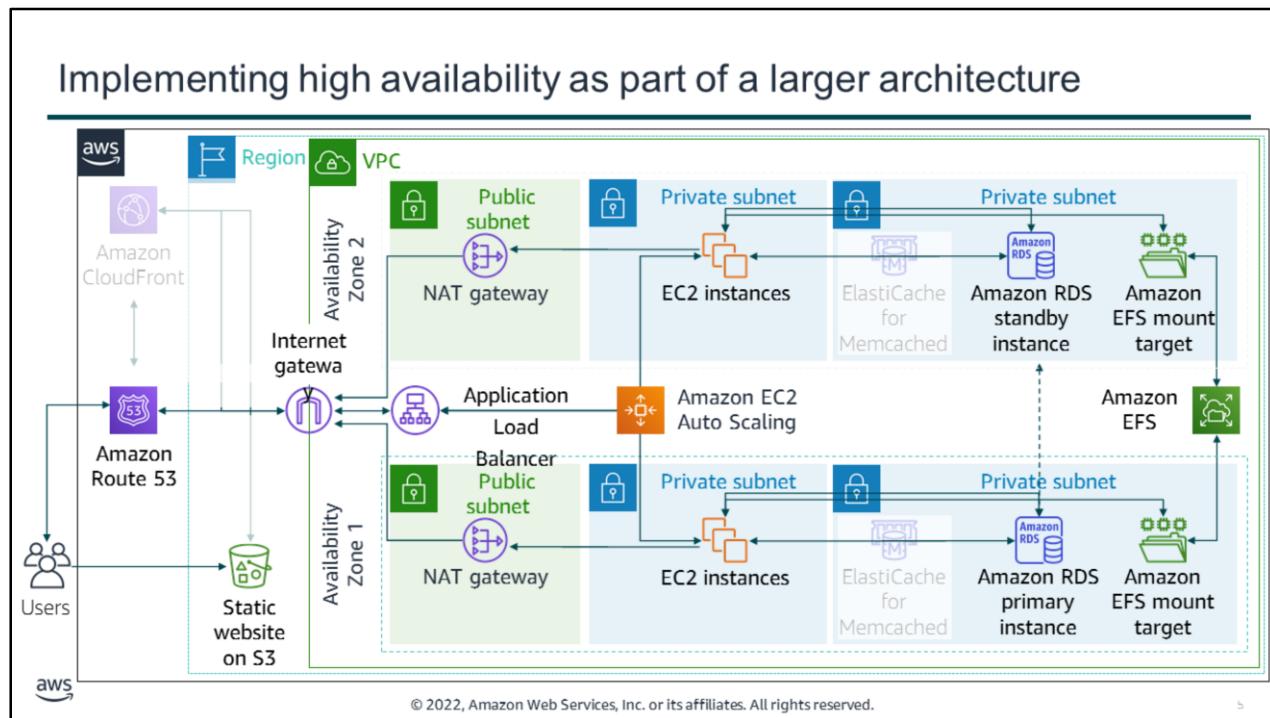
Section 1: Architectural need

Module 9: Implementing Elasticity, High Availability, and Monitoring



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

Introducing Section 1: Architectural need.



In this module, you will learn how to implement a reactive architecture that is elastic, resilient, and responsive. The components that make this architecture scalable and highly available (such as the second Availability Zone, the Application Load Balancer, Amazon EC2 Auto Scaling, and Amazon Route 53) are discussed.

Café business requirement

The café will be featured in a famous TV food show. When it airs, the architecture must handle significant increases in capacity.



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

6

The café will soon be featured in a famous TV food show. When it airs, Sofía and Nikhil anticipate that the café’s web server will experience a temporary spike in the number of users—perhaps even up to tens of thousands of users. Currently, the café’s web server is deployed in one Availability Zone, and they are worried that it won’t be able to handle the expected increase in traffic. They want to ensure that their customers have a great experience when they visit the website, and that they don’t experience any issues, such as lags or delays in placing orders.

To ensure this experience, the website must be responsive, scale both up and down to meet fluctuating customer demand, and be highly available. It must also incorporate load balancing. Instead of overloading a single server, the architecture must distribute customer order requests across multiple application servers so it can handle the increase in demand.

Reactive architectures



Elastic
and scalable



Resilient



Responsive



Message-driven



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

Modern applications must be able to handle massive amounts of data, with no downtime and with sub-second response times. To meet these requirements, you can implement a [reactive system](#) that is elastic, resilient, responsive, and message-driven. A well-designed reactive architecture can save you money and provide a better experience for your users.

In this module, you learn how to build reactive architectures on AWS that are elastic, resilient, and responsive. You learn about message-driven components to build decoupled architectures in a later module.

Section 2: Scaling your compute resources

Module 9: Implementing Elasticity, High Availability, and Monitoring



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

Introducing Section 2: Scaling your compute resources.

What is elasticity?

An elastic infrastructure can expand and contract as capacity needs change.

Examples:

- Increasing the number of web servers when traffic spikes
- Lowering write capacity on your database when traffic goes down
- Handling the day-to-day fluctuation of demand throughout your architecture



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

9

One characteristic of a reactive architecture is elasticity. *Elasticity* means that the infrastructure can expand and contract when capacity needs change. You can acquire resources when you need them and release resources when you do not.

Elasticity enables you to:

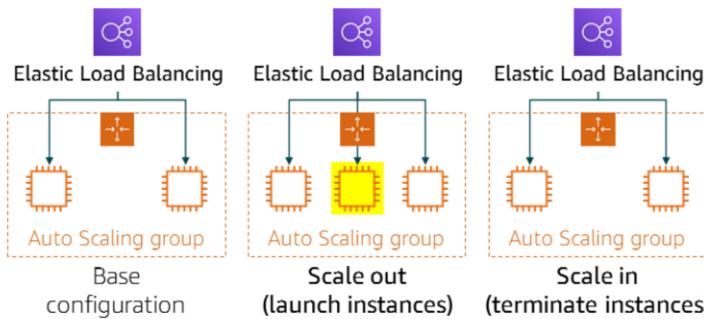
- Increase the number of web servers when traffic to your application spikes
- Lower the write capacity on your database when traffic goes down
- Handle the day-to-day fluctuation of demand throughout your architecture

In the café example, elasticity is important because after the TV show airs, the website might see an immediate increase in traffic. The traffic might drop to normal levels after a week, or might increase again during holiday seasons.

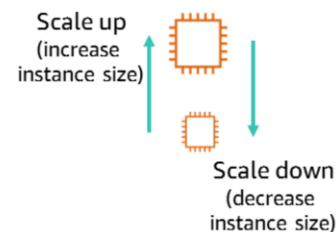
What is scaling?

A technique that is used to achieve elasticity

Horizontal scaling



Vertical scaling



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

10

Scaling is a technique that is used to achieve elasticity. Scaling is the ability to increase or decrease the compute capacity of your application.

Scaling has two types:

- *Horizontal scaling* is where you add or remove resources. For example, you might need to add more hard drives to a storage array or add more servers to support an application. Adding resources is referred to as *scaling out*, and terminating resources is referred to as *scaling in*. Horizontal scaling is a good way to build internet-scale applications that take advantage of the elasticity of cloud computing.
- *Vertical scaling* is where you increase or decrease the specifications of an individual resource. For example, you could upgrade a server so it has a larger hard drive or a faster CPU. With Amazon Elastic Compute Cloud (Amazon EC2), you can stop an instance and resize it to an instance type that has more RAM, CPU, I/O, or networking capabilities. Vertical scaling can eventually reach a limit, and it is not always a cost-efficient or highly available approach. However, it is easy to implement and can be sufficient for many use cases, especially in the short term.

Amazon EC2 Auto Scaling



- Launches or terminates instances based on specified conditions
- Automatically registers new instances with load balancers when specified
- Can launch across Availability Zones



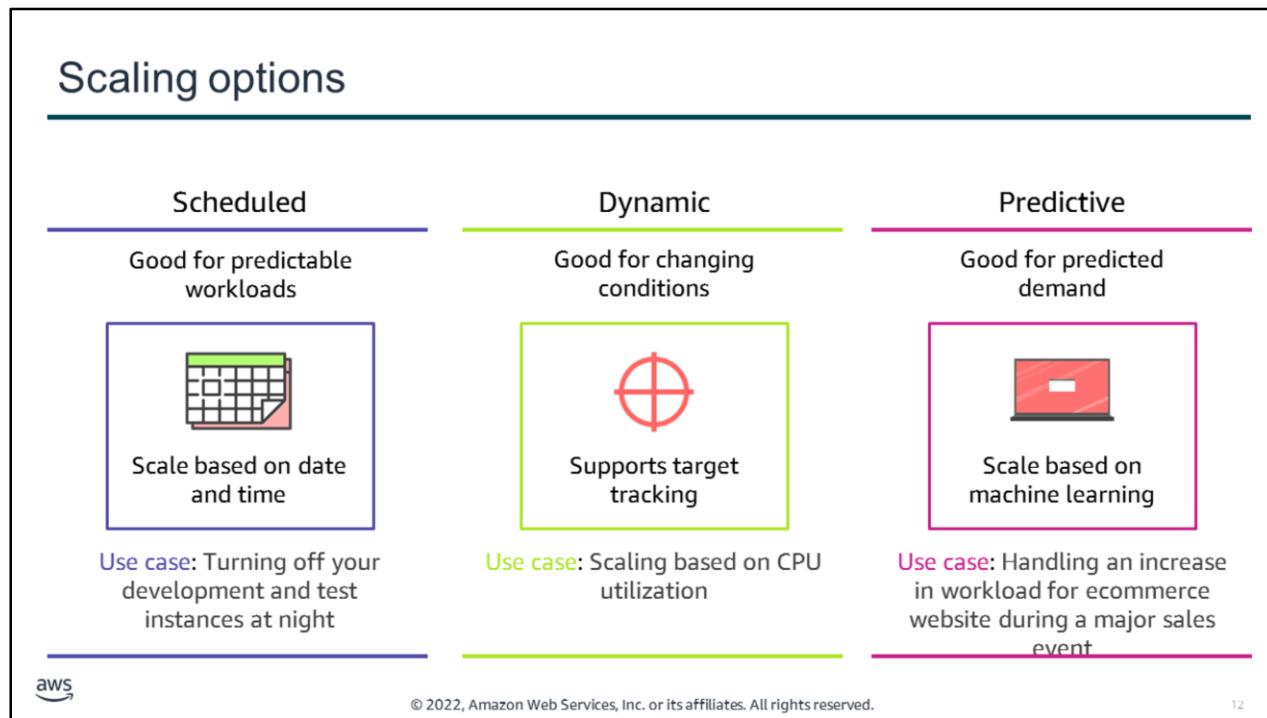
© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

11

In the cloud, scaling can be handled automatically. [Amazon EC2 Auto Scaling](#) helps you maintain application availability and enables you to automatically add or remove EC2 instances according to policies that you define, schedules, and health checks. If you specify scaling policies, then Amazon EC2 Auto Scaling can launch or terminate instances when demand on your application increases or decreases.

Amazon EC2 Auto Scaling integrates with Elastic Load Balancing—it automatically registers new instances with load balancers to distribute incoming traffic across the instances.

Amazon EC2 Auto Scaling enables you to build highly available architectures that span multiple Availability Zones in a Region. You will learn more about high availability later in this module. If one Availability Zone becomes unhealthy or unavailable, Amazon EC2 Auto Scaling launches new instances in an unaffected Availability Zone. When the unhealthy Availability Zone returns to a healthy state, Amazon EC2 Auto Scaling automatically redistributes the application instances evenly across all the designated Availability Zones.



Amazon EC2 Auto Scaling provides several ways to adjust scaling to best meet the needs of your applications:

- **Scheduled scaling** – With scheduled scaling, scaling actions are performed automatically as a function of date and time. This feature is useful for predictable workloads when you know exactly when to increase or decrease the number of instances in your group. For example, suppose that every week, the traffic to your web application starts to increase on Wednesday, remains high on Thursday, and starts to decrease on Friday. You can plan your scaling actions based on the predictable traffic patterns of your web application. To implement scheduled scaling, you create a [scheduled action](#).
- **Dynamic, on-demand scaling** – This approach is a more advanced way to scale your resources. It enables you to define parameters that control the scaling process. For example, you have a web application that currently runs on two EC2 instances. You want the CPU utilization of the Auto Scaling group to stay close to 50 percent when the load on the application changes. This option is useful for scaling in response to changing conditions, when you don't know when those conditions are going to change. Dynamic scaling gives you extra capacity to handle traffic spikes without maintaining an excessive number of idle resources. You can configure your Auto Scaling group to scale automatically to meet this need.
- **Predictive scaling** – You can use Amazon EC2 Auto Scaling with AWS Auto Scaling to implement predictive scaling, where your capacity scales based on predicted demand. Predictive scaling uses data that is collected from your actual Amazon EC2 usage, and the data is further informed by billions of data points that are drawn from observations by AWS. AWS then uses well-trained machine learning models to predict your expected traffic (and

Amazon EC2 usage), including daily and weekly patterns. The model needs at least 1 day of historical data to start making predictions. It is re-evaluated every 24 hours to create a forecast for the next 48 hours. The prediction process produces a scaling plan that can drive one or more groups of automatically scaled EC2 instances.

Dynamic scaling and predictive scaling can be used together to scale your infrastructure faster. Finally, you can also add or remove EC2 instances manually. With [manual scaling](#), you specify only the change in the maximum, minimum, or desired capacity of your Auto Scaling group.

Dynamic scaling policy types

- **Simple scaling** – Single scaling adjustment
 - Example use cases: New workloads, spiky workloads
- **Step scaling** – Adjustment depends on size of alarm breach
 - Example use case: Predictable workloads
- **Target tracking scaling** – Target value for specific metric
 - Example use case: Horizontally scalable applications, such as load-balanced applications and batch data-processing applications



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

13

Dynamic scaling means adjusting your application's capacity to meet changing demand so that you can optimize availability, performance, and cost. The [scaling policy type](#) determines how the scaling action is performed:

- With **step scaling** and **simple scaling** policies, you choose scaling metrics and threshold values for the CloudWatch alarms that trigger the scaling process. You also define how your Auto Scaling group should be scaled when a threshold is in breach for a specified number of evaluation periods. The main difference is that with step scaling, the current capacity of the Auto Scaling group increases or decreases based on a set of scaling adjustments, known as *step adjustments*, that vary based on the size of the alarm breach.
- **Target tracking scaling** policies increase or decrease the current capacity of the group based on a target value for a specific metric. This type of scaling is similar to the way that your thermostat maintains the temperature of your home: you select a temperature, and the thermostat does the rest. With target tracking scaling policies, you select a scaling metric and set a target value. Amazon EC2 Auto Scaling creates and manages the CloudWatch alarms that trigger the scaling policy and calculates the scaling adjustment based on the metric and the target value. The scaling policy adds or removes capacity as required to keep the metric at, or close to, the specified target value. In addition to keeping the metric close to the target value, a target tracking scaling policy also adjusts to the changes in the metric due to a changing load pattern.

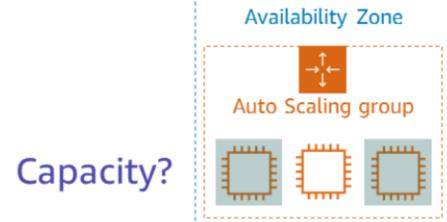
To learn more about target tracking scaling policies, see the following resources:

- [Target Tracking Scaling Policies for Amazon EC2 Auto Scaling](#)
- [Set it and Forget it Auto Scaling Target Tracking Policies](#)
- [AWS re: Invent 2017: Auto Scaling Prime Time: Target Tracking Hits the Bullseye at Netflix](#)

Auto Scaling groups

An Auto Scaling group defines:

- Minimum capacity
- Maximum capacity
- Desired capacity*



Capacity?

*The [desired capacity](#) reflects the number of instances that are running and can fluctuate in response to events.



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

14

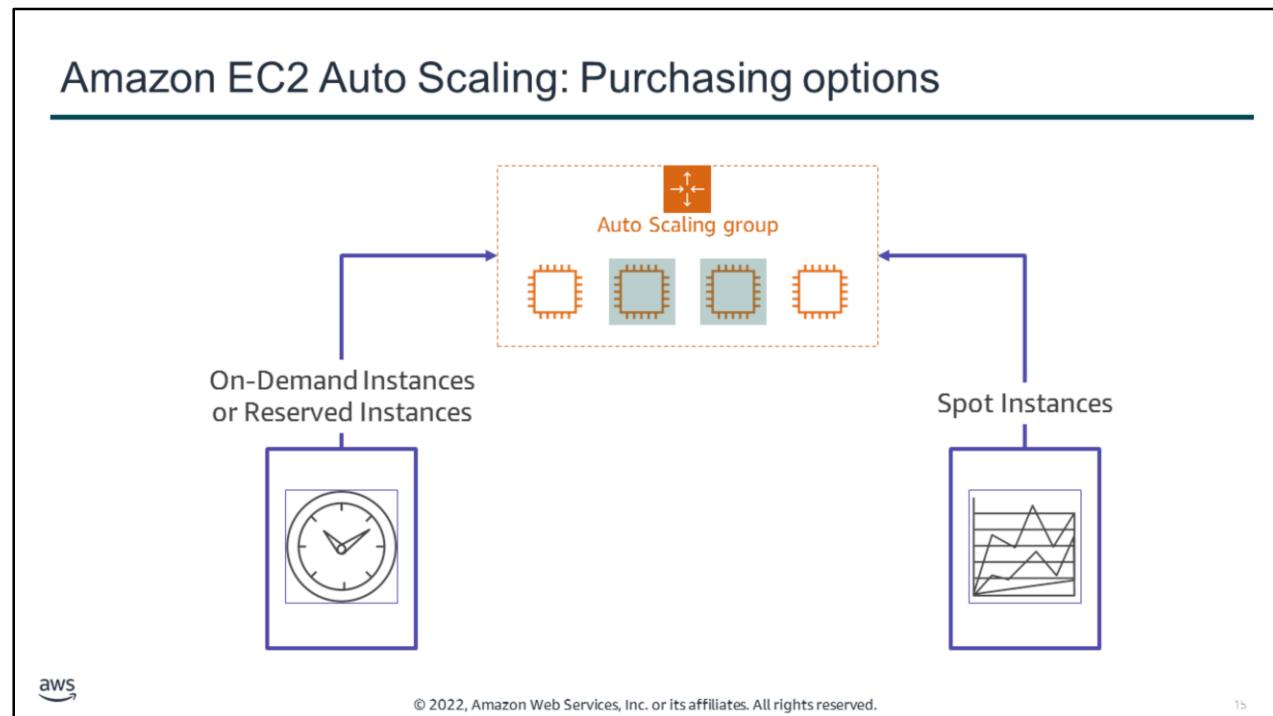
Amazon EC2 Auto Scaling helps you ensure that you have the correct number of Amazon EC2 instances available to handle the load for your application.

You create collections of EC2 instances, called *Auto Scaling groups*. You can specify the *minimum* number of instances in each Auto Scaling group, and Amazon EC2 Auto Scaling helps ensure that your group never goes below this size. You can specify the *maximum* number of instances in each Auto Scaling group, and Amazon EC2 Auto Scaling helps ensure that your group never goes above this size. If you specify the *desired* capacity, either when you create the group or at any time after you create it, Amazon EC2 Auto Scaling helps ensure that your group has this many instances.

Note that the desired capacity is a trigger-based setting and can fluctuate in response to events like a threshold being breached. It reflects the number of instances that are running at that point. It can never be lower than the min value or greater than the max value. The role of the scaling policy is to act as your automated representative and make decisions on how to adjust the desired capacity. Amazon EC2 Auto Scaling then responds to the change in the desired capacity configuration.

Initially, you set the desired capacity to tell the Auto Scaling group how many instances you want to be running at a particular time. The number of instances that are currently running may be different than the desired value until Amazon EC2 Auto Scaling spins them up or tears them

down.



Amazon EC2 Auto Scaling enables you to scale out and scale in your infrastructure in response to changing conditions. When you configure an Auto Scaling group, you can specify the [EC2 instance types](#) that it uses. You can also specify what percentage of the desired capacity should be fulfilled with [On-Demand Instances, Reserved Instances, and Spot Instances](#). Amazon EC2 Auto Scaling then provisions the lowest price combination of instances to meet the desired capacity based on these preferences.

You can use only one instance type. However, it is a best practice to use a few instance types to avoid trying to launch instances from instance pools that have insufficient capacity. If the Auto Scaling group's request for Spot Instances cannot be fulfilled in one Spot Instance pool, it keeps trying in other Spot Instance pools instead of launching On-Demand Instances.

For more information about purchasing options, see [Auto Scaling Groups with Multiple Instance Types and Purchase Options](#).

Automatic scaling considerations

- Multiple types of automatic scaling
- Simple, step, or target tracking scaling
- Multiple metrics (not just CPU)
- When to scale out and scale in
- Use of lifecycle hooks



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

16

Things for you to consider when you use Amazon EC2 Auto Scaling to scale your architecture are as follows:

- Multiple types of automatic scaling – You might need to implement a combination of scheduled, dynamic, and predictive scaling.
- Dynamic scaling policy type – Simple scaling policies increase or decrease the current capacity of the group based on a single scaling adjustment. Step scaling policies increase or decrease the current capacity of the group based on a set of scaling adjustments, known as *step adjustments*, that vary based on the size of the alarm breach. Target tracking scaling policies increase or decrease the current capacity of the group based on a target value for a specific metric.
- Multiple metrics – Some architectures must scale on two or more metrics (not just CPU). AWS recommends that you use a target tracking scaling policy to scale on a metric, like average CPU utilization or the RequestCountPerTarget metric from the Application Load Balancer. Metrics that decrease when capacity increases—and increase when capacity decreases—can be used to proportionally scale out or in the number of instances that use target tracking. This type of metric helps to ensure that Amazon EC2 Auto Scaling follows the demand curve for your applications closely.
- When to scale out and scale in – Try to scale out early and fast, and scale in slowly over time.
- Use of lifecycle hooks – Lifecycle hooks enable you to perform custom actions by *pausing* instances when an Auto Scaling group launches or terminates them. When an instance is paused, it remains in a wait state either until you complete the lifecycle action by using the complete-lifecycle-action command or the CompleteLifecycleAction operation, or until the timeout period ends (1 hour by default).

Demonstration: Creating Scaling Policies for Amazon EC2 Auto Scaling



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

17

Now, the educator might choose to demonstrate how to create target tracking and step scaling policies for Amazon EC2 Auto Scaling.

Section 2 key takeaways



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

18

- An **elastic infrastructure** can expand and contract as capacity needs change
- **Amazon EC2 Auto Scaling** automatically adds or removes EC2 instances according to policies that you define, schedules, and health checks
- Amazon EC2 Auto Scaling provides **several scaling options** to best meet the needs of your applications
- When you configure an Auto Scaling group, you can specify the **EC2 instance types** and the combination of **pricing models** that it uses

Some key takeaways from this section of the module include:

- An elastic infrastructure can expand and contract as capacity needs change
- Amazon EC2 Auto Scaling automatically adds or removes EC2 instances according to policies that you define, schedules, and health checks
- Amazon EC2 Auto Scaling provides several scaling options to best meet the needs of your applications
- When you configure an Auto Scaling group, you can specify the EC2 instance types and the combination of pricing models that it uses

Section 3: Scaling your databases

Module 9: Implementing Elasticity, High Availability, and Monitoring



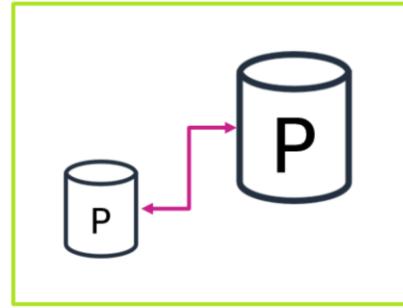
© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

Introducing Section 3: Scaling your databases.

In the last section, you learned how to scale your EC2 instances. You can also scale your database instances. In this section, you learn how to scale your relational and nonrelational databases.

Vertical scaling with Amazon RDS: Push-button scaling

- Scale DB instances **vertically** up or down
- From **micro to 24xlarge** and everything in between
- Scale vertically with **minimal downtime**



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

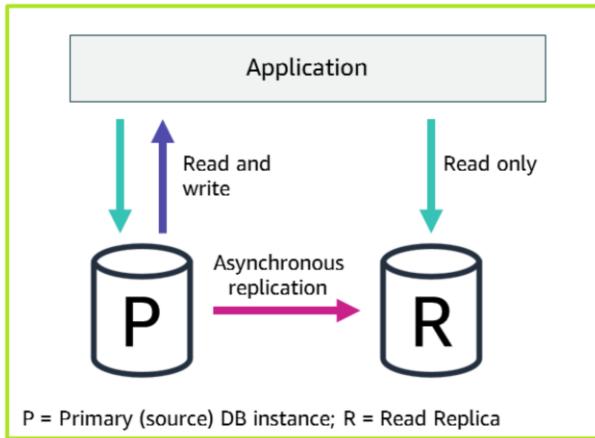
20

First, consider scaling relational databases. As a managed service, Amazon Relational Database Service ([Amazon RDS](#)) scales your relational database so that it can keep up with the increasing demands of your application.

You can vertically scale your Amazon RDS database (DB) instance by changing the Amazon RDS [instance class](#). If you decide to scale the compute resources that are available to your DB instance up or down, be aware that your database is temporarily unavailable while the DB instance class is modified. This period of unavailability typically lasts only a few minutes. It occurs during the maintenance window for your DB instance, unless you specify that the modification should be applied immediately.

When you scale your database instance up or down, your storage size remains the same and it is not affected by the change. You can separately modify your DB instance to increase the allocated storage space or improve the performance by changing the storage type—for example, from General Purpose solid state drive (SSD) to Provisioned input/output operations per second (IOPS) SSD. Instead of manually provisioning storage, you can also use [Amazon RDS Storage Autoscaling](#) to automatically scale storage capacity in response to growing database workloads.

Horizontal scaling with Amazon RDS: Read replicas



- Horizontally scale for **read-heavy** workloads
- Up to **five read replicas** and up to **15 Aurora replicas**
- Replication is **asynchronous**
- Available for Amazon RDS for MySQL, MariaDB, PostgreSQL, and Oracle



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

21

In addition to vertical scaling, you can scale your database horizontally. Amazon RDS uses the MariaDB, MySQL, Oracle, and PostgreSQL DB engines' built-in replication functionality to create a special type of DB instance called a *read replica* from a source DB instance. Updates that are made to the source DB instance are asynchronously copied to the read replica. You can reduce the load on your source DB instance by routing read queries from your applications to the read replica.

To improve the performance of read-heavy database workloads, you can use read replicas to horizontally scale your source DB instance. You can create one or more replicas (up to five) of a given source DB instance and serve high-volume application-read traffic from multiple copies of your data, which increases aggregate read throughput.

If there is a disaster, you can increase the availability of your database by promoting a read replica to a standalone DB instance.

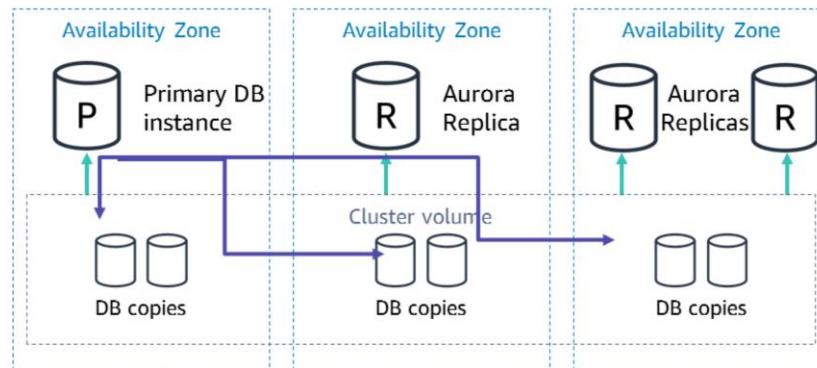
For more information about read replicas, see [Working with Read Replicas](#).

For more information about how to scale your RDS DB instance, see the following resources:

- [Modifying an Amazon RDS DB Instance](#) in the AWS Documentation
- [Scaling Your Amazon RDS Instance Vertically and Horizontally](#) AWS Database Blog post

Scaling with Amazon Aurora

Each Aurora DB cluster can have up to 15 Aurora replicas



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

22

[Amazon Aurora](#) further extends the benefits of read replicas by employing an SSD-backed virtualized storage layer that is purpose-built for database workloads. Amazon Aurora is a MySQL and PostgreSQL-compatible relational database engine built for the cloud. Amazon RDS manages your Amazon Aurora databases and handles tasks such as provisioning, patching, backup, recovery, failure detection, and repair. Amazon Aurora features a distributed, fault-tolerant, self-healing storage system that automatically scales up to 64 TB per database instance. It delivers high performance and availability, point-in-time recovery, continuous backup to Amazon Simple Storage Service (Amazon S3), and replication across three Availability Zones.

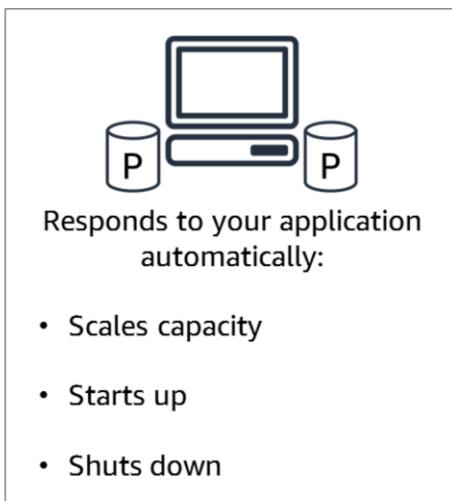
An Amazon Aurora *DB cluster* consists of one or more DB instances and the cluster volume that manages the data for those DB instances.

Two types of DB instances make up an Aurora DB cluster:

- **Primary DB instance** – Supports read and write operations, and performs all the data modifications to the cluster volume. Each Aurora DB cluster has one primary DB instance.
- **Aurora Replica** – Connects to the same storage volume as the primary DB instance and supports only read operations. Each Aurora DB cluster can have up to 15 Aurora replicas in addition to the primary DB instance.

You can choose your DB instance class size and add Aurora replicas to increase read throughput. If your workload changes, you can modify the DB instance class size and change the number of Aurora replicas. This model works well when the database workload is predictable, because you can adjust capacity manually based on the expected workload.

Amazon Aurora Serverless



Pay for the number of Aurora capacity units (ACUs) that are used

Good for intermittent and unpredictable workloads



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

25

However, in some environments, workloads can be intermittent and unpredictable. There can be periods of heavy workloads that might last only a few minutes or hours, and also long periods of light activity, or even no activity. Some examples are retail websites with intermittent sales events, reporting databases that produce reports when needed, development and testing environments, and new applications with uncertain requirements. In these cases and many others, it can be difficult to configure the correct capacity at the right time. It can also result in higher costs when you pay for capacity that isn't used.

[Aurora Serverless](#) is an on-demand, automatically scaling configuration for Amazon Aurora. Aurora Serverless enables your database to automatically start up, shut down, and scale capacity up or down based on your application's needs. It enables you to run your database in the cloud without managing any database instances. Aurora Serverless can be used for infrequent, intermittent, or unpredictable workloads.

You create a database endpoint without specifying the DB instance class size. You specify Aurora capacity units (ACUs). Each ACU is a combination of processing and memory capacity. Database storage automatically scales from 10 Gibibytes (GiB) to 64 Tebibytes (TiB), the same as storage in a standard Aurora DB cluster. The database endpoint connects to a proxy fleet that routes the workload to a fleet of resources. Aurora Serverless scales the resources automatically based on the minimum and maximum capacity specifications.

You pay on a per-second basis for the database capacity that you use when the database is active, and you can migrate between standard and serverless configurations. You pay for the number of ACUs used.

For more information about Aurora Serverless, see [How Aurora Serverless Works](#).

Horizontal scaling: Database sharding

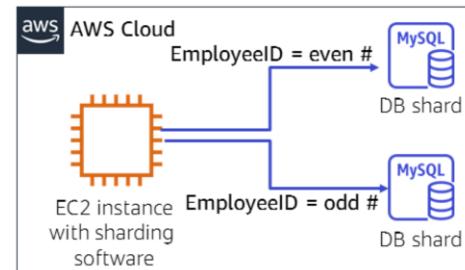
Without shards, all data resides in **one partition**.

- Example: Employee IDs in one database

With **sharding**, data is split into **large chunks (shards)**.

- Example: Even-numbered employee IDs in one database, and odd-numbered employee IDs in another database

In many circumstances, sharding **improves write performance**.



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

24

Sharding, also known as *horizontal partitioning*, is a popular scale-out approach for relational databases that improves write performance.

Sharding is a technique that splits data into smaller subsets and distributes them across a number of physically separated database servers. Each server is referred to as a database *shard*. Database shards usually have the same type of hardware, database engine, and data structure to generate a similar level of performance. However, they have no knowledge of each other, which is the key characteristic that differentiates sharding from other scale-out approaches, such as database clustering or replication.

A sharded database architecture offers scalability and fault tolerance. With sharding, data can be split across as many database servers as necessary. Further, if one database shard has a hardware issue or goes through failover, no other shards are impacted because the single point of failure or slowdown is physically isolated. The drawback to this approach is that because the data is spread across shards, data mapping and routing logic must be specifically engineered to read or join data from multiple shards. These types of queries can incur higher latency compared to a nonsharded database.

For more information about sharding with Amazon RDS, read this [AWS Database Blog post](#).

Scaling with Amazon DynamoDB: On-Demand

On-Demand

Pay per request



No more provisioning

Use case: Spiky, unpredictable workloads.
Rapidly accommodates to need.



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

25

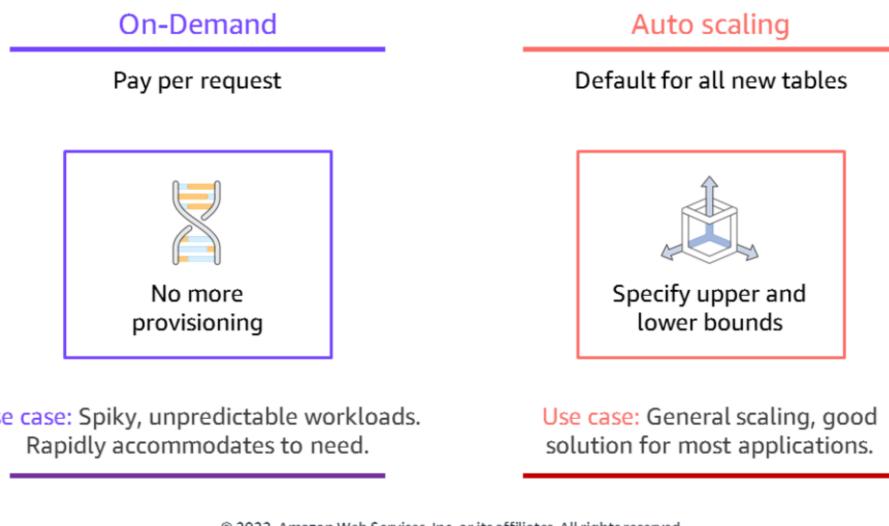
For unpredictable workloads that require a nonrelational database, Amazon DynamoDB On-Demand is a flexible billing option for DynamoDB. It can serve thousands of requests per second without capacity planning. It has a pay-per-request pricing model, instead of a provisioned pricing model.

DynamoDB On-Demand can observe any increase or scale of traffic level. If the level of traffic hits a new peak, DynamoDB adapts rapidly to accommodate the workload. This feature is useful if your workload is difficult to predict or has large spikes over a short duration.

You can change a table from provisioned capacity to on-demand once per day. You can go from on-demand capacity to provisioned as often as you want.

To learn more about Amazon DynamoDB On-Demand, read this [AWS News Blog post](#).

Scaling with Amazon DynamoDB: Auto scaling



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

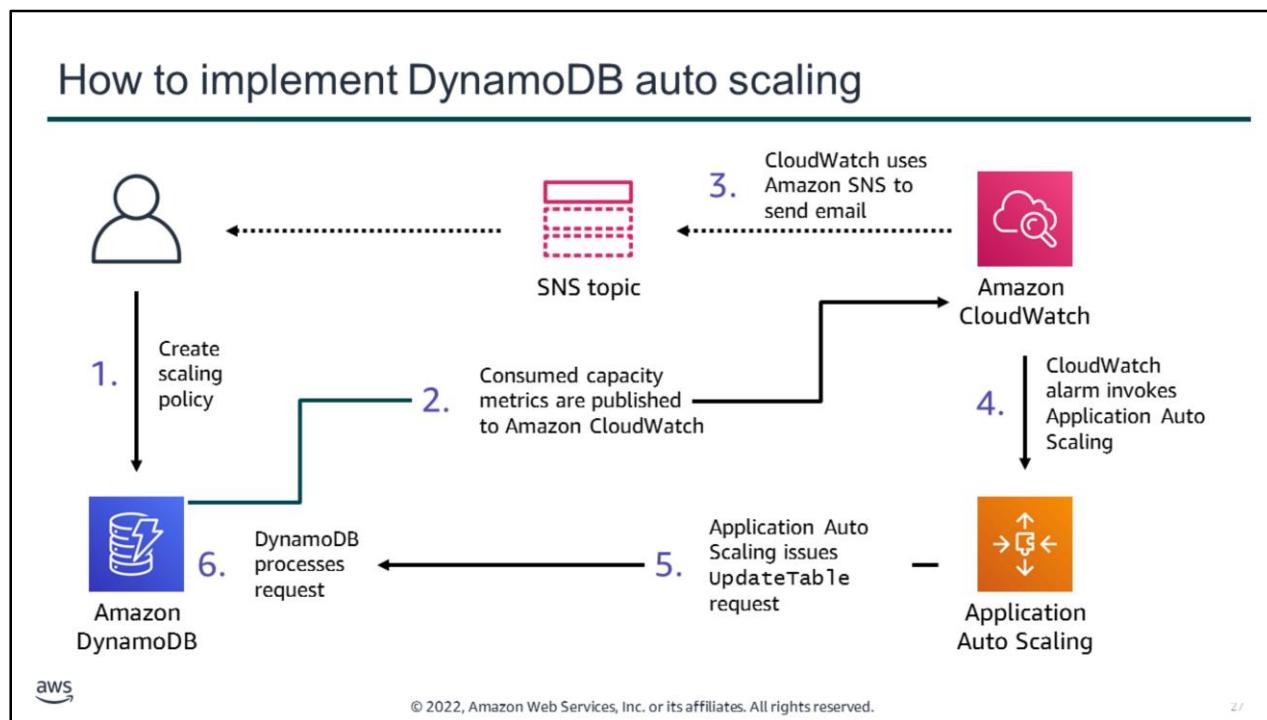
26

Amazon DynamoDB also has a feature called *auto scaling* that's enabled by default. Amazon DynamoDB auto scaling automatically adjusts read and write throughput capacity in response to dynamically changing request volumes, with zero downtime. With DynamoDB auto scaling, you set your desired throughput utilization target and minimum and maximum limits—DynamoDB auto scaling handles the rest.

DynamoDB auto scaling works with Amazon CloudWatch to continuously monitor actual throughput consumption. It automatically scales capacity up or down when actual utilization deviates from your target.

The use of DynamoDB auto scaling incurs no additional cost, beyond what you already pay for DynamoDB and CloudWatch alarms.

For more information about DynamoDB auto scaling, see [Managing Throughput Capacity Automatically with DynamoDB Auto Scaling](#).



Amazon DynamoDB auto scaling uses the *Application Auto Scaling* service to dynamically adjust provisioned throughput capacity on your behalf, in response to actual traffic patterns. This service enables a table or a global secondary index (GSI) to increase its provisioned read and write capacity to handle sudden increases in traffic, without throttling. When the workload decreases, Application Auto Scaling decreases the throughput so that you don't pay for unused provisioned capacity.

To implement DynamoDB auto scaling, perform the following steps:

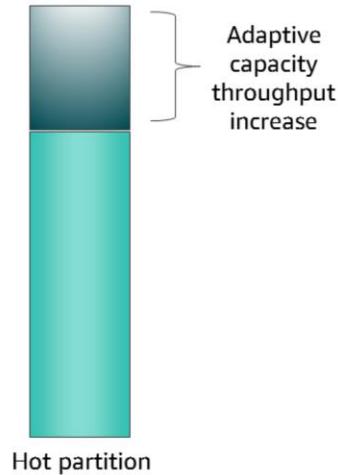
1. Create a *scaling policy* for your DynamoDB table or GSI. The scaling policy specifies whether you want to scale read capacity or write capacity (or both), and the minimum and maximum provisioned capacity unit settings for the table or index.
2. DynamoDB publishes consumed capacity metrics to Amazon CloudWatch.
3. If the table's consumed capacity exceeds your target utilization (or falls below the target) for a specific length of time, Amazon CloudWatch triggers an alarm. You can use Amazon Simple Notification Service (Amazon SNS) to view the alarm in the Amazon CloudWatch console and receive notifications.
4. The CloudWatch alarm invokes Application Auto Scaling to evaluate your scaling policy.
5. Application Auto Scaling issues an *UpdateTable* request to DynamoDB to adjust your table's provisioned throughput.
6. DynamoDB processes the *UpdateTable* request and dynamically increases (or decreases) the table's provisioned throughput capacity so that it approaches your target utilization.

For more information about how to implement DynamoDB auto scaling, [Managing Throughput Capacity Automatically with DynamoDB Auto Scaling](#).

Scaling throughput capacity: DynamoDB adaptive capacity

- Enables reading and writing to hot partitions **without throttling**
- Automatically increases throughput capacity for partitions that receive more traffic*
- Is **enabled automatically** for every DynamoDB table

*Traffic cannot exceed the table's total provisioned capacity or the partition's maximum capacity.



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

28

It's not always possible to distribute read and write activity evenly across partitions all the time. When data access is imbalanced, one partition can receive a higher volume of read and write traffic compared to other partitions (called a *hot partition*). In extreme cases, throttling can occur if a single partition receives more than 3,000 read capacity units (RCUs) or 1,000 write capacity units (WCUs).

To better accommodate uneven access patterns, DynamoDB adaptive capacity enables your application to continue reading and writing to hot partitions without being throttled. The traffic, however, cannot exceed your table's total provisioned capacity or the partition maximum capacity. Adaptive capacity works by automatically increasing throughput capacity for partitions that receive more traffic.

Adaptive capacity is enabled automatically for every DynamoDB table, so you don't need to explicitly enable or disable it.

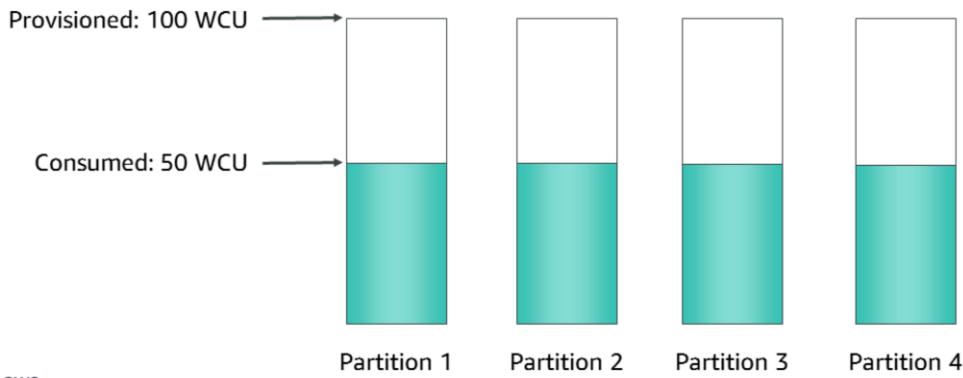
Adaptive capacity example (1 of 3)

Example table with **adaptive** capacity

Total provisioned capacity = 400

WCUs

Total consumed capacity = 200 WCUs



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

29

This diagram illustrates how adaptive capacity works.

The example table is provisioned with 400 WCUs evenly shared across four partitions, so that each partition can sustain up to 100 WCUs per second.

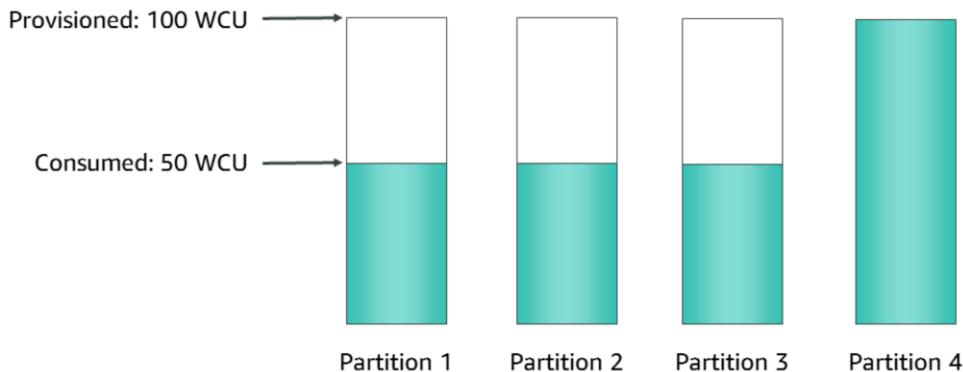
Adaptive capacity example (2 of 3)

Example table with **adaptive** capacity

Total provisioned capacity = 400

WCUs

Total consumed capacity = 250 WCUs



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

50

Partitions 1, 2, and 3 each receive write traffic of 50 WCUs per second. Partition 4 receives 150 WCUs per second. This hot partition can accept write traffic when it still has unused burst capacity, but eventually it throttles traffic that exceeds 100 WCUs per second.

Adaptive capacity example (3 of 3)

Example table with **adaptive** capacity

Total provisioned capacity = 400

WCUs

Total consumed capacity = 300 WCUs

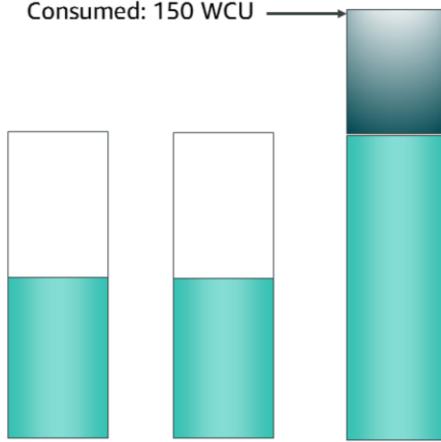
Provisioned: 100 WCU →



Consumed: 50 WCU →



Consumed: 150 WCU →



Adaptive capacity throughput increase

Partition 1

Partition 2

Partition 3

Partition 4



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

31

DynamoDB adaptive capacity responds by increasing the capacity for Partition 4 so that it can sustain the higher workload of 150 WCUs per second without being throttled.

For more information about DynamoDB adaptive capacity, see [Understanding DynamoDB Adaptive Capacity](#).

Adaptive capacity does not fix hot keys and hot partitions

Partition key value	Uniformity
User ID, where the application has many users	Good
Status code, where there are only a few possible status codes	Bad
Item creation date, rounded to the nearest time period (for example, day, hour, or minute)	Bad
Device ID, where each device accesses data at relatively similar intervals	Good
Device ID, where even if many devices are tracked, one is much more popular than all the others	Bad



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

52

The partition key portion of a table's primary key determines the logical partitions where a table's data is stored. This determination in turn affects the underlying physical partitions. Provisioned I/O capacity for the table is divided evenly among these physical partitions. Therefore, a partition key design that doesn't distribute I/O requests evenly can create hot partitions that result in throttling and use your provisioned I/O capacity inefficiently.

The optimal usage of a table's provisioned throughput depends on the workload patterns of individual items, and also on the design of the partition key. It doesn't mean that you must access all partition key values to achieve an efficient throughput level. It doesn't even mean that the percentage of accessed partition key values must be high. It *does* mean that the more distinct partition key values that your workload accesses, the more those requests are spread across the partitioned space. In general, you use your provisioned throughput more efficiently when the ratio of accessed partition key values to total partition key values increases.

The table shows a comparison of the provisioned throughput efficiency of some common partition key schemas.

For more information about designing partition keys, see [Designing Partition Keys to Distribute Your Workload Evenly](#).

Section 3 key takeaways



- You can use [push-button scaling](#) to vertically scale compute capacity for your RDS DB instance
- You can use [read replicas](#) or [shards](#) to horizontally scale your RDS DB instance
- With [Amazon Aurora](#), you can choose the DB instance class size and number of Aurora replicas (up to 15)
- [Aurora Serverless](#) scales resources automatically based on the minimum and maximum capacity specifications
- Amazon DynamoDB [On-Demand](#) offers a pay-per-request pricing model
- DynamoDB [auto scaling](#) uses Amazon Application Auto Scaling to dynamically adjust provisioned throughput capacity
- DynamoDB [adaptive capacity](#) works by automatically increasing throughput capacity for partitions that receive more traffic

© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

33

Some key takeaways from this section of the module include:

- You can use push-button scaling to vertically scale compute capacity for your RDS DB instance
- You can use read replicas or shards to horizontally scale your RDS DB instance
- With Amazon Aurora, you can choose the DB instance class size and number of Aurora replicas (up to 15)
- Aurora Serverless scales resources automatically based on the minimum and maximum capacity specifications
- Amazon DynamoDB On-Demand offers a pay-per-request pricing model
- DynamoDB auto scaling uses Amazon Application Auto Scaling to dynamically adjust provisioned throughput capacity
- DynamoDB adaptive capacity works by automatically increasing throughput capacity for partitions that receive more traffic

Section 4: Designing an environment that's highly available

Module 9: Implementing Elasticity, High Availability, and Monitoring



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

Introducing Section 4: Designing an environment that's highly available.

Highly available systems

- Can withstand some measure of degradation while remaining available
- Have minimized downtime
- Require minimal human intervention
- Recover from failure or roll over to secondary source in an acceptable amount of degraded performance time

Percentage of Uptime	Maximum Downtime Per Year	Equivalent Downtime Per Day
90%	36.5 days	2.4 hours
99%	3.65 days	14 minutes
99.9%	8.76 hours	86 seconds
99.99%	52.6 minutes	8.6 seconds
99.999%	5.25 minutes	0.86 seconds



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

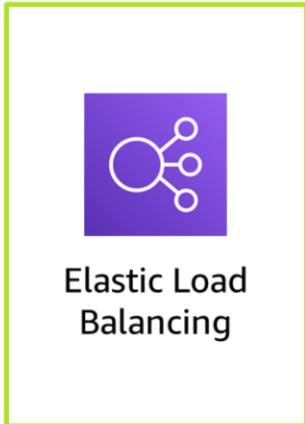
35

It is a best practice to avoid single points of failure when you design and build solutions. To follow this best practice, you want to design your architecture to be highly available.

A *highly available* system is one that can withstand some measure of degradation while remaining available. In a highly available system, downtime is minimized as much as possible, and minimal human intervention is required.

A highly available system enables resiliency in a reactive architecture. A resilient workload can recover when it's stressed by load (more requests for service), attacks, or component failure. A resilient workload recovers from a failure, or it rolls over to a secondary source within an acceptable amount of degraded performance time.

Elastic Load Balancing



A **managed load balancing service** that distributes incoming application traffic across multiple EC2 instances, containers, IP addresses, and Lambda functions.

- Can be **external-facing** or **internal-facing**
- Each load balancer receives a **DNS name**
- Recognizes and responds to **unhealthy instances**



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

36

Elastic Load Balancing is a key component of creating a highly available architecture.

ELB automatically distributes incoming application traffic across multiple targets, such as EC2 instances, containers, IP addresses, and Lambda functions. It can handle the varying load of your application traffic in a single Availability Zone or across multiple Availability Zones.

ELB offers three types of load balancers. Load balancers can be external-facing and distribute inbound public traffic. They can also be internal-facing and distribute private traffic.

Each load balancer receives a default Domain Name System (DNS) name.

Finally, ELB automatically distributes incoming traffic across multiple targets in multiple Availability Zones and only sends traffic to healthy targets. To discover the availability of your EC2 instances, the load balancer periodically sends pings, attempts connections, or sends requests to test the EC2 instances. These tests are called *health checks*. Each registered EC2 instance must respond to the target of the health check with an HTTP status code 200 to be considered healthy by your load balancer.

Types of load balancers		
Application Load Balancer	Network Load Balancer	Classic Load Balancer
 <ul style="list-style-type: none">• Flexible application management• Advanced load balancing of HTTP and HTTPS traffic• Operates at the request level (Layer 7)	 <ul style="list-style-type: none">• Ultra-high performance and static IP address for your application• Load balancing of TCP, UDP, and TLS traffic• Operates at the connection level (Layer 4)	<p>PREVIOUS GENERATION for HTTP, HTTPS, TCP, and SSL</p> <ul style="list-style-type: none">• Load balancing across multiple EC2 instances• Operates at both the request level and connection level



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

37

ELB offers three types of load balancers that all feature the high availability, automatic scaling, and robust security needed to make your applications fault-tolerant.

- An *Application Load Balancer* operates at the application level (Open Systems Interconnection, or OSI, model layer 7). It routes traffic to targets—EC2 instances, containers, IP addresses, and Lambda functions—based on the content of the request. It works well for the advanced load balancing of HTTP and Secure HTTP (HTTPS) traffic. An Application Load Balancer provides advanced request routing that is targeted at the delivery of modern application architectures, including microservices and container-based applications. An Application Load Balancer simplifies and improves the security of your application by ensuring that the latest Secure Sockets Layer/Transport Layer Security (SSL/TLS) ciphers and protocols are used at all times.
- A *Network Load Balancer* operates at the network transport level (OSI model layer 4), routing connections to targets such as EC2 instances, containers, and IP addresses. It works well for load balancing both Transmission Control Protocol (TCP) and User Datagram Protocol (UDP) traffic. A Network Load Balancer can handle millions of requests per second, while maintaining ultra-low latencies. A Network Load Balancer is optimized to handle sudden and volatile network traffic patterns.
- A *Classic Load Balancer* provides basic load balancing across multiple EC2 instances, and it operates at both the application level and network transport level. A Classic Load Balancer supports the load balancing of applications that use HTTP, HTTPS, TCP, and SSL. The Classic Load Balancer is an older implementation. When possible, AWS recommends that you use a dedicated Application Load Balancer or Network Load Balancer.

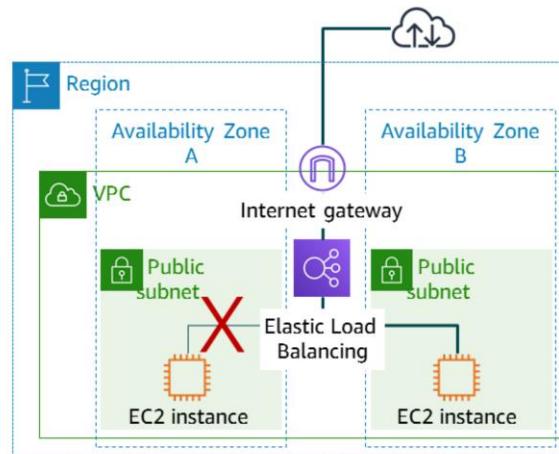
Using VPC peering, you can access internal load balancers (including Classic Load Balancers, Application Load Balancers, and Network Load Balancers) from another VPC. VPC peering is available for intra-Region and inter-Region connectivity for local or cross-account VPCs.

To learn more about the differences between the three types of load balancers, see *Product comparisons* on the [Elastic Load Balancing features page](#).

Implementing high availability

Start with two Availability Zones per AWS Region.

If resources in one Availability Zone are unreachable, your application shouldn't fail.



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

58

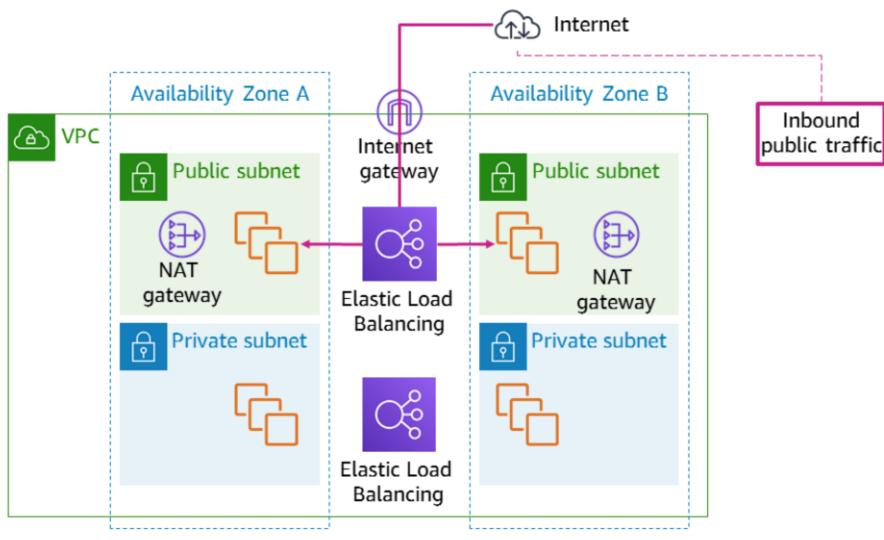
To build a highly available application, it is a best practice to launch resources in multiple Availability Zones and use a load balancer to distribute traffic among those resources. Running your applications across multiple Availability Zones provides greater availability if there is a failure in a data center.

In the basic pattern here, two web servers run on EC2 instances that are in different Availability Zones. The instances are positioned behind an Elastic Load Balancing load balancer that distributes traffic between them. If one server becomes unavailable, the load balancer is configured to stop distributing traffic to the unhealthy instance and start routing traffic to the healthy instance. That way, the application is still available if there is a data center failure in one of the Availability Zones.

Most applications can be designed to support two Availability Zones per AWS Region. If you use data sources that only support primary or secondary failover, then your application might not benefit from the use of more Availability Zones. Because Availability Zones are spread out physically, you won't receive much benefit from duplicating your resources in three or more Availability Zones in one AWS Region.

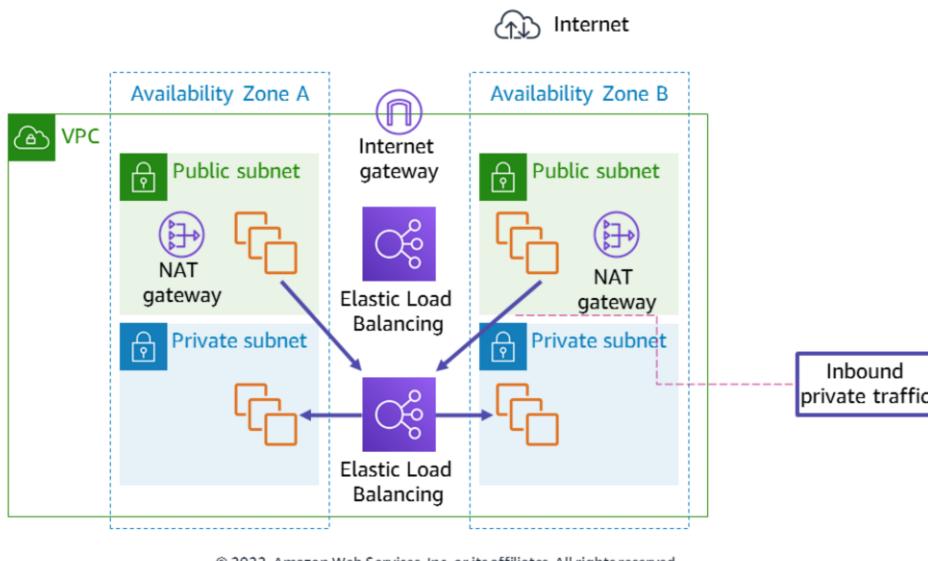
For heavy Amazon EC2 Spot Instance usage—or data sources that go beyond active/passive, such as Amazon DynamoDB—there might be a benefit to using more than two Availability Zones.

Example of a highly available architecture (1 of 3)



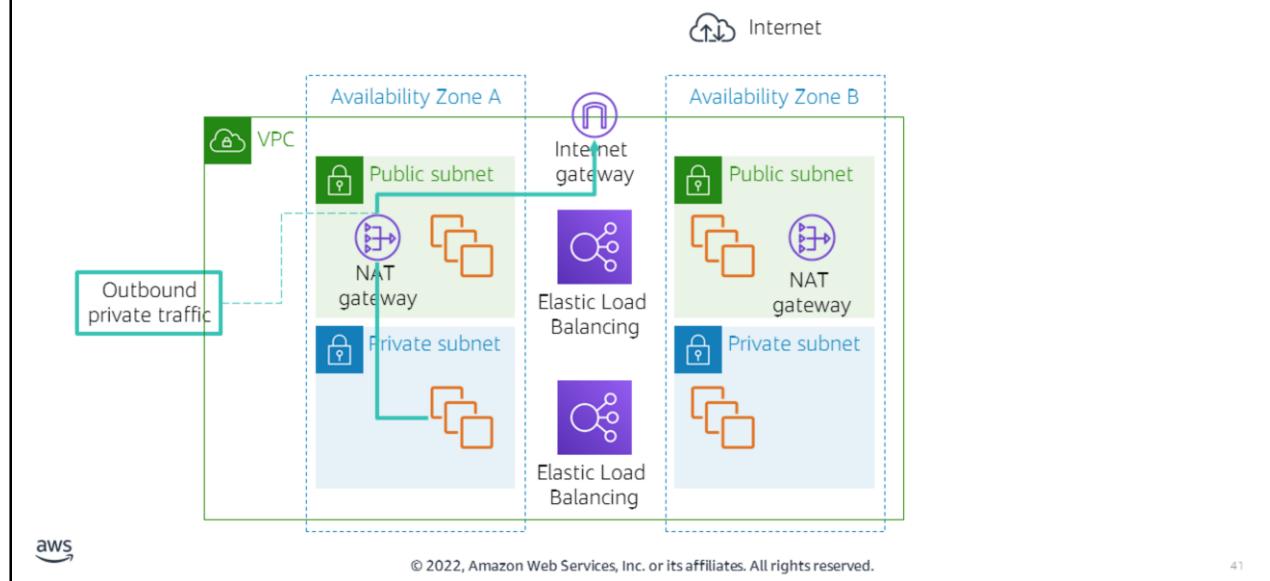
Like the architecture that you just considered, EC2 instances are positioned behind a load balancer, which distributes inbound public traffic between them. If one of the servers becomes unavailable, the load balancer is configured to stop distributing traffic to the unhealthy instances and start routing traffic to the healthy ones.

Example of a highly available architecture (2 of 3)



You can include a second load balancer in your architecture to route inbound traffic from the instances in the public subnets to the instances in the private subnets.

Example of a highly available architecture (3 of 3)



If you have resources in multiple Availability Zones and they share one NAT gateway—and if the NAT gateway's Availability Zone is down—resources in the other Availability Zones lose internet access. It's a best practice to have NAT gateways in both Availability Zones to ensure high availability.

Demonstration: Creating a Highly Available Web Application

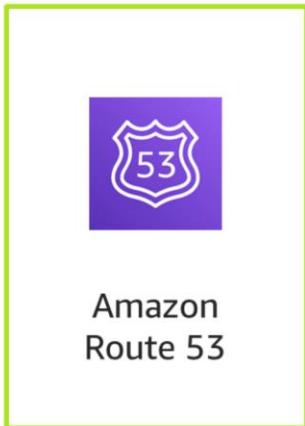


© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

42

Now, the educator might choose to demonstrate how to create a highly available web application by deploying web servers behind an Application Load Balancer across multiple Availability Zones.

Amazon Route 53



Amazon Route 53 is a highly available and scalable cloud DNS service.

- Translates domain names into IP addresses
- Connects user requests to infrastructure that runs inside and outside of AWS
- Can be configured to route traffic to healthy endpoints, or to monitor the health of your application and its endpoints
- Offers registration for domain names
- Has multiple routing options



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

44

You can also implement multi-Region high availability and fault tolerance in your network architecture by using Amazon Route 53.

Amazon Route 53 is a highly available and scalable cloud DNS service. The service is designed to provide a reliable and cost-effective way to route users to internet applications. It translates names like *example.com* into the numeric IP addresses (such as *192.0.2.1*) that computers use to connect to each other.

Route 53 effectively connects user requests to infrastructure that runs in AWS—such as EC2 instances, ELB load balancers, or S3 buckets. You can also use Route 53 to route users to infrastructure outside of AWS.

You can use Route 53 to configure DNS health checks to route traffic to healthy endpoints, or to independently monitor the health of your application and its endpoints.

Route 53 also offers domain name registration. You can purchase and manage domain names such as *example.com*, and Amazon Route 53 automatically configures DNS settings for your domains.

Amazon Route 53 offers various routing options, which can be combined with DNS failover to enable low-latency, fault-tolerant architectures. For details about Amazon Route 53 routing

options, see [Choosing a Routing Policy](#).

Amazon Route 53 supported routing

- Simple routing
- Weighted round robin routing
- Latency-based routing
- Geolocation routing
- Geoproximity routing
- Failover routing
- Multivalue answer routing



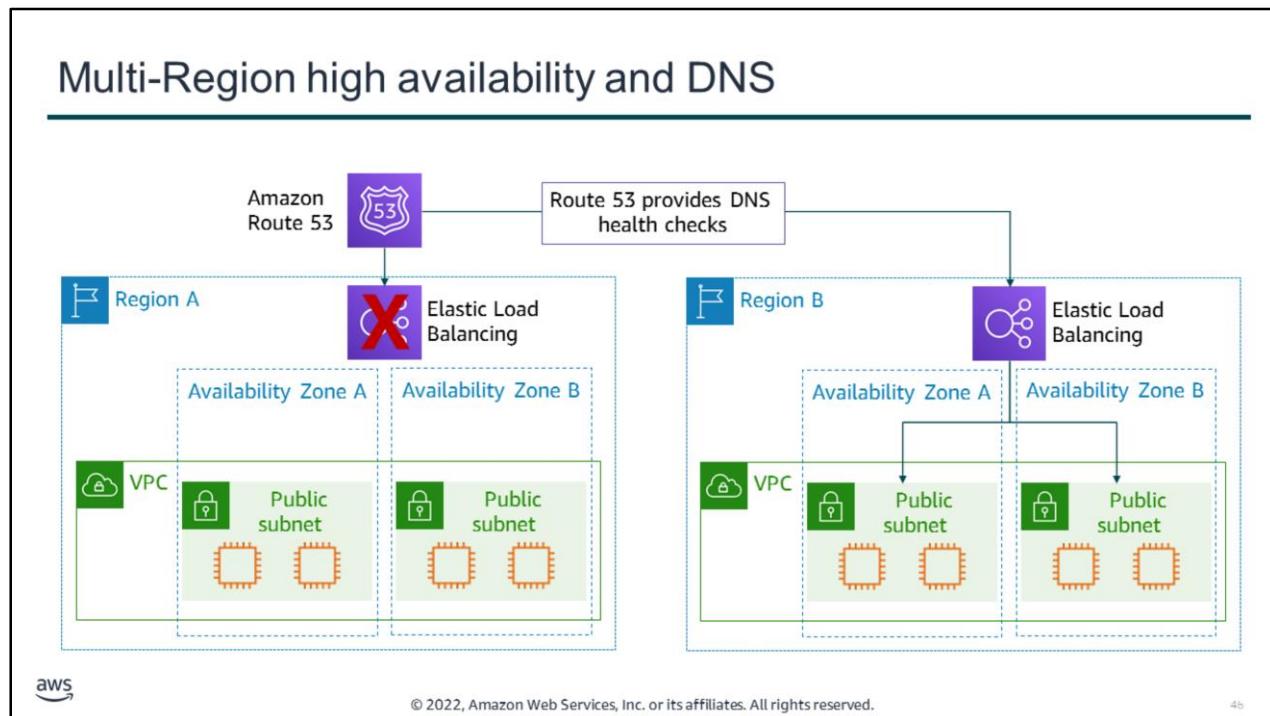
© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

45

Amazon Route 53 supports several types of routing policies, which determine how Amazon Route 53 responds to queries:

- *Simple routing (round robin)* – Distributes the number of requests as evenly as possible between all participating servers.
- *Weighted round robin routing* – Enables you to assign weights to resource record sets to specify the frequency that different responses are served at. You might want to use this capability to do A/B testing, which is when you send a small portion of traffic to a server where you made a software change. For instance, suppose you have two record sets that are associated with one DNS name: one with weight 3 and one with weight 1. In this case, 75 percent of the time, Amazon Route 53 will return the record set with weight 3, and 25 percent of the time, Amazon Route 53 will return the record set with weight 1. Weights can be any number between 0 and 255.
- *Latency-based routing (LBR)* – Use when you have resources in multiple AWS Regions and you want to route traffic to the Region that provides the best latency. Latency routing works by routing your customers to the AWS endpoint (for example, EC2 instances, Elastic IP addresses, or load balancers) that provides the fastest experience based on actual performance measurements of the different AWS Regions where your application runs.
- *Geolocation routing* – Enables you to choose the resources that serve your traffic based on the geographic location of your users (the origin of the DNS queries). When you use geolocation routing, you can localize your content and present some or all of your website in the language of your users. You can also use geolocation routing to restrict the distribution of content to only the locations where you have distribution rights. Another possible use is for balancing the load across endpoints in a predictable, easy-to-manage way, so that each user location is consistently routed to the same endpoint.

- *Geoproximity routing* – Enables you to route traffic based on the physical distance between your users and your resources, if you are using Route 53 Traffic Flow. You can also route more or less traffic to each resource by specifying a positive or negative bias. When you create a traffic flow policy, you can specify either an AWS Region (if you are using AWS resources) or the latitude and longitude for each endpoint.
- *Failover routing (DNS failover)* – Use when you want to configure active-passive failover. Route 53 can help detect when your website has an outage, and redirect your users to alternate locations where your application is operating properly. When you enable this feature, Route 53 health-checking agents monitor each location or endpoint of your application to determine its availability. You can use this feature to increase the availability of your customer-facing application.
- *Multivalue answer routing* – Use if you want to route traffic approximately randomly to multiple resources, such as web servers. You can create one multivalue answer record for each resource. You can also optionally associate a Route 53 health check with each record. For example, suppose that you manage an HTTP web service with 12 web servers that each have their own IP address. No one web server could handle all of the traffic. However if you create a dozen multivalue answer records, Route 53 responds to DNS queries with up to eight healthy records in response to each DNS query. Route 53 gives different answers to different DNS resolvers. If a web server becomes unavailable after a resolver caches a response, client software can try another IP address in the response.



With *DNS failover routing*, Route 53 can help detect an outage of your website and redirect your users to alternate locations where your application is operating properly. When you enable this feature, Route 53 health-checking agents monitor each location or endpoint of your application to determine its availability. You can use this feature to increase the availability of your customer-facing application.

Demonstration: Amazon Route 53



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

47

Now, the educator might choose to play a video that demonstrates simple routing, failover routing, and geolocation routing with Route 53.

Section 4 key takeaways



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

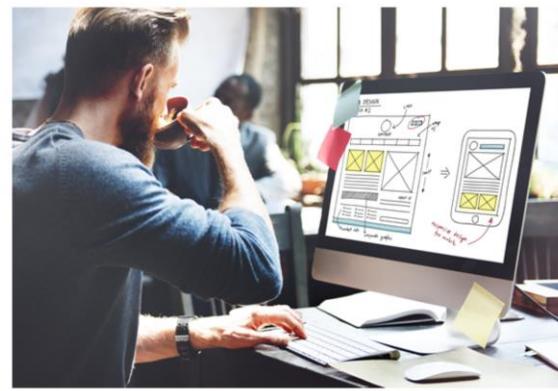
48

- You can design your network architectures to be highly available and avoid single points of failure
- Route 53 offers various routing options that can be combined with DNS failover to enable low-latency, fault-tolerant architectures

Some key takeaways from this section of the module include:

- You can design your network architectures to be highly available and avoid single points of failure
- Route 53 offers a variety of routing options that can be combined with DNS failover to enable a variety of low-latency, fault-tolerant architectures

Module 9 – Guided Lab: Creating a Highly Available Environment



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

49

You now complete the Module 9 – Guided Lab: Creating a Highly Available Environment.

Guided lab: Tasks

1. Inspect a provided VPC
2. Create an Application Load Balancer
3. Create an Auto Scaling group
4. Test the application for high availability



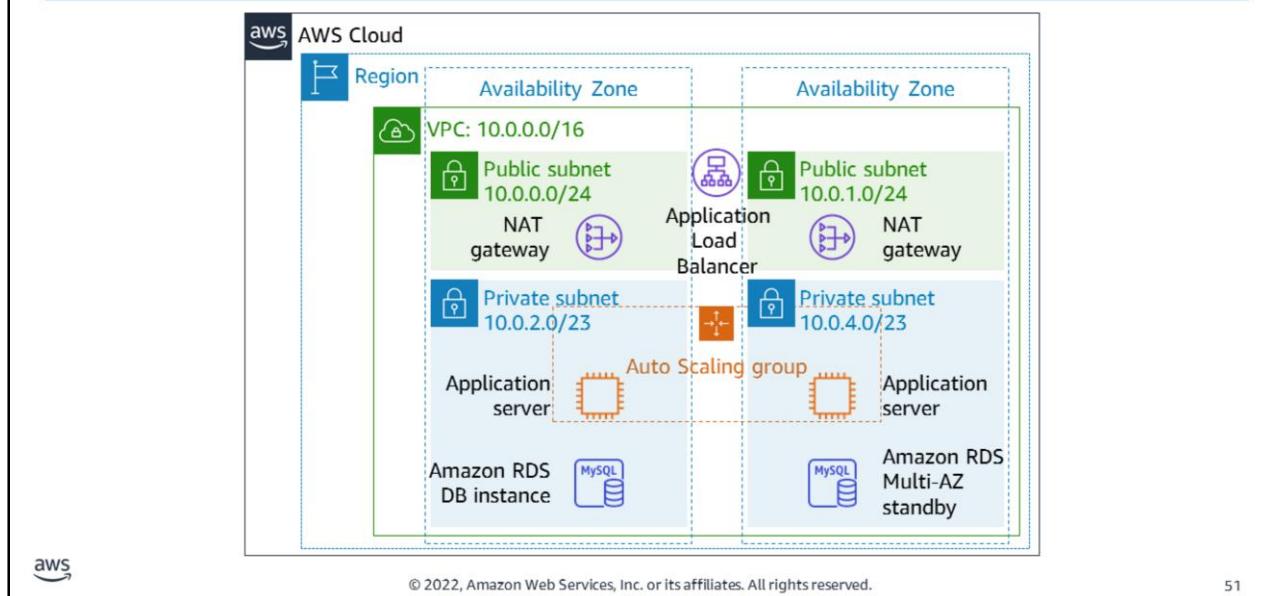
© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

50

In this guided lab, you will complete the following tasks:

1. Inspect a provided VPC
2. Create an Application Load Balancer
3. Create an Auto Scaling group
4. Test the application for high availability

Guided lab: Final product



The diagram summarizes what you are going to build in the lab.



A timer icon indicating a duration of ~ 40 minutes.

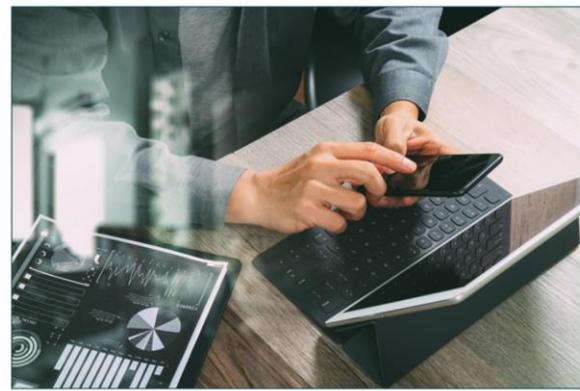
Begin Module 9 – Guided Lab: Creating a Highly Available Environment

aws

© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

It is now time to start the guided lab.

Guided lab debrief: Key takeaways



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

53

The educator might choose to lead a conversation about the key takeaways from the guided lab after you have completed it.

Section 5: Monitoring

Module 9: Implementing Elasticity, High Availability, and Monitoring



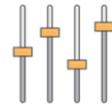
© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

Introducing Section 5: Monitoring.

Monitoring usage, operations, and performance



Operational Health



Resource Utilization



Application Performance



Security Auditing



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

55

Monitoring is an essential component of a reactive architecture.

Monitoring can help you:

- Track how your resources are operating and performing
- Track resource utilization and application performance to make sure that your infrastructure is meeting demand
- Decide what permissions to set for your AWS resources to achieve the security goals that you want

Monitoring your costs

To create a more flexible and elastic architecture, you should know where you are spending money.

AWS Cost Explorer



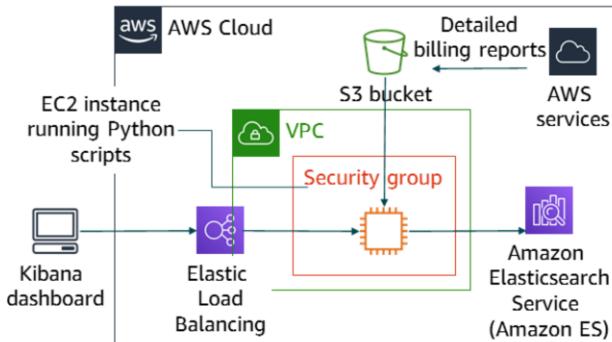
AWS Budgets



AWS Cost and Usage Report



Cost Optimization Monitor



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

56

Monitoring can also help you to understand and manage the cost of your AWS infrastructure. AWS provides monitoring and reporting tools, including the following:

- [AWS Cost Explorer](#) helps you visualize, understand, and manage your AWS costs and usage with daily or monthly granularity. You can use it to view data up to the last 13 months, which enables you to see patterns in how you spend AWS resources over time.
- [AWS Budgets](#) enables you to set custom budgets that alert you when your costs or usage exceed (or are forecasted to exceed) your budgeted amount.
- [AWS Cost and Usage Report](#) contains the most comprehensive set of AWS cost and usage data available, including additional metadata about AWS services, pricing, and reservations.
- The [Cost Optimization Monitor](#) is a solution architecture that automatically processes detailed billing reports to provide granular metrics that you can search, analyze, and visualize in a dashboard that you can customize. The solution provides you with insight into service usage and cost, which you can break down by period, account, resource, or tags.

To learn more about how you can monitor the cost of your AWS infrastructure, see [AWS Cost Management Services](#).

Amazon CloudWatch



- Collects and tracks metrics for your resources and applications
- Helps you correlate, visualize, and analyze metrics and logs
- Enables you to create alarms and detect anomalous behavior
- Can send notifications or make changes to resources that you are monitoring



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

57

Amazon CloudWatch is a monitoring and observability service that is built for DevOps engineers, developers, site reliability engineers, and information technology managers. CloudWatch provides you with data and actionable insights to monitor your applications, respond to system-wide performance changes, optimize resource utilization, and get a unified view of operational health.

You can use CloudWatch to collect and track metrics, which are variables that you can measure for your resources and applications. You can create CloudWatch alarms that watch metrics and send notifications. Also, when a threshold is breached, CloudWatch can automatically change the resources that you are monitoring.

For example, you can monitor the CPU usage and the disk reads and writes of your EC2 instances. You can then use this data to determine whether you should launch additional instances to handle increased load. You can also use this data to stop under-used instances to save money.

In addition to monitoring the built-in metrics that come with AWS, you can monitor your own custom metrics. With CloudWatch, you gain system-wide visibility into resource utilization, application performance, and operational health.

For more information about CloudWatch, see [What is Amazon CloudWatch?](#)

How CloudWatch responds



Metrics



Logs



Alarms



Events



Rules



Targets



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

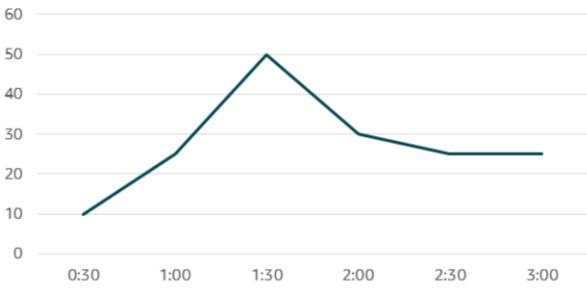
58

You can use several CloudWatch components to monitor your resources and applications, and to respond to events.

CloudWatch metrics

-  Metrics
-  Logs
-  Alarms
-  Events
-  Rules
-  Targets

Average CPU Utilization



Time	Average CPU Utilization
0:30	10
1:00	25
1:30	50
2:00	30
2:30	25
3:00	25

Metric data is kept for 15 months

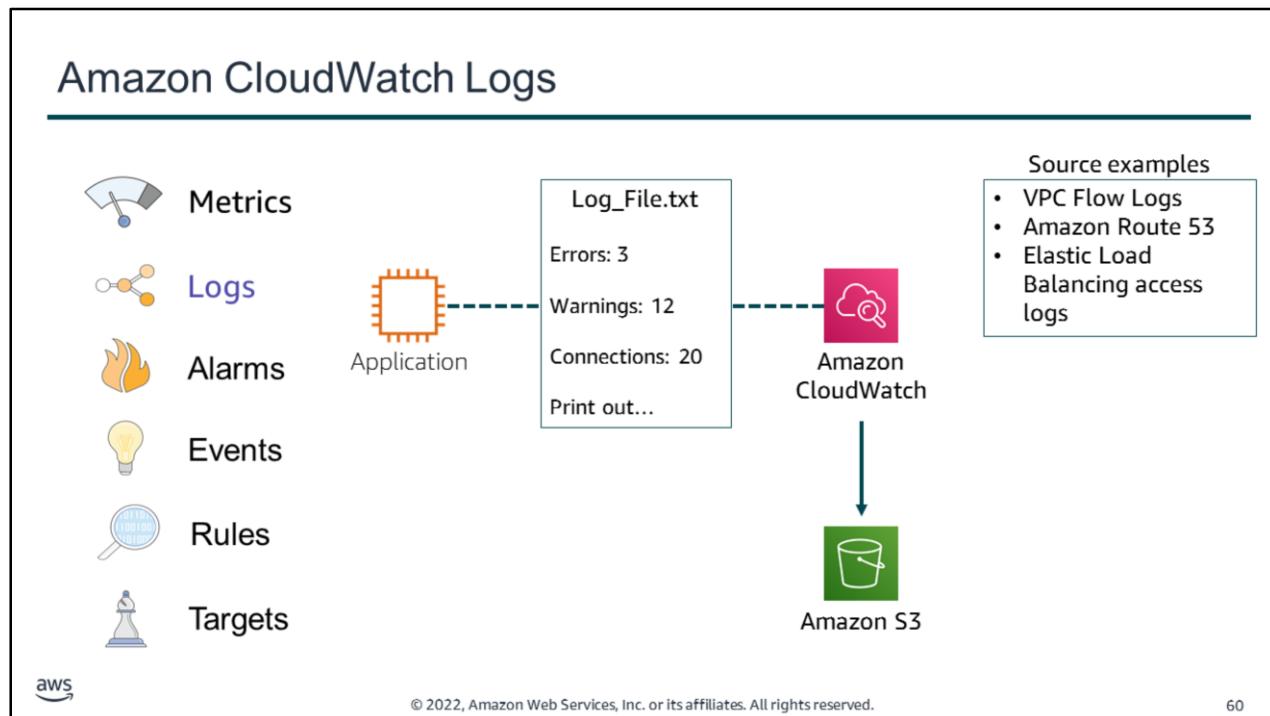
© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

59

Metrics are data about the performance of your systems. By default, many AWS services provide metrics for resources, such as EC2 instances, Amazon Elastic Block Store (Amazon EBS) volumes, and Amazon RDS DB instances. You can also enable detailed monitoring of some resources, such as your EC2 instances, or publish your own application metrics. CloudWatch can load all the metrics in your account (both AWS resource metrics and application metrics that you provide) for search, graphing, and alarms.

Metric data is kept for 15 months, which enables you to view both up-to-the-minute data and historical data.

For more information about metrics, see [Using Amazon CloudWatch Metrics](#).



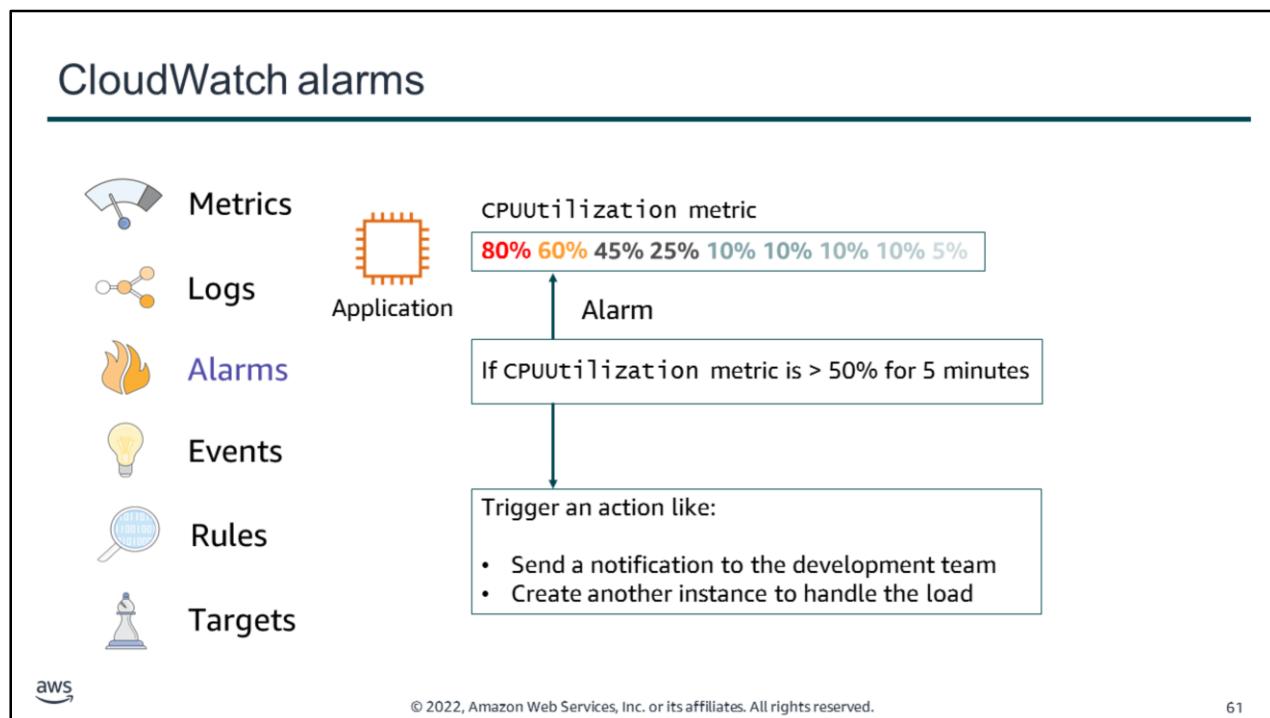
You can use Amazon CloudWatch Logs to monitor, store, and access your log files from sources such as EC2 instances, AWS CloudTrail, Route 53, and other AWS services.

For example, you can use CloudWatch Logs to monitor applications and systems by using log data. CloudWatch Logs can track the number of errors that occur in your application logs. It sends a notification when the rate of error exceeds a threshold that you specify.

Additionally, you can use CloudWatch Logs Insights to analyze your logs in seconds. It gives you fast, interactive queries and visualizations. You can use line or stacked area charts to visualize query results, and add those queries to a CloudWatch Dashboard.

For more information about CloudWatch Logs, see the following resources:

- [Amazon CloudWatch Logs](#)
- [Amazon CloudWatch Logs Insights](#)



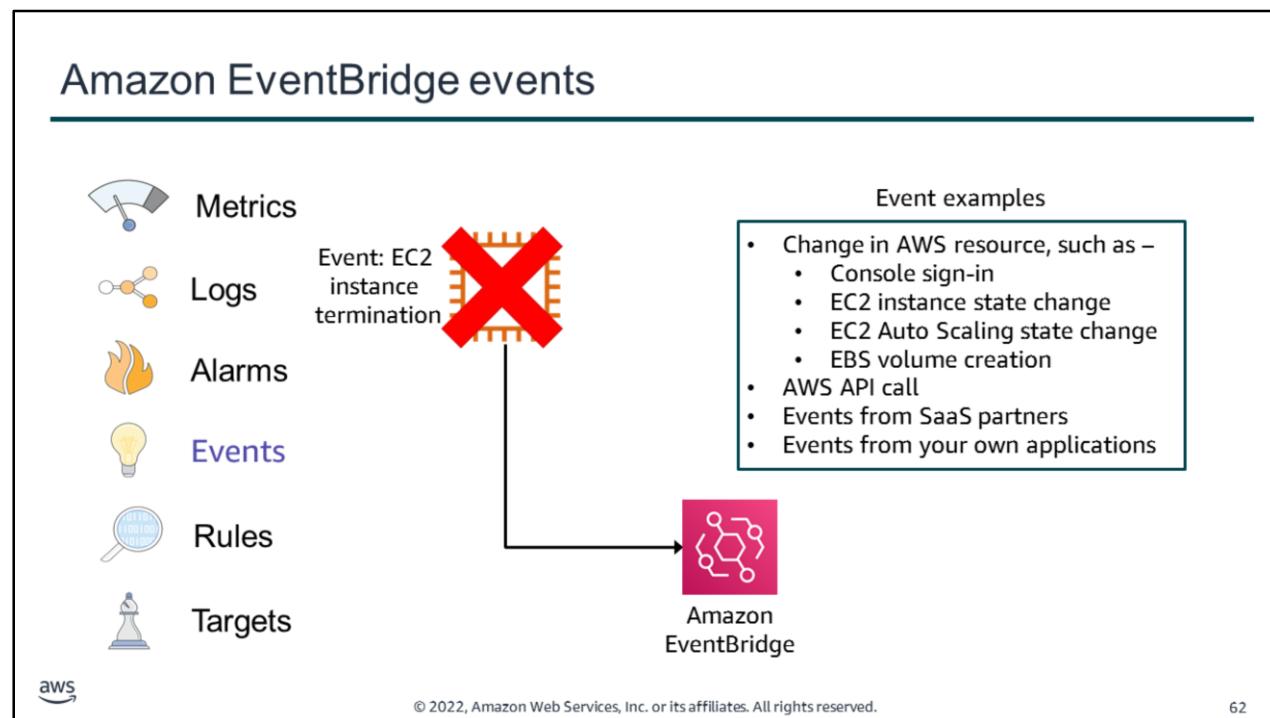
You can use an alarm to automatically initiate actions on your behalf. An *alarm* watches a single metric over a specified time period. It performs one or more specified actions, based on the value of the metric relative to a threshold over time. The action is a notification that is sent to an Amazon SNS topic or an Auto Scaling policy. You can also add alarms to dashboards.

Alarms invoke actions for sustained state changes only. CloudWatch alarms do not invoke actions merely because they are in a particular state. The state must have changed and been maintained for a specified number of periods.

In the example that is shown, an alarm is triggered when the [CPUUtilization metric](#) (that is, the percentage of allocated EC2 compute units that are currently in use on the instance) is greater than 50 percent for 5 minutes. The alarm triggers an action, such as running an Auto Scaling policy or sending a notification to the development team.

Actions can also be run when an alarm is *not* triggered.

For more information about CloudWatch alarms, see [Using Amazon CloudWatch Alarms](#).

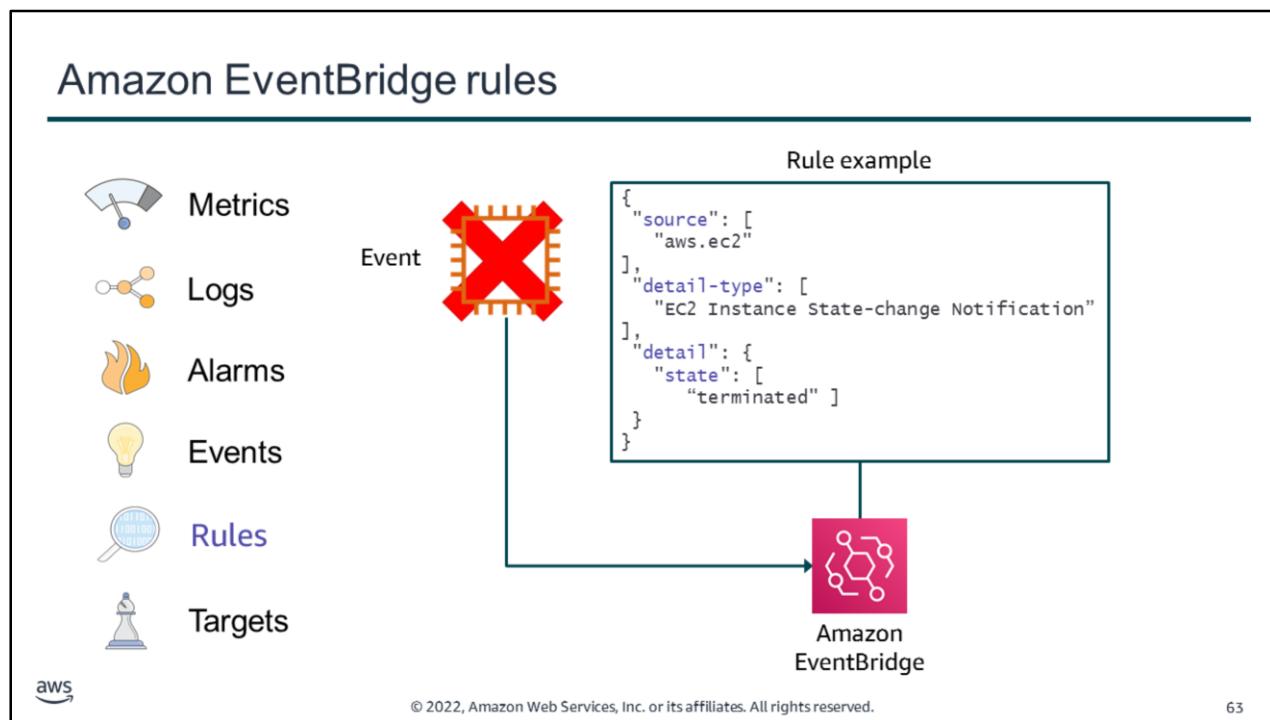


Amazon EventBridge (formerly called Amazon CloudWatch Events) ingests a stream of real-time data from your own applications, software-as-a-service (SaaS) applications, and AWS services. It then routes that data to targets, such as AWS Lambda.

An *event* indicates a change in an environment. It can be an AWS environment, a SaaS partner service or application, or one of your own custom applications or services. For example, Amazon EC2 generates an event when the state of an EC2 instance changes from *pending* to *running*, and Amazon EC2 Auto Scaling generates events when it launches or terminates instances. AWS CloudTrail publishes events when you make API calls. You can also set up scheduled events that are generated on a periodic basis.

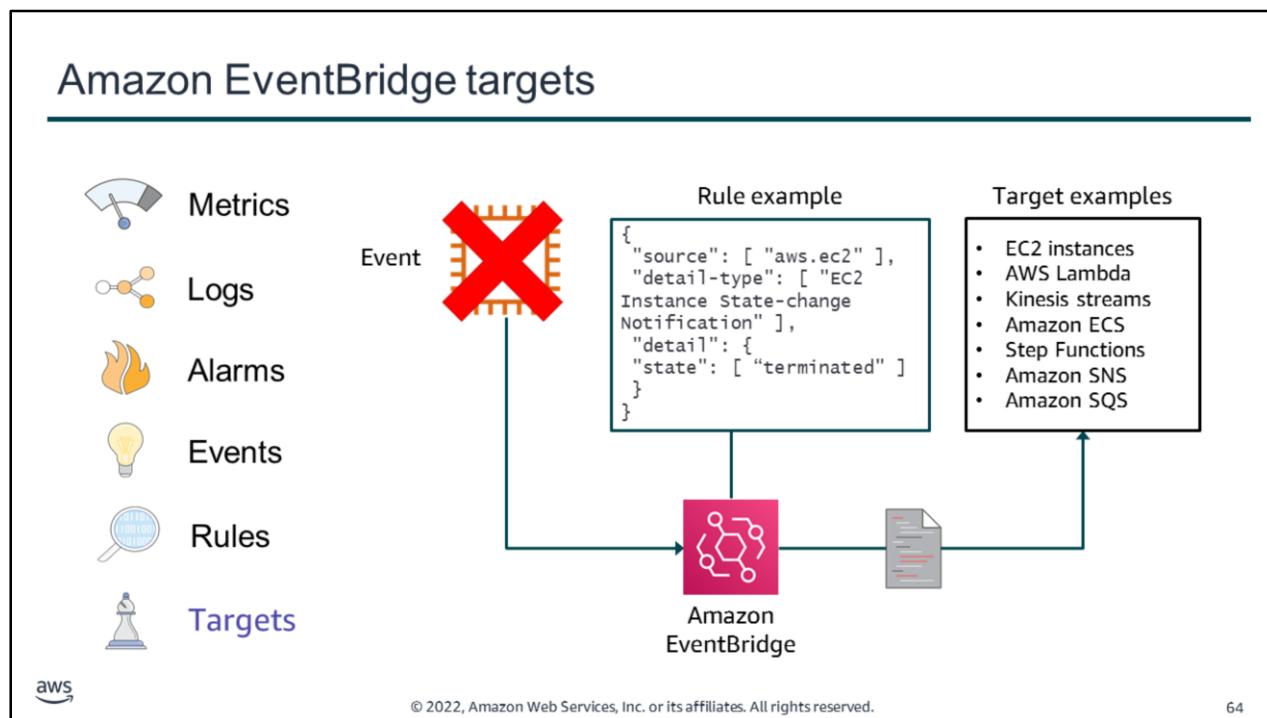
Existing CloudWatch Events users can access their existing default bus, rules, and events in the new EventBridge console and in the CloudWatch Events console. EventBridge uses the same CloudWatch Events API, so all your existing CloudWatch Events API usage remains the same.

For more information about EventBridge, see [What Is Amazon EventBridge?](#)



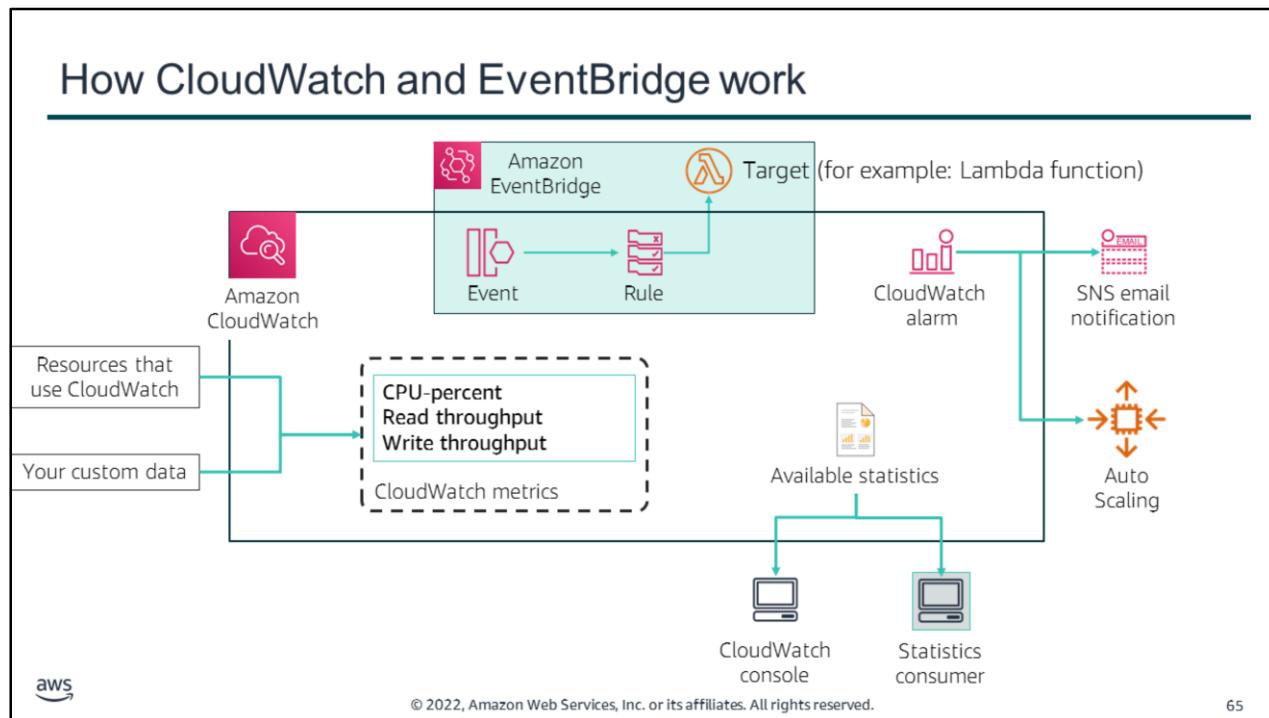
You can set up routing rules to determine where to send your data to build application architectures that react—in real time—to all your data sources.

A *rule* matches incoming events and routes them to targets for processing. A single rule can route to multiple targets, which are all processed in parallel. Rules are not processed in a particular order. Thus, different parts of an organization can look for and process the events that are of interest to them. A rule can customize the JavaScript Object Notation (JSON) that is sent to the target by passing only certain parts, or by overwriting it with a constant.



A *target* processes events. Targets can include EC2 instances, Lambda functions, Amazon Kinesis streams, Amazon Elastic Container Service (Amazon ECS) tasks, AWS Step Functions state machines, SNS topics, Amazon Simple Queue Service (Amazon SQS) queues, and built-in targets. A target receives events in JSON format.

When you create a rule, you associate it with a specific *event bus*, and the rule is matched only to events received by that event bus.



This architecture shows how CloudWatch and EventBridge work, at a high level.

CloudWatch acts as a metrics repository. An AWS service—such as Amazon EC2—puts metrics into the repository, and you retrieve statistics based on those metrics. If you put your own custom metrics into the repository, you can also retrieve statistics on these metrics.

You can use the metrics to calculate statistics, and then present the data graphically in the CloudWatch console.

You can create a rule in EventBridge that matches incoming events and routes them to targets for processing.

You can configure alarm actions to stop, start, or terminate an EC2 instance when certain criteria are met. In addition, you can create alarms that initiate Amazon EC2 Auto Scaling and Amazon SNS actions on your behalf.

Section 5 key takeaways



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

66

- AWS Cost Explorer, AWS Budgets, AWS Cost and Usage Report, and the Cost Optimization Monitor can help you understand and manage the cost of your AWS infrastructure.
- CloudWatch collects monitoring and operational data in the form of logs, metrics, and events. It visualizes the data by using automated dashboards so you can get a unified view of your AWS resources, applications, and services that run in AWS and on-premises.
- EventBridge is a serverless event bus service that connects your applications with data from various sources. EventBridge delivers a stream of real-time data from your own applications, SaaS applications, and AWS services. It then routes that data to targets.

Some key takeaways from this section of the module include:

- AWS Cost Explorer, AWS Budgets, AWS Cost and Usage Report, and the Cost Optimization Monitor can help you understand and manage the cost of your AWS infrastructure.
- CloudWatch collects monitoring and operational data in the form of logs, metrics, and events. It visualizes the data by using automated dashboards so you can get a unified view of your AWS resources, applications, and services that run in AWS and on-premises.
- EventBridge is a serverless event bus service that makes it easy to connect your applications with data from a variety of sources. EventBridge ingests a stream of real-time data from your own applications, SaaS applications, and AWS services. It then routes that data to targets.

Module 9 – Challenge Lab:

Creating a Scalable and Highly Available Environment for the Café



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

67

You will now complete Module 9 – Challenge Lab: Creating a Scalable and Highly Available Environment for the Café.

The business need: A scalable and highly available environment



- The café will soon be featured in a famous TV food show.
- Sofía and Nikhil want to make sure that the café's website can handle the expected increase in traffic.



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

68

The café will soon be featured in a famous TV food show. When it airs, Sofía and Nikhil anticipate that the café's web server will experience a temporary spike in the number of users—perhaps even up to tens of thousands of users. Currently, the café's web server is deployed in one Availability Zone, and they are worried that it won't be able to handle the expected increase in traffic. They want to ensure that their customers have a great experience when they visit the website, and that they don't experience any issues, such as lags or delays in placing orders.

To ensure this experience, the website must be responsive, scale both up and down to meet fluctuating customer demand, and be highly available. It must also incorporate load balancing. Instead of overloading a single server, the architecture must distribute customer order requests across multiple application servers so it can handle the increase in demand.

Challenge lab: Tasks

1. Creating a NAT gateway for the second Availability Zone
2. Creating a bastion host instance in a public subnet
3. Creating a launch template
4. Creating an Auto Scaling group
5. Creating a load balancer
6. Testing the web application
7. Testing automatic scaling under load



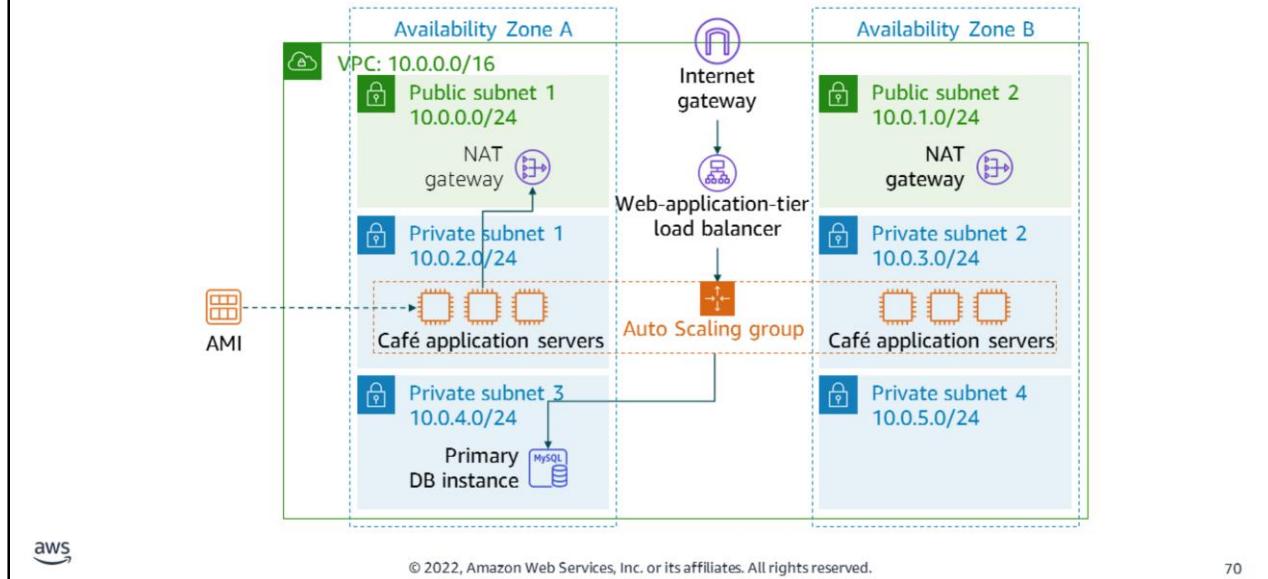
© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

69

In this challenge lab, you will complete the following tasks:

1. Creating a NAT gateway for the second Availability Zone
2. Creating a bastion host instance in a public subnet
3. Creating a launch template
4. Creating an Auto Scaling group
5. Creating a load balancer
6. Testing the web application
7. Testing automatic scaling under load

Challenge lab: Final product



The diagram summarizes what you will have built after you complete the lab.



A timer icon with the text "~ 90 minutes" indicating the duration of the challenge lab.

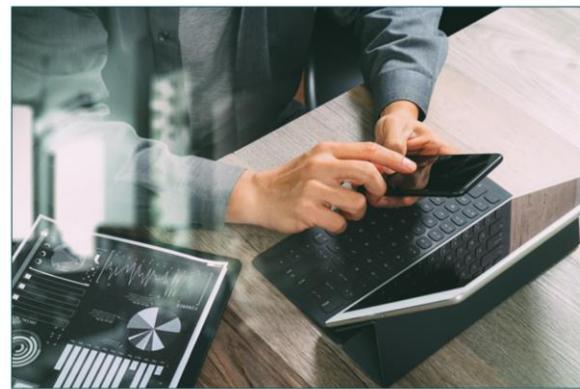
Begin Module 9 – Challenge Lab: Creating a Scalable and Highly Available Environment for the Café

aws

© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

It is now time to start the challenge lab.

Challenge lab debrief: Key takeaways



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

72

Your educator might choose to lead a conversation about the key takeaways from this challenge lab after you have completed it.

Module wrap-up

Module 9: Implementing Elasticity, High Availability, and Monitoring



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

It's now time to review the module and wrap-up with a knowledge check and discussion of a practice certification exam question.

Module summary

In summary, in this module, you learned how to:

- Use Amazon EC2 Auto Scaling within an architecture to promote elasticity
- Explain how to scale your database resources
- Deploy an Application Load Balancer to create a highly available environment
- Use Amazon Route 53 for DNS failover
- Create a highly available environment
- Design architectures that use Amazon CloudWatch to monitor resources and react accordingly



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

74

In summary, in this module, you learned how to:

- Use Amazon EC2 Auto Scaling within an architecture to promote elasticity
- Explain how to scale your database resources
- Deploy an Application Load Balancer to create a highly available environment
- Use Amazon Route 53 for DNS failover
- Create a highly available environment
- Design architectures that use Amazon CloudWatch to monitor resources and react accordingly

Complete the knowledge check



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

/5

It is now time to complete the knowledge check for this module.



Sample exam question

A web application enables customers to upload orders to an S3 bucket. The resulting Amazon S3 events trigger a Lambda function that inserts a message into an SQS queue. A single EC2 instance reads the messages from the queue, processes them, and stores them in a DynamoDB table partitioned by unique order ID. Next month, traffic is expected to increase by a factor of 10 and a Solutions Architect is reviewing the architecture for possible scaling problems.

Which component is MOST likely to need re-architecting to be able to scale to accommodate the new traffic?

Choice	Response
A	Lambda function
B	SQS queue
C	EC2 instance
D	DynamoDB table

© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

76

Look at the answer choices and rule them out based on the keywords.



Sample exam question answer

A web application enables customers to upload orders to an S3 bucket. The resulting Amazon S3 events trigger a Lambda function that inserts a message into an SQS queue. A single EC2 instance reads the messages from the queue, processes them, and stores them in a DynamoDB table partitioned by unique order ID. Next month, traffic is expected to increase by a factor of 10 and a Solutions Architect is reviewing the architecture for possible scaling problems.

Which component is MOST likely to need re-architecting to be able to scale to accommodate the new traffic?

The correct answer is C.

The keywords in the question are “a single EC2 instance” and “able to scale”

© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

77

The following are the keywords to recognize: **“a single EC2 instance” and “able to scale”**.

The correct answer is C: EC2 instance. A single EC2 instance does not scale and is a single point of failure in the architecture. A better solution would be to have EC2 instances in an Auto Scaling group across two Availability Zones, and have them read messages from the queue. The other responses are all managed services that can be configured to scale, or that scale automatically.

Additional resources

- [Set it and Forget it: Auto Scaling Target Tracking Policies](#)
- [Introduction to Amazon Elastic Load Balancer – Application](#)
- [Configuring Auto Scaling Group with ELB Elastic Load Balancer](#)
- [What Is an Application Load Balancer?](#)



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

78

If you want to learn more about the topics covered in this module, you might find the following additional resources helpful:

- [Set it and Forget it: Auto Scaling Target Tracking Policies](#)
- [Introduction to Amazon Elastic Load Balancer – Application](#)
- [Configuring Auto Scaling Group with ELB Elastic Load Balancer](#)
- [What Is an Application Load Balancer?](#)



Thank you

Corrections, feedback, or other questions?
Contact us at <https://support.aws.amazon.com/#/contacts/aws-academy>.
All trademarks are the property of their owners.

© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

79

Thank you for completing this module.



**AWS Academy Cloud Architecting
Module 10 Student Guide
Version 2.0.8**

200-ACACAD-20-EN-SG

© 2022 Amazon Web Services, Inc. or its affiliates. All rights reserved.

This work may not be reproduced or redistributed, in whole or in part, without prior written permission from Amazon Web Services, Inc. Commercial copying, lending, or selling is prohibited.

All trademarks are the property of their owners.

Contents

Module 10: Automating Your Architecture	4
---	---



Module 10: Automating Your Architecture

AWS Academy Cloud Architecting

© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

Welcome to Module 10: Automating Your Architecture.

Module overview

Sections

1. Architectural need
2. Reasons to automate
3. Automating your infrastructure
4. Automating deployments
5. AWS Elastic Beanstalk

Demonstration

- Analyzing AWS CloudFormation Template Structure and Creating a Stack

Labs

- Guided Lab: Automating Infrastructure Deployment with AWS CloudFormation
- Challenge Lab: Automating Infrastructure Deployment



Knowledge check



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

2

This module contains the following sections:

1. Architectural need
2. Reasons to automate
3. Automating your infrastructure
4. Automating deployments
5. AWS Elastic Beanstalk

This module also includes:

- An educator-led *demonstration* that starts with an analysis of the structure of an AWS CloudFormation template, and then creates a stack from the template.
- A *guided lab* that provides hands-on practice with using AWS CloudFormation to create resources in your AWS account.
- A *challenge lab* where you use AWS CloudFormation to create Amazon Web Services (AWS) resources that support the Café use case.

Finally, you will be asked to complete a *knowledge check* that will test your understanding of key concepts covered in this module.

Module objectives

At the end of this module, you should be able to:

- Recognize when to automate and why
- Identify how to model, create, and manage a collection of AWS resources using AWS CloudFormation
- Use the Quick Start AWS CloudFormation templates to set up an architecture
- Indicate how to use AWS System Manager and AWS OpsWorks for infrastructure and deployment automation
- Indicate how to use AWS Elastic Beanstalk to deploy simple applications



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

3

At the end of this module, you should be able to:

- Recognize when to automate and why
- Identify how to model, create, and manage a collection of AWS resources using AWS CloudFormation
- Use the Quick Start AWS CloudFormation templates to set up an architecture
- Indicate how to use AWS System Manager and AWS OpsWorks for infrastructure and deployment automation
- Indicate how to use AWS Elastic Beanstalk to deploy simple applications



Introducing Section 1: Architectural need.

Café business requirement

The café now has locations in multiple countries and must start automating to keep growing. Their organization has many different architectures and needs a way to consistently deploy, manage, and update them.



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

5

Up to this point, the café created their AWS resources and configured their applications manually--mostly by using the AWS Management Console. This approach worked well as a way for the café to quickly develop a web presence and build out an infrastructure that supports the needs of employees and customers. However, they find it challenging to replicate their deployments to new AWS Regions so they can support new cafe locations in multiple countries.

They would also like to have separate development and production environments that reliably have matching configurations. They realize that they must start automating to support continued growth. Their organization has many different architectures, and it needs a way to consistently deploy, manage, and update these architectures quickly, consistently, and reliably.

In this module, you will learn about AWS services that provide automation, including AWS CloudFormation. By using AWS CloudFormation, you will be able to help the cafe meet these new business requirements.

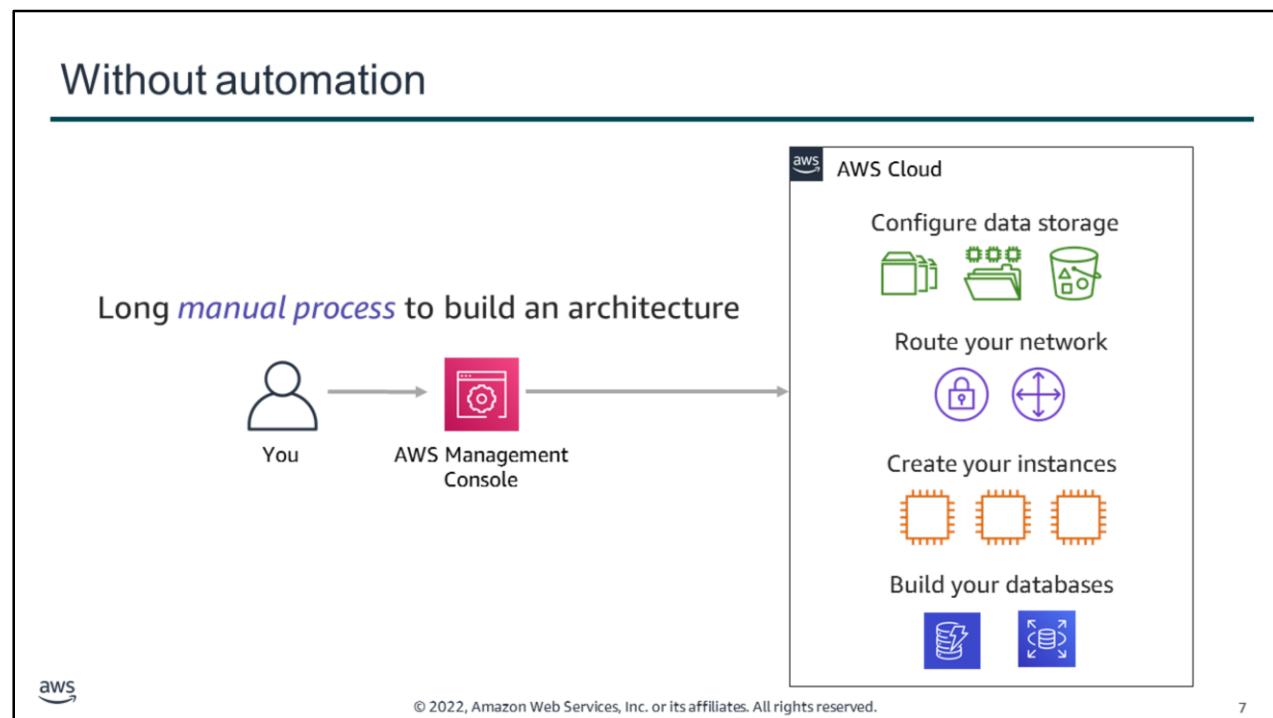
Section 2: Reasons to automate

Module 10: Automating Your Architecture



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

Introducing Section 2: Reasons to automate.



7

It takes significant time and energy to build a large-scale computing environment.

Many organizations will start using AWS by manually creating an Amazon Simple Storage Service (Amazon S3) bucket, or launching an Amazon Elastic Compute Cloud (Amazon EC2) instance and running a web server on it. Then, over time, they manually add more resources as they find that expanding their use of AWS can meet additional business needs. Soon, however, it can become challenging to manually manage and maintain these resources.

Some questions to ask include:

- Where do you want to put your efforts—into the design or the implementation? What are the risks of manual implementations?
- How would you ideally update production servers? How will you roll out deployments across multiple geographic regions? When things break—and they will—how will you manage the rollback to the last known good version?
- How will you debug deployments? Can you fix the bugs in your application before you roll the deployment out to the customer? How will you discover what is wrong and then fix it so that it remains fixed?
- How will you manage dependencies on the various systems and subsystems in your organization?
- Finally, is it realistic that you will be able to do all these tasks through manual configurations?

Risks from manual processes



Does not support repeatability at scale

- How will you replicate deployments to multiple Regions?



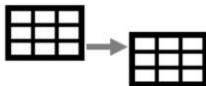
No version control

- How will you roll back the production environment to a prior version?



Lack of audit trails

- How will you ensure compliance? How will you track changes to configuration details at the resource level?



Inconsistent data management

- For example, how will you ensure matching configurations across multiple Amazon Elastic Compute Cloud (Amazon EC2) instances?



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

8

Manually creating resources and adding new features and functionality to your environment *does not scale*. If you are responsible for a large corporate application, there might not be enough people to manually sail the ship.

Also, creating architecture and applications from scratch does not have inherent *version control*. If there is an emergency, it's helpful to be able to roll back the production stack to a previous version—but that is not possible when you create your environment manually.

Having an *audit trail* is important for many compliance and security situations. It's dangerous to allow anyone in your organization to manually control and edit your environments.

Finally, *consistency* is critical when you want to minimize risks. Automation enables you to maintain consistency.

Complying with AWS Well-Architected Framework principles

- Operational excellence design principles
 - Perform operations as code
 - Make frequent, small, reversible changes
- Reliability pillar design principles
 - Manage change in automation



Creating and maintaining AWS resources and deployments by following a **manual approach** does not enable you to meet these guidelines.



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

9

Also consider how well manual approaches comply with the design principles of the AWS Well-Architected Framework. One of the six design principles of [operational excellence](#) is to perform operations as code. In the cloud, you can apply the same engineering discipline that you use for application code to your entire environment. You can define your entire workload (applications, infrastructure, and other resources) as code and update it with code. You can script your operations procedures and automate when they run by triggering them in response to events. By performing operations as code, you limit human error and enable consistent responses to events.

Another operational excellence design principle is to make frequent, small, reversible changes. That is, design workloads to enable regular updates to components so that you can increase the flow of beneficial changes into your workload. Make changes in small increments that can be reversed if they do not help you identify and resolve issues that were introduced to your environment. When possible, try not to affect customers when you make these changes.

Finally, one of the design principles of the Well-Architected [reliability pillar](#) is to manage change in automation. Changes to your infrastructure should be done via automation. Therefore, the changes that must be managed are changes to the automation. Making changes to production systems is one of the largest risk areas for many organizations. In operations, use automation wherever it's practical, like for testing and deploying changes, adding or removing capacity, and migrating data.

Section 3: Automating your infrastructure

Module 10: Automating Your Architecture



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

Introducing Section 3: Automating your infrastructure.

Automating your infrastructure



- AWS CloudFormation provides a simplified way to **model, create, and manage** a collection of **AWS resources**
 - Collection of resources is called an **AWS CloudFormation stack**
 - No extra charge (pay only for resources you create)
- Can create, update, and delete stacks
- Enables orderly and predictable **provisioning** and updating of resources
- Enables **version control** of AWS resource deployments



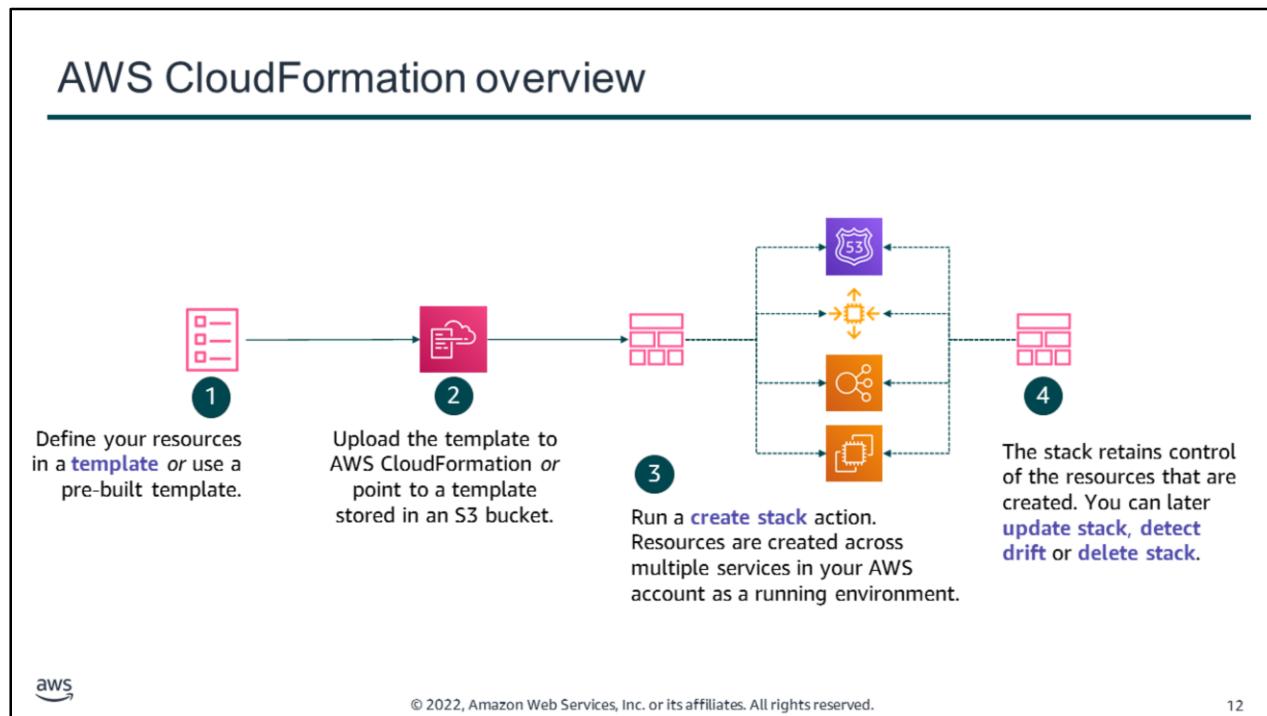
© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

11

AWS CloudFormation provisions resources in a repeatable manner. It enables you to build and rebuild your infrastructure and applications without needing to perform manual actions or write custom scripts. With AWS CloudFormation, you author a document that describes what your infrastructure should be, including all the AWS resources that should be a part of the deployment. You can think of this document as a *model*. You then use the model to create the reality, because AWS CloudFormation can actually create the resources in your account.

When you use AWS CloudFormation to create resources, it is called an **AWS CloudFormation stack**. You create a stack, update a stack, or delete a stack. Thus, you can provision resources in an orderly and predictable way.

Using AWS CloudFormation enables you to treat your infrastructure as code (IaC). Author it with any code editor, check it into a *version control* system such as GitHub or AWS CodeCommit, and review files with team members before you deploy into the appropriate environments. If the AWS CloudFormation document that you create to model your deployment is checked in to a version control system, you could always delete a stack, check out an older version of the document, and create a stack from it. With version control, you can use essential rollback capabilities.



This diagram demonstrates how AWS CloudFormation works. First, you define the AWS resources that you want to create. In the example here, it creates a few EC2 instances, a load balancer, an Auto Scaling group, and an Amazon Route 53 hosted zone. You define the resources in an AWS CloudFormation template. You can create the template from scratch, or you can use a pre-built template. Many [sample templates](#) are also available.

Although AWS CloudFormation offers broad support for AWS services, not all resources can be created by AWS CloudFormation. See the list of [resources that are supported by AWS CloudFormation](#) for details.

Next, you upload the template to AWS CloudFormation. Alternatively, you can store the template on Amazon S3 and point AWS CloudFormation to the location where it is stored.

Third, you run the create stack action. When you do this, the AWS CloudFormation service reads through what is specified in the template and creates the desired resources in your AWS account. A single stack can create and configure resources in a single Region across multiple AWS services.

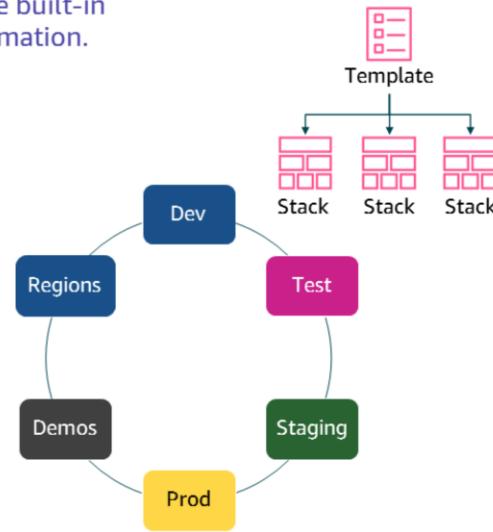
Finally, you can observe the progress of the stack-creation process. After the stack has successfully completed, the AWS resources that it created exist in your account. The stack object remains, and it acts like a handle to all the resources that it created. This is helpful when you want to take actions later. For example, you might want to update the stack (to create additional

AWS resources or modify existing resources) or delete the stack (which will clean up and delete the resources that were created by the stack).

Infrastructure as code (IaC)

For AWS Cloud development, the built-in choice for IaC is AWS CloudFormation.

- IaC is the process of provisioning and managing your cloud resources by writing a template file that is –
 - Human readable
 - Machine consumable
- It is infrastructure you can replicate, re-deploy, re-purpose
- You can roll back to the last good state on failures



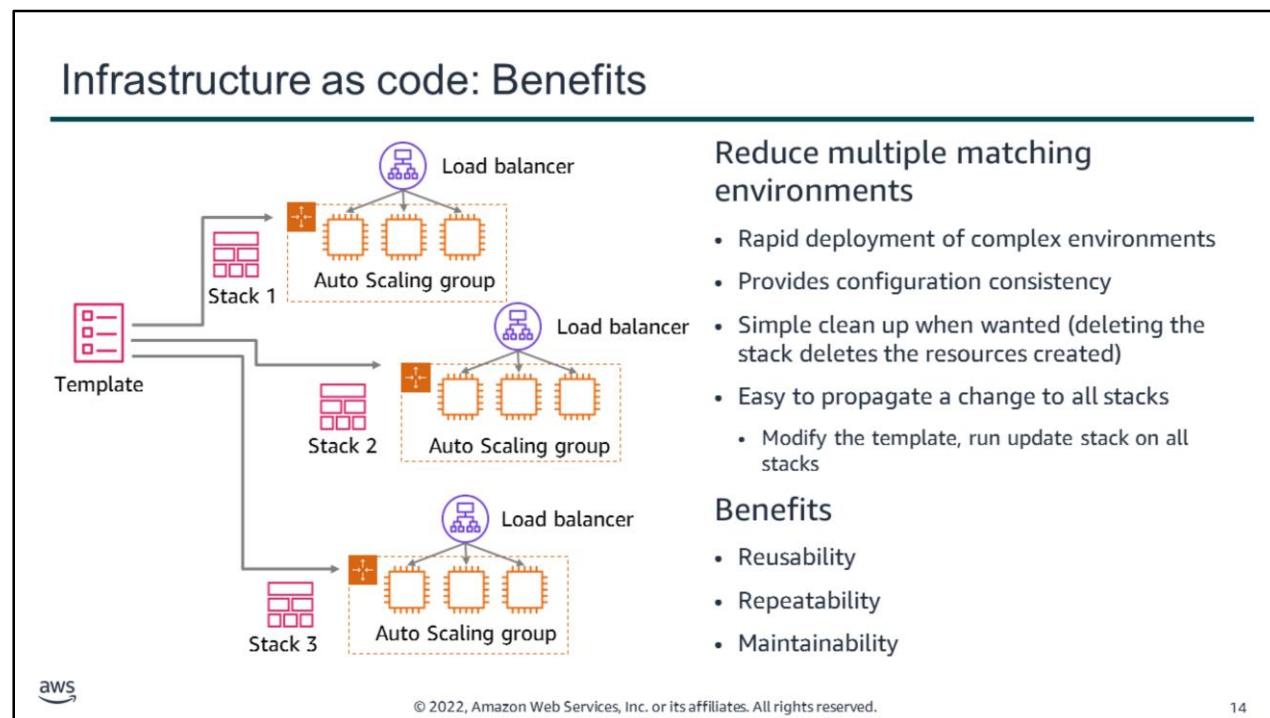
© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

13

Infrastructure as code (IaC) is an industry term that refers to the process of provisioning and managing cloud resources by defining them in a template file that is both human-readable and machine-consumable.

IaC continues to grow in popularity because it provides a workable solution to challenges, like how to replicate, re-deploy, and re-purpose infrastructure, easily, reliably, and consistently.

The transactional nature of AWS CloudFormation is one of its biggest advantages from a customer perspective. AWS CloudFormation is transactional—the service will *roll back* to the last good state on failures.



Now, consider some of the benefits of IaC in more detail. If you build infrastructure with code, you gain benefits like the ability to rapidly deploy complex environments. With one template (or a combination of templates), you can build the same complex environments repeatedly.

In the example here, a single template can be used to create three different stacks. Each stack can be created rapidly, usually in a matter of minutes. Each stack replicates complex configuration details consistently.

If Stack 2 is your test environment and Stack 3 is your production environment, you can have greater confidence that if your test jobs performed well in the test environment, that they will also perform well in the production environment. The template minimizes the risk that the test environment is configured differently from the production environment.

Also, if you must make a configuration update in the test environment, you update the template with the change and update all the stacks. This process helps ensure that the modifications to a single environment will be reliably propagated to all environments that should receive the update.

Another benefit is that it easier to clean up all the resources that were created in your account to support a test environment after you no longer need them. This helps reduce cost associated with no resources that you no longer need, and helps keep your account clean of cruft.

AWS CloudFormation template syntax

AWS CloudFormation templates

- Author in JavaScript Object Notation (JSON) or YAML Ain't Markup Language (YAML)
- YAML advantages –
 - Less verbose (no {}, "", characters)
 - Supports embedded comments
- JSON advantages –
 - More widely used by other computer systems (for example, APIs)
- Recommendation – Treat templates as source code
 - Store them in a code repository



```
{ "AWSTemplateFormatVersion": "2010-09-09", "Resources": { "awsexamplebucket1": { "Type": "AWS::S3::Bucket" } } }
```

JSON example

```
AWSTemplateFormatVersion: 2010-09-09 Resources: awsexamplebucket1: Type: AWS::S3::Bucket
```

YAML example

Templates can also be authored in the [AWS CloudFormation Designer](#)—a graphical design interface in the AWS Management Console.



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

15

An AWS CloudFormation template can be authored in either JavaScript Object Notation (JSON) or YAML Ain't Markup Language (YAML).

YAML is optimized for readability. The same data that is stored in JSON will take fewer lines to store in YAML because YAML does not use braces ({}), and it uses fewer quotation marks (""). Another advantage of YAML is that it natively supports embedded comments. You might find it easier to debug YAML documents compared with JSON. With JSON, it can be difficult to track down missing or misplaced commas or braces. YAML does not have this issue.

Despite the many advantages of YAML, JSON offers some unique advantages. First, it is widely used by computer systems. Its ubiquity is an advantage because data that is stored in JSON can reliably be used with many systems without needing transformation. Also, it is usually easier to generate and parse JSON than it is to generate and parse YAML.

The AWS Management Console provides a graphical interface, called the AWS CloudFormation Designer, which can be used to author or view the contents of AWS CloudFormation templates. It can also be used to convert a valid JSON template to YAML or to convert YAML to JSON. It provides a drag-and-drop interface for authoring templates that can be output as either JSON or YAML.

Simple template: Create an EC2 instance

```
{  
  "AWSTemplateFormatVersion": "2010-09-09",  
  "Description": "Create EC2 instance",  
  "Parameters": {  
    "KeyPair": {  
      "Description": "SSH Key Pair",  
      "Type": "String"}},  
  "Resources": {  
    "Ec2Instance": {  
      "Type": "AWS::EC2::Instance",  
      "Properties": {  
        "ImageId": "ami-9d23aeea",  
        "InstanceType": "m3.medium",  
        "KeyName": {"Ref": "KeyPair"}  
      }},  
  },  
  "Outputs": {  
    "InstanceId": {  
      "Description": "InstanceId",  
      "Value": {"Ref": "Ec2Instance"}  
    }  
  }  
}
```

← **Parameters** – Specify what values can be set at runtime when you create the stack

- Example uses: Region-specific settings, or production versus test environment settings

← **Resources** – Define what needs to be created in the AWS account

- Example: Create all components of a virtual private cloud (VPC) in a Region, and then create EC2 instances in the VPC
- Can reference parameters

← **Outputs** – Specify values returned after the stack is created

- Example use: Return the instanceId or the public IP address of an EC2 instance



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

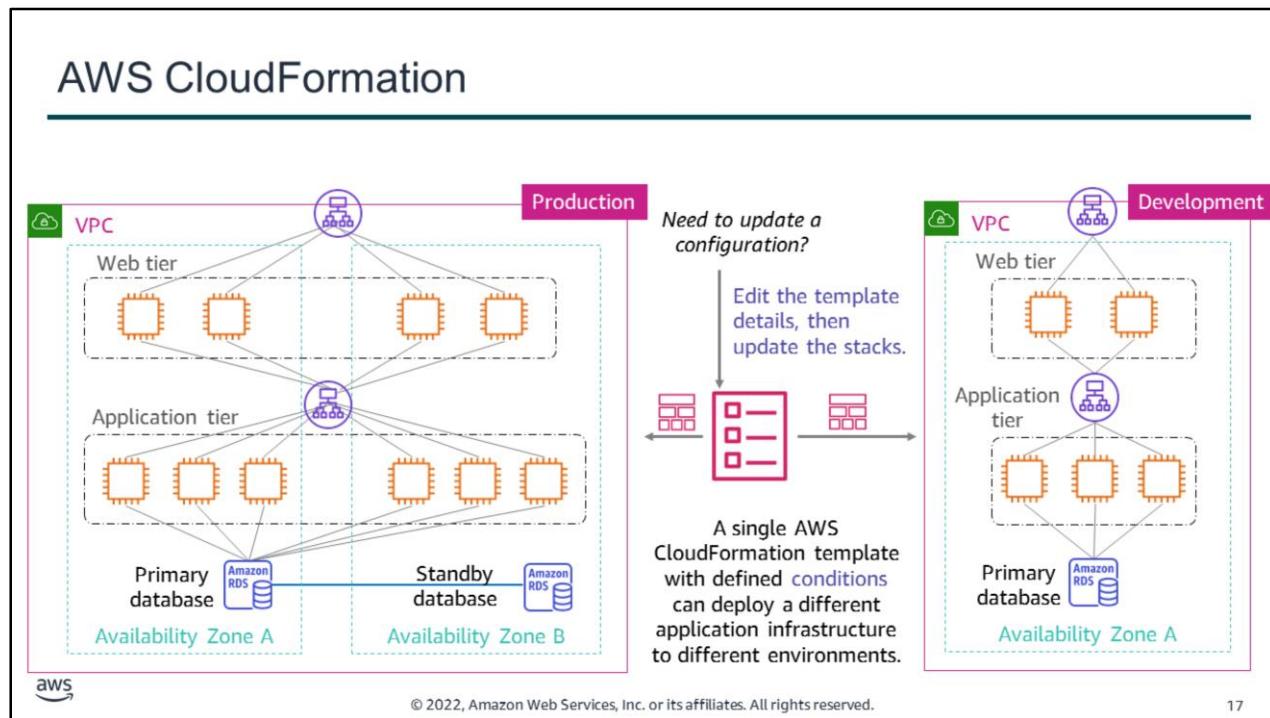
16

This example AWS CloudFormation template creates an EC2 instance. Although this example does not illustrate all possible sections of a template, it does highlight some of the most commonly used sections, including parameters, resources, and outputs.

Parameters is an optional section of the template. Parameters are values that are passed to your template at runtime (when you create or update a stack). You can refer to parameters from the Resources and Outputs sections of the template. A parameter's name and description appear in the Specify Parameters page when a user launches the Create Stack wizard in the console.

Resources is a required section for any template. Use it to specify the AWS resources to create, along with their properties. In this example, a resource of type AWS::EC2::Instance is specified, which creates an EC2 instance. The example resource includes both statically defined properties (ImageId and InstanceType) and a reference to the KeyPair parameter.

Finally, the example shows the Outputs section. Outputs describes the values that are returned when you view your stack's properties. In the example, an InstanceId output is declared. After the stack is created, you can see this value in the stack details in the AWS CloudFormation console, by running the aws cloudformation describe-stacks AWS Command Line Interface (AWS CLI) command, or by using the AWS software development kits (SDKs) to retrieve this value.



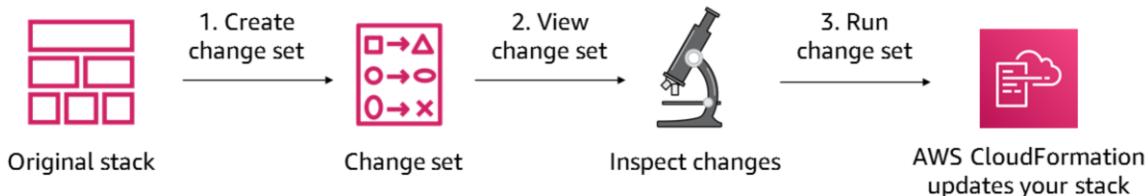
You can use the same AWS CloudFormation template to create both your production environment and development environment. This approach can help ensure that (for example) the same application binaries, same Java version, and same database version are used in both development and production. Thus, the template can help ensure that your application behaves in production the same way that it behaved in the development environment.

In the example, you see that both the production and development environments are created from the same template. However, the production environment is configured to run across two Availability Zones and the development environment runs in a single Availability Zone. These kinds of deployment-specific differences can be accomplished by using *conditions*. You can use a *Conditions* statement in AWS CloudFormation templates to help ensure that—though they are different in size and scope—development, test, and production environments are otherwise configured identically.

You might need several testing environments for functional testing, user acceptance testing, and load testing. Creating those environments manually is risky. However, creating them with AWS CloudFormation helps ensure consistency and repeatability.

AWS CloudFormation change sets

Change sets enable you to **preview changes** before you implement them.



Use the [DeletionPolicy](#) attribute to preserve or backup a resource when its stack is deleted or updated.



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

18

One way to update a stack (and thus update your AWS resources) is to update the AWS CloudFormation template that you used to create the stack, and then run the **Update Stack** option.

However, you might want to gain additional insight about the specific changes that AWS CloudFormation will implement if you run that command—before you actually run an update. If you want this type of insight, you can use a change set.

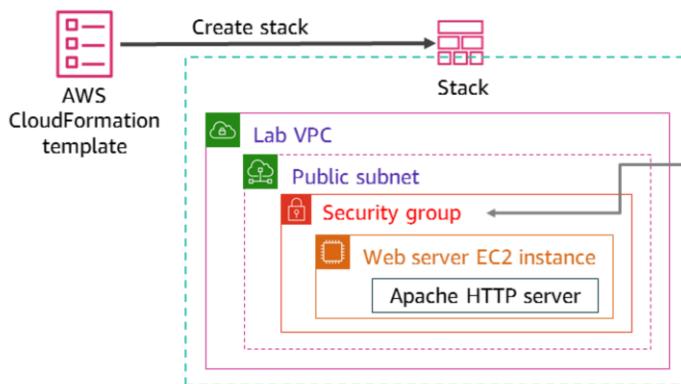
Change sets enable you to preview the changes, verify that they align with your expectations, and then approve the updates before you proceed.

Follow this basic workflow to use AWS CloudFormation change sets:

1. Create a change set by submitting changes for the stack that you want to update.
2. View the change set to see which stack settings and resources will change. If you want to consider other changes before you decide which changes to make, create additional change sets.
3. Run the change set. AWS CloudFormation updates your stack with those changes.

If you use change sets, you might want to set a deletion policy on some resources. The [DeletionPolicy](#) attribute can be used to preserve (or, in some cases, back up) a resource when its stack is deleted or updated. If a resource has no [DeletionPolicy](#) attribute, AWS CloudFormation deletes the resource.

Drift detection



Scenario:

1. An application environment is created by an AWS CloudFormation stack.
2. Later, someone **manually modifies the security group** and opens a new inbound TCP port.
3. Drift detection is run on the stack.
4. All resources except the security group show the result **IN_SYNC**, but the security group shows a status of **MODIFIED**, with details.

Question: In this scenario, what would be a better approach if the team wants to modify the security group setting?

Answer: Modify the AWS CloudFormation template security group settings. Then, run Update Stack. AWS CloudFormation will update the security group. Keeps the *model* deployment synchronized with the actual deployment.



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

19

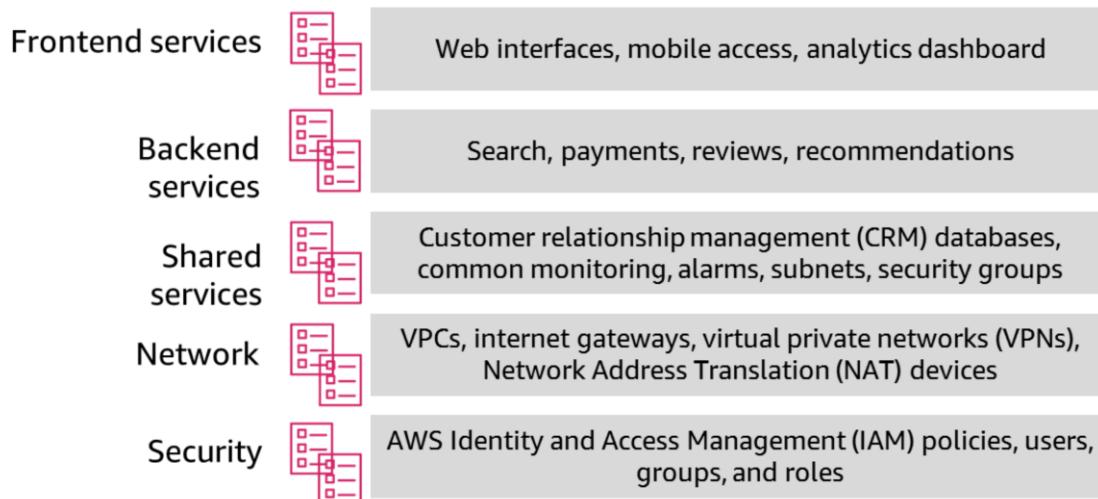
Consider this scenario. An application environment is created by running an AWS CloudFormation stack. Then, someone decides to manually modify the deployed environment settings. They create a new inbound rule in the security group that was created by the stack. However, they make this change outside the context of AWS CloudFormation—for example, by using the Amazon EC2 console. As the architect of this application, you would want to know that your deployed environment no longer matches the model environment that is defined in the AWS CloudFormation template.

How would you know which resources were modified so that they no longer exactly conform with the specifications in the stack?

Drift detection can be run on a stack by choosing Detect Drift from the Stack actions menu in the console. Performing drift detection on a stack shows you whether the stack has drifted from its expected template configuration. Drift detection returns detailed information about the drift status of each resource that supports drift detection in the stack.

When you delete a stack that has drift, the drift is not handled by the AWS CloudFormation resource cleanup process. If the stack has unresolved resource dependencies, they might cause the delete stack action to fail. In such cases, you might need to manually resolve the issue. For details, see the list of [resources that support drift detection](#).

Scoping and organizing templates



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

20

As your organization starts to use more AWS CloudFormation templates, it will be important for you to have a strategy for templates. This strategy will define the scope of what a single template should create, and the general characteristics that would make you want to define your AWS infrastructure in more than one template.

The diagram offers some ideas about how you might organize your templates so that they are easier to maintain, and so that they can be used in combination with each other in a way that makes sense. A good strategy is to group your resource definitions in templates similar to the way that you would organize the functionality of a large enterprise application into different parts.

Think about the more tightly connected components of your infrastructure, and consider putting them in the same templates. In this example, AWS CloudFormation templates are scoped to create and maintain AWS resources in one of five areas: frontend services, backend services, shared services, network, and security. In each area, you might maintain a template that is scoped to a single application or a single department's needs.

Regardless of how you organize and scope each AWS CloudFormation template, treat your templates as code that needs version control. Store your templates in a source control system.

AWS Quick Starts

For AWS CloudFormation templates built by AWS solutions architects, visit the AWS Quick Starts.



- Are gold-standard deployments
- Are based on AWS best practices for security and high availability
- Can be used to create entire architectures with one click in less than an hour
- Can be used for experimentation and as the basis for your own architectures

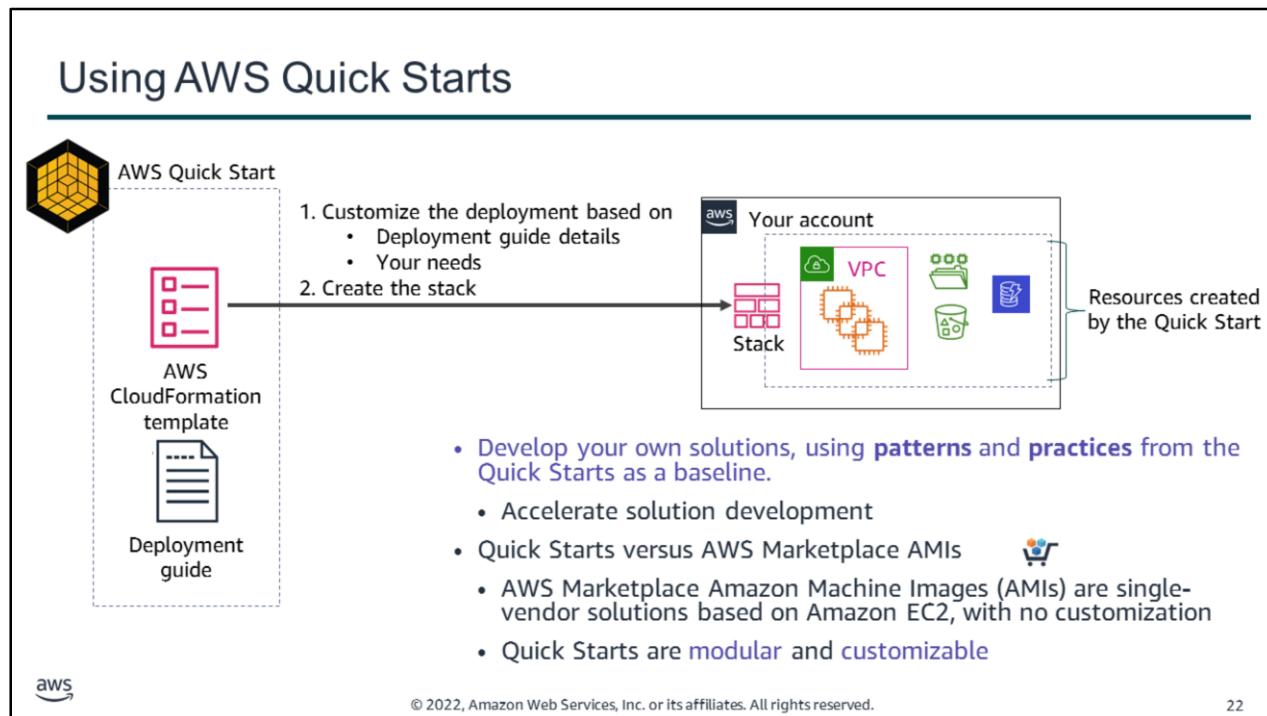


© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

21

AWS Quick Starts provide AWS CloudFormation templates. The Quick Starts are built by AWS solutions architects and partners to help you deploy popular solutions on AWS, based on AWS best practices for security and high availability. These reference deployments can be provisioned in your AWS account in less than an hour, often in only minutes. They help you build Well-Architected test or production environment in a few steps. You can use them to create entire architectures, or you can use them to experiment with new deployment approaches.

Go to the [AWS Quick Starts](#) page for details.



Each Quick Start consists of an AWS CloudFormation template and a deployment guide. The guide provides details about deployment options and how to configure the deployment to match your needs.

Customize the deployment to match your needs and create the stack. Depending on the AWS resources that must be created, the Quick Start will finish deploying in a matter of minutes or a few hours.

Even if you do not use the Quick Starts, you may find it helpful to look at a few of them to see the types of *patterns* and *practices* that the Quick Starts follow. You might find that they help accelerate your own template development when you borrow a section of a Quick Start and embed it in your own template.

AWS Marketplace Amazon Machine Images (AMIs) are another solution that people sometimes use. These resources can be launched from the Amazon EC2 console. AWS Marketplace AMIs provide single-vendor solutions that run on EC2 instances. In contrast, AWS Quick Starts are modular and more customizable solutions that might (or might not) use Amazon EC2.

**Demonstration:
Analyzing AWS
CloudFormation
Template Structure and
Creating a Stack**



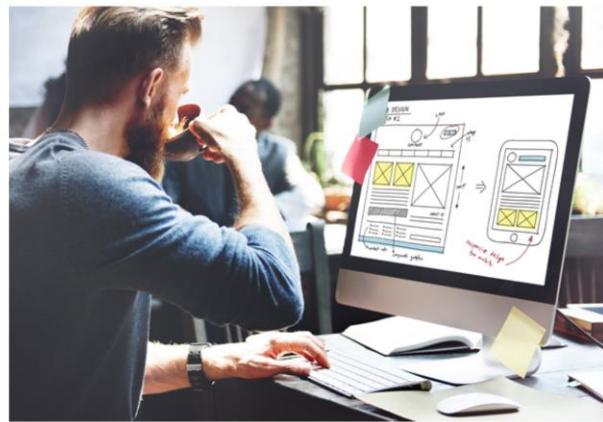
© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

23

The educator might choose to demonstrate the structure of an AWS CloudFormation template and then create an AWS CloudFormation stack by using the template.

Module 10 – Guided Lab:

Automating Infrastructure Deployment with AWS CloudFormation



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

24

You will now complete Module 10 - Guided Lab: Automating Infrastructure Deployment with AWS CloudFormation.

Guided lab: Tasks

1. Deploying a networking layer
2. Deploying an application layer
3. Updating a stack
4. Exploring templates with AWS CloudFormation Designer
5. Deleting the stack



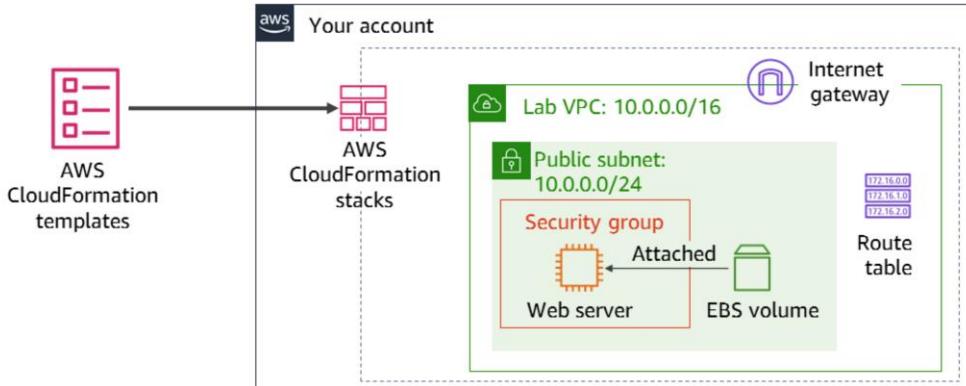
© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

25

In this guided lab, you will complete the following tasks:

1. Deploying a networking layer
2. Deploying an application layer
3. Updating a stack
4. Exploring templates with AWS CloudFormation Designer
5. Deleting the stack

Guided lab: Final product



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

26

By the end of this guided lab, you will have used AWS CloudFormation to create the resources in the diagram. The resources in the network layer are created when you create the first stack. The EC2 instance, security group, and Amazon Elastic Block Store (Amazon EBS) volume are created when you create the second stack. The security group settings are then updated in the template that is used to create the second stack. This modification is applied to the security group resource when you run the update stack action.



A timer icon with the text "~ 20 minutes" indicating the duration of the guided lab.

Begin Module 10 – Guided Lab: Automating Infrastructure Deployment with AWS CloudFormation

aws

© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

27

It is now time to start the guided lab.

Guided lab debrief: Key takeaways



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

28

Your educator might choose to lead a conversation about the key takeaways from the guided lab after you have completed it.

Section 3 key takeaways



- AWS CloudFormation is an infrastructure as code (IaC) service that enables you to model, create, and manage a collection of AWS resources
- AWS CloudFormation IaC is defined in templates that are authored in JSON or YAML
- A stack is what you create when you use a template to create AWS resources
- Actions that are available on an existing stack include update stack, detect drift, and delete stack
- AWS Quick Starts provide AWS CloudFormation templates that are built by solutions architects that reflect AWS best practices

© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

29

Some key takeaways from this section of the module include:

- AWS CloudFormation is an infrastructure as code (IaC) service that enables you to model, create, and manage a collection of AWS resources
- AWS CloudFormation IaC is defined in templates that are authored in JSON or YAML
- A stack is what you create when you use a template to create AWS resources
- Actions that are available on an existing stack include update stack, detect drift, and delete stack
- AWS Quick Starts provides AWS CloudFormation templates that are built by solutions architects and that reflect AWS best practices

Section 4: Automating deployments

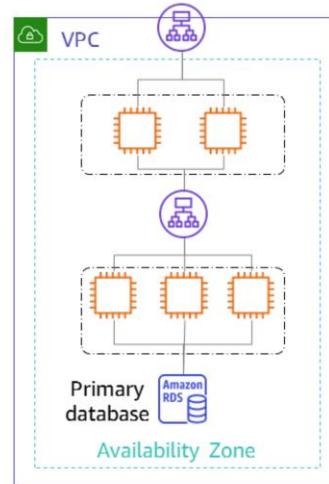
Module 10: Automating Your Architecture



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

Introducing Section 4: Automating deployments.

How will you keep your fleet updated?



You might have hundreds of instances to manage.
How do you apply guest operating system (OS) patches
and update the software that is installed on them?



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

31

In the previous section, you learned how to create an entire infrastructure automatically by using AWS CloudFormation. This capability is powerful, but there are still some important questions to ask.

How will you update the software on your fleet of EC2 instances? Are you supposed to remotely log in to each instance and run update commands yourself? How will you revert a change if something goes wrong? What if you have hundreds or even thousands of servers that run many different applications?

Traditional tools can help with these scenarios, but a ready to use solution would be more convenient.

AWS Systems Manager

Gain operational insights and take action on AWS resources.



- Automates operational tasks
 - Example: Apply OS patches and software upgrades across a fleet of EC2 instances
- Simplifies resource and application management
 - Manage software inventory
 - View detailed system configurations across the fleet
- Manages servers on-premises and in the cloud



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

32

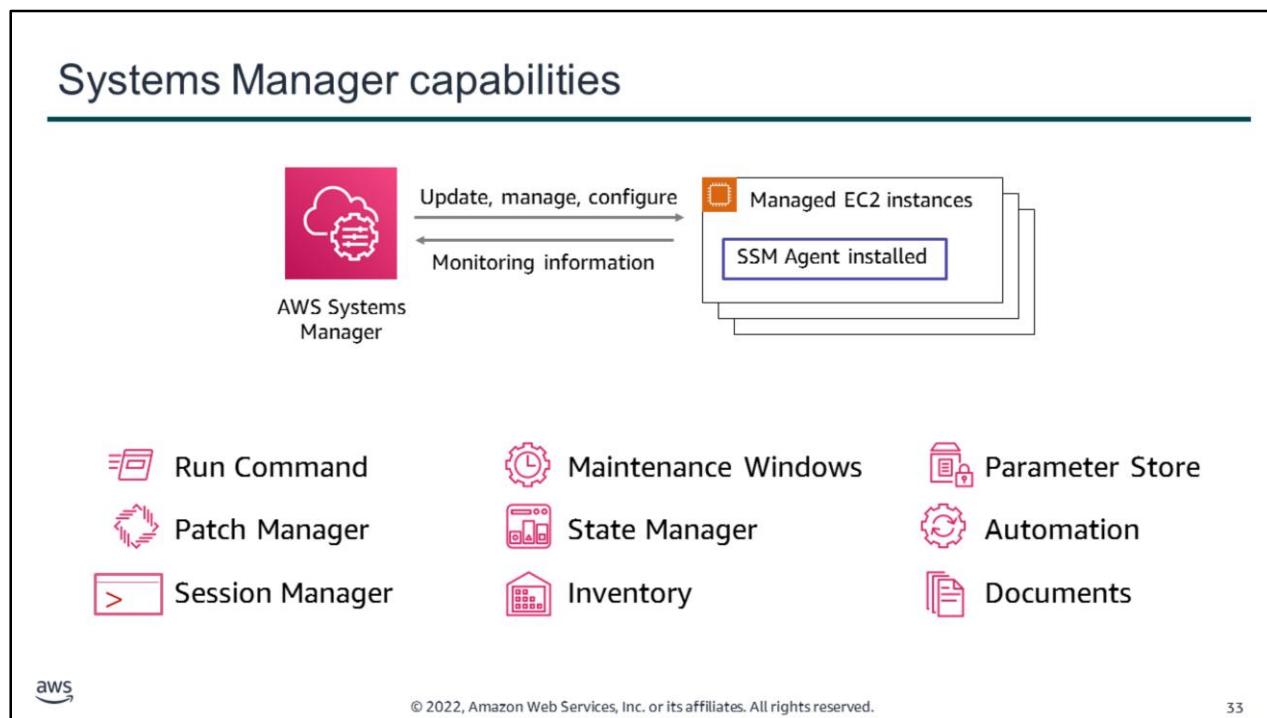
Even if you use an IaC tool like AWS CloudFormation to create and maintain your AWS resource deployments, it is helpful to have other tools that you can use. For example, these tools can address the environment's ongoing needs for configuration management. These needs can occur both after infrastructure resources are provisioned and after the infrastructure is up and running. AWS Systems Manager is a service that addresses this challenge.

AWS Systems Manager is a management service that is designed to be highly focused on automation. It enables the configuration and management of systems that run on-premises or in AWS. AWS Systems Manager enables you to identify the instances that you want to manage, and then define the management tasks that you want to perform on those instances. AWS Systems Manager is available at no cost, and it can manage both your Amazon EC2 and on-premises resources.

Some tasks you can accomplish with AWS Systems Manager include:

- Collecting software inventory
- Applying operating system (OS) patches
- Creating system images
- Configuring Microsoft Windows and Linux operating systems

These capabilities help you define and track system configurations, prevent drift, and maintain the software compliance of your Amazon EC2 and on-premises configurations.



This example shows how Systems Manager can be used to update, manage, and configure a fleet of EC2 instances.

You can install an AWS Systems Manager Agent (SSM Agent) on an EC2 instance, or even on an on-premises server, or a virtual machine (VM). Once the SSM Agent is installed, it will be possible for Systems Manager to update, manage, and configure the server on which it is installed. The agent processes requests from Systems Manager and then runs them in accordance with the specification provided in the request. The agent then sends status and relevant information back to Systems Manager.

SSM Agent is preinstalled, by default, on most Microsoft Windows Server AMIs, all Amazon Linux and Amazon Linux 2 AMIs, and some Ubuntu AMIs. However, you must manually install the agent on EC2 instances that were created from other Linux AMIs. For full details, see the [Working with SSM Agent](#) AWS documentation.

AWS Systems Manager provides various tools:

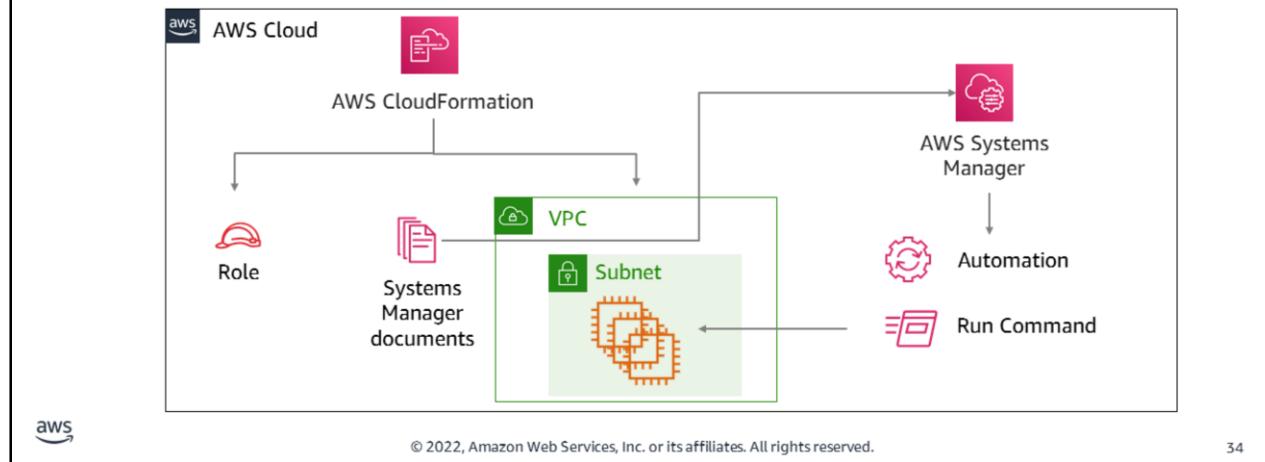
- Run Command enables you to remotely and securely manage the configuration of your managed instances. Commands can run without Secure Shell (SSH) or Remote Desktop Protocol (RDP) access, so you can use them to reduce the need for a bastion host. Run Bash, PowerShell, Salt, or Ansible scripts.
- Maintenance Windows enable you to define a schedule for performing potentially disruptive actions on your instances. Examples include patching an operating system, updating drivers, or installing software or patches.
- Parameter Store provides secure storage for configuration data and secrets management. For example, you can store passwords, database strings, and license codes as parameter values.
- Patch Manager automates the process of patching managed instances with both security-

related updates and other types of updates.

- State Manager automates the process of keeping your Amazon EC2 and hybrid infrastructure in a state that you define.
- Automation enables you to build automation workflows to configure and manage instances and AWS resources.
- Session Manager enables you to manage your EC2 instances through an interactive, browser-based shell.
- Inventory provides visibility into your Amazon EC2 and on-premises computing environments. You can use Inventory to collect metadata from your managed instances.
- Documents define the actions that Systems Manager performs on your managed instances. You can use more than a dozen pre-configured documents by specifying parameters at runtime. You can also define your own documents in JSON or YAML, and specify steps and parameters.

AWS CloudFormation and Systems Manager complement each other

AWS CloudFormation works well for **defining AWS Cloud resources**.
Systems Manager works well for **automating within guest operating systems**.



Now that you learned about the features of both AWS CloudFormation and AWS Systems Manager, consider how the two services complement each other.

Systems Manager works well for automating *within* a guest OS. In contrast, AWS CloudFormation works well for defining AWS Cloud resources.

You can use AWS CloudFormation at the AWS Cloud layer to define AWS resources. As the diagram demonstrates, you can then use AWS Systems Manager to configure the OS of the instances that were created by the AWS CloudFormation stack.

By maintaining the cloud resources with AWS CloudFormation, you keep the ability to deploy a stack and then delete a stack from a single template. In contrast, Systems Manager gives you a way to perform ongoing tasks, such as updating the EC2 instance guest OS with patch updates and centrally aggregating logs to Amazon CloudWatch.

For more information about an example solution that uses these two services together, see the [Using AWS Systems Manager Automation and AWS CloudFormation together](#) blog post.

AWS OpsWorks

AWS OpsWorks is a configuration management service.



- Automate how servers are configured, deployed, and managed
- Provides managed instances of **Chef** and **Puppet**
 - Chef and Puppet are popular automation platforms
- Three versions available –
 - AWS OpsWorks for Chef Automate
 - AWS OpsWorks for Puppet Enterprise
 - AWS OpsWorks Stacks



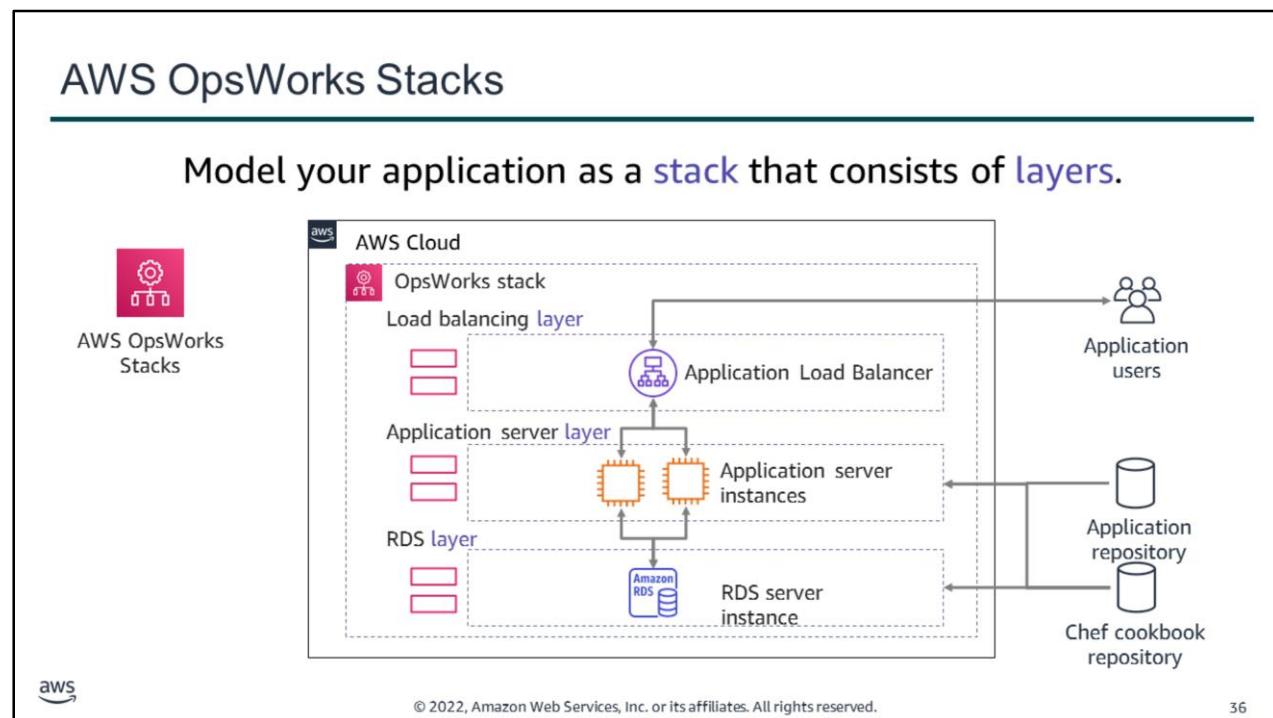
© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

35

AWS OpsWorks is a service for configuration management. You can use OpsWorks to automate how EC2 instances are configured, deployed, and managed.

AWS OpsWorks comes in three different versions:

- **AWS OpsWorks for Chef Automate** provides a fully managed Chef Automate server that provides workflow automation for continuous deployment, and automated testing for compliance and security. The Chef Automate platform handles operational tasks, such as software and operating system configurations, continuous compliance, package installations, database setups, and more. You can use Chef Automate to create and manage dynamic infrastructure that runs on the AWS Cloud. A Chef Automate server manages the configuration of nodes in your environment by telling the chef-client which Chef recipes to run on the nodes. It also stores information about nodes, and serves as a central repository for your Chef cookbooks.
- **AWS OpsWorks for Puppet Enterprise** provides a managed Puppet Enterprise server and a suite of automation tools that provide workflow automation for orchestration, automated provisioning, and visualization for traceability. With Puppet Enterprise, you can define configurations for your servers in a format that you can maintain and version like your application source code. The primary Puppet servers are designed to consistently configure and maintain your other Puppet servers (or nodes). You can also configure your nodes dynamically based on the state of other nodes.
- **AWS OpsWorks Stacks** is a configuration management service that helps you configure and operate applications of all kinds and sizes by using Chef. You can define the application's architecture and the specification of each component—including package installation, software configuration, and resources (such as storage).



This example demonstrates how a basic application might be managed with AWS OpsWorks Stacks. The fundamental unit of creation in an OpsWorks Stacks application is the *stack*.

After a stack is created, you can add multiple *layers* to that stack. You can thus build out your application as a set of interacting layers of related functionality.

In this case, a group of application servers runs in an *application server layer*. The applications run behind an Elastic Load Balancing load balancer that is defined in a *load balancing layer*. This example also includes a backend Amazon Relational Database Service (Amazon RDS) database server, which is defined in an *RDS layer*.

Layers depend on [Chef recipes](#) to handle tasks like installing packages on instances, deploying applications, running scripts, and so on. OpsWorks Stacks uses Chef cookbooks to handle tasks like installing and configuring packages and deploying applications. Your custom cookbooks must be stored in an online repository—an archive (such as a .zip file), or a source control manager (such as Git).

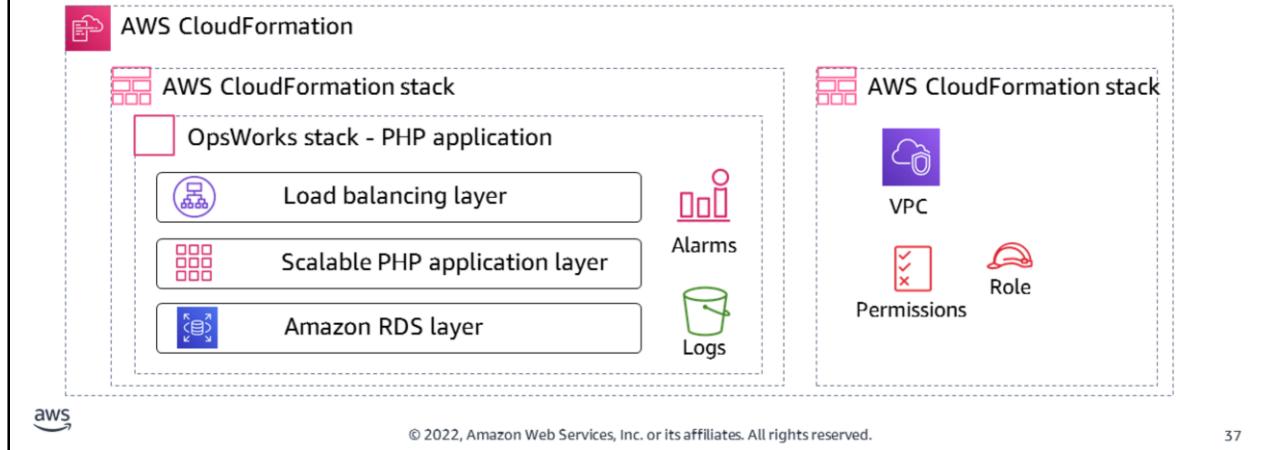
One key OpsWorks Stacks feature is a set of *lifecycle events*—including Setup, Configure, Deploy, Undeploy, and Shutdown—which automatically run a specified set of recipes at the appropriate time on each instance.

Each layer can have a set of recipes that are assigned to each lifecycle event. These recipes handle

various tasks for that event and layer.

OpsWorks Stacks complements AWS CloudFormation

1. Use **AWS CloudFormation** to create the **infrastructure** (VPC, IAM roles, and so on).
2. Deploy the **application layer** with **OpsWorks Stacks**.



Because OpsWorks Stacks can be created via AWS CloudFormation, the use of the two technologies is complementary.

For example, you could use one AWS CloudFormation template to create the AWS resources infrastructure for your environment, including a VPC. You could then use another AWS CloudFormation template to create the OpsWorks stack that will be deployed in that VPC. After both AWS CloudFormation stacks are created in your account, you can use the OpsWorks stack to manage the application.

Section 5: AWS Elastic Beanstalk

Module 10: Automating Your Architecture



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

Introducing Section 5: AWS Elastic Beanstalk.

General challenges



Managing infrastructure around application deployment can be *difficult*



It can be *time-consuming* to manage and configure servers



You might have *lack of consistency* across multiple projects or applications



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

39

Now, consider some general challenges that you might face as you approach managing your cloud infrastructure.

First, consider that deploying an application can be *difficult*. How will you ensure that it is highly available and that it can support user requests even during peak usage? How can you make sure that the application is resilient, and that regular backups are taken for disaster recovery (DR) purposes? It can be *time-consuming* to manage and configure servers. Meanwhile, you would like to maintain *consistency* across projects and applications, but this consistency can be difficult to achieve.

AWS Elastic Beanstalk

- Easy way to get [web applications](#) up and running



- [Managed service](#) that automatically handles –

- Infrastructure provisioning and configuration
- Deployment
- Load balancing
- Automatic scaling
- Health monitoring
- Analysis and debugging
- Logging



- No additional charge for using it

- Pay only for the underlying resources that are used



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

40

AWS Elastic Beanstalk is another AWS compute service option. It is a platform as a service (PaaS) offering that facilitates the quick deployment, scaling, and management of your web applications and services. It addresses many of the challenges that you just learned about.

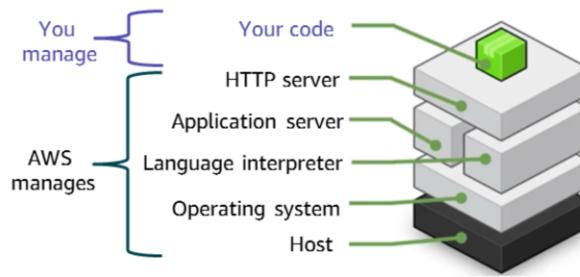
With Elastic Beanstalk, you remain in control of your code, while AWS maintains the underlying infrastructure. The required AWS resources are created and deployed by using a simple wizard in the AWS Management Console. The wizard asks you to choose the instance type and size, the database type and size, and what automatic scaling settings you would like to use. It provides you access to the server log files, and enable Secure HTTP (HTTPS) on the load balancer.

You upload your code and Elastic Beanstalk automatically handles the deployment—including capacity provisioning, load balancing, automatic scaling, and monitoring application health. At the same time, you retain full control over the AWS resources that power your application, and you can access the underlying resources at any time.

There is no additional charge for AWS Elastic Beanstalk. You pay for the AWS resources that you create to store and run your application, such EC2 instances or S3 buckets. You only pay for what you use, when you use it.

AWS Elastic Beanstalk deployments

- It supports web applications written for common platforms
 - Java, .NET, PHP, Node.js, Python, Ruby, Go, and Docker
- You upload your code
 - Elastic Beanstalk automatically handles the deployment
 - Deploys on servers such as Apache, NGINX, Passenger, Puma, and Microsoft Internet Information Services (IIS)



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

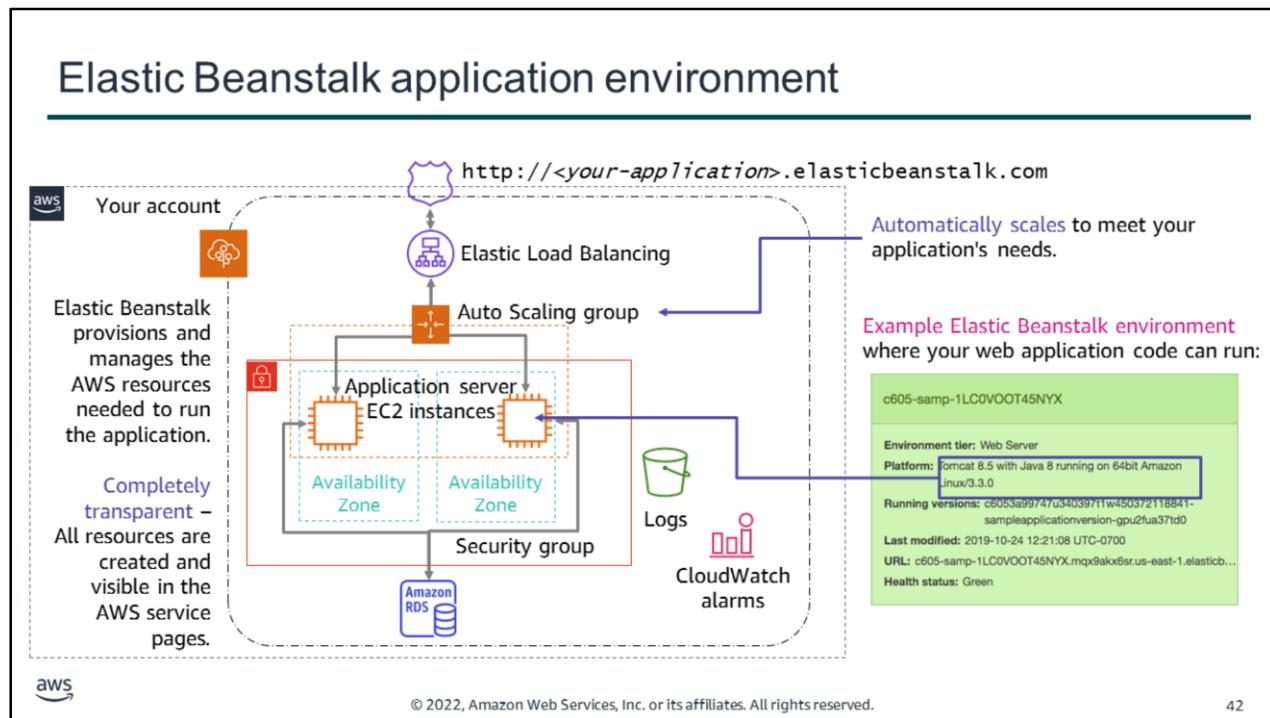
41

Elastic Beanstalk configures each EC2 instance in your environment with the components that are needed to run applications for the selected platform. You don't need to worry about logging in to instances to install and configure your application stack.

The only thing that you must create is your code. Elastic Beanstalk is designed to make deploying your application a quick and easy process. It supports a range of platforms, including Docker, Go, Java, .NET, Node.js, PHP, Python, and Ruby.

AWS Elastic Beanstalk deploys your code on:

- Apache Tomcat for Java applications
- Apache HTTP Server for PHP and Python applications
- NGINX or Apache HTTP Server for Node.js applications
- Passenger or Puma for Ruby applications
- Microsoft Internet Information Services (IIS) for .NET, Java SE, Docker, and Go applications.

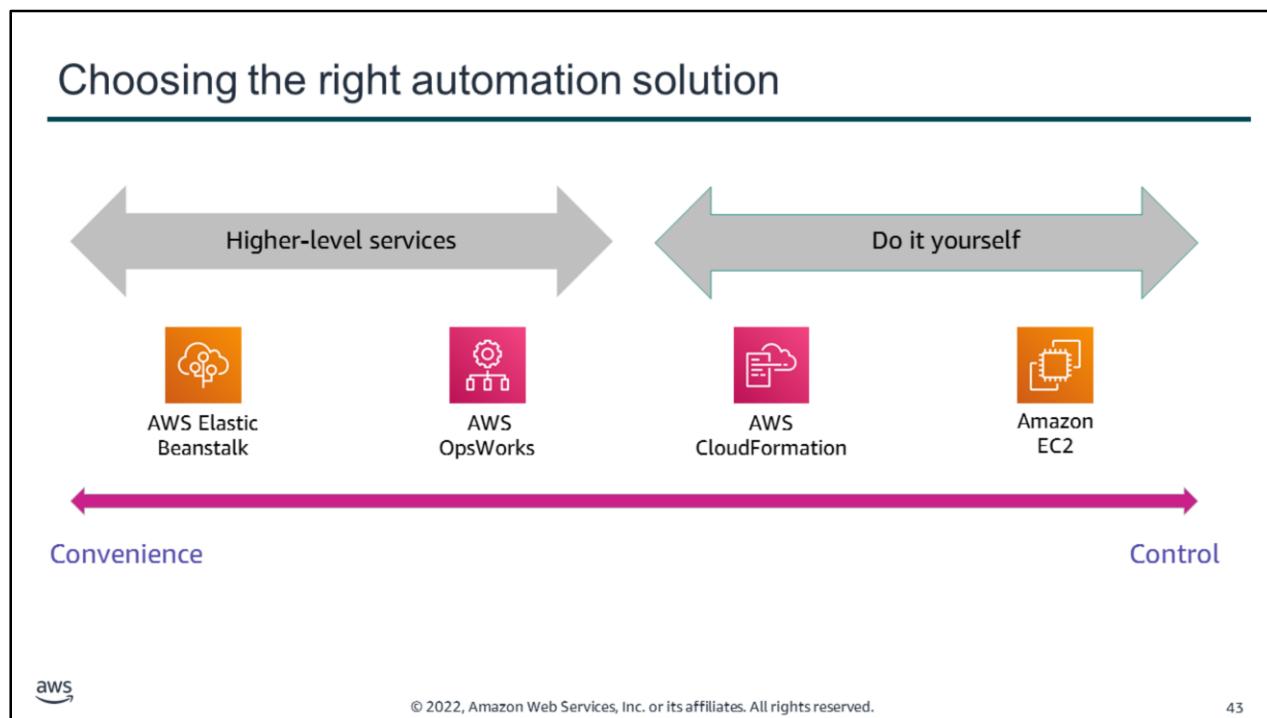


You can choose from two types of environments when you work with Elastic Beanstalk. The single-instance environment enables you to launch a single EC2 instance and it does not include load balancing or automatic scaling. The other type of environment—which is in this example—can launch multiple EC2 instances, and it includes load balancing and an automatic scaling configuration. A managed database layer is optional.

The AWS resources that are created by Elastic Beanstalk are visible in your AWS account. For example, after you create an Elastic Beanstalk application, open the AWS Management Console and then open the Amazon EC2 console. You will see the instances that Elastic Beanstalk is managing on your behalf. In the example, two EC2 instances were created. Each EC2 instance runs the Amazon Linux guest OS, with Java 8 and the Apache Tomcat 8.5 web server installed. Your application code will run on these servers. Automatic scaling is configured, so if the load starts to stress the resources on these two instances (such as excess CPU utilization for more than 5 minutes), more application server instances will be launched automatically. This example also shows that an RDS database instance is available and accessible from the EC2 instances. You can store your application data in the database and use structured query language (SQL) in your application code to access and update this data. Elastic Beanstalk manages the database instance and helps maintain connectivity between the EC2 instances and the database.

Elastic Beanstalk creates and manages scalable environments. You can configure the Auto Scaling group to automatically scale your application up to handle massive traffic loads. It also provides

you with a unique domain name for your application environment. The URL syntax is <your-application>.elasticbeanstalk.com. You can also resolve your own domain name to the provided domain name by using Amazon Route 53.



You were introduced to at least four AWS services in this module. One frequently asked question is about the multiple services that provide application management capabilities: where is the line between them, or which service should be used under which circumstances? Your decision should depend on the relative level of convenience and control that you need.

Elastic Beanstalk is an easy-to-use application service for building web applications that run on Java, PHP, Node.js, Python, Ruby, or Docker. If you want to upload your code and don't need to customize your environment, Elastic Beanstalk might be a good choice for you.

OpsWorks enables you to launch an application, define its architecture, and define the specification for each component, including package installation, software configuration, and resources (such as storage). You can use templates for common technologies (applications servers, databases, and others), or you can build your own template.

Both Elastic Beanstalk and OpsWorks provide a higher level of service than authoring and maintaining AWS CloudFormation templates to create stacks, or managing EC2 instances directly. However, the correct choice of service (or combinations of services) to use depends on your needs. These tools are all available to you. As an architect, you must decide which services will be the most appropriate for your use case.

Section 5 key takeaways



- Elastic Beanstalk creates and manages a scalable and highly available web application environment that enables you to focus on the application code
- You can author your Elastic Beanstalk application code in Java, .NET, PHP, Node.js, Python, Ruby, Go, or Docker
- AWS resources that are created by Elastic Beanstalk are fully transparent—they are visible in the AWS Management Console service page views
- No extra charge for Elastic Beanstalk – you pay only for the underlying resources that are used



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

44

Some key takeaways from this section of the module include:

- AWS Elastic Beanstalk creates and manages a scalable and highly available web application environment that enables you to focus on the application code
- You can author your Elastic Beanstalk application code in Java, .NET, PHP, Node.js, Python, Ruby, Go, or Docker
- AWS resources that are created by Elastic Beanstalk are fully transparent—they are visible in the AWS Management Console service page views
- No extra charge for Elastic Beanstalk – you pay only for the underlying resources that are used

Module 10 – Challenge Lab: Automating Infrastructure Deployment



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

45

You will now complete the Module 10 – Challenge Lab: Automating Infrastructure Deployment.

The business need: Implement IaC

- The café now has locations in multiple countries and must start automating to keep growing.
- They need a way to consistently deploy, manage, and update café resources across multiple AWS services.
- They want to be able to reliably create repeatable environments across AWS Regions, to serve both development and production needs.



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

46

For too long, the café has been creating their AWS resources and configured their applications manually. That approach worked well as a way for the café to quickly develop a web presence and build out an infrastructure that supports the needs of employees and customers. However, they find it challenging to replicate their deployments to new AWS Regions so that they can support new cafe locations in multiple countries.

The café would also like to have separate development and production environments that reliably have matching configurations. They realize that they must start automating to support continued growth.

Challenge lab: Tasks

1. Creating an AWS CloudFormation template from scratch
2. Configuring the bucket as a website and updating the stack
3. Cloning a CodeCommit repository that contains AWS CloudFormation templates
4. Creating a new network layer with AWS CloudFormation, CodeCommit, and CodePipeline
5. Updating the network stack
6. Defining an EC2 instance resource and creating the application stack
7. Duplicating the café network and website to another AWS Region



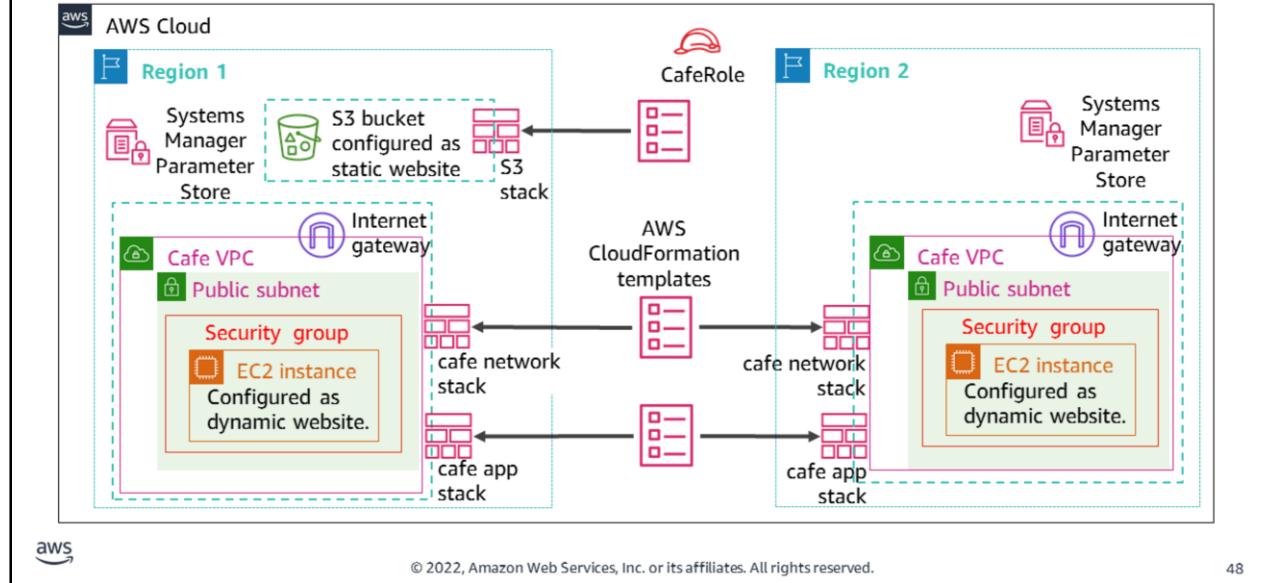
© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

47

In this challenge lab, you will complete the following tasks:

1. Creating an AWS CloudFormation template from scratch
2. Configuring the bucket as a website and updating the stack
3. Cloning a CodeCommit repository that contains AWS CloudFormation templates
4. Creating a new network layer with AWS CloudFormation, CodeCommit, and CodePipeline
5. Updating the network stack
6. Defining an EC2 instance resource and creating the application stack
7. Duplicating the café network and website to another AWS Region

Challenge lab: Final product



This diagram shows the completed architecture that you will build in the challenge lab.



It is now time to start the challenge lab.

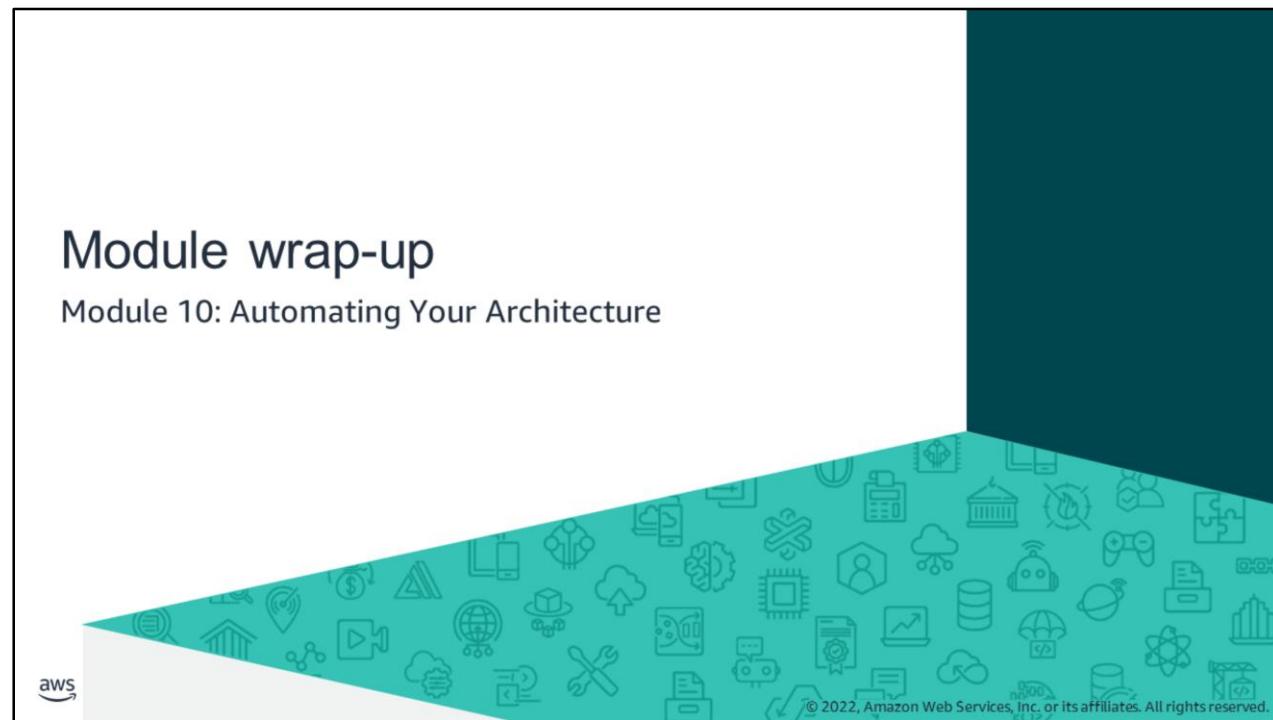
Challenge lab debrief: Key takeaways



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

50

Your educator might choose to lead a conversation about the key takeaways from this guided lab after you have completed it.



It's now time to review the module and wrap up with a knowledge check and discussion of a practice certification exam question.

Module summary

In summary, in this module, you learned how to:

- Recognize when to automate and why
- Identify how to model, create, and manage a collection of AWS resources using AWS CloudFormation
- Use the Quick Start AWS CloudFormation templates to set up an architecture
- Indicate how to use AWS System Manager and AWS OpsWorks for infrastructure and deployment automation
- Indicate how to use AWS Elastic Beanstalk to deploy simple applications



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

52

In summary, in this module, you learned how to:

- Recognize when to automate and why
- Identify how to model, create, and manage a collection of AWS resources using AWS CloudFormation
- Use the Quick Start AWS CloudFormation templates to set up an architecture
- Indicate how to use AWS System Manager and AWS OpsWorks for infrastructure and deployment automation
- Indicate how to use AWS Elastic Beanstalk to deploy simple applications

Complete the knowledge check



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

53

It is now time to complete the knowledge check for this module.



Sample exam question

Consider a situation where you want to create a single AWS CloudFormation template that is capable of creating both a production environment that spans two Availability Zones, and a development environment that exists in a single Availability Zone.

Which optional section of the AWS CloudFormation template will you want to make use of to configure the logic that will support this?

Choice	Response
A	Resources
B	Outputs
C	Conditions
D	Description

© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

54

Look at the answer choices and rule them out based on the keywords.

Sample exam question answer



Consider a situation where you want to create a single AWS CloudFormation template that is capable of creating both a production environment that spans two Availability Zones, and a development environment that exists in a single Availability Zone.

Which optional section of the AWS CloudFormation template will you want to make use of to configure the logic that will support this?

The correct answer is C.

The keywords in the question are a single AWS CloudFormation template, two Availability Zones, single Availability Zone, and optional section.

© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

55

The following are the keywords to recognize: **a single AWS CloudFormation template, two Availability Zones, single Availability Zone, and optional section.**

The correct answer is C – Conditions. The optional Conditions section contains statements that define the circumstances under which entities are created or configured.

Resources is not an optional section of an AWS CloudFormation template. The Outputs section cannot affect how many AWS resources will be deployed by the template when the stack is run. The Description section does not affect the configuration.

Additional resources

- [Overview of Deployment Options on AWS](#)
- [Working with AWS CloudFormation Templates](#)
- [AWS CloudFormation Sample Templates](#)
- [AWS OpsWorks Stacks FAQs](#)
- [AWS Systems Manager Features](#)
- [AWS Elastic Beanstalk FAQs](#)

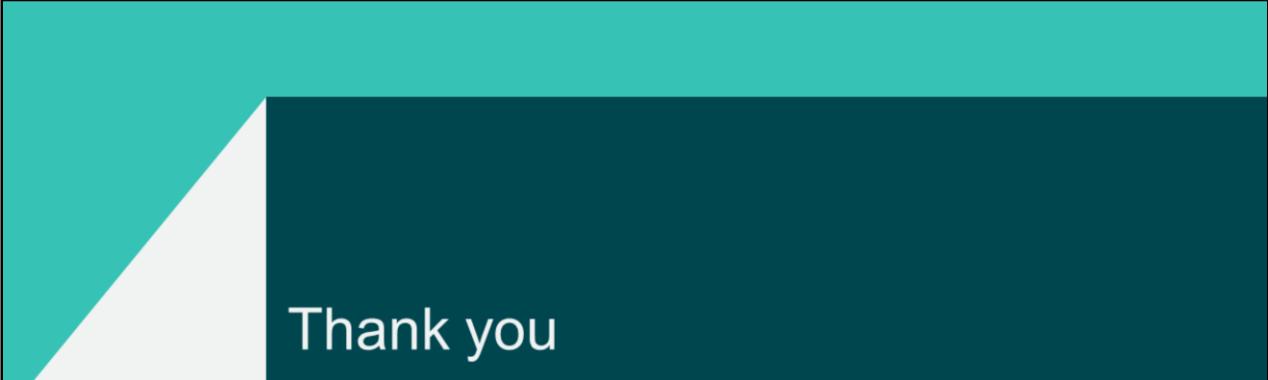


© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

56

If you want to learn more about the topics covered in this module, you might find the following additional resources helpful:

- [Overview of Deployment Options on AWS](#)
- [Working with AWS CloudFormation Templates](#)
- [AWS CloudFormation Sample Templates](#)
- [AWS OpsWorks Stacks FAQs](#)
- [AWS Systems Manager Features](#)
- [AWS Elastic Beanstalk FAQs](#)



Thank you

Corrections, feedback, or other questions?
Contact us at <https://support.aws.amazon.com/#/contacts/aws-academy>.
All trademarks are the property of their owners.



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

57

Thank you for completing this module.



**AWS Academy Cloud Architecting
Module 11 Student Guide
Version 2.0.8**

200-ACACAD-20-EN-SG

© 2022 Amazon Web Services, Inc. or its affiliates. All rights reserved.

This work may not be reproduced or redistributed, in whole or in part, without prior written permission from Amazon Web Services, Inc. Commercial copying, lending, or selling is prohibited.

All trademarks are the property of their owners.

Contents

Module 11: Caching Content	4
----------------------------	---



Module 11: Caching Content

AWS Academy Cloud Architecting

© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

Welcome to Module 11: Caching Content.

Module overview

Sections

1. Architectural need
2. Overview of caching
3. Edge caching
4. Caching web sessions
5. Caching databases

Lab

- Guided Lab: Streaming Dynamic Content Using Amazon CloudFront



Knowledge check



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

2

This module includes the following sections:

1. Architectural need
2. Overview of caching
3. Edge caching
4. Caching web sessions
5. Caching databases

The module also includes a guided lab in which you will learn how to stream dynamic content by using Amazon CloudFront.

Finally, you will be asked to complete a knowledge check that will test your understanding of key concepts covered in this module.

Module objectives

At the end of this module, you should be able to:

- Identify how caching content can improve application performance and reduce latency
- Identify how to design architectures that use edge locations for distribution and distributed denial of service (DDoS) protection
- Create architectures that use Amazon CloudFront to cache content
- Recognize how session management relates to caching
- Describe how to design architectures that use Amazon ElastiCache



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

3

At the end of this module, you should be able to:

- Identify how caching content can improve application performance and reduce latency
- Identify how to design architectures that use edge locations for distribution and distributed denial of service (DDoS) protection
- Create architectures that use Amazon CloudFront to cache content
- Recognize how session management relates to caching
- Describe how to design architectures that use Amazon ElastiCache

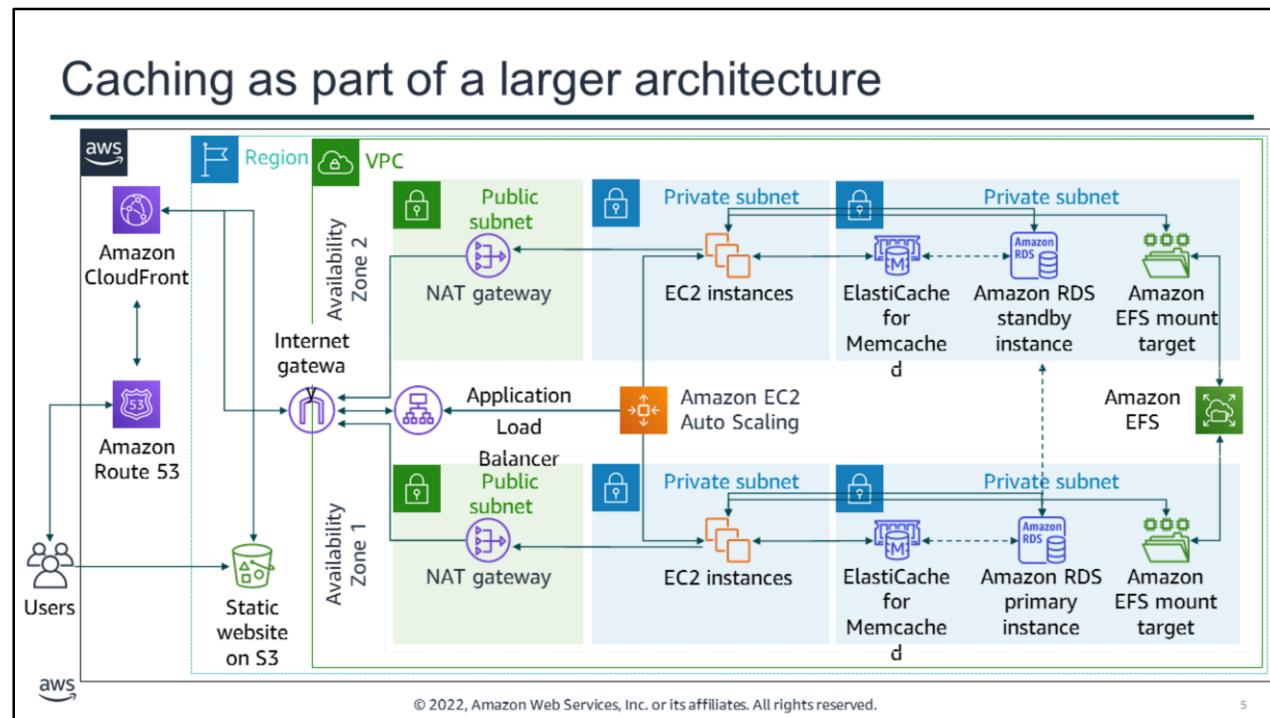
Section 1: Architectural need

Module 11: Caching Content



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

Introducing Section 1: Architectural need.



In this module, you will learn how to implement caching in your networking environment. The final components of the architecture diagram (that is, Amazon ElastiCache and Amazon CloudFront) are introduced in this module, and have been revealed. This module will also cover database caching with Amazon DynamoDB.

Café business requirement

The capacity of the café's infrastructure is constantly being overloaded with the same requests. This inefficiency is increasing cost and latency.



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

6

The capacity of the café's infrastructure is constantly being overloaded with the same requests for static content, such as images of menu items. This situation is increasing both cost and latency. Sofía and Nikhil want to identify and cache frequently accessed static content to reduce latency and improve the customer experience. Every time a customer loads the menu, it is served from the cache. However, when menu items change, the request is routed to the database, and the cache is updated.

In addition, local celebrities are starting to endorse the café through video testimonials. Sofía and Nikhil must use edge caching for streaming data so they can serve appropriate, regionally-appealing content based on the geolocation of the user who is loading the site.

Section 2: Overview of caching

Module 11: Caching Content



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

Introducing Section 2: Overview of caching.

Caching: Trading capacity for speed



- Is a high-speed data storage layer
- Stores a subset of data
- Increases data retrieval performance
- Reduces the need to access the underlying slower storage layer



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

8

Speed is important, whether your application serves the latest news, a top-10 leaderboard, a product catalog, or sells tickets to an event. The speed at which you deliver content affects the success of your application. If someone wants data, whether for a webpage or a report that drives business decisions, you can deliver that data much faster if it's cached.

In computing, a *cache* is a high-speed data storage layer. Unlike a database, which usually stores data in a complete and durable form, a cache transiently stores a subset of data. The primary purpose of a cache is to increase the performance of data retrieval by reducing the need to access the underlying, slower storage layer. Future requests for cached data are served faster than requests that access the data's primary storage location.

Caching trades capacity for speed, and it enables you to efficiently reuse previously retrieved or computed data.

The data in a cache is generally stored in fast-access hardware, such as random access memory (RAM).

Cache example (1 of 2)

Travel time = 30 minutes



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

9

To illustrate how a cache improves performance, consider the example of making a trip to a hardware store.

If the store is miles away, it takes you considerable effort to go there every time you need something.

Cache example (2 of 2)

Travel time = 2 minutes



Your house



Storage unit



Hardware store



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

10

Instead, you can store the supplies that you use regularly in a storage unit close to your house. Thus, it takes you less time to access these supplies than it would to go all the way to the hardware store.

However, you could still go to the store to refresh your supplies.

In this example, the storage unit is analogous to a cache.

What should you cache?



Data that requires a slow and expensive query to acquire



Relatively static and frequently accessed data—for example, a profile for your social media website



Information that can be stale for some time, such as a publicly traded stock price



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

11

When you decide what data to cache, consider these factors:

Speed and expense – Time-consuming database queries and frequently used, complex queries often create bottlenecks in applications. Generally, if the data requires a slow and expensive query to acquire, it's a candidate for caching. For example, queries that perform joins on multiple tables are slower and more expensive than simple queries on single tables. However, even data that requires a relatively quick and simple query might still be a candidate for caching, depending on other factors.

Data and access patterns – Determining what to cache also involves understanding the data itself and its access patterns. For example, it doesn't make sense to cache webpages that return search results that are unusually dynamic in nature. For caching to provide meaningful benefits, the data should be relatively static and frequently accessed, such as a personal profile on a social media site. Conversely, you don't want to cache data if caching it provides no speed or cost advantage. For example, it doesn't make sense to cache webpages that return the results of a search because these queries and results are almost always unique.

Staleness – By definition, cached data is stale data. Even if it's not stale in certain circumstances, cached data should be considered and treated as stale. When you determine whether your data is a candidate for caching, you must also determine your application's tolerance for stale data. Your application might be able to tolerate stale data in one context, but not another. For example,

consider an application that serves a publicly traded stock price on a website. With this application, a short delay might be acceptable if it has a disclaimer that prices might be delayed up to n minutes. But when an application must serve the stock price to a broker who is making a sale or purchase, you want real-time data.

Benefits of caching



Improves application speed



Reduces response latency



Reduces database access time



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

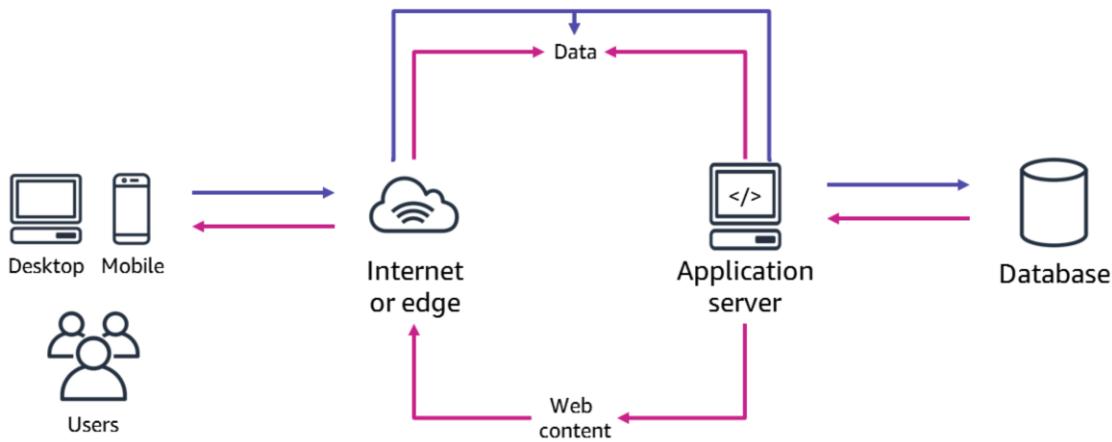
12

A cache provides high throughput, low-latency access to commonly accessed application data by storing the data in memory.

Caching can:

- Improve the speed of your application.
- Reduce the response latency that users experience with your application.
- Reduce application processing time and database access time for read-heavy workloads, such as social networking, gaming, media sharing, and Q&A portals. Write-heavy applications typically do not see as great a benefit from caching. However, even write-heavy applications normally have a read/write ratio greater than 1, which implies that read caching can still be beneficial.

Caching throughout the data journey



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

13

This simple web application architecture shows how data flows from and to the user. You can use caching at each layer to improve the overall performance and usability of your application. These layers include operating systems, networking layers like content delivery networks (CDNs) and Domain Name Systems (DNS), web applications, and databases.

In this architecture, you can use caching to:

- Accelerate retrieval of information from websites
- Store a mapping of domain names to IP addresses
- Accelerate retrieval of web content from web servers or application servers
- Accelerate application performance and data access
- Reduce latency that is associated with database query requests

In this module, you will learn how different AWS services support caching at these various layers.

Section 2 key takeaways



- A cache provides high throughput, low-latency access to commonly accessed application data by storing the data in memory
- When you decide what data to cache, consider speed and expense, data and access patterns, and your application's tolerance for stale data
- Caches can be applied and used throughout various layers of technology, including operating systems, networking layers, web applications, and databases

© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

14

Some key takeaways from this section of the module include:

- A cache provides high throughput, low-latency access to commonly accessed application data by storing the data in memory
- When you decide what data to cache, consider speed and expense, data and access patterns, and your application's tolerance for stale data
- Caches can be applied and used throughout various layers of technology, including operating systems, networking layers, web applications, and databases

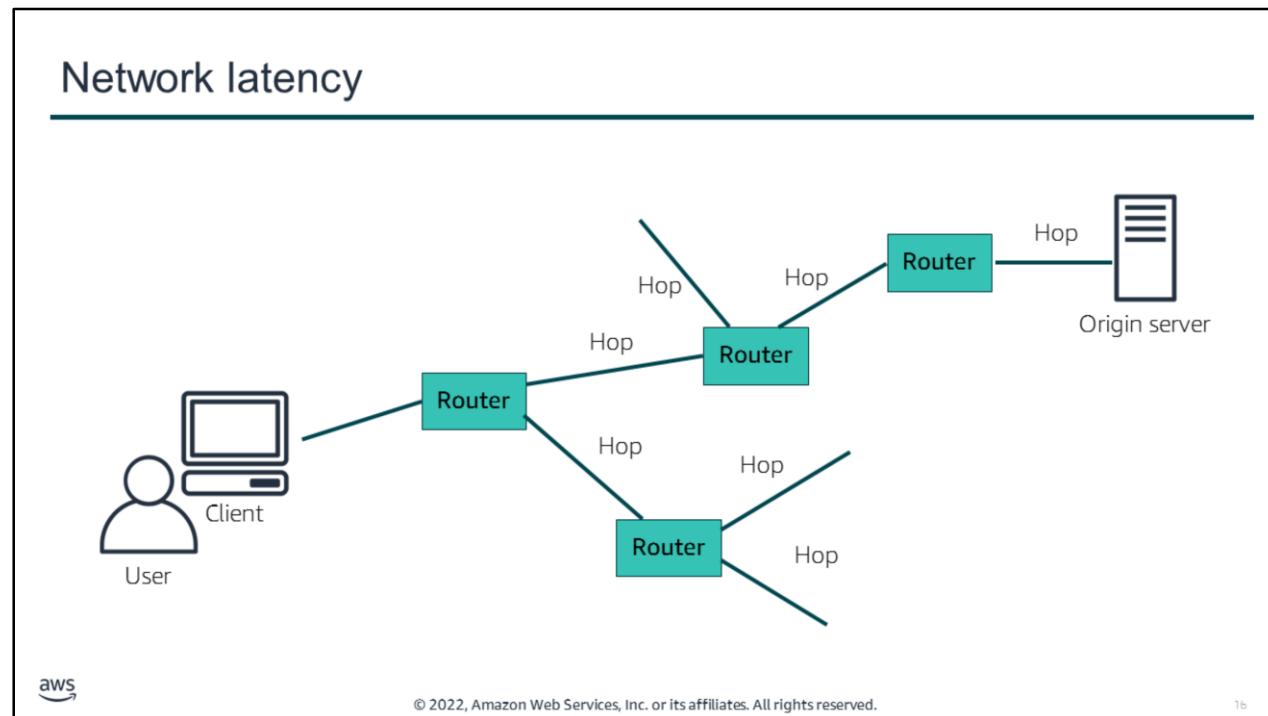
Section 3: Edge caching

Module 11: Caching Content



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

Introducing Section 3: Edge caching.



When someone browses to your website or uses your application, their request is routed through many different networks to reach your origin server. The origin server—which is also referred to as *origin*—stores the original, definitive versions of your objects (for example, web objects, images, and media files). The number of network hops and the distance that the request travels can significantly affect the performance and responsiveness of your website.

Further, network latency can depend on the geographic location of the origin server. When your web traffic is geographically dispersed, it's not always feasible (or cost-effective) to replicate your entire infrastructure across the globe. In this case, a content delivery network (CDN) can be useful.

Content delivery network (CDN)

- Is a globally distributed system of caching servers
- Caches copies of commonly requested files (static content)
- Delivers a local copy of the requested content from a nearby cache edge or Point of Presence
- Improves application performance and scaling



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

17

A content delivery network (CDN) is a globally distributed system of caching servers. A CDN caches copies of commonly requested files that are hosted on the application origin server. These files can include static content, such as HTML, CSS, JavaScript, image, and video files. The CDN delivers a local copy of the requested content from a cache edge or Point of Presence (PoP) that provides the fastest delivery to the requester.

For more information about caching with CDNs, see [Content Delivery Network \(CDN\) Caching](#).

Amazon CloudFront



Amazon
CloudFront

- Is the Amazon global CDN
- Is optimized for all delivery use cases, with a multi-tier cache by default and extensive flexibility
- Provides an extra layer of security for your architectures
- Supports WebSockets and HTTP or HTTPS methods



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

18

Amazon CloudFront is a global CDN service that accelerates the delivery of content to users. Such content might be static and dynamic content, media files that use HTTP or HTTPS, and streaming video (both video on demand and live streaming). Like other AWS services, CloudFront is a self-service, pay-per-use offering, which requires no long-term commitments or minimum fees.

The CDN offers a multi-tier cache by default. Regional edge caches improve latency and lower the load on your origin servers when the object is not already cached at the edge. In addition, the CDN offers multiple options for streaming your media, both pre-recorded files and live events. It offers them at the sustained, high throughput that is required for 4K delivery to global viewers.

CloudFront provides both network-level and application-level protection. Your traffic and applications benefit through various built-in protections, such as AWS Shield Standard, at no extra cost. You can also use configurable features such as AWS Certificate Manager (ACM) to create and manage custom SSL certificates at no extra cost. CloudFront supports Secure Sockets Layer/Transport Layer Security (SSL/TLS) protocols.

CloudFront supports real-time, bidirectional communication over the WebSocket protocol. This persistent connection enables clients and servers to send real-time data to one another without the cost of repeatedly opening connections. It is especially useful for communications applications, such as chat, collaboration, gaming, and financial trading. CloudFront also supports HTTP methods (DELETE, GET, HEAD, OPTIONS, PATCH POST, PUT), which improve the

performance of dynamic websites. These websites have web forms, comment and login boxes, add-to-cart buttons, and other features that upload data from users. Thus, you can use a single domain name to deliver your entire website through CloudFront, which accelerates both the download and upload parts of your website.

What type of content can you cache in an edge cache?

The diagram illustrates a screenshot of an Amazon.com webpage with annotations explaining what content can be cached:

- Secure**: Points to the HTTPS:// URL in the browser bar.
- Dynamic**: Points to the search bar and search results.
- User input**: Points to the search bar.
- Web objects**: Points to the main content area. A callout box states: "Can be cached!"
- Image**: Points to a thumbnail image of a Kindle Fire tablet. A callout box states: "Can be cached!"
- Video**: Points to a video player interface. A callout box states: "Can be cached!"

Annotations on the page itself:

- "Revolutionary on-device tech support"
- "Exclusively on Kindle Fire HDX tablets—live on-device tech support from an Amazon expert is just a tap away with the new "Mayday" button"
- "Live Support with Mayday"
- "NEW—Simply tap the "Mayday" button to be connected for free to an Amazon expert who can co-pilot you through any feature by drawing on your screen or pointing to your device. Or, if you prefer, do it yourself, or doing it for you—whatever works best. Mayday is available 24x7, 365 days a year, and it's FREE! Just tap the "Mayday" button to see your Amazon Tech advisor live on your screen, but they won't see you. 15 seconds or less is the Mayday response time goal."
- "Watch it in Action"
- "Kindle Fire HDX is easy to use right out of the box. But when you need extra assistance, the "Mayday" button is there—device works in these short commercials."

aws

© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

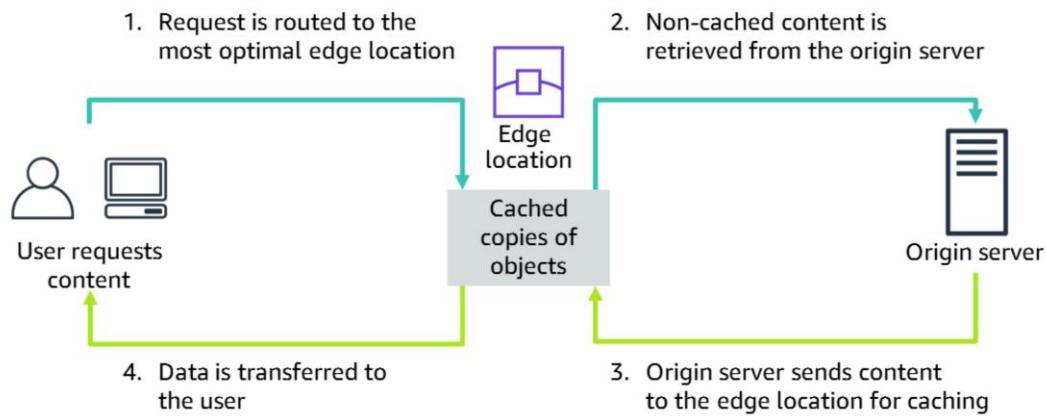
19

This example of an Amazon.com webpage shows how static and dynamic content can compose a dynamic web application. This web application is delivered through the HTTPS protocol for the encryption of user page requests and the pages that are returned from a web server. You can use a CDN or edge cache to cache static content. Such content might include web objects (for example, HTML documents, CSS style sheets, or JavaScript files), image files, and video files.

You cannot cache dynamically generated content or user-generated data. However, you can configure CloudFront to deliver this information from an application that runs on a custom origin. For example, it might be an EC2 instance or a web server.

Additionally, you can configure CloudFront to require that viewers use HTTPS to request your objects, so that connections are encrypted when CloudFront communicates with viewers. You also can configure CloudFront to use HTTPS to get objects from your origin, so that connections are encrypted when CloudFront communicates with your origin.

How caching works in Amazon CloudFront



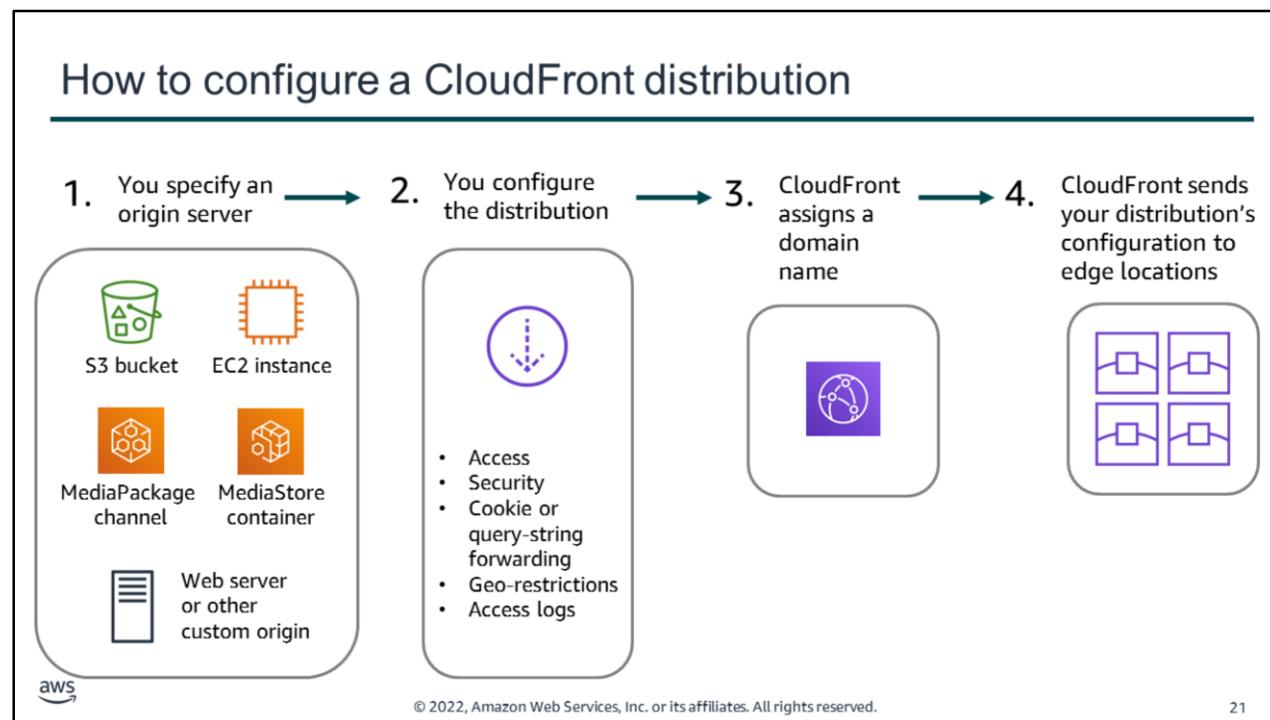
© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

20

Amazon CloudFront delivers your content to users through a worldwide network of data centers that are called *edge locations*.

When a user requests content that you are serving with CloudFront, DNS routes the request to the edge location that best serves the request. Typically, it is the nearest edge location that has the shortest latency. CloudFront checks the cache for the requested content (step 1). If the content is in the cache, CloudFront delivers it immediately to the user (step 4). The content might not be currently in the cache. If not, CloudFront forwards the request to the origin server that you identified as the source for the definitive version of your content (step 2). The origin server sends the content back to the edge location (step 3), and CloudFront then forwards the content to the user (step 4). It also adds the content to the cache in the edge location for the next time someone requests it.

As objects become less popular, individual edge locations might remove those objects to make room for more popular content. For the less popular content, CloudFront has *regional edge caches*. Regional edge caches are CloudFront locations that are deployed globally and close to your viewers. They are located between your origin server and the global edge locations that serve content directly to viewers. A regional edge cache has a larger cache than an individual edge location, so objects remain in the regional edge cache longer. This arrangement helps to keep more of your content closer to your viewers. It reduces the need for CloudFront to go back to your origin server, and it improves overall performance for viewers.



When you want to use CloudFront to distribute your content, you create a *distribution*.

1. You specify the origin server that hosts your files. Your origin server can be an S3 bucket, an AWS Elemental MediaPackage channel, an AWS Elemental MediaStore container, or a custom origin. For example, a custom origin might be an EC2 instance or your own web server.
2. You then specify details about how to track and manage content delivery. For example, you can specify whether you want your files to be available to everyone or only to certain users. You can also specify whether you want CloudFront to perform the following functions: create access logs that show user activity, forward cookies or query strings to your origin, or require users to use HTTPS to access your content.
3. CloudFront assigns a domain name to your new distribution.
4. CloudFront sends your distribution's configuration—but not the content—to all edge locations.

How to expire content

- Time to Live (TTL) –
 - Fixed period of time (expiration period)
 - Set by you
 - GET request to origin from CloudFront uses **If-Modified-Since** header
- Change object name –
 - Header-v1.jpg becomes Header-v2.jpg
 - New name forces **immediate** refresh
- Invalidate object –
 - Last resort: inefficient and expensive



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

22

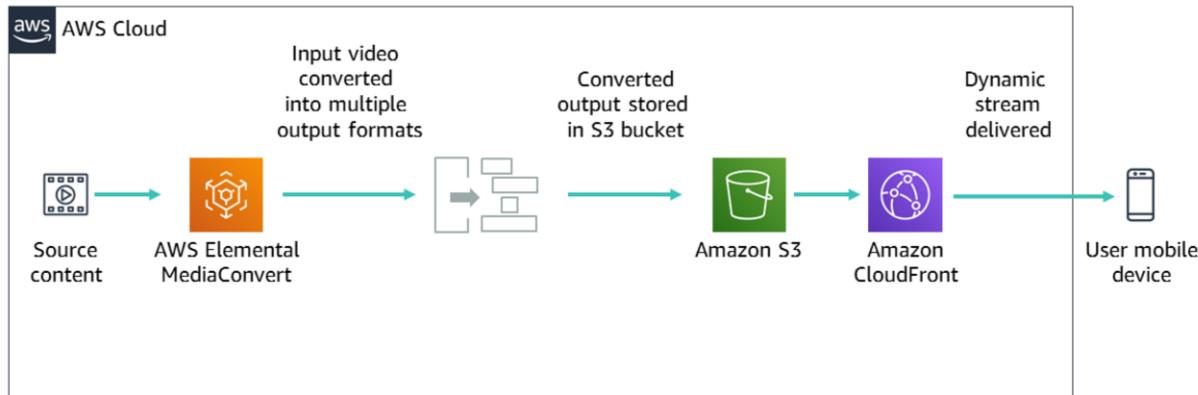
You can expire cached content in three ways:

- Time to Live (TTL) – With this method, you can control how long your files stay in a CloudFront cache before CloudFront forwards another request to your origin. Reducing the duration enables you to serve dynamic content. Increasing the duration means that your users get better performance because your files are more likely to be served directly from the edge cache. A longer duration also reduces the load on your origin. If you set the TTL for a particular origin to 0, CloudFront still caches the content from that origin. It then makes a GET request with an If-Modified-Since header. Thus, the origin has a chance to signal that CloudFront can continue to use the cached content if it hasn't changed at the origin. TTL is a good method to use if the replacement doesn't need to be immediate.
- Change object name – This method requires more effort, but replacement is immediate. Although you *can* update existing objects in a CloudFront distribution and use the same object names, it is not recommended. CloudFront distributes objects to edge locations only when the objects are requested—not when you put new or updated objects in your origin. For example, you might update an existing object in your origin with a newer version that has the same name. In that case, an edge location won't get that new version from your origin until both of the listed events occur.
- Invalidate object – This method is a bad solution because the system must forcibly interact with all edge locations. You should use this method sparingly and only for individual objects.

For more information about how to expire cache content, see [Managing How Long Content Stays](#)

[in an Edge Cache \(Expiration\).](#)

Example: Video on demand streaming



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

23

As you have learned, you can use CloudFront to deliver streaming video—both video on demand and live streaming.

For on-demand video streaming, you must use an encoder to format and package video content before CloudFront can distribute it. Examples of encoders include [AWS Elemental MediaConvert](#) and [Amazon Elastic Transcoder](#). The packaging process creates *segments*, which are static files that contain your audio, video, and captions content. It also generates manifest files, which describe what segments to play and the specific order to play them in. Package formats include Dynamic Adaptive Streaming over HTTP (DASH, or MPEG-DASH), Apple HTTP Live Streaming (HLS), Microsoft Smooth Streaming, and Common Media Application Format (CMAF).

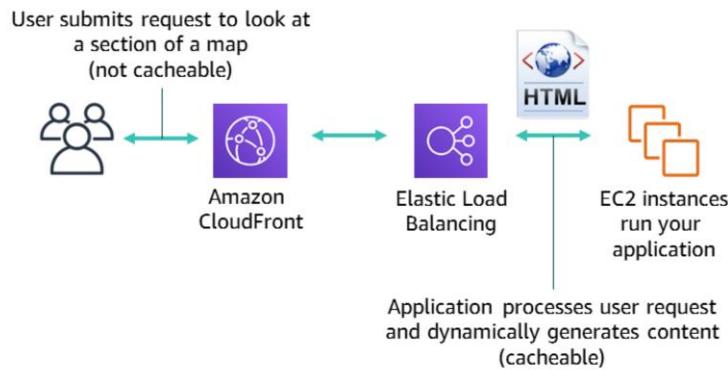
After converting the video into the output formats, you host the converted content in an S3 bucket, which is your origin server. You then use CloudFront to deliver the segment files to users around the world.

For more information on video streaming with CloudFront, see [Video on Demand and Live Streaming Video with CloudFront](#).

Example: Dynamically generated content

Use case: Map tiles

Problem: Need faster DB response time



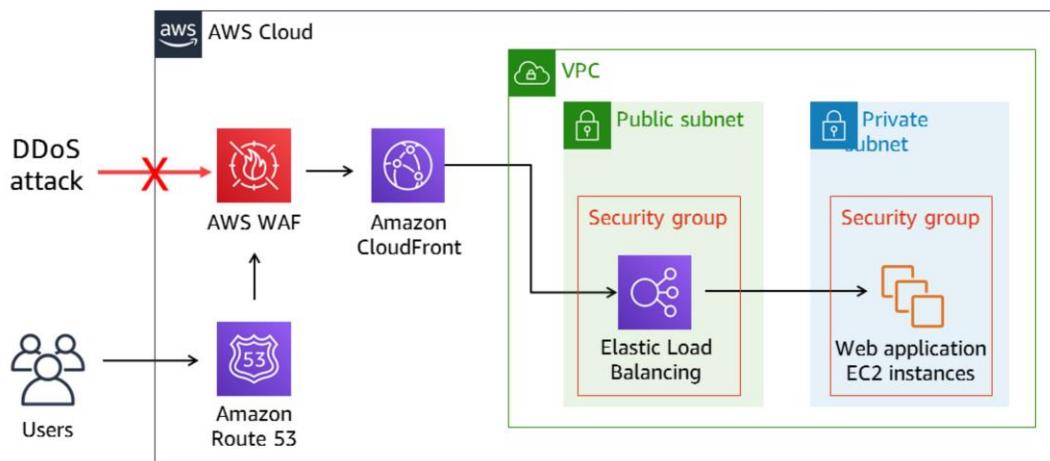
© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

24

In general, you cache only static content. However, you can have content that *appears* static (because of its URL), but which is built dynamically the first time that it's needed. This dynamic build can be useful when the content is reusable, but it can be expensive to create, and it can change infrequently.

Map tiles are the classic example. In this case, you cache places that people frequently view (such as major cities), but not places like remote areas. Generating all possible combinations of tiles would be prohibitively expensive and wasteful because most tiles would almost never be requested. Instead, the path component of each tile's URL can include the parameters needed to generate the tile. If the tile is already present in a particular CloudFront edge location, then it is served up directly. Otherwise, it is generated, returned to the edge location, and then used to satisfy future requests.

Example: DDoS mitigation



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

25

You can use CloudFront to improve the resiliency of your applications that run on AWS from distributed denial of service (DDoS) attacks. A DDoS attack is a deliberate attempt to make your website or application unavailable to users, for example, by flooding it with network traffic. To achieve this end, attackers use multiple sources to orchestrate an attack against a target. These sources might include distributed groups of malware-infected computers, routers, Internet of Things (IoT) devices, and other endpoints.

The following example shows a resilient architecture that can help prevent or mitigate DDoS attacks.

A DNS service, such as Amazon Route 53, can effectively connect users' requests to a CloudFront distribution. The CloudFront distribution then proxies requests for dynamic content to the infrastructure that hosts your application's endpoints. Both Route 53 DNS requests and subsequent application traffic that are routed through CloudFront are inspected inline. Always-on monitoring, anomaly detection, and mitigation against common infrastructure DDoS attacks are built into both Route 53 and CloudFront.

Common infrastructure attacks include synchronize/acknowledge (SYN/ACK) floods, User Datagram Protocol (UDP) floods, and reflection attacks. When the SYN flood attack threshold is exceeded, SYN cookies are activated to avoid dropping connections from legitimate clients.

Deterministic packet filtering drops malformed TCP packets and invalid DNS requests, and permits traffic to pass only if the traffic is valid for the service. Heuristics-based anomaly detection evaluates attributes such as type, source, and composition of traffic. Traffic is scored across many dimensions, and only the most suspicious traffic is dropped.

This method enables you to avoid false positives while you protect application availability. Route 53 is also designed to withstand DNS query floods. DNS query floods are real DNS requests that can continue for hours and attempt to exhaust DNS server resources. Route 53 uses shuffle sharding and anycast striping to spread DNS traffic across edge locations and help protect the availability of the service.

[AWS WAF](#) is a web application firewall. It enables you to monitor the HTTP and HTTPS requests that are forwarded to an Amazon API Gateway API, CloudFront, or Application Load Balancer. AWS WAF also enables you to control access to your content. For example, you can specify conditions such as the IP addresses that requests originate from or the values of query strings. Based on these conditions, API Gateway, CloudFront, or an Application Load Balancer responds with either the requested content or an HTTP 403 status code (Forbidden). You also can configure CloudFront to return a custom error page when a request is blocked.

For more information about how to improve the resiliency of your applications that run on AWS against DDoS attacks, see the following resources:

- [AWS Best Practices for DDoS Resiliency](#) AWS whitepaper
- [How to Help Protect Dynamic Web Applications Against DDoS Attacks by Using Amazon CloudFront and Amazon Route 53](#) AWS Security Blog post

Section 3 key takeaways



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

26

- Amazon CloudFront is a [global CDN service](#) that accelerates the delivery of content, including static and video, to users with no minimum usage commitments.
- CloudFront uses a global network that comprises [edge locations](#) and [regional edge caches](#) to deliver content to your users.
- To use CloudFront to deliver your content, you specify an [origin server](#) and configure a CloudFront [distribution](#). CloudFront assigns a domain name and sends your distribution's configuration to all of its edge locations.
- You can use Amazon CloudFront to [improve the resilience](#) of your applications that run on AWS from DDoS attacks.

Some key takeaways from this section of the module include:

- Amazon CloudFront is a global CDN service that accelerates the delivery of content, including static and video, to users with no minimum usage commitments.
- CloudFront uses a global network that comprises edge locations and regional edge caches to deliver content to your users.
- To use CloudFront to deliver your content, you specify an origin server and configure a CloudFront distribution. CloudFront assigns a domain name and sends your distribution's configuration to all of its edge locations.
- You can use Amazon CloudFront to improve the resilience of your applications that run on AWS from DDoS attacks.

Module 11 – Guided Lab: Streaming Dynamic Content Using Amazon CloudFront



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

27

You will now complete Module 11 – Guided Lab: Streaming Dynamic Content Using Amazon CloudFront.

Guided lab: Scenario

In this lab, you use [Amazon Elastic Transcoder](#) to convert a source video into multiple bitrates. You use [Amazon CloudFront](#) to deliver the dynamic, multiple bitrate stream to a connected device by using Apple HTTP Live Streaming (HLS) protocol.



Amazon Elastic
Transcoder



Amazon
CloudFront



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

28

In this lab, you use Amazon Elastic Transcoder to convert a source video into multiple bitrates. You use Amazon CloudFront to deliver the dynamic, multiple-bitrate stream to a connected device by using Apple HTTP Live Streaming (HLS) protocol. The stream can be played on any browser that supports the HLS protocol.

Apple HLS can dynamically adjust movie playback quality to match the available speed of wired or wireless networks by using an ordinary web server. It works by creating different quality streams. Each stream is then broken into chunks that are streamed sequentially to a client device. On the client's end, you can select streams of varying bitrates, which enable streaming sessions to adapt to different network speeds.

Guided lab: Tasks

1. Create an Amazon CloudFront distribution
2. Create an Amazon Elastic Transcoder pipeline
3. Test playback of the dynamic (multiple bitrate) stream



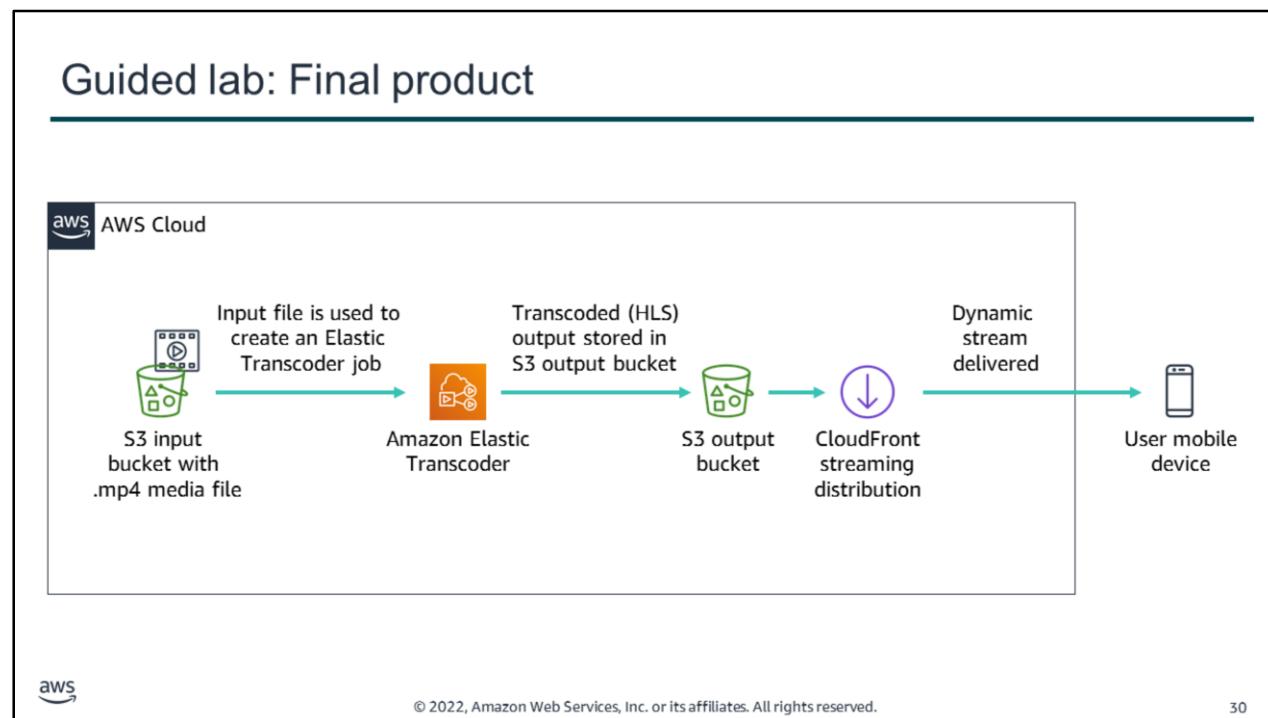
© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

29

In this guided lab, you will complete the following tasks:

1. Create an Amazon CloudFront distribution
2. Create an Amazon Elastic Transcoder pipeline
3. Test playback of the dynamic (multiple bitrate) stream

Guided lab: Final product



The diagram summarizes what you will have built after you complete the lab.



~ 30 minutes

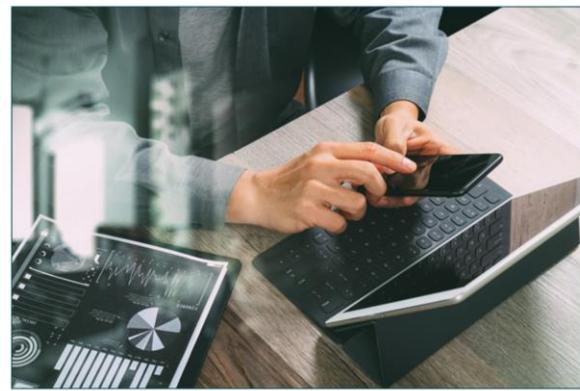
Begin Module 11 – Guided Lab: Streaming Dynamic Content Using Amazon CloudFront

aws

© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

It is now time to start the guided lab.

Guided lab debrief: Key takeaways



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

32

Your educator might choose to lead a conversation about the key takeaways from this guided lab after you have completed it.

Section 4: Caching web sessions

Module 11: Caching Content



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

Introducing Section 4: Caching web sessions.

Session management: Sticky sessions

Elastic Load
Balancing



Sticky sessions

Feature that enables a load balancer to route a request to the specific server that manages the user's session.

- Use client-side cookies
- Are cost-effective
- Speed up retrieval of sessions
- Have disadvantages –
 - Loss of sessions when you have an instance failure
 - Limit scalability: Uneven load distribution and increased latency



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

34

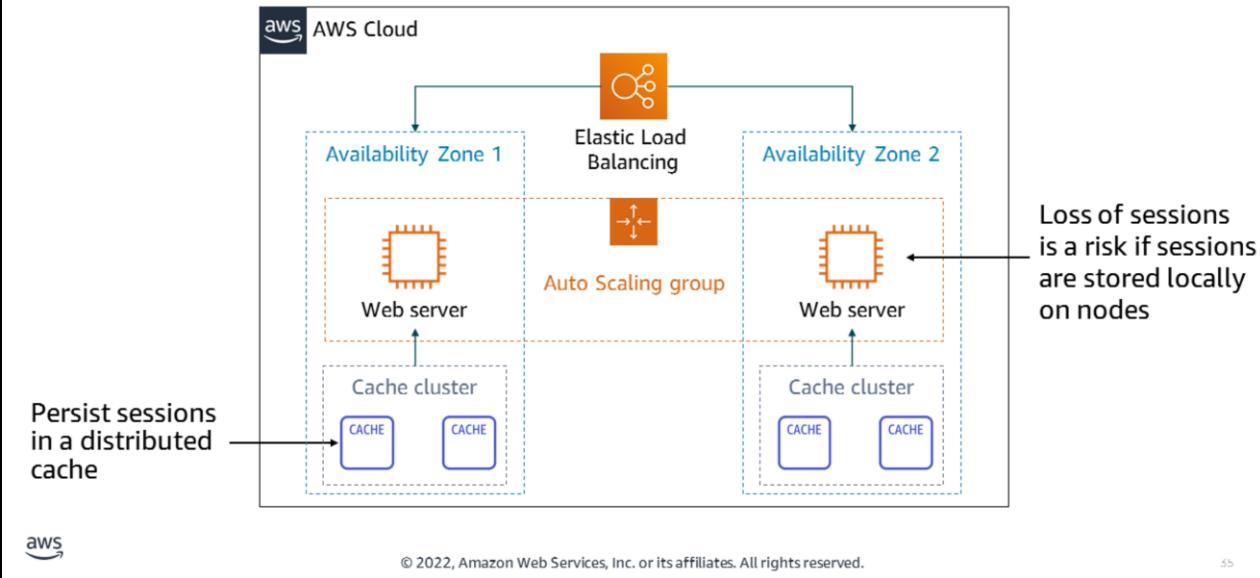
When a user or service interacts with a web application, it sends an HTTP request, and the application returns a response. A sequence of such transactions is called a *session*. Every request is independent of previous transactions. Therefore, sessions are used to manage user authentication and store user data while the user interacts with the application. For example, by using sessions, your users are not required to send their credentials for every request that they make to your server.

You can manage user sessions in various ways. (By default, a load balancer routes each request independently to the registered instance with the smallest load.) To use sticky sessions, the client must support cookies.

Sticky sessions are cost-effective because the sessions are stored on the web servers that are running your applications. Therefore, sticky sessions eliminate network latency and speed up retrieval of those sessions. However, in the event of instance failure, you are likely to lose the sessions that are stored on that instance.

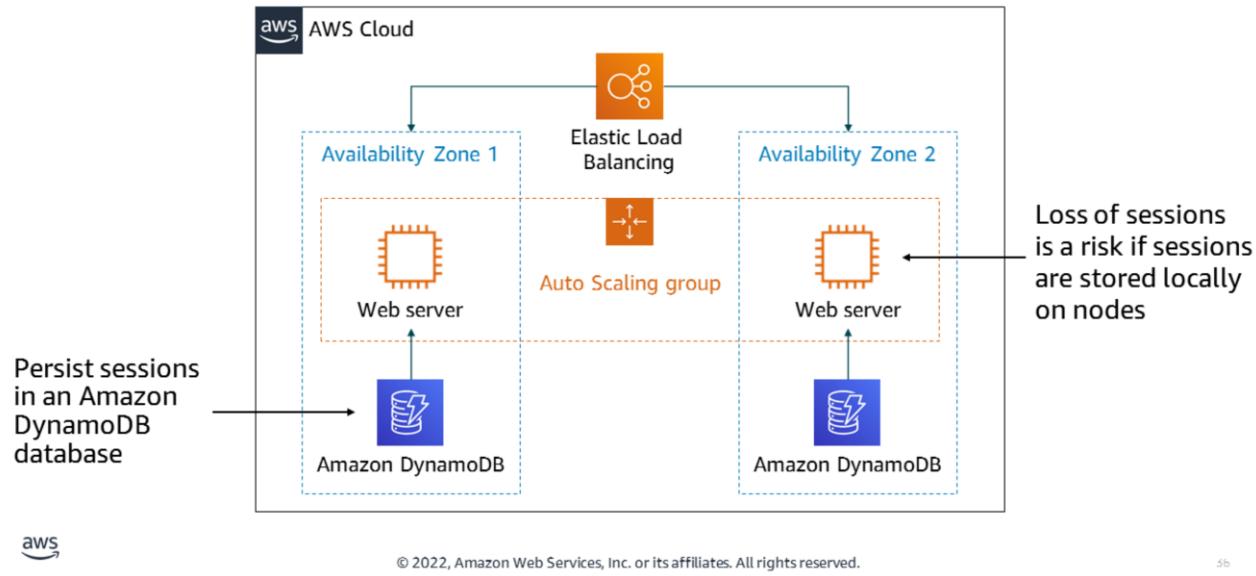
Another disadvantage of sticky sessions is that they can limit your application's scalability. With sticky sessions, the load balancer is unable to truly balance the load each time that it receives a request from a client. Sticky sessions force the load balancer to send all the requests to the original server where the session state was created. If that server is heavily loaded, then receiving so many requests can lead to unequal load across servers and affect user response time.

Instead of sticky sessions: Persist sessions inside a distributed cache

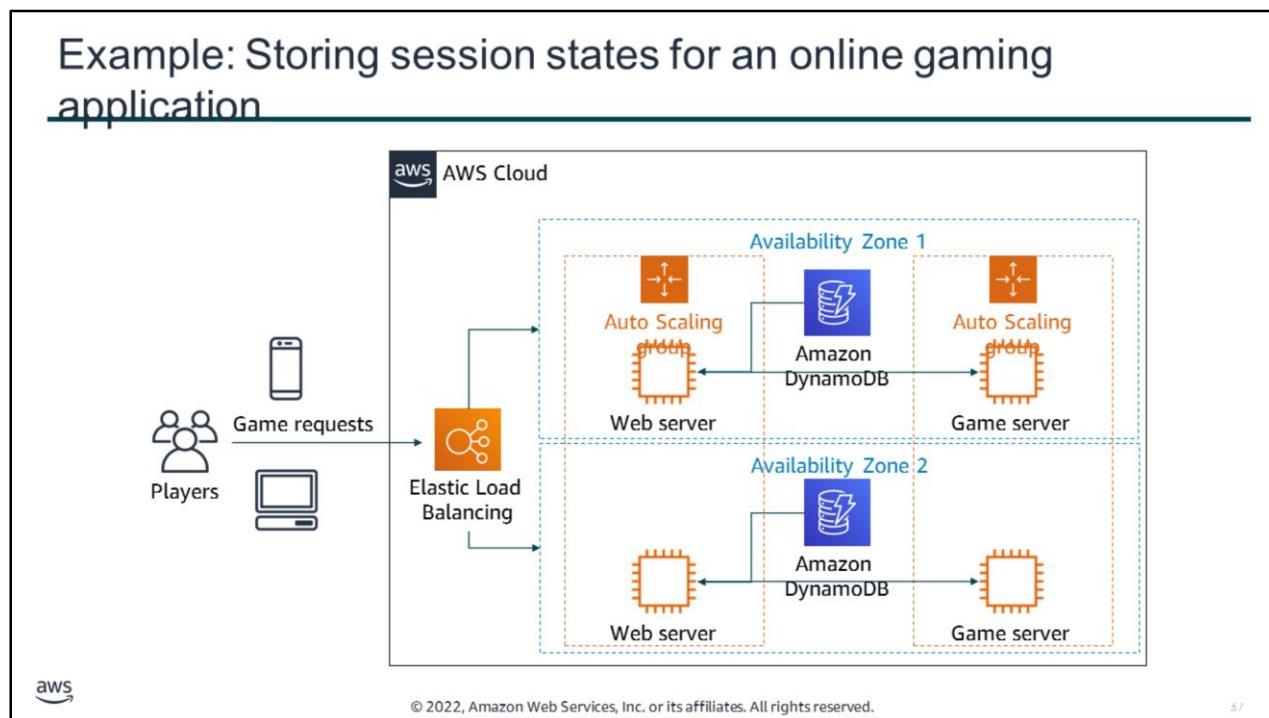


Instead of using sticky sessions, you can designate a layer in your architecture that can store sessions in a scalable and robust manner outside the instance. One option is to persist session data in a distributed cache, as this architecture diagram shows. It makes sense to implement this architecture in a dynamic environment when the number of web servers changes to accommodate load, and you don't want to risk losing sessions.

Instead of sticky sessions: Persist sessions inside a DynamoDB table



Another option is to persist session data in an Amazon DynamoDB database, as this diagram shows. With Amazon DynamoDB, you can set your scaling policy and have DynamoDB scale capacity up and down as necessary.



This architecture might support an online gaming application, where fast session retrieval is imperative. Game data is continuously updated as a player collects items, defeats enemies, receives gold, unlocks levels, and completes achievements. Each session event must be written to your database layer so that it isn't lost. Game makers store session history and other time-oriented data in DynamoDB for fast lookup by player, date, and time.

Each DynamoDB database table is associated with a throughput capacity. You can specify 1,000 writes per second, and DynamoDB scales the database in the background. As your needs change, you can update the capacity—and Amazon DynamoDB re-allocates resources as needed. This elasticity helps game developers: if your game becomes popular, you might suddenly scale from a few thousand players to millions of players. You can quickly and easily scale back down if needed.

DynamoDB maintains predictable, low-latency performance at any scale, which is crucial if your game grows to millions of latency-sensitive customers. You won't spend any time tuning the performance of DynamoDB.

To see a similar gaming architecture that includes DynamoDB, see this [AWS Big Data Blog post](#).

Section 4 key takeaways



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

38

- Sessions are used to manage user authentication and store user data while the user interacts with the application.
- You can manage sessions with sticky sessions, which is a feature of Elastic Load Balancing load balancers. Sticky sessions route requests to the specific server that's managing the user's session.
- You can also manage sessions by persisting session data outside the web server instance—for example, in a distributed cache or DynamoDB table.

Some key takeaways from this section of the module include:

- Sessions are used to manage user authentication and store user data while the user interacts with the application.
- You can manage sessions with sticky sessions, which is a feature of Elastic Load Balancing load balancers. Sticky sessions route requests to the specific server that is managing the user's session.
- You can also manage sessions by persisting session data outside the web server instance—for example, in a distributed cache or DynamoDB table.

Section 5: Caching databases

Module 11: Caching Content



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

Introducing Section 5: Caching databases.

When should you cache your database?



You are concerned about response times for your customer.



You have a high volume of requests that are inundating your database.



You would like to reduce your database costs.



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

40

As you learned earlier in this module, time-consuming database queries and complex queries can create bottlenecks in applications.

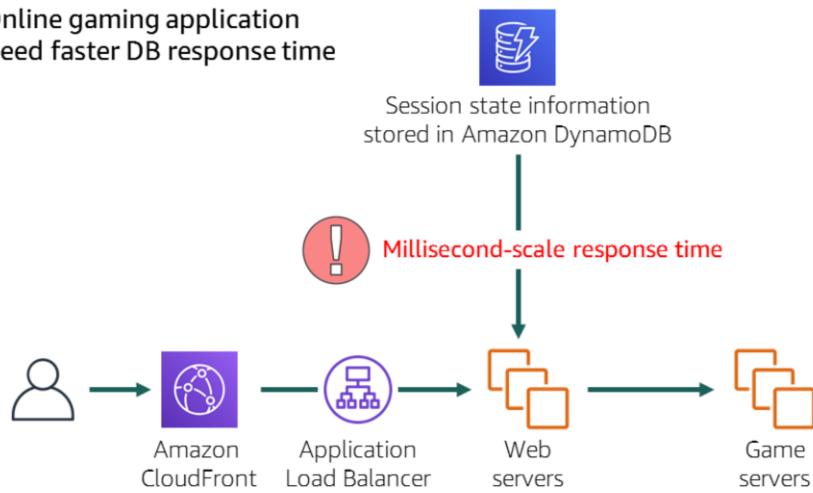
You should consider caching your database when you:

- Are concerned about response times for your customer. You might have latency-sensitive workloads that you want to speed up. Caching can help you increase throughput and reduce data retrieval latency, thus improving the performance of your application.
- Have a high volume of requests that inundate your database. You might have a large amount of traffic, and thus not getting the throughput that you need for that workload. Putting a caching layer next to your database can increase your throughput and help you achieve higher performance.
- Would like to reduce your database costs. Whether data is distributed in a disk-based NoSQL database or vertically scaled up in a relational database, scaling for high reads can be costly. A number of database read replicas might be necessary to match what a single in-memory cache node can deliver in terms of requests per second.

A database cache supplements your primary database by removing unnecessary pressure on it, typically in the form of frequently accessed read data. The cache itself can be in a number of areas, including your database, application, or as a standalone layer.

Using DynamoDB for state information

Use case: Online gaming application
Problem: Need faster DB response time



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

41

Consider a simplified version of the architecture for an online gaming application, where you are storing session state information in DynamoDB. In some cases, you might find that a millisecond-scale response time isn't fast enough for your application. Database caching can help with this problem.

Amazon DynamoDB Accelerator



Amazon
DynamoDB
Accelerator

Fully managed, highly available, in-memory cache for DynamoDB

- Extreme performance (microsecond-scale response time)
- Highly scalable
- Fully managed
- Integrated with DynamoDB
- Flexible
- Secure



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

42

Amazon DynamoDB Accelerator (DAX) is a fully managed, highly available, in-memory cache for DynamoDB. It delivers up to a performance improvement of up to 10 times—from milliseconds to microseconds—even at millions of requests per second. DAX does all the heavy lifting required to add in-memory acceleration to your DynamoDB tables. You are not required to manage cache invalidation, data population, or cluster management.

DAX provides the following benefits:

- **Extreme performance** – DynamoDB offers consistent single-digit millisecond latency. When DynamoDB and DAX are used together, you can achieve response times in microseconds for millions of requests per second for read-heavy workloads.
- **Highly scalable** – DAX has on-demand scaling. You can start with a three-node DAX cluster and add capacity as necessary, up to a 10-node cluster.
- **Fully managed** – Like DynamoDB, DAX is fully managed. DAX takes care of management tasks including provisioning, setup and configuration, software patching, and replicating data over nodes during scaling operations. DAX automates common administrative tasks such as failure detection, failure recovery, and software patching.
- **Integrated with DynamoDB** – DAX is API-compatible with DynamoDB, and it's not necessary to make any functional application code changes. Provision a DAX cluster, and use the DAX client software development kit (SDK) to point existing DynamoDB API calls at the DAX cluster. DAX handles the rest.
- **Flexible** – You can provision one DAX cluster for multiple DynamoDB tables, multiple DAX

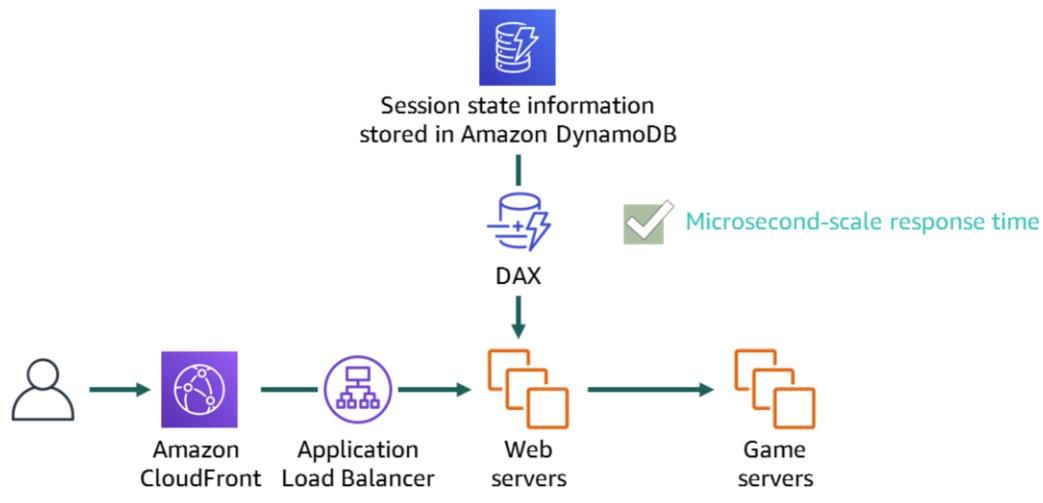
clusters for a single DynamoDB table, or a combination of both.

- **Secure** – DAX fully integrates with AWS services to enhance security. You can use AWS Identity and Access Management (IAM) to assign unique security credentials to each user and control each user's access to services and resources. Amazon CloudWatch enables you to gain system-wide visibility into resource utilization, application performance, and operational health. Integration with AWS CloudTrail enables you to easily log and audit changes to your cluster configuration. DAX supports Amazon Virtual Private Cloud (Amazon VPC) for secure and easy access from your existing applications. Tagging provides you more visibility to help you manage your DAX clusters.

The retrieval of cached data reduces the read load on existing DynamoDB tables. As a result, it might reduce provisioned read capacity and lower overall operational costs.

For more information about DAX, see this [AWS Database Blog post](#).

Using DynamoDB with DAX to accelerate response time

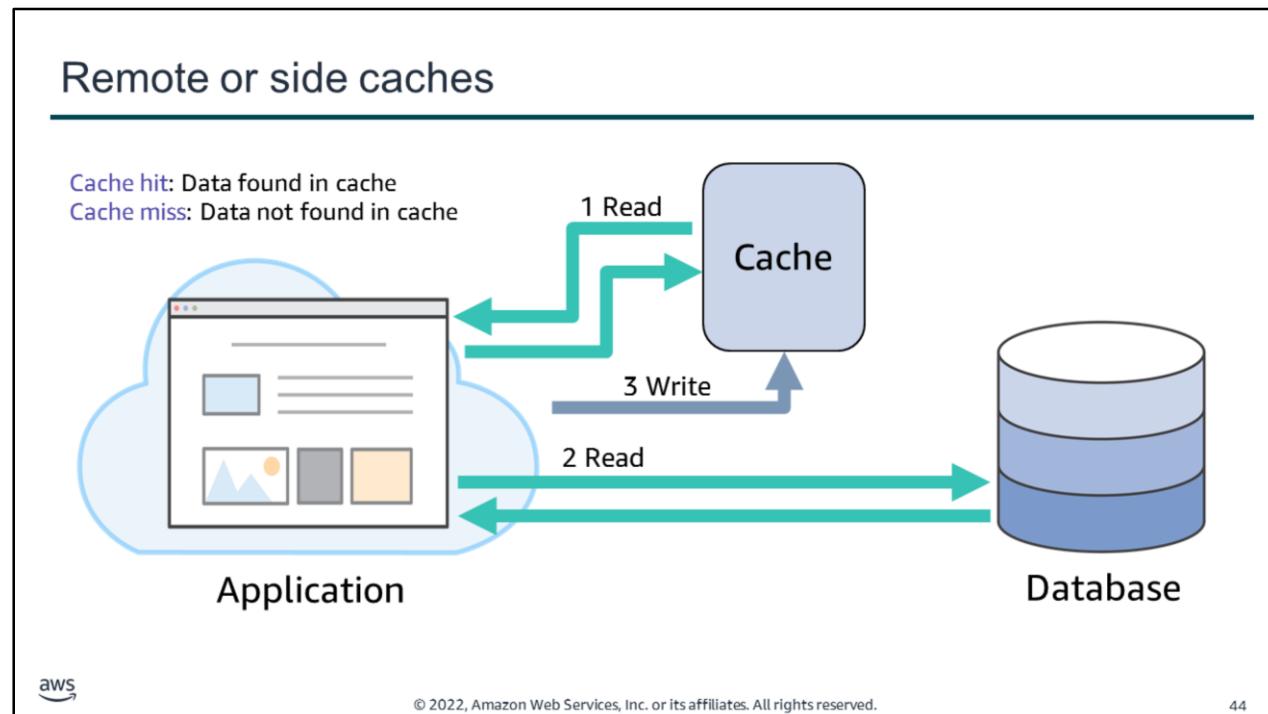


aws

© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

45

In the gaming example, the acceleration that you get by adding DAX to your architecture comes without needing to make any major changes in the game code. In this way, it simplifies deployment into your architecture. The one thing you must do is re-initialize your DynamoDB client with a new endpoint that points to DAX. It isn't necessary to make any changes to the rest of the code. DAX handles cache invalidation and data population without your intervention. This cache can help speed responsiveness when you run events that might cause a spike in players. An example of such an event is a seasonal downloadable content (DLC) offering or a new patch release.



DAX is a transparent cache. Another way to implement a database cache deployment is to use a remote or side cache. A side cache isn't directly connected to the database—instead, it's used adjacently to the database. Side caches are typically built on key-value NoSQL stores such as Redis or Memcached. They provide hundreds of thousands of requests, up to a million requests per second per cache node.

Side caches are typically used for read-heavy workloads. They work as follows:

1. For a given key-value pair, an application first tries to read the data from the cache. If the cache contains the data (called a *cache hit*), the value is returned.
2. If the intended key-value pair is not found in the cache (called a *cache miss*), the application fetches the data from the underlying database.
3. It's important that the data is present when the application needs it again. To ensure that it is, the key-value pair that's obtained from the database is then written to the cache.

Amazon ElastiCache



Amazon
ElastiCache

ElastiCache provides web applications with an in-memory data store in the cloud.

- Works as an in-memory data store and cache
- Offers high performance
- Is fully managed
- Is scalable
- Supports Redis and Memcached

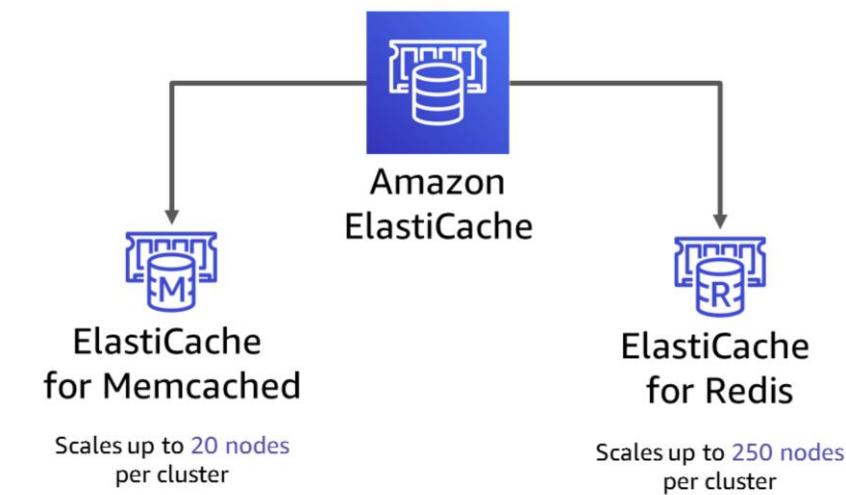


© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

45

Amazon ElastiCache is a side cache that works as an in-memory data store to support the most demanding applications, which require sub-millisecond response times. With ElastiCache, you don't need to perform management tasks such as hardware provisioning, software patching, setup, configuration, monitoring, failure recovery, and backups. ElastiCache continuously monitors your clusters to keep your workloads up and running so that you can focus on higher-value application development. Amazon ElastiCache can scale out, scale in, and scale up to meet fluctuating application demands. Write and memory scaling is supported with sharding. Replicas provide read scaling. ElastiCache supports two open-source in-memory databases: Redis and Memcached.

Redis and Memcached



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

46

ElastiCache for Memcached can scale up to 20 nodes per cluster. In contrast, ElastiCache for Redis can scale up to 250 nodes for increased data access performance. ElastiCache supports Amazon VPC, which enables you to isolate your cluster to the IP ranges that you choose for your nodes.

ElastiCache runs on the same highly reliable infrastructure that other AWS services use. ElastiCache for Redis provides high availability through Multi-AZ deployments with automatic failover. For Memcached workloads, data is partitioned across all nodes in the cluster. Thus, you can scale out to better handle more data when demand grows.

Memcached versus Redis comparison

Feature	Memcached	Redis
Sub-millisecond latency	Yes	Yes
Ability to scale horizontally for writes and storage	Yes	No
Multi-threaded performance	Yes	No
Advanced data structures	No	Yes
Sorting and ranking datasets	No	Yes
Publish/subscribe messaging	No	Yes
Multi-AZ deployments with automatic failover	No	Yes
Persistence	No	Yes



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

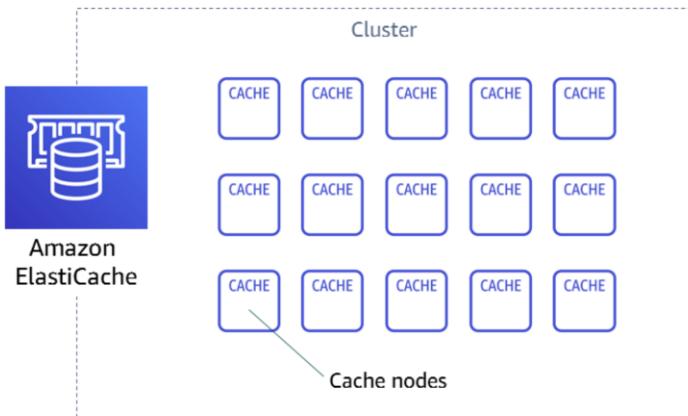
4 /

The Memcached and Redis engines are simple caches that can be used to offload DB burden. Each engine provides certain advantages. This table compares some key features between Memcached and Redis.

- **Sub-millisecond latency** – Both engines offer sub-millisecond response times. By storing data in memory, they can read data more quickly than disk-based databases.
- **Ability to scale horizontally** – Memcached enables you to scale out and in, adding and removing nodes as demand on your system increases and decreases.
- **Multi-threaded performance** – Because Memcached is multithreaded, it can use multiple processing cores, which means that you can handle more operations by scaling up compute capacity.
- **Advanced data structures** – Redis supports complex data types, such as strings, hashes, lists, sets, sorted sets, and bitmaps.
- **Sorting or ranking datasets** – You can use Redis to sort or rank in-memory datasets. For example, you can use Redis sorted sets to implement a game leaderboard that keeps a list of players that is sorted by their rank.
- **Publish/subscribe messaging** – Redis supports publish/subscribe messaging with pattern matching, which you can use for high-performance chat rooms, real-time comment streams, social media feeds, and server intercommunication.
- **Multi-AZ deployments with automatic failover** – ElastiCache for Redis provides high availability through Multi-AZ deployments with automatic failover, in case your primary node fails.

- **Persistence** – Redis enables you to persist your key store. By contrast, the Memcached engine does not support persistence. For example, perhaps a node fails and is replaced with a new, empty node; or you might terminate a node or scale one down. In this case, you lose the data that's stored in cache memory.

ElastiCache components



- A **node** is the smallest block of an ElastiCache deployment
- Each node has its own DNS name and port
- A **cluster** is a logical grouping of one or more nodes



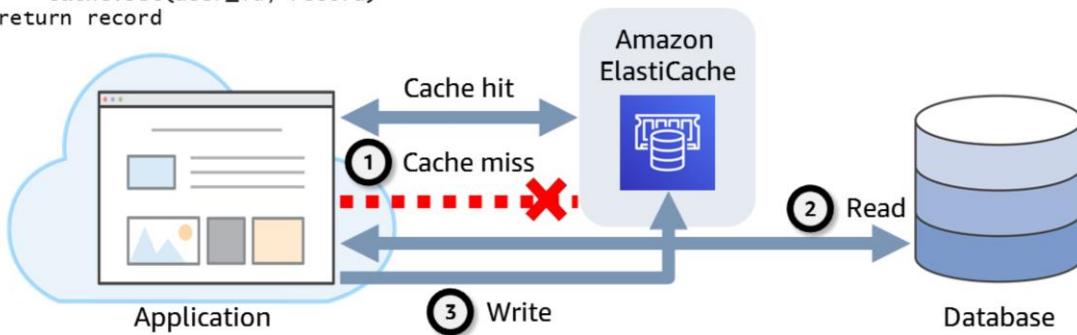
© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

48

A cache *node* is the smallest building block of an ElastiCache deployment. It is a fixed-size chunk of secure, network-attached RAM. Each node runs the engine that was chosen when the cluster or replication group was created or last modified. Each node has its own DNS name and port. It can exist in isolation from other nodes, or in a grouping with other nodes, which is also known as a *cluster*.

Caching strategies: Lazy loading

```
def get_user(user_id):
    # Check the cache
    record = cache.get(user_id)
    if record is None:
        # Run a DB query
        record = db.query("select * from users where id = ?", user_id)
        # Populate the cache
        cache.set(user_id, record)
    return record
```



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

49

You can employ two caching strategies with ElastiCache.

The first strategy, *lazy loading*, is a caching strategy that loads data into the cache only when necessary. In this case, when your application requests data, it first makes the request to the ElastiCache cache. If the data exists in the cache and is current, a cache hit occurs and ElastiCache returns the data to your application. Otherwise (in the case of a cache miss), your application requests the data from your data store, which returns the data to your application. Your application then writes the data to the cache so that it can be retrieved more quickly the next time that it's requested.

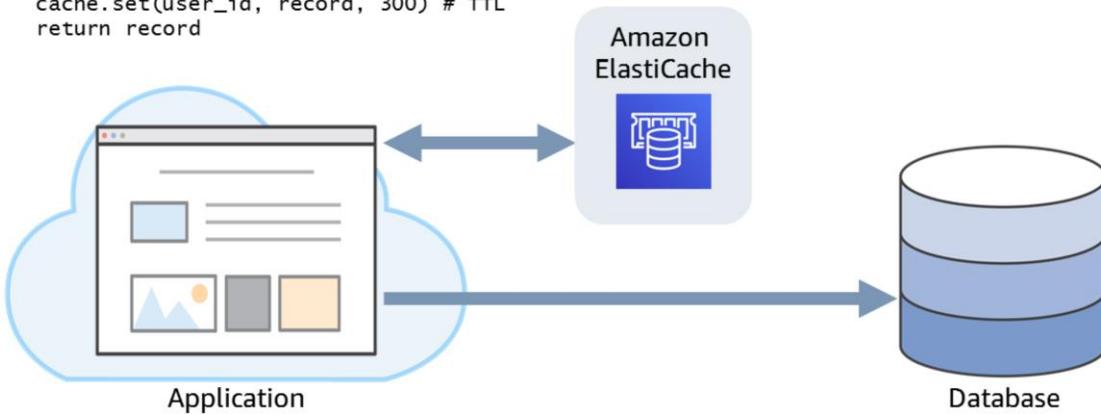
Use lazy loading when you have data that will be read often, but written infrequently. For example, in a typical web or mobile application, a user's profile rarely changes, but it is accessed throughout the application. A person might update his or her profile only a few times per year. However, the profile might be accessed dozens or hundreds of times a day, depending on the user.

The advantage of lazy loading is that only requested data is cached. Because most data is never requested, lazy loading avoids filling up the cache with unnecessary data. However, a cache miss incurs a penalty. Each cache miss results in three trips, which can cause a noticeable delay in data getting to the application. Also, if data is written to the cache only when a cache miss occurs, data in the cache can become stale. Lazy loading does not provide updates to the cache when

data is changed in the database.

Caching strategies: Write-through

```
def save_user(user_id, values):
    # Save to DB
    record = db.query("update users...where id = ?", user_id, values)
    # Push into cache
    cache.set(user_id, record, 300) # TTL
    return record
```



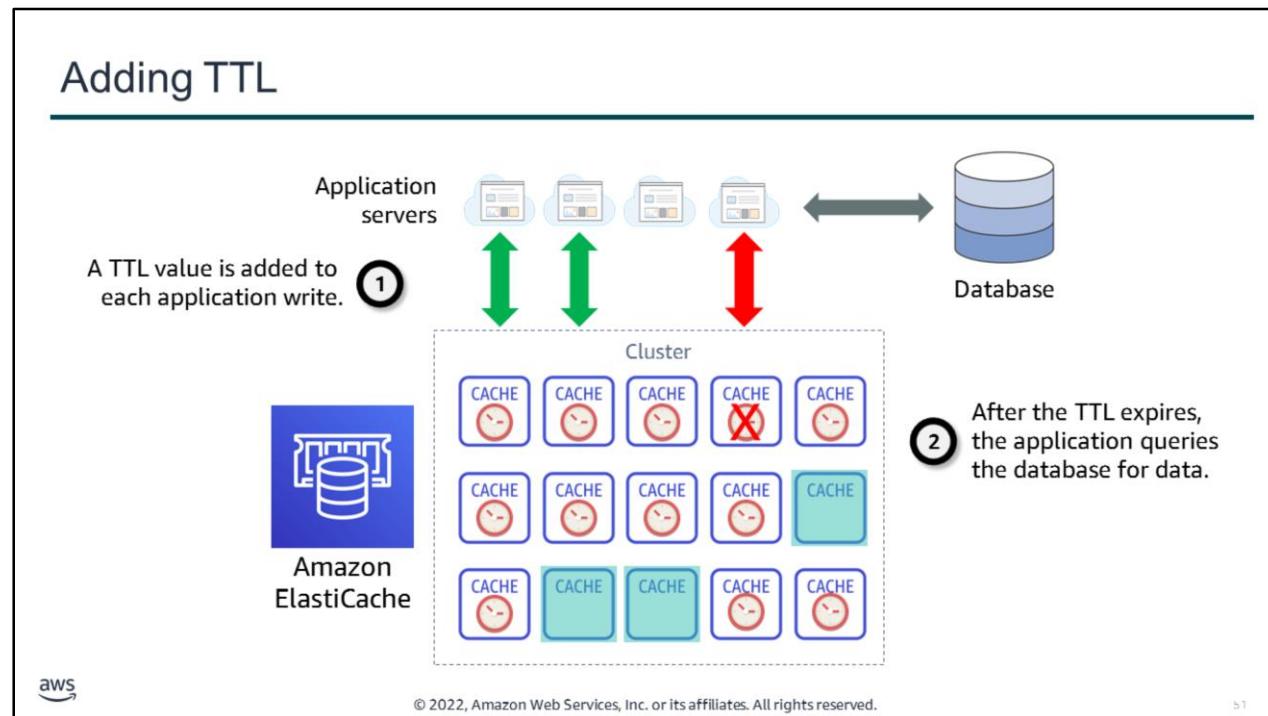
© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

50

The *write-through* strategy is the second caching strategy. It adds data or updates data in the cache when data is written to the database. Use a write-through caching strategy when you have data that must be updated in real time. This approach is proactive: you can avoid unnecessary cache misses when you have data that you know is going to be accessed. A good example for this situation is any type of aggregate, such as a top-100-game leaderboard, the top-10-most-popular news stories, or recommendations. Because this data is typically updated through a specific piece of application or background job code, it's straightforward to update the cache along with it.

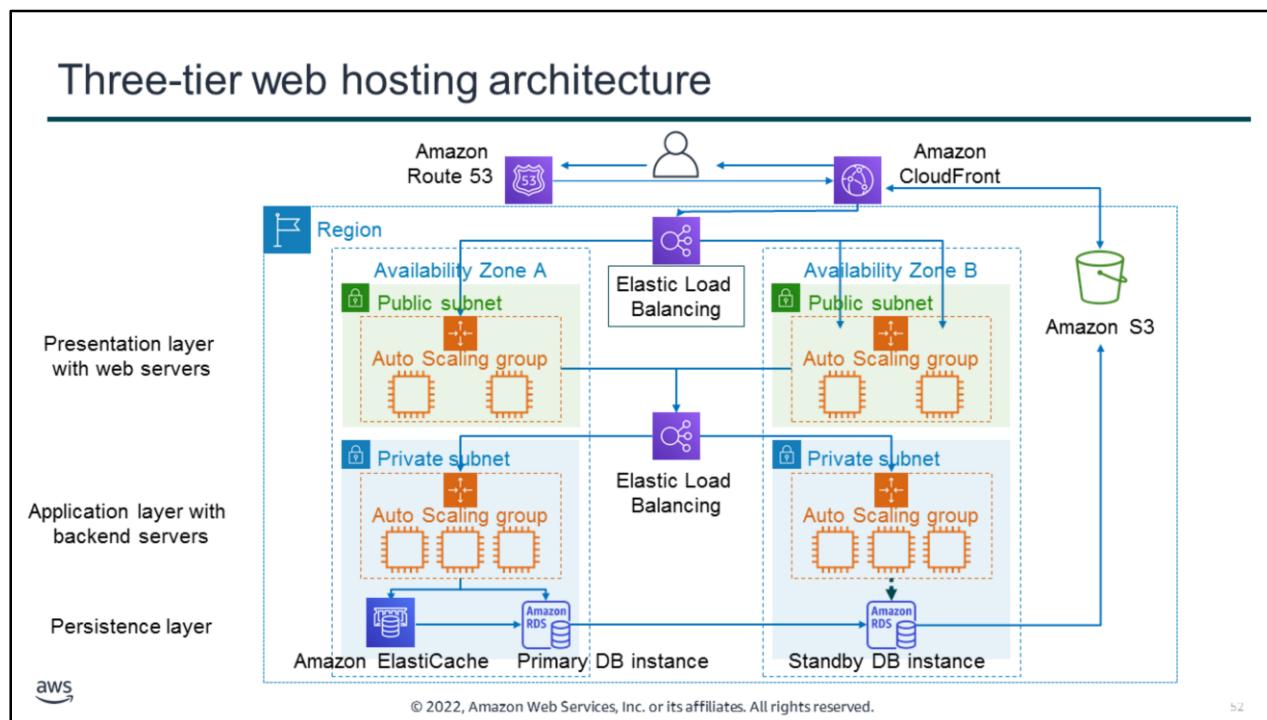
The advantage of this approach is that it increases the likelihood that your application will find that value in the cache when it looks for it. The disadvantage is that you are potentially caching data that you don't need, so this approach can cause added costs.

In practice, both approaches are used together, so it's important for you to understand the frequency of data change and use the appropriate TTLs.



Lazy loading permits stale data. Write-through ensures that data is always fresh, but this approach might populate the cache with unnecessary data.

By adding a TTL value to each write, you enjoy the advantages of each strategy and avoid cluttering the cache with data. TTL is an integer value or key that specifies the number of seconds or milliseconds, depending on the in-memory engine, until the key expires. When an application attempts to read an expired key, it's treated as though the data isn't found in the cache. As a result, the database is queried and the cache is updated. This way, the data doesn't get too stale, and values in the cache are occasionally refreshed from the database.



One scenario where you might want to use Amazon ElastiCache is in a traditional, three-tier web hosting architecture. In this scenario, you want to run a public web application while you still maintain private backend servers in a private subnet. You can create a public subnet for your web servers that have access to the internet. At the same time, you can place your backend infrastructure in a private subnet with no internet access. The database tier of your backend infrastructure might include Amazon Relational Database Service (Amazon RDS) DB instances and an ElastiCache cluster that provides the in-memory layer.

In this web hosting architecture diagram:

- *Amazon Route 53* enables you to map your zone apex (such as *example.com*) DNS name to your load balancer DNS name.
- *Amazon CloudFront* provides edge caching for high-volume content.
- A load balancer spreads traffic across web servers in Auto Scaling groups in the presentation layer.
- Another load balancer spreads traffic across backend application servers in the Auto Scaling groups that are in the application layer.
- *Amazon ElastiCache* provides an in-memory data cache for the application, which removes load from the database tier.

Section 5 key takeaways



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

53

- A [database cache](#) supplements your primary database by removing unnecessary pressure on it, typically in the form of frequently accessed read data
- [DAX](#) is a fully managed, highly available, in-memory cache for DynamoDB that delivers a performance improvement of up to 10 times—from milliseconds to microseconds
- [Amazon ElastiCache](#) is a side cache that works as an in-memory data store to support the most demanding applications that require sub-millisecond response times

Some key takeaways from this section of the module include:

- A database cache supplements your primary database by removing unnecessary pressure on it, typically in the form of frequently accessed read data
- DAX is a fully managed, highly available, in-memory cache for DynamoDB that delivers a performance improvement of up to 10 times—from milliseconds to microseconds
- Amazon ElastiCache is a side cache that works as an in-memory data store to support the most demanding applications requiring submillisecond response times

Module wrap-up

Module 11: Caching Content



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

It's now time to review the module and wrap up with a knowledge check and discussion of a practice certification exam question.

Module summary

In summary, in this module, you learned how to:

- Identify how caching content can improve application performance and reduce latency
- Create architectures that use Amazon CloudFront to cache content
- Identify how to design architectures that use edge locations for distribution and distributed denial of service (DDoS) protection
- Recognize how session management relates to caching
- Describe how to design architectures that use Amazon ElastiCache



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

55

In summary, in this module, you learned how to:

- Identify how caching content can improve application performance and reduce latency
- Create architectures that use Amazon CloudFront to cache content
- Identify how to design architectures that use edge locations for distribution and distributed denial of service (DDoS) protection
- Recognize how session management relates to caching
- Describe how to design architectures that use Amazon ElastiCache

Complete the knowledge check



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

55

It is now time to complete the knowledge check for this module.

Sample exam question



A company is developing a highly available web application that uses stateless web servers. Which services are suitable for storing session state data? (Select TWO.)

Choice	Response
A	Amazon CloudWatch
B	Amazon DynamoDB
C	Elastic Load Balancing
D	Amazon ElastiCache
E	AWS Storage Gateway

© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

57

Look at the answer choices and rule them out based on the keywords.

Sample exam question answer



A company is developing a highly available web application that uses stateless web servers. Which services are suitable for storing session state data? (Select TWO.)

The correct answers are B and D.

The keywords in the question are "storing session state data".

© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

58

The following are the keywords to recognize: "**storing session state data**".

The correct answers are B (Amazon DynamoDB) and D (Amazon ElastiCache): Both DynamoDB and ElastiCache provide high-performance storage of key-value pairs. CloudWatch and Elastic Load Balancing are not storage services. AWS Storage Gateway is a storage service, but it's a hybrid storage service that enables on-premises applications to use cloud storage.



Thank you

Corrections, feedback, or other questions?
Contact us at <https://support.aws.amazon.com/#/contacts/aws-academy>.
All trademarks are the property of their owners.

© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

59

Thank you for completing this module.



**AWS Academy Cloud Architecting
Module 12 Student Guide
Version 2.0.8**

200-ACACAD-20-EN-SG

© 2022 Amazon Web Services, Inc. or its affiliates. All rights reserved.

This work may not be reproduced or redistributed, in whole or in part, without prior written permission from Amazon Web Services, Inc. Commercial copying, lending, or selling is prohibited.

All trademarks are the property of their owners.

Contents

[Module 12: Building Decoupled Architectures](#)

4



Module 12: Building Decoupled Architectures

AWS Academy Cloud Architecting

© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

Welcome to Module 12: Building Decoupled Architectures.

Module overview

Sections

1. Architectural need
2. Decoupling your architecture
3. Decoupling with Amazon Simple Queue Service (Amazon SQS)
4. Decoupling with Amazon Simple Notification Service (Amazon SNS)
5. Sending messages between cloud applications and on-premises with Amazon MQ



Knowledge check



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

2

This module includes the following sections:

1. Architectural need
2. Decoupling your architecture
3. Decoupling with Amazon Simple Queue Service (Amazon SQS)
4. Decoupling with Amazon Simple Notification Service (Amazon SNS)
5. Sending messages between cloud applications and on-premises with Amazon MQ

At the end of this module, you will be asked to complete a knowledge check that will test your understanding of key concepts covered in this module.

Module objectives

At the end of this module, you should be able to:

- Differentiate between tightly and loosely coupled architectures
- Identify how Amazon SQS works and when to use it
- Identify how Amazon SNS works and when to use it
- Describe Amazon MQ



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

3

At the end of this module, you should be able to:

- Differentiate between tightly and loosely coupled architectures
- Identify how Amazon SQS works and when to use it
- Identify how Amazon SNS works and when to use it
- Describe Amazon MQ

Section 1: Architectural need

Module 12: Building Decoupled Architectures



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

Introducing Section 1: Architectural need.

Café business requirement

The café's architecture now supports hundreds of thousands of users. However, it's difficult to make changes to one layer of the application without affecting the other layers.



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

5

The café's architecture now supports hundreds of thousands of users. However, the café's systems are too tightly coupled. It's difficult to make changes to one layer of the application without affecting the other layers. For example, daily ordering reports are generated from the same web server that also serves the café's website to customers.

In addition, Frank mentioned that he's not getting the regular 17:00 report on Fridays. After some investigation, Sofía and Nikhil observe that the scheduled maintenance window coincides with the time that the reporting system attempts to generate the report.

They speak with Olivia, who recommends that they decouple the architecture. By moving the reporting process to another system, the reporting data will not be lost, even if the web server becomes temporarily unavailable. Also, the need to generate a report will be queued and handled.

Section 2: Decoupling your architecture

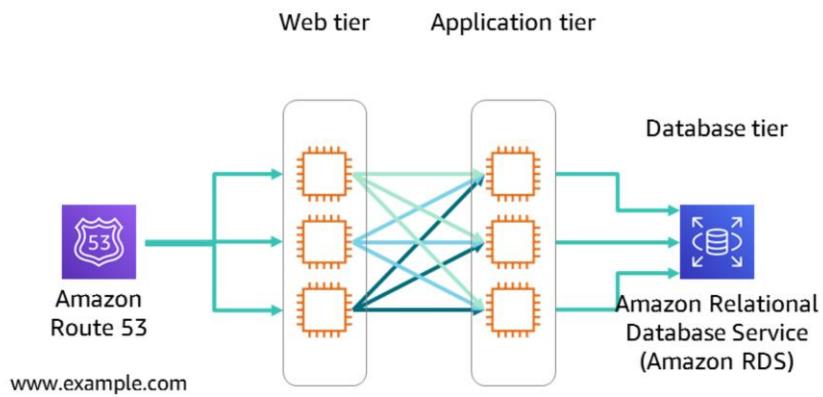
Module 12: Building Decoupled Architectures



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

Introducing Section 2: Decoupling your architecture.

Tightly coupled architectures



Components are **strongly connected** to each other.



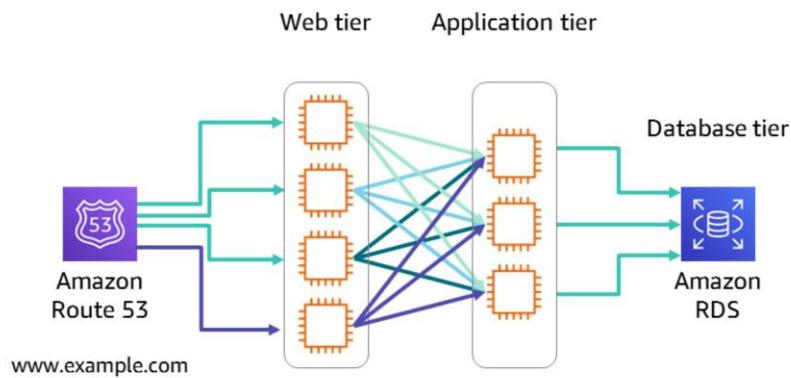
© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

7

Traditional infrastructures have chains of tightly integrated components. Each component has a specific purpose. When one component goes down, the disruption to the system can be fatal.

Consider this example of a three-tier architecture for a web application that processes customer orders. Each instance in the web tier communicates with each instance in the application tier. Each instance in the application tier persists data to a backend database. An instance failure in the web or application tiers would also cause failure in the persistence of some customer order data.

Tightly coupled architectures impede scaling



Adding resources **increases complexity** and **impedes scaling**.



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

8

In a tightly coupled system, scaling is also impeded: if you add servers at one layer, you must also connect them to servers in every connecting layer. Continuing with the three-tier architecture example, an instance that's added to the web tier must be connected to every instance in the application tier.

Forms of system coupling

Application-level coupling:

Relates to managing incoming and outgoing dependencies

Platform coupling:

Relates to interoperability of heterogeneous systems components

Spatial coupling:

Relates to managing components at network-topology level or protocol level

Temporal (runtime) coupling:

Refers to the ability of a system component to do meaningful work while it performs a synchronous, blocking operation



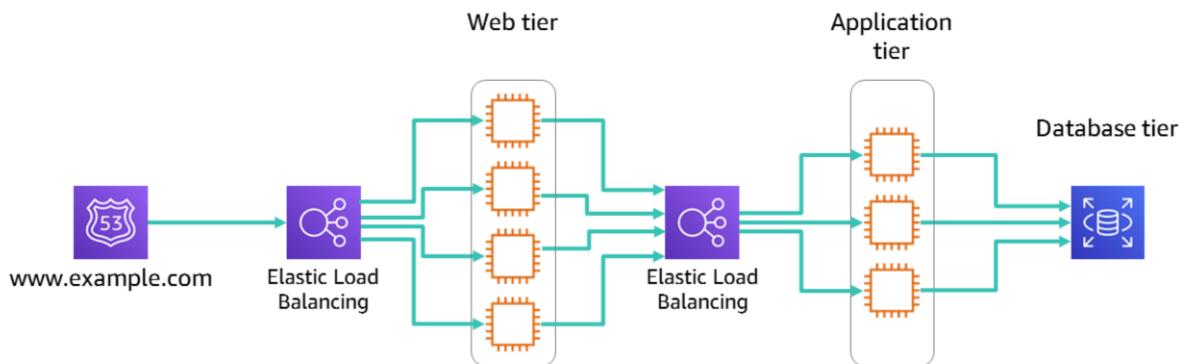
© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

9

A system can be coupled in many ways, and you should consider them when you build distributed applications in the cloud:

- *Application-level* coupling relates to managing incoming and outgoing dependencies
- *Platform* coupling relates to the interoperability of heterogeneous systems components
- *Spatial* coupling relates to managing components at a network-topology level or protocol level
- *Temporal* (or runtime) coupling refers to the ability of a system component to do meaningful work while it performs a synchronous, blocking operation

Loosely coupled architectures



Use **managed solutions** as **intermediaries** between layers.



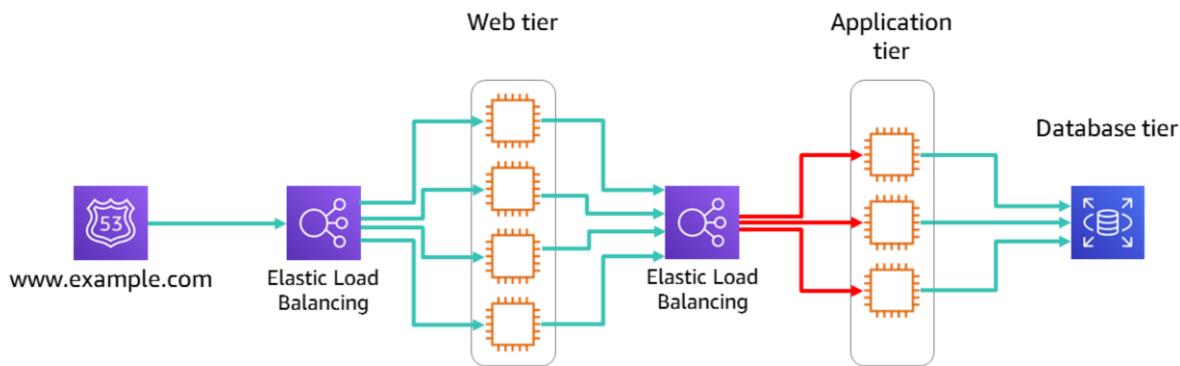
© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

10

To help ensure that your application scales with increasing load and that there are no bottlenecks or single points of failure in your system, implement loose coupling. With loose coupling, you reduce dependencies in your system by using managed solutions as intermediaries between the layers of your system. This way, failures and the scaling of components or layers are automatically handled by the intermediary.

Consider the three-tier web application architecture again. You can implement loose coupling by adding load balancers in front of the web and application tiers to distribute traffic at each layer. If one server goes down, the load balancer will automatically direct traffic to the healthy instances.

Considerations in loosely coupled architectures



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

11

In the business use case of the order processing workflow, one potential point of vulnerability is saving order data to the database. If the business requires that order data must persist in the database, then various scenarios—such as a potential deadlock, race condition, or network issue—could cause the persistence of the order to fail. In this case, the order would be lost and you could not restore it.

Section 2 key takeaways



- Tightly coupled systems have chains of tightly integrated components and impede scaling
- You can implement loose coupling in your system by using managed solutions (such as Elastic Load Balancing) as intermediaries between layers



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

12

Some key takeaways from this section of the module include:

- Tightly coupled systems have chains of tightly integrated components and impede scaling
- You can implement loose coupling in your system by using managed solutions (such as Elastic Load Balancing) as intermediaries between layers

Section 3: Decoupling with Amazon SQS

Module 12: Building Decoupled Architectures

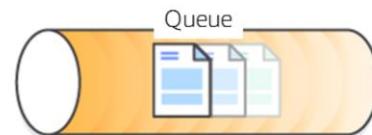


© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

Introducing Section 3: Decoupling with Amazon SQS.

Message queues for decoupling architectures

Producer – Application component
that produces messages and
adds them to queue



Consumer – Application component
that polls queue for messages
and processes them



A **message** is for communication between software components—not between people (that is, it's not an email or text message).

Pull mechanism



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

14

Message queues are another component that help you implement a decoupled architecture. Message queues provide communication and coordination for these distributed applications.

A message queue is a temporary repository for messages that are waiting to be processed. Messages are usually small, and can be things like requests, replies, error messages, or plain information. Examples of messages include customer records, product orders, invoices, patient records, among others.

To send a message, a component that is called a *producer* adds a message to the queue. The message is stored in the queue until another component that is called a *consumer* retrieves and processes it.

Amazon SQS



Amazon Simple
Queue Service
(Amazon SQS)

- Fully managed message queueing service
- Uses a pull mechanism
- Messages are encrypted and stored until they are processed and deleted
- Acts as a buffer between producers and consumers



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

15

Amazon Simple Queue Service (Amazon SQS) is a fully managed, message queuing service that enables you to decouple application components so they run independently. It lets web service applications queue messages that are generated by one application component to be consumed by another component.

A queue is a temporary repository for messages that are waiting to be processed. Messages are stored until they are processed and deleted (from 1 through 14 days; the default is 4 days). Messages can contain up to 256 KB of text in any format. Amazon SQS works on a massive scale and processes billions of messages per day. It stores all message queues and messages within a single, highly available AWS Region with multiple redundant Availability Zones. No single computer, network, or Availability Zone failure can make messages inaccessible. Messages can be sent and read simultaneously.

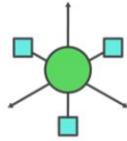
You can securely share Amazon SQS queues anonymously or with specific AWS accounts. You can also restrict queue sharing by IP address and the time of day. Messages in SQS queues are encrypted with server-side encryption (SSE) by using keys that are managed in the AWS Key Management Service (AWS KMS). Amazon SQS decrypts messages only when they are sent to an authorized consumer.

Amazon SQS supports multiple producers and consumers that interact with the same queue. Amazon SQS can be used with several AWS services, including: Amazon Elastic Compute Cloud

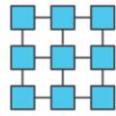
(Amazon EC2), Amazon Simple Storage Service (Amazon S3), Amazon Elastic Container Service (Amazon ECS), AWS Lambda, and Amazon DynamoDB.

Achieve loose coupling with Amazon SQS

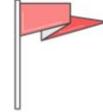
With Amazon SQS, you can:



Use **asynchronous processing** to get your responses from each step quickly



Handle performance and service requirements by increasing the number of job instances



Easily **recover from failed steps** because messages will remain in the queue



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

16

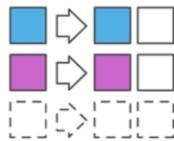
Amazon SQS enables you to achieve loose coupling in your architecture.

- It performs asynchronous processing so that you can get responses from each step quickly
- It can handle performance and service requirements by increasing the number of job instances
- Your application can easily recover from failed steps because messages will remain in the queue

Amazon SQS general use cases



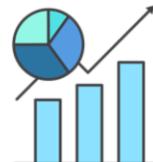
Work queues



Buffering batch operations



Request offloading

Trigger
Amazon EC2
Auto Scaling

© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

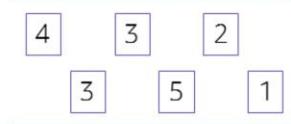
17

Amazon SQS can be used in many different ways:

- *Work queues* – Decouple components of a distributed application that might not all process the same amount of work simultaneously.
- *Buffering batch operations* – Add scalability and reliability to your architecture, and smooth out temporary volume spikes without losing messages or increasing latency.
- *Request offloading* – Queue requests to move slow operations off of interactive request paths
- *Trigger Amazon EC2 Auto Scaling* – Use SQS queues to help determine the load on an application. When they are combined with Amazon EC2 Auto Scaling, you can scale the number of EC2 instances out or in, depending on the volume of traffic.

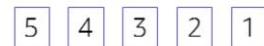
Queue types

Standard queues



- At-least-once delivery
- Best-effort ordering
- Nearly unlimited throughput

First in, first out (FIFO) queues



- First-in-first-out delivery
- Exactly once processing
- High throughput



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

18

There are two types of SQS queues:

Standard queues offer:

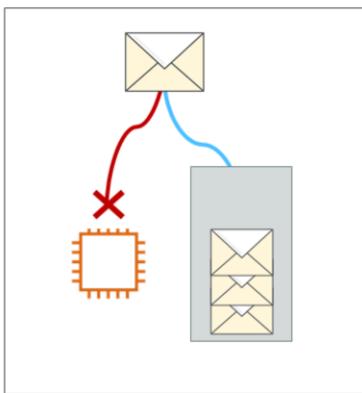
- At-least-once delivery – A message is delivered at least once, but occasionally more than one copy of a message is delivered.
- Best-effort ordering – Occasionally, messages might be delivered in an order that is different from the order that they were sent in.
- Nearly unlimited throughput – Standard queues support a nearly unlimited number of transactions per second (TPS) per API action.

First in, first out (FIFO) queues:

- Are designed to guarantee that messages are processed exactly once, in the exact order that they are sent and received.
- Provide high throughput – FIFO queues support up to 300 messages per second (300 send, receive, or delete operations per second). When you batch 10 messages per operation (maximum), FIFO queues can support up to 3,000 messages per second.

Amazon SQS features (1 of 3)

Dead-letter queue support



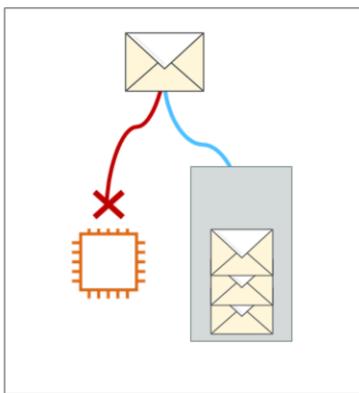
© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

19

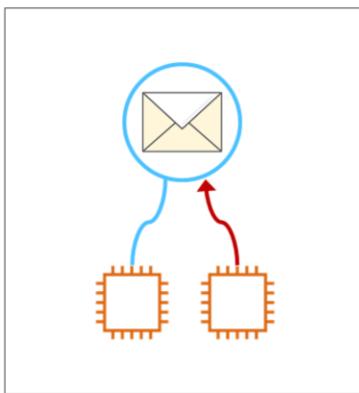
Amazon SQS features dead-letter queue support. A *dead-letter queue* (DLQ) is a queue of messages that could not be processed. It receives messages after the maximum number of processing attempts has been reached. A DLQ is like any other SQS queue: messages can be sent to it and received from it. You can create a DLQ from the Amazon SQS API and the console.

Amazon SQS features (2 of 3)

Dead-letter queue support



Visibility timeout



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

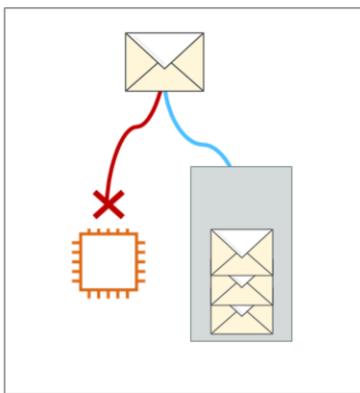
20

Another feature of Amazon SQS is the visibility timeout. *Visibility timeout* is the period of time when Amazon SQS prevents other consumers from receiving and processing the same message. The timeout helps ensure that a job does not get processed multiple times and cause duplication. During the visibility timeout, the component that received the message processes it and then deletes it from the queue. The default visibility timeout for a message is 30 seconds, and the maximum is 12 hours.

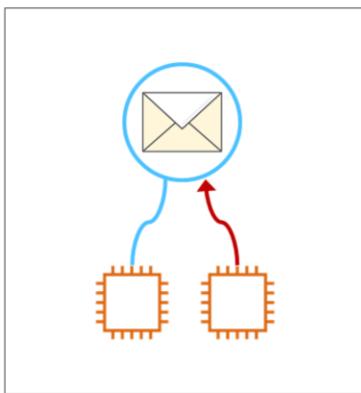
If the consumer fails to process and delete the message before the visibility timeout expires, the message becomes visible to other consumers and it might be processed again. Typically, you should set the visibility timeout to the maximum time that it takes your application to process and delete a message from the queue.

Amazon SQS features (3 of 3)

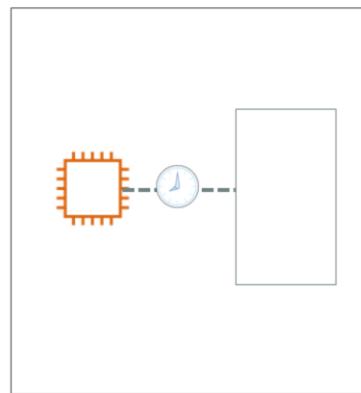
Dead-letter queue support



Visibility timeout



Long polling



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

21

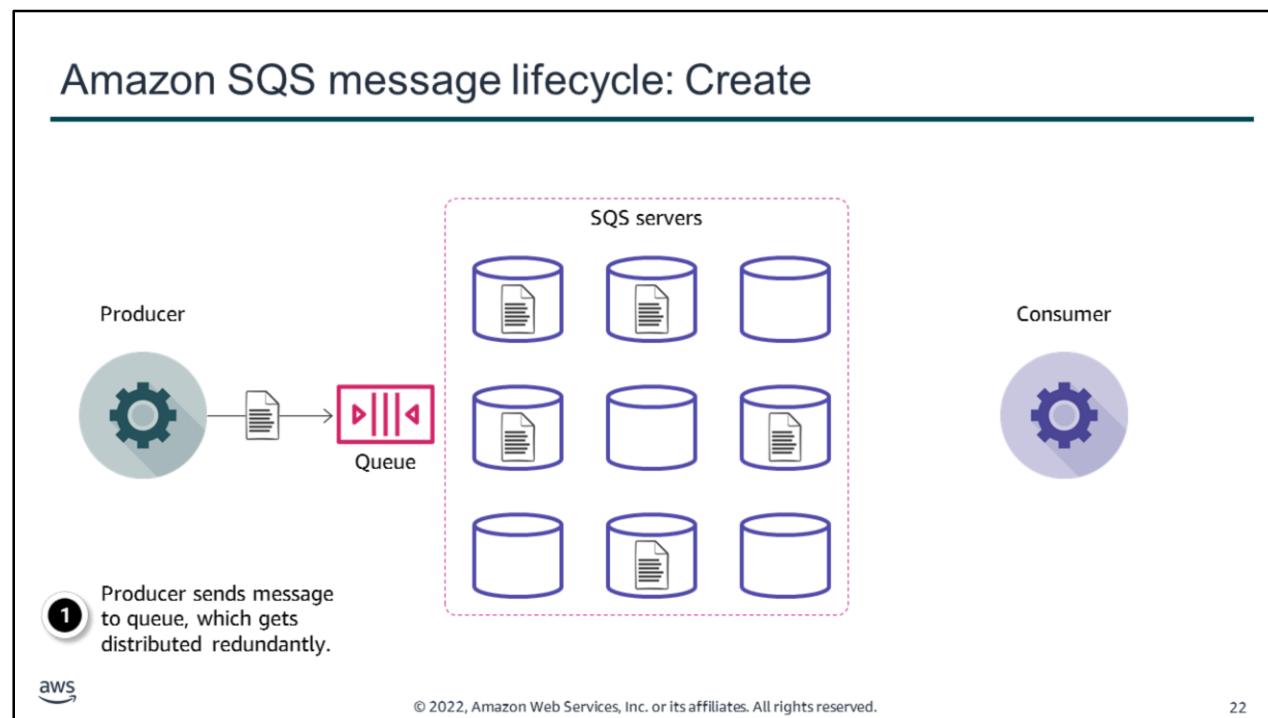
Finally, Amazon SQS supports both support short polling and long polling to retrieve messages from your SQS queues. By default, queues use short polling.

Short polling queries only a subset of the servers (based on a weighted random distribution) to find messages can be included in the response. Amazon SQS sends the response immediately, even if the query found no messages.

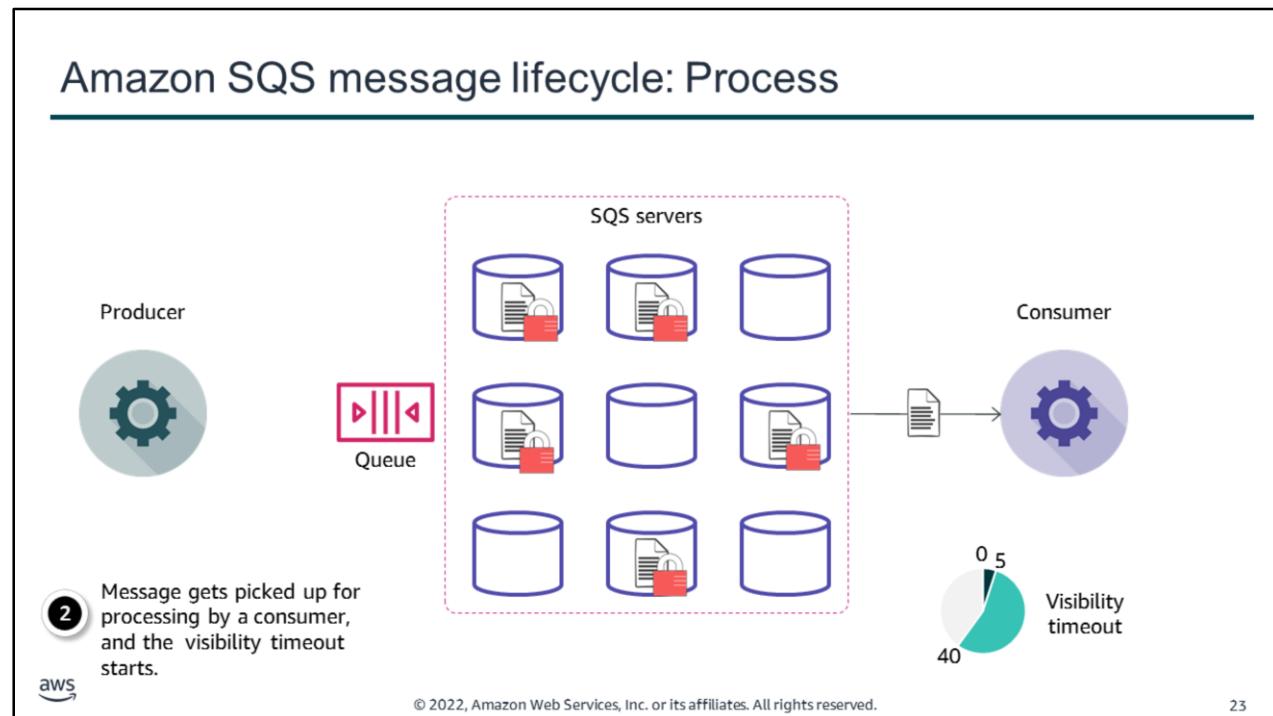
By contrast, long polling queries all the servers for messages. Amazon SQS sends the response after it collects the maximum number of messages for the response, or the polling wait time expires.

Long polling makes it inexpensive to retrieve messages from your SQS queue as soon as the messages are available. Long polling might reduce the cost of using Amazon SQS because you can reduce the number of empty receives.

Amazon SQS message lifecycle: Create

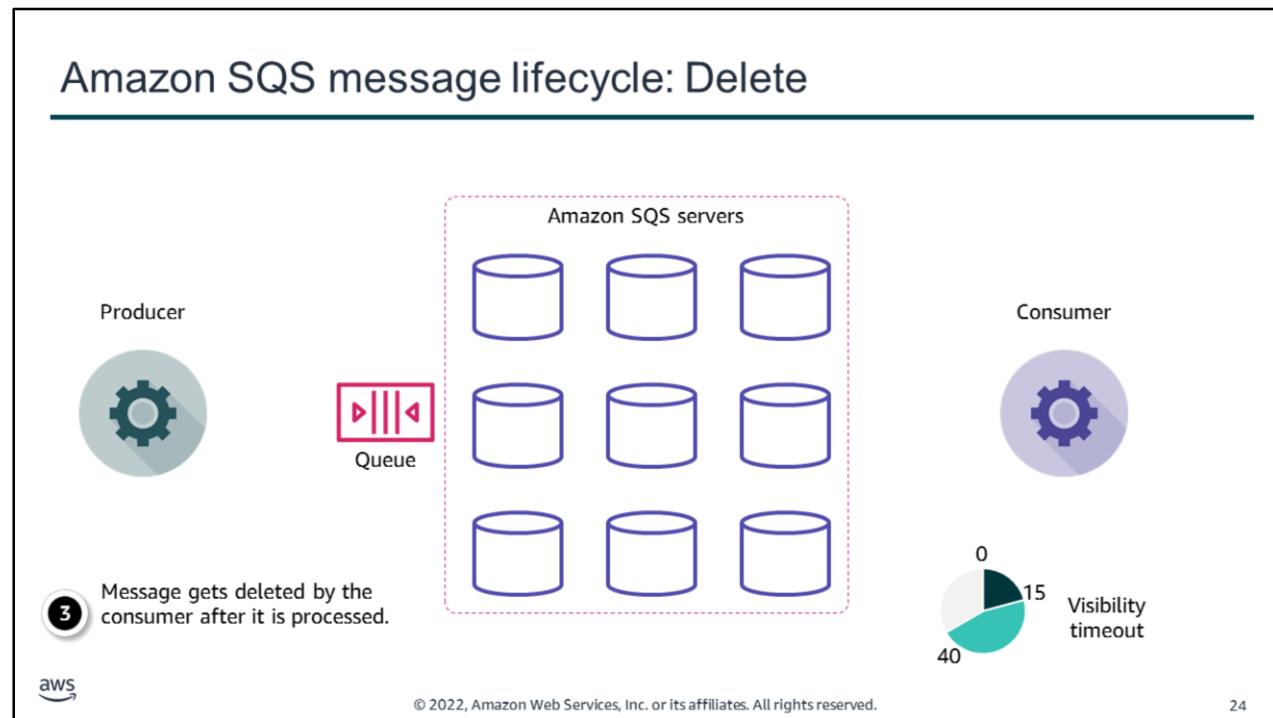


The lifecycle of a message in an SQS queue can be illustrated with the following scenario. First, a producer sends a message to a queue, and the message is distributed across the SQS servers redundantly.



When a consumer is ready to process the message, it retrieves the message from the queue. While the message is being processed, it remains in the queue.

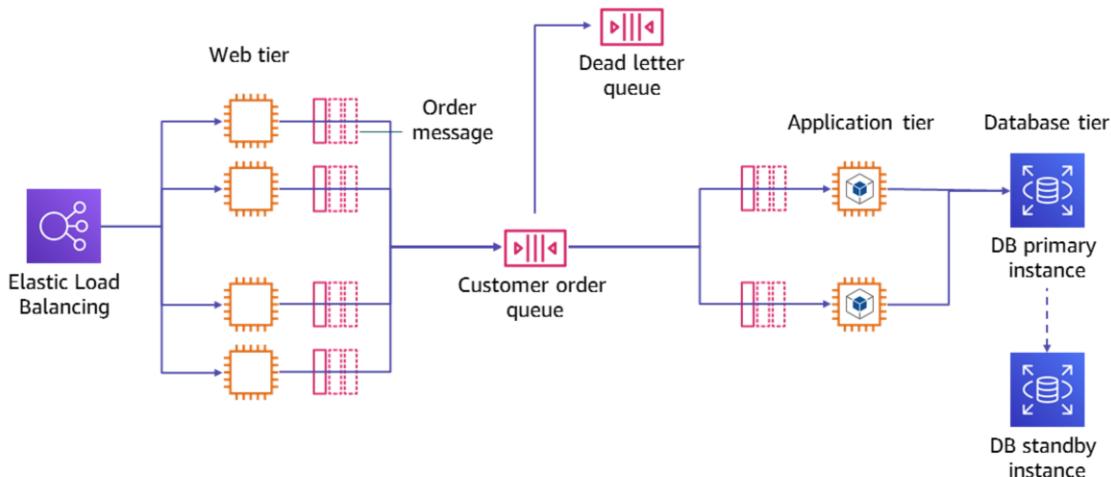
During the visibility timeout, other consumers cannot process the message. In this example, the visibility timeout is *40 seconds*.



After processing the message, the consumer deletes the message from the queue. This action prevents the message from being received and processed again when the visibility timeout expires.

Amazon SQS doesn't automatically delete the message. Because Amazon SQS is a distributed system, there's no guarantee that the consumer actually receives the message (for example, because of a connectivity issue, or an issue in the consumer application). Therefore, the consumer must delete the message from the queue after receiving and processing it.

Decoupling example: Using Amazon SQS



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

25

Again, consider the ordering processing application that you learned about in the previous section. You can decouple the architecture for this application by introducing an SQS queue. You can use the queue to isolate the processing logic into its own component so that it runs in a separate process from the web application.

This design enables the system to be more resilient to spikes in traffic. Further, it enables work to be performed only as fast as necessary to manage costs.

In addition, you now have a mechanism for persisting orders as messages (with the queue acting as a temporary database). You also moved the scope of your transaction with your database further down the stack. If an application exception or transaction fails, this design helps to ensure that the order processing can be retried or redirected to a dead letter queue for re-processing at a later stage.

For more information about this use case, see [this AWS Compute Blog post](#).

Message queue use cases (1 of 2)



Service-to-service communication



Asynchronous work items



State change notifications



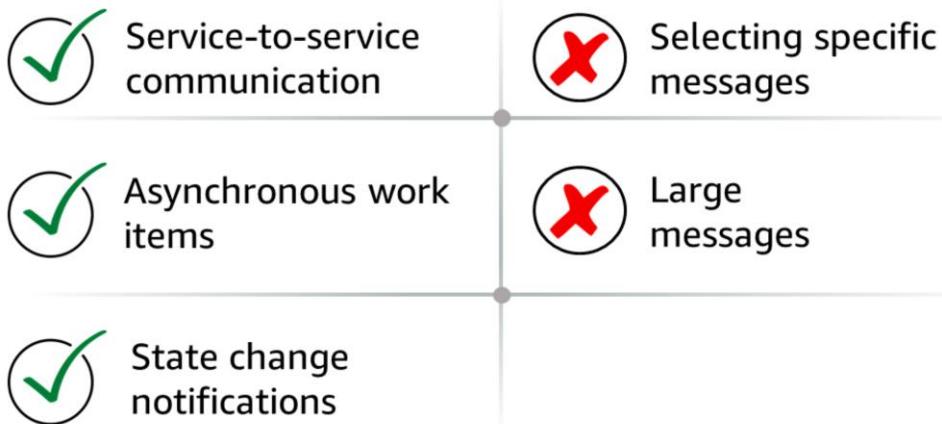
© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

26

These common use cases demonstrate where a message queue is a great fit:

- *Service-to-service communication* – For example, consider a frontend website that must update a customer's delivery address in a backend customer relationship management (CRM) service. You can have the code of the frontend website send messages to an SQS queue, and have the backend CRM service consume them.
- *Asynchronous work items* – For example, say that a hotel booking system must cancel a reservation, which is a process that takes a long time. You can send messages to an SQS queue and have the same hotel booking system consume those messages and perform asynchronous cancellations.
- *State change notifications* – For example, say that you have a service that manages some resource. You want other services to receive updates about changes to that resource. An inventory system might publish notifications when a certain item is low and needs to be ordered.

Message queue use cases (2 of 2)



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

27

It's also important to know when a particular technology will *not* fit your use case. Messaging has its own set of commonly encountered anti-patterns.

- *Selecting specific messages* – You might want to selectively receive messages from a queue that match a particular set of attributes, or match an ad hoc logical query. For example, a service requests a message with a particular attribute because it contains a response to another message that the service sent out. In this scenario, the queue can have messages in it that no one is polling for and are never consumed.
- *Large messages* – Most messaging protocols and implementations work best with reasonably sized messages (in the tens or hundreds of KBs). As message sizes grow, it's best to use a dedicated storage system (such as Amazon S3), and pass a reference to an object in the store that is in the message itself.

Section 3 key takeaways



- Amazon SQS is a fully managed, message-queuing service that enables you to decouple application components so they run independently.
- Amazon SQS supports standard and FIFO queues.
- A producer sends a message to a queue. A consumer processes and deletes the message during the visibility timeout.
- Messages that cannot be processed can be sent to a dead letter queue.
- Long polling is a way to retrieve a large number of messages from your SQS queues.

© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

28

Some key takeaways from this section of the module include:

- Amazon SQS is a fully managed, message-queuing service that enables you to decouple application components so they run independently.
- Amazon SQS supports standard and FIFO queues.
- A producer sends a message to a queue. A consumer processes and deletes the message during the visibility timeout.
- Messages that cannot be processed can be sent to a dead letter queue.
- Long polling is a way to retrieve a large number of messages from your SQS queues.

Section 4: Decoupling with Amazon SNS

Module 12: Building Decoupled Architectures

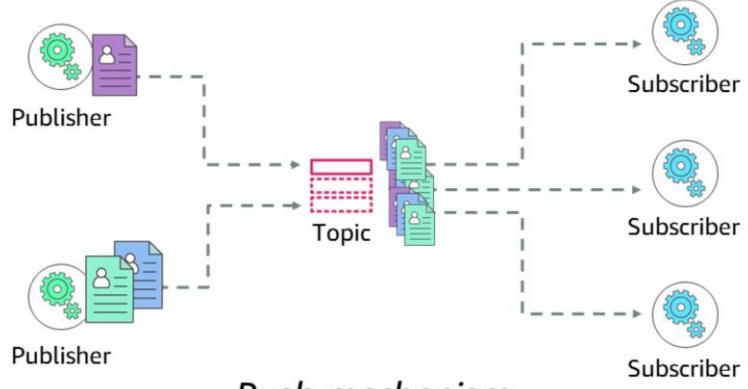


© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

Introducing Section 4: Decoupling with Amazon SNS.

Pub/sub messaging

Publisher – Component that pushes a message to a topic



Push mechanism



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

30

In modern cloud architecture, applications are decoupled into smaller, independent building blocks that are easier to develop, deploy, and maintain. Publish/subscribe (pub/sub) messaging provides instant event notifications for distributed applications.

The pub/sub model enables messages to be broadcast to different parts of a system asynchronously. A message topic provides a lightweight mechanism to broadcast asynchronous event notifications. It also provides endpoints that enable software components to connect to the topic so that they can send and receive those messages.

To broadcast a message, a component called a publisher pushes a message to the topic. Unlike message queues, which batch messages until they are retrieved, message topics transfer messages with no or little queuing, and push them out immediately to all subscribers. A subscriber will receive every message that is broadcast, unless it sets a message filtering policy. Examples of subscribers include web servers, email addresses, Amazon SQS queues, and AWS Lambda functions.

The subscribers to the message topic often perform different functions, and can each do something different with the message in parallel. The publisher doesn't need to know who is using the information that it broadcasts, and the subscribers don't need to know who the message comes from. This style of messaging is a little different from message queues, where the component that sends the message often knows the destination it is sending to.

In the pub/sub messaging paradigm, notifications are delivered to clients by using a push mechanism that eliminates the need to periodically check or poll for new information and updates.

Amazon SNS



- Is a highly available, durable, secure, and fully managed pub/sub messaging service
- Uses a push mechanism
- Supports encrypted topics using customer master keys (CMKs)



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

31

Amazon Simple Notification Service (Amazon SNS) is a web service that you can use to set up, operate, and send notifications from the cloud. The service follows the pub/sub messaging paradigm, where notifications are delivered to clients by using a push mechanism. Amazon SNS is designed to meet the needs of the largest and most demanding applications, and it enables applications to publish an unlimited number of messages at any time.

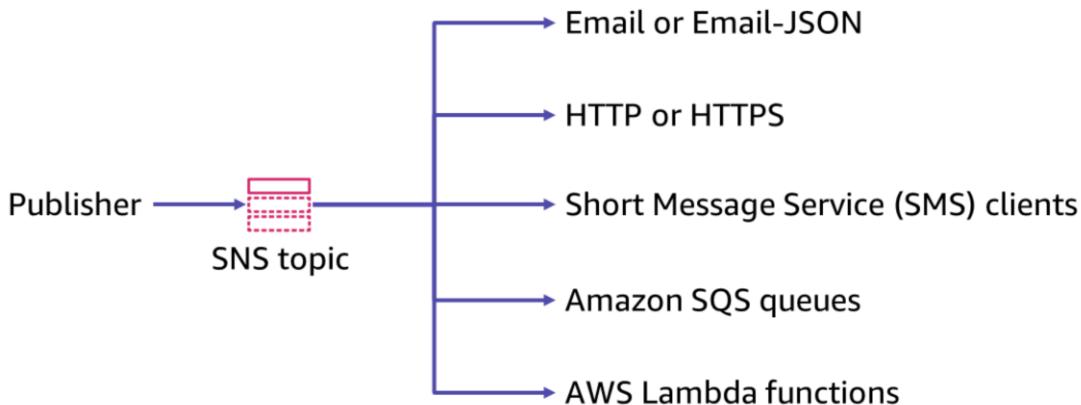
When you use Amazon SNS, you create a topic and set policies that restrict who can publish or subscribe to the topic. A publisher sends messages to topics that they have either created or that they have permission to publish to. Amazon SNS matches the topic to a list of subscribers who have subscribed 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 that are published to the topics that they subscribe to, and all subscribers to a topic receive the same messages.

Amazon SNS supports encrypted topics. After you publish messages to encrypted topics, Amazon SNS uses customer master keys (CMKs) to encrypt your messages. CMKs are the primary resources in AWS KMS. Amazon SNS supports both customer managed and AWS managed CMKs.

When Amazon SNS receives your messages, they are encrypted by using a 256-bit Advanced Encryption Standard-Galois/Counter Mode (AES-GCM) algorithm. The encrypted messages are stored redundantly across multiple servers and data centers, and across multiple Availability

Zones for durability. Messages are decrypted just before they are delivered to subscribed endpoints. For more information about encrypting messages published to Amazon SNS, read this [AWS Compute blog post](#).

Supported transport protocols



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

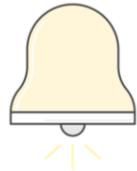
32

Amazon SNS supports the following transport protocols for delivering messages:

- *Email or Email-JSON* – Messages are emailed to registered addresses. Email-JSON sends notifications as a JavaScript Object Notation (JSON) object, and Email sends text-based email message.
- *Hypertext Transfer Protocol (HTTP) or Secure HTTP (HTTPS)* – During subscription registration, subscribers specify a URL. Messages are delivered through an HTTP POST request to the specified URL.
- *Short Message Service (SMS)* – Messages are sent to registered phone numbers as SMS text messages.
- *Amazon SQS queues* – Users specify an SQS standard queue as the endpoint. Amazon SNS will enqueue a notification message to the specified queue. FIFO queues are not currently supported.
- *AWS Lambda functions* – Messages are delivered to AWS Lambda functions, which handle message customizations, enable message persistence, or communicate with other AWS services.

General use cases for Amazon SNS

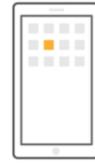
Application and system alerts



Push email and text messaging



Mobile push notifications



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

33

There are many ways to use Amazon SNS:

- *Application and system alerts* – You can use Amazon SNS to receive immediate notification when an event occurs, such as a change to an Auto Scaling group.
- *Push email and text messaging* – You can use Amazon SNS to push targeted news headlines to subscribers by email or SMS.
- *Mobile push notifications* – You can use Amazon SNS to send notifications to an application, indicating that an update is available. The notification message can include a link to download and install the update.

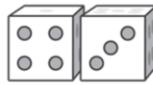
Amazon SNS considerations



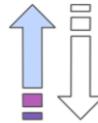
Single published message



No recall options



Order and delivery not guaranteed



Retry policy for each delivery protocol



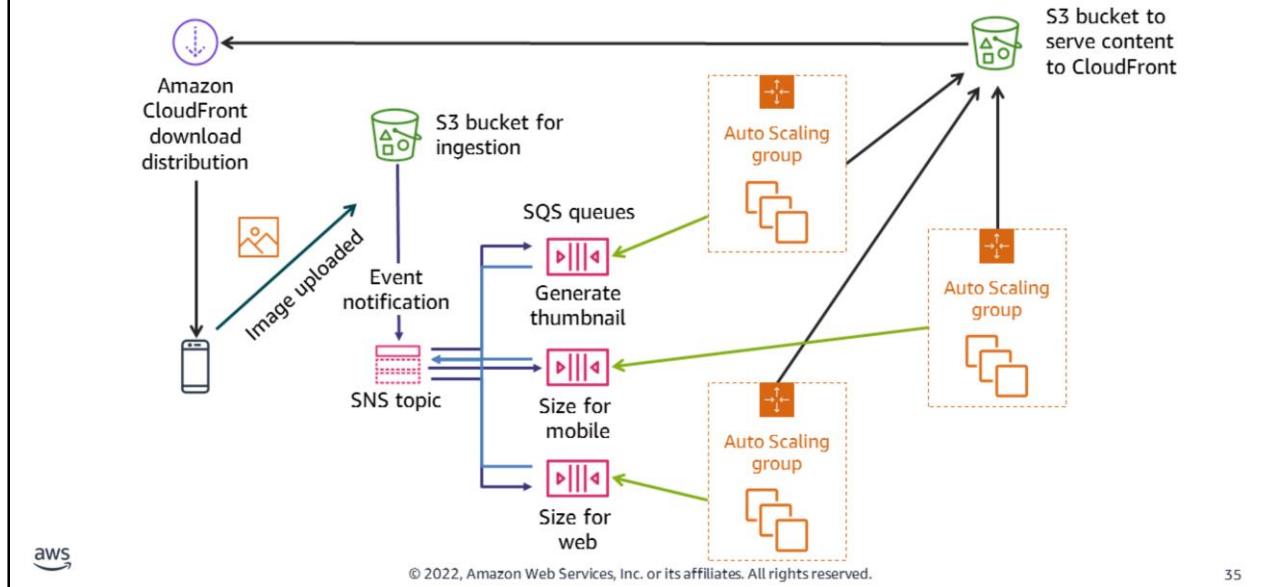
© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

34

If you are planning to use Amazon SNS, consider the following points:

- Each notification message contains a single published message.
- When a message is delivered successfully, there is no way to recall it.
- Amazon SNS will attempt to deliver messages from the publisher in the order that they were published into the topic. However, network issues could potentially result in out-of-order messages at the subscriber end.
- Amazon SNS defines a delivery policy for each delivery protocol. The delivery policy defines how Amazon SNS retries the delivery of messages when server-side errors occur (that is, when the system that hosts the subscribed endpoint becomes unavailable). If a message cannot be successfully delivered on the first attempt, Amazon SNS uses a four-phase retry policy:
 1. Retries with no delay in between attempts;
 2. Retries with minimum delay between attempts;
 3. Retries according to a back-off model;
 4. Retries with maximum delay between attempts. When the message delivery retry policy is exhausted, Amazon SNS can move the message to a DLQ.

Decoupling example: Using Amazon S3 with Amazon SNS



With Amazon SNS, you can use topics to decouple message publishers from subscribers, fan-out messages to multiple recipients at one time, and eliminate polling in your applications.

You can use Amazon SNS to send messages in a single account or to resources in different accounts.

AWS services (such as Amazon EC2, Amazon S3, and Amazon CloudWatch) can publish messages to your SNS topics to trigger event-driven computing and workflows. In this example, after an image is uploaded to an S3 bucket, Amazon S3 triggers an event notification, which automatically sends the message to the SNS topic. Amazon SNS then delivers the S3 event notification to SQS queue subscribers. Auto Scaling groups of EC2 instances process the messages in the SQS queues and publish the processed images to an S3 bucket that serves the content to Amazon CloudFront.

Amazon SNS versus Amazon SQS

Feature	Amazon SNS (Publisher/Subscriber)	Amazon SQS (Producer/Consumer)
Producer/consumer	Publish/subscribe	Send/receive
Delivery mechanism	Push (passive)	Poll (active)
Distribution model	Many to many	One to one
Message persistence	No	Yes



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

36

- Amazon SNS uses a pub/sub messaging paradigm and enables applications to send time-critical messages to multiple subscribers through a push mechanism.
- Amazon SQS uses a send/receive messaging paradigm and exchanges messages through a polling model—sending and receiving components are decoupled.
- Amazon SQS provides flexibility for distributed components of applications—they can send and receive messages without requiring that each component must be concurrently available.

Section 4 key takeaways



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

37

- Amazon SNS is a web service that you can use to set up, operate, and send notifications from the cloud
- Amazon SNS follows the pub/sub messaging paradigm
- When you use Amazon SNS, you create a topic and set policies that restrict who can publish or subscribe to the topic
- You can use topics to decouple message publishers from subscribers, fan-out messages to multiple recipients at one time, and eliminate polling in your applications
- AWS services can publish messages to your SNS topics to trigger event-driven computing and workflows

Some key takeaways from this section of the module include:

- Amazon SNS is a web service that you can use to set up, operate, and send notifications from the cloud
- Amazon SNS follows the pub/sub messaging paradigm
- When you use Amazon SNS, you create a topic and set policies that restrict who can publish or subscribe to the topic
- You can use topics to decouple message publishers from subscribers, fan-out messages to multiple recipients at one time, and eliminate polling in your applications
- AWS services can publish messages to your SNS topics to trigger event-driven computing and workflows

Section 5: Sending messages between cloud applications and on-premises with Amazon MQ

Module 12: Building Decoupled Architectures



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

Introducing Section 5: Sending messages between cloud applications and on-premises with Amazon MQ.

Amazon MQ



Amazon
MQ

- Is a managed message broker service for Apache ActiveMQ
- Manages the provisioning, setup, and maintenance of ActiveMQ
- Simplifies message migration to the cloud
- Is compatible with open-standard APIs and protocols
 - JMS, NMS, AMQP, STOMP, MQTT, and WebSockets



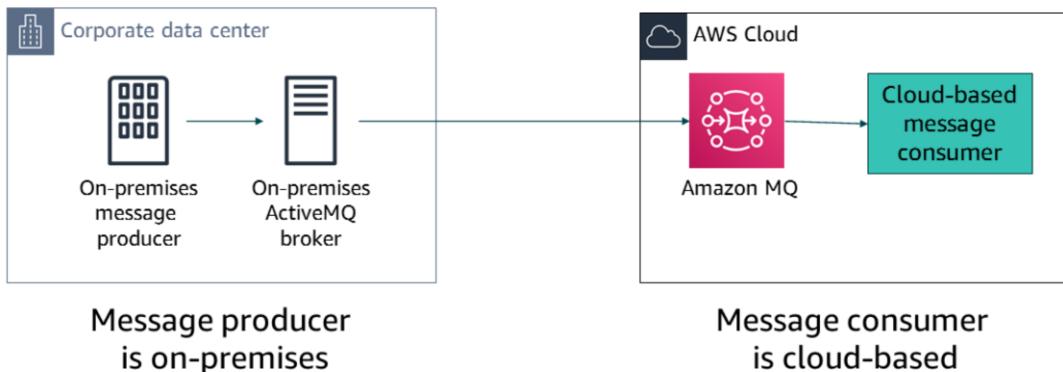
© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

39

Amazon MQ is a managed message broker service for Apache ActiveMQ that enables you to set up and operate message brokers in the cloud. Message brokers enable different software systems—which often use different programming languages on different platforms—to communicate and exchange information. Amazon MQ reduces your operational load by managing the provisioning, setup, and maintenance of ActiveMQ, a popular open-source message broker.

With Amazon MQ, you can migrate messaging to the cloud while preserving the existing connections between your applications. It supports open standard APIs and protocols for messaging, including Java Message Service (JMS), .NET Message Service (NMS), Advanced Message Queuing Protocol (AMQP), Streaming Text Oriented Messaging Protocol (STOMP), MQ Telemetry Transport (MQTT), and WebSockets. You can move from any message broker that uses these standards to Amazon MQ, usually without the need to rewrite any messaging code. In most cases, you can update the endpoints of your applications to connect to Amazon MQ and start sending messages.

Amazon MQ use case: Hybrid cloud environment



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

40

Many organizations, particularly enterprises, rely on message brokers to connect and coordinate different systems. Message brokers enable distributed applications to communicate with each other. They serve as the technological backbone for their IT environment and, ultimately, their business services. Applications depend on messaging to work.

In many cases, these organizations have started to build new cloud-native applications or to lift-and-shift applications to AWS. There are some applications, such as mainframe systems, that are too costly to migrate. In these cases, the on-premises applications must still interact with cloud-based components.

Amazon MQ enables organizations to send messages between applications in the cloud and applications that are on premises to enable hybrid environments and application modernization. For example, you can invoke AWS Lambda from queues and topics that are managed by Amazon MQ brokers to integrate legacy systems with serverless architectures.

The example shows that you can use Amazon MQ to integrate on-premises and cloud environments by using the network of brokers feature of ActiveMQ. The diagram shows the message lifecycle from the on-premises producer to the on-premises broker, which traverses the hybrid connection between the on-premises broker and Amazon MQ. Finally, the message moves to consumption within the AWS Cloud.

For more information about how to use Amazon MQ to integrate on-premises and cloud environments, read this [AWS Compute blog post](#).

Amazon MQ versus Amazon SQS and Amazon SNS

Amazon MQ	Amazon SQS and SNS
For application migration	For born-in-the-cloud applications
Protocols: JMS, NMS, AMQP, STOMP, MQTT, and WebSockets	Protocol: HTTPS
Feature-rich	Nearly unlimited throughput
Pay per hour and pay per GB	Pay per request
Can do pub/sub	Cannot do pub/sub in Amazon SQS, but you can do pub/sub in Amazon SNS



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

41

Amazon MQ is a managed message broker service that provides compatibility with many popular message brokers. Amazon SQS and Amazon SNS are queue and topic services, respectively, that are highly scalable, simple to use, and don't require you to set up message brokers.

- If you use messaging with existing applications, and want to move your messaging to the cloud, AWS recommends using Amazon MQ. It supports open standard APIs and protocols. You can switch from any standards-based message broker to Amazon MQ without the need to rewrite the messaging code in your applications.
- If you are building new applications in the cloud, AWS recommends using Amazon SQS and Amazon SNS.

Section 5 key takeaways



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

42

- Amazon MQ is a managed message-broker service for Apache ActiveMQ that enables you to set up and operate message brokers in the cloud
- Amazon MQ manages the provisioning, setup, and maintenance of ActiveMQ, which is a popular open-source message broker
- Amazon MQ is compatible with open standard APIs and protocols (that is, JMS, NMS, AMQP, STOMP, MQTT, and WebSockets)
- You can use Amazon MQ to integrate on-premises and cloud environments by using the network of brokers feature of ActiveMQ

Some key takeaways from this section of the module include:

- Amazon MQ is a managed message broker service for Apache ActiveMQ that enables you set up and operate message brokers in the cloud
- Amazon MQ manages the provisioning, setup, and maintenance of ActiveMQ, which is a popular open-source message broker
- Amazon MQ is compatible with open standard APIs and protocols (that is, JMS, NMS, AMQP, STOMP, MQTT, and WebSockets)
- You can use Amazon MQ to integrate on-premises and cloud environments by using the network of brokers feature of ActiveMQ

Module wrap-up

Module 12: Building Decoupled Architectures



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

It's now time to review the module and wrap up with a knowledge check and discussion of a practice certification exam question.

Module summary

In summary, in this module, you learned how to:

- Differentiate between tightly and loosely coupled architectures
- Identify how Amazon SQS works and when to use it
- Identify how Amazon SNS works and when to use it
- Describe Amazon MQ



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

44

In summary, in this module, you learned how to:

- Differentiate between tightly and loosely coupled architectures
- Identify how Amazon SQS works and when to use it
- Identify how Amazon SNS works and when to use it
- Describe Amazon MQ

Complete the knowledge check



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

45

It is now time to complete the knowledge check for this module.



Sample exam question

A company must perform asynchronous processing, and implemented Amazon Simple Queue Service (Amazon SQS) as part of a decoupled architecture. The company wants to ensure that the number of empty responses from polling requests are kept to a minimum.

What should a Solutions Architect do to ensure that empty responses are reduced?

Choice	Response
A	Increase the maximum message retention period for the queue
B	Increase the maximum receives for the redrive policy for the queue
C	Increase the default visibility timeout for the queue
D	Increase the long polling wait time for the queue

© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

46

Look at the answer choices and rule them out based on the keywords.

Sample exam question answer



A company must perform asynchronous processing, and implemented Amazon Simple Queue Service (Amazon SQS) as part of a decoupled architecture. The company wants to ensure that the number of empty responses from polling requests are kept to a minimum.

What should a Solutions Architect do to ensure that empty responses are reduced?

The correct answer is D.

The keywords in the question are ensure that empty responses are reduced and increase the long polling wait time for the queue.

© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

47

The following are the keywords to recognize: ensure that empty responses are reduced and increase the long polling wait time for the queue.

The correct answer is D: “Increase the long polling wait time for the queue.” When the ReceiveMessageWaitTimeSeconds property of a queue is set to a value greater than zero, long polling is in effect. Long polling reduces the number of empty responses by enabling Amazon SQS to wait until a message is available before it sends a response to a ReceiveMessage request.

Additional resources

- [Building Loosely Coupled, Scalable, C# Applications with Amazon SQS and Amazon SNS](#)
- [Amazon SQS resources](#)
- [Amazon SNS resources](#)
- [Amazon MQ resources](#)



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

48

If you want to learn more about the topics covered in this module, you might find the following additional resources helpful:

- [Building Loosely Coupled, Scalable, C# Applications with Amazon SQS and Amazon SNS](#)
- [Amazon SQS resources](#)
- [Amazon SNS resources](#)
- [Amazon MQ resources](#)



Thank you

Corrections, feedback, or other questions?
Contact us at <https://support.aws.amazon.com/#/contacts/aws-academy>.
All trademarks are the property of their owners.

© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

49

Thank you for completing this module.



**AWS Academy Cloud Architecting
Module 13 Student Guide
Version 2.0.8**

200-ACACAD-20-EN-SG

© 2022 Amazon Web Services, Inc. or its affiliates. All rights reserved.

This work may not be reproduced or redistributed, in whole or in part, without prior written permission from Amazon Web Services, Inc. Commercial copying, lending, or selling is prohibited.

All trademarks are the property of their owners.

Contents

[Module 13: Building Microservices and Serverless Architectures](#)

4



Module 13: Building Microservices and Serverless Architectures

AWS Academy Cloud Architecting

© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

Welcome to Module 13: Building Microservices and Serverless Architectures.

Module overview

Sections

1. Architectural need
2. Introducing microservices
3. Building microservice applications with AWS container services
4. Introducing serverless architectures
5. Building serverless architectures with AWS Lambda
6. Extending serverless architectures with Amazon API Gateway
7. Orchestrating microservices with AWS Step Functions



Demonstrations

- Creating an AWS Lambda function
- Using AWS Lambda with Amazon S3

Labs

- (Optional) Guided Lab 1: Breaking a Monolithic Node.js Application into Microservices
- Guided Lab 2: Implementing a Serverless Architecture on AWS
- Challenge Lab: Implementing a Serverless Architecture for the Café



Knowledge check

© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

2

This module includes the following sections:

Architectural need

- 1.Introducing microservices
- 2.Building microservice applications with AWS container services
- 3.Introducing serverless architectures
- 4.Building serverless architectures with AWS Lambda
- 5.Extending serverless architectures with Amazon API Gateway
- 6.Orchestrating microservices with AWS Step Functions

This module also includes:

- Two AWS Lambda demonstrations
- An optional guided lab where you refactor a monolithic application into microservices
- A guided lab where you implement a serverless architecture on AWS with Amazon S3, AWS Lambda, Amazon DynamoDB, and Amazon SNS
- A challenge lab where you use AWS Lambda and Amazon Simple Notification Service (Amazon SNS) to generate and send a daily sales report for the café.

Finally, you will be asked to complete a knowledge check that will test your understanding of key concepts covered in this module.

Module objectives

At the end of this module, you should be able to:

- Indicate the characteristics of microservices
- Refactor a monolithic application into microservices and use Amazon ECS to deploy the containerized microservices
- Explain serverless architecture
- Implement a serverless architecture with AWS Lambda
- Describe a common architecture for Amazon API Gateway
- Describe the types of workflows that AWS Step Functions supports



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

3

At the end of this module, you should be able to:

- Indicate the characteristics of microservices
- Refactor a monolithic application into microservices and use Amazon ECS to deploy the containerized microservices
- Explain serverless architecture
- Implement a serverless architecture with AWS Lambda
- Describe a common architecture for Amazon API Gateway
- Describe the types of workflows that AWS Step Functions supports

Section 1: Architectural need

Module 13: Building Microservices and Serverless Architectures



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

Introducing Section 1: Architectural need.

Café business requirement

The café wants to get daily reports via email about all the orders that were placed on the website. They want this information so they can anticipate demand and bake the correct number of desserts going forward (reducing waste). They also want to identify any patterns in their business (analytics).



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

5

Frank and Martha want to get daily reports via email about all the orders that were placed on the website. Frank wants to anticipate demand so he can bake the correct number of desserts going forward (reducing waste). Martha wants to identify any patterns in the café's business (analytics). Currently, Sofía has set up a cron job on the web server instance that sends these daily order report email messages to Frank and Martha. However, the cron job is resource-intensive and reduces web server performance.

Olivia advises Sofía and Nikhil that non-business-critical reporting tasks should be kept separate. Sofía and Nikhil want to further decouple the architecture and move the cron job into a managed, serverless environment that will scale well and reduce costs.

Section 2: Introducing microservices

Module 13: Building Microservices and Serverless Architectures



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

Introducing Section 2: Introducing microservices.

What are microservices?

Applications that are composed of **independent services** that communicate over **well-defined APIs**

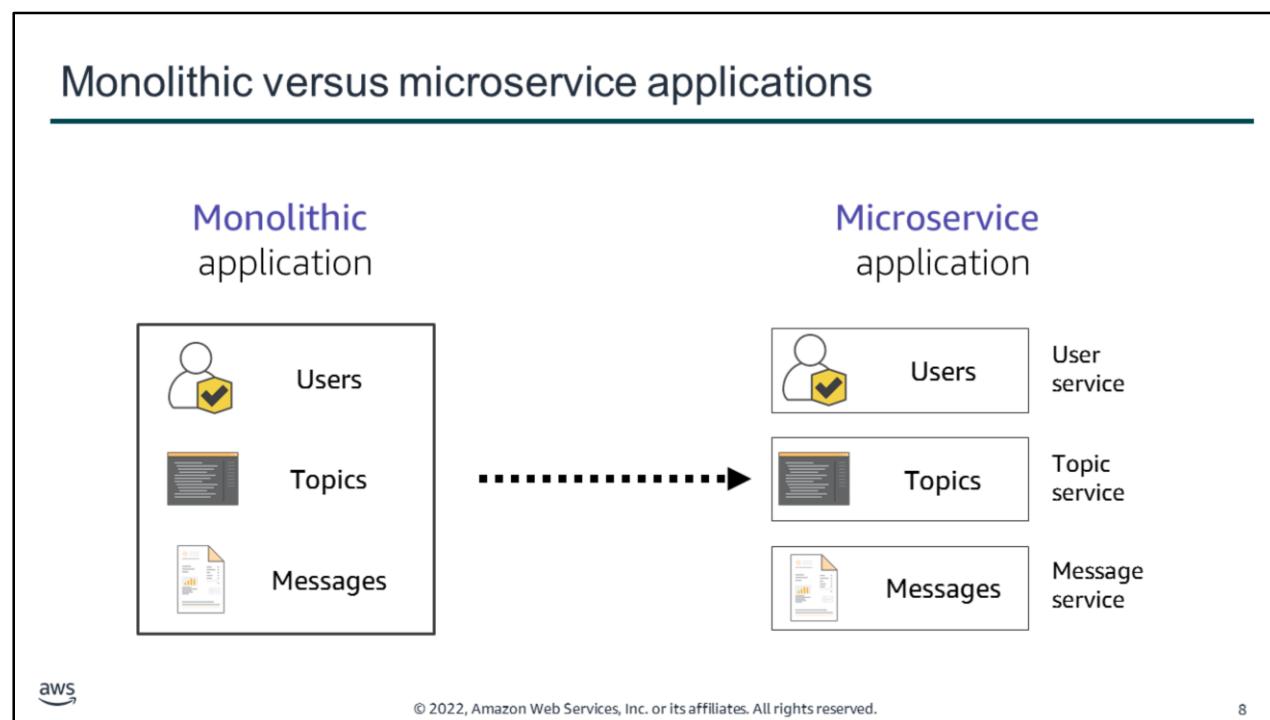


© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

7

Microservices are an architectural and organizational approach to software development where applications are composed of independent services that communicate over well-defined application programming interfaces (APIs). This approach is designed to speed up deployment cycles.

The microservices approach fosters innovation and ownership, and improves the maintainability and scalability of software applications.



To understand the benefits of microservices, consider first a monolithic application.

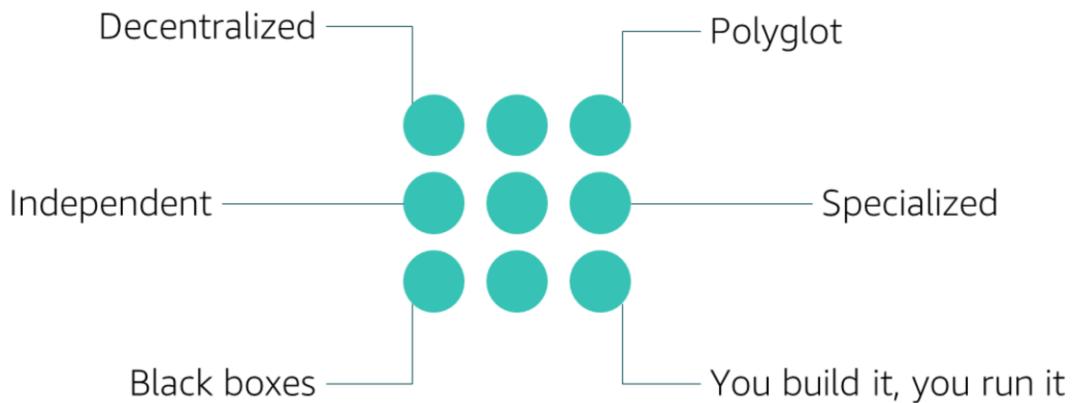
In the example on the left, the three processes (users, topics, and messages) of a monolithic forum application are tightly coupled. They run as a single service. If one process of the application experiences a spike in demand, the entire architecture must be scaled. Adding or improving features becomes more complex as the code base grows, which limits experimentation and makes it difficult to implement new ideas. The availability of monolithic applications is also at risk because many dependent and tightly coupled processes increase the impact of a single process failure.

Now, suppose that the same application runs in a microservice architecture. Each process of the application is built as an independent component that runs as a service. The services communicate by using lightweight API operations. Each service performs a single function that can support multiple applications. Because the services run independently, they can be updated, deployed, and scaled to meet the demand for specific functions of an application.

A microservice architecture provides much quicker iteration, automation, and overall agility. Start fast, fail fast, and recover fast.

For an overview of microservices on AWS, see [What are Microservices?](#)

Characteristics of microservices



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

9

Microservices share some common characteristics:

- Decentralized – Microservice architectures are distributed systems with decentralized data management. They don't rely on a unifying schema in a central database. Each microservice has its own view about data models. Microservices are also decentralized in the way they are developed, deployed, managed, and operated.
- Independent – Each component service in a microservice architecture can be changed, upgraded, or replaced independently without affecting the function of other services. Services do not need to share any of their code or implementation with other services. Similarly, the teams responsible for different microservices can act independently from each other.
- Specialized – Each component service is designed for a set of capabilities and focuses on a specific domain. If the code for a particular component service reaches a certain level of complexity, then the service can be split into two or more services.
- Polyglot – Microservices don't follow a single approach. Teams have the freedom to choose the best tool for their specific problem. As a consequence, microservice architectures take a heterogeneous approach to operating systems, programming languages, data stores, and tools. This approach is called polyglot persistence and programming.
- Black boxes – Individual component services are designed as black boxes, which mean that the details of their complexity are hidden from other components. Any communication between services happens through well-defined APIs to prevent implicit and hidden dependencies.
- You build it, you run it – DevOps is a key organizational principle for microservices, where the team responsible for building a service is also responsible for operating and maintaining it in

production.

Section 2 key takeaways



- Microservice applications are composed of independent services that communicate over well-defined APIs
- Microservices share the following characteristics –
 - Decentralized
 - Independent
 - Specialized
 - Polyglot
 - Black boxes
 - You build it, you run it

© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

10

Some key takeaways from this section of the module include:

- Microservice applications are composed of independent services that communicate over well-defined APIs
- Microservices share the following characteristics –
 - Decentralized: Microservices are decentralized in the way they are developed, deployed, managed, and operated
 - Independent: Each component service in a microservices architecture can be developed, deployed, operated, and scaled without affecting the function of other services
 - Specialized: Each component service is designed for a set of capabilities and focuses on solving a specific problem
 - Polyglot: Microservice architectures take a heterogeneous approach to operating systems, programming languages, data stores, and tools
 - Black boxes: The details of the complexity of microservice components are hidden from other components
 - You build it, you run it: DevOps is a key organizational principle for microservices

Section 3: Building microservice applications with AWS container services

Module 13: Building Microservices and Serverless Architectures

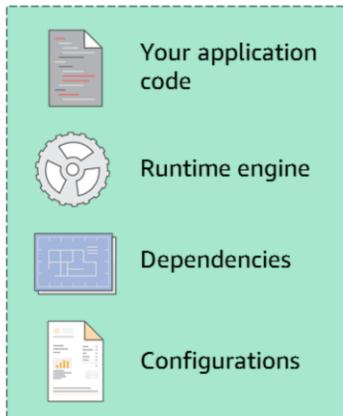


© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

Introducing Section 3: Building microservice applications with AWS container services.

What is a container?

Your container



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

12

When you build a microservice architecture, you can use containers for the processing power.

Containers are a method of operating system virtualization that enables you to run an application and its dependencies in resource-isolated processes. A container is a lightweight, standalone software package. It contains everything that a software application needs to run, such as the application code, runtime engine, system tools, system libraries, and configurations.

A problem that containers solve

Getting software to run reliably in different work environments



Developer's
workstation



Production
environment



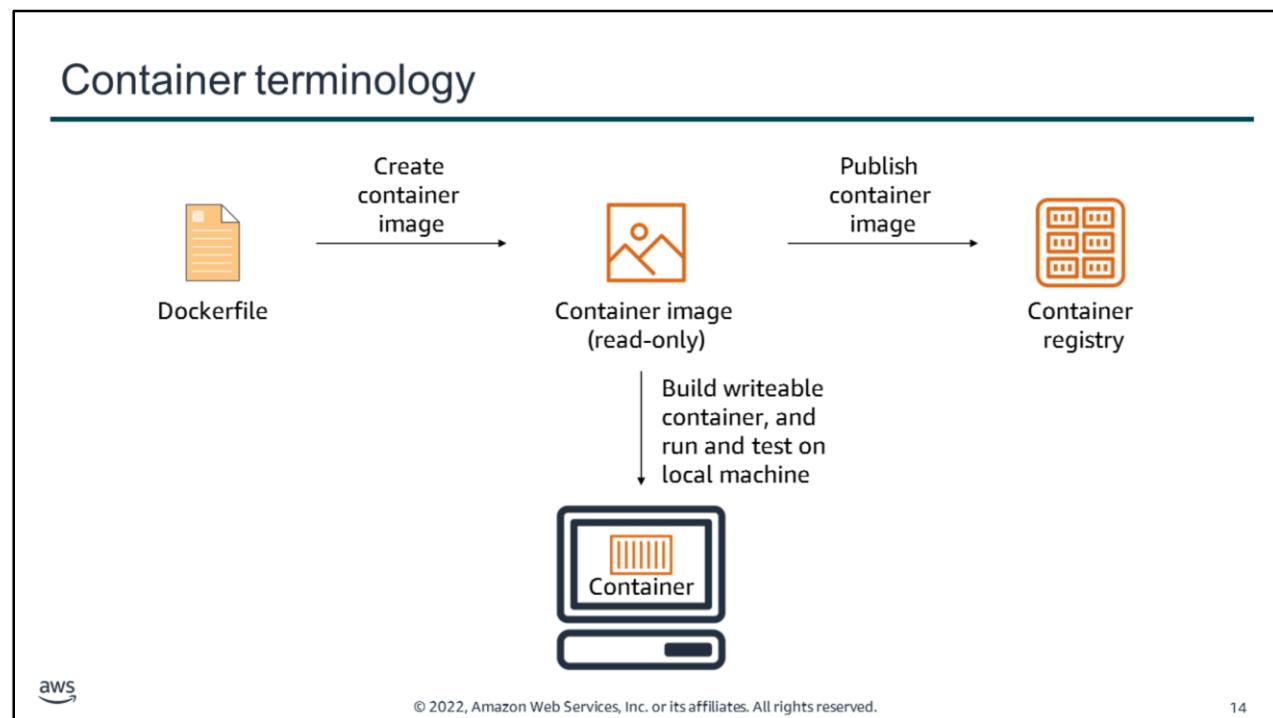
Test
environment



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

13

Containers can help ensure that applications deploy quickly, reliably, and consistently, regardless of deployment environment. Containers also give you more granular control over resources, which improves the efficiency of your infrastructure.

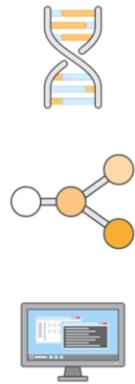


A container is created from a read-only template that is called an *image*. Images are typically built from a Dockerfile, which is a plaintext file that specifies all the components that are included in the container. You can create images from scratch, or you can use images that others created and published to a public or private container registry.

A container image is the snapshot of the file system that is available to the container. For example, you might have the Debian operating system as a container image. When you run this container, a Debian operating system is available to it. You can also package all your code dependencies in the container image and use it as your code artifact.

Container images are stored in a *registry*. You can download the images from the registry and run them on your cluster. Registries can exist in or outside your AWS infrastructure.

Amazon ECS



Orchestrates when containers run

Maintains and scales the fleet of instances that run your containers

Removes the complexity of standing up the infrastructure



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

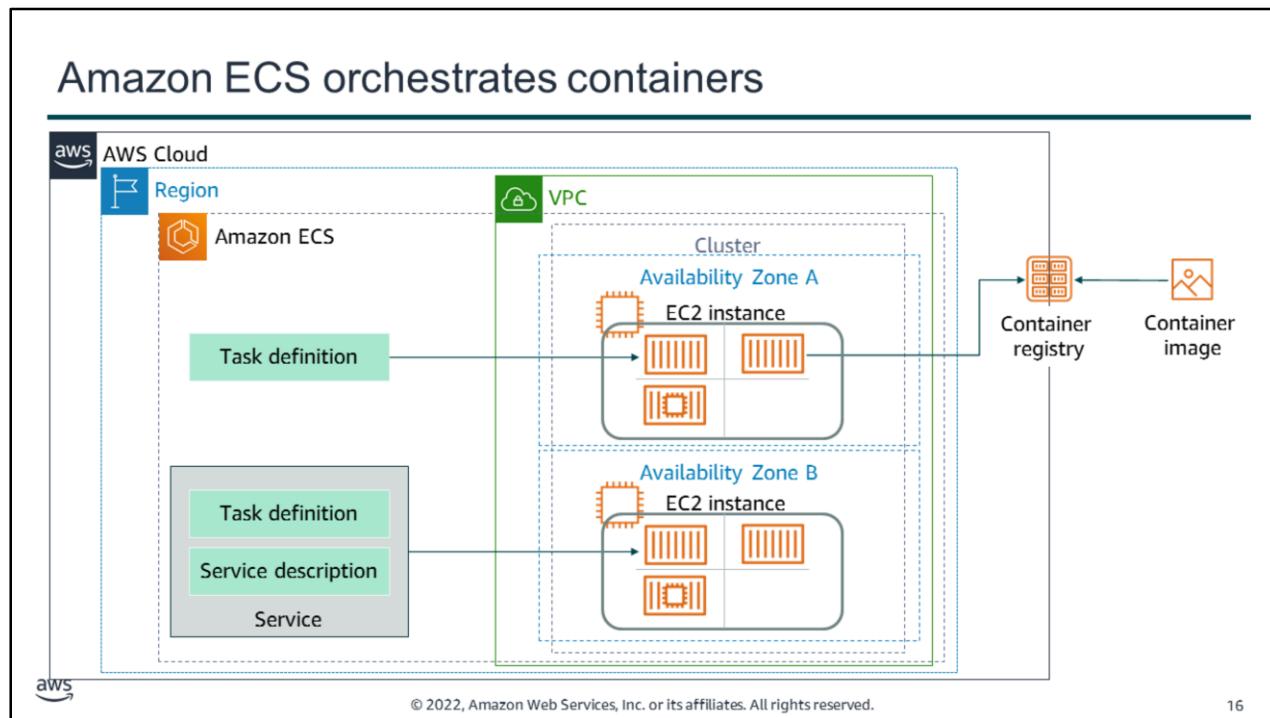
15

You can run your containers on Amazon Elastic Container Service (Amazon ECS). Amazon ECS is a highly scalable, high-performance, container-management service. It supports Docker containers and enables you to easily run applications on a managed cluster of Amazon Elastic Compute Cloud (Amazon EC2) instances.

Amazon ECS is a scalable cluster service for hosting containers that:

- Can scale up to thousands of Docker containers in seconds
- Monitors container deployment
- Manages the state of the cluster that runs the containers
- Schedules containers by using a built-in scheduler or third-party scheduler (Apache Mesos, Blox)
- Is extensible by using APIs
- Can be launched with either AWS Fargate or Amazon EC2 [launch types](#)

You can run ECS clusters at scale by mixing Spot Instances with On-Demand Instances and Reserved Instances.



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 ECS clusters in a new or existing virtual private cloud (VPC). A cluster is a logical grouping of resources.

After a cluster is up and running, you can define task definitions and services that specify which Docker container images to run across your clusters.

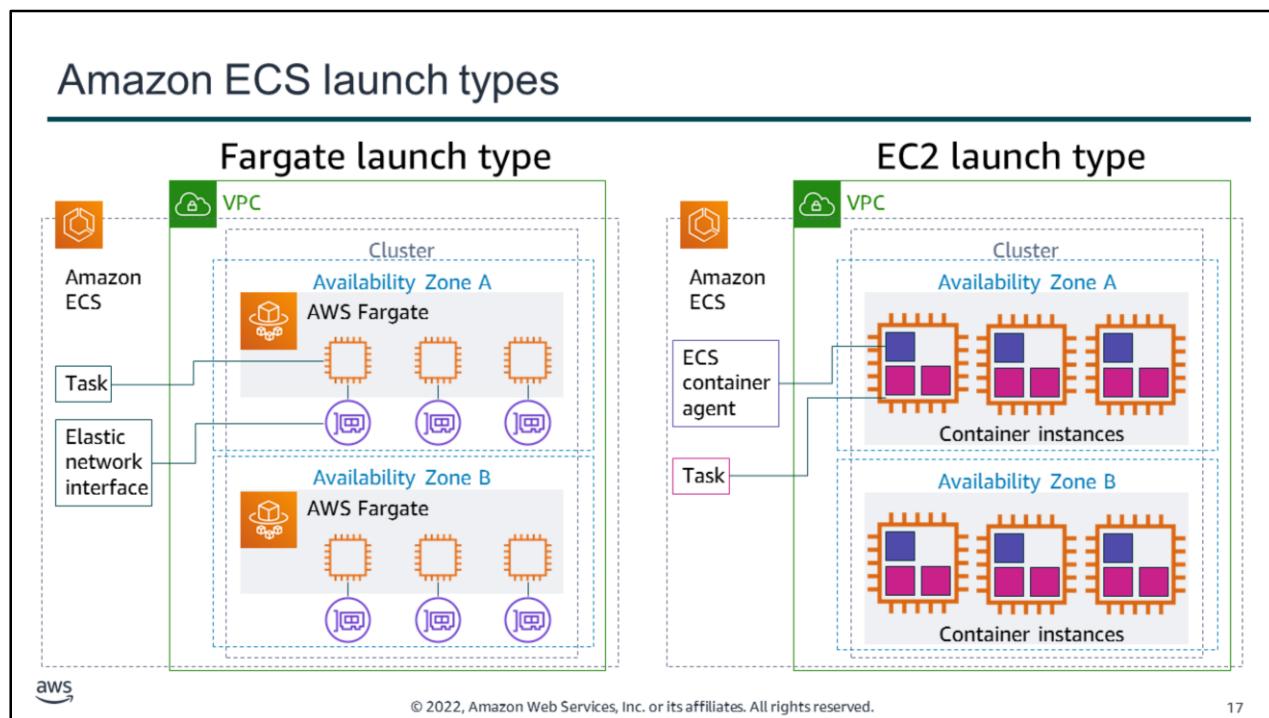
A task definition is a text file in JavaScript Object Notation (JSON) format. It describes one or more containers, up to a maximum of 10, that form your application. You can think of it as a blueprint for your application. Task definitions specify parameters for your application—for example, which containers and launch type to use. Other parameters include which ports should be opened for your application and what data volumes should be used with the containers in the task.

A service enables you to specify how many copies of your task definition to run and maintain in a cluster. You can optionally use an Elastic Load Balancing load balancer to distribute incoming traffic to containers in your service. Amazon ECS maintains that number of tasks and coordinates task scheduling with the load balancer.

After you create a task definition for your application, you can specify the number of tasks that will run on your cluster. A task is the instantiation of a task definition within a cluster. When you

use Amazon ECS to run tasks, you place them in a cluster.

Amazon ECS downloads your container images from a registry that you specify, and runs those images within your cluster.



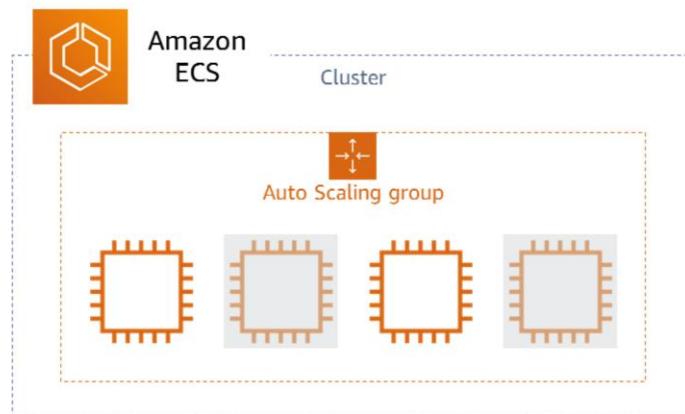
Amazon ECS offers two launch types for hosting your containerized applications.

You can use the Fargate launch type to host your cluster on a serverless infrastructure that Amazon ECS manages. You only need to package your application in containers, specify the CPU and memory requirements, define networking and AWS Identity and Access Management (IAM) policies, and launch the application.

Alternatively, if you want more control, you can use the EC2 launch type to host your tasks on a cluster of EC2 container instances that *you* manage. A *container instance* is an EC2 instance that is running the *Amazon ECS container agent*. You can use Amazon ECS to schedule the placement of containers across your cluster based on your resource needs, isolation policies, and availability requirements. For information about different scheduling options, see [Scheduling Amazon ECS Tasks](#). Amazon ECS keeps track of all the CPU, memory, and other resources in your cluster. It also finds the best server for a container to run on based on your specified resource requirements.

For more information about the Fargate and EC2 launch types, see [Amazon ECS Launch Types](#).

Amazon ECS cluster auto scaling



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

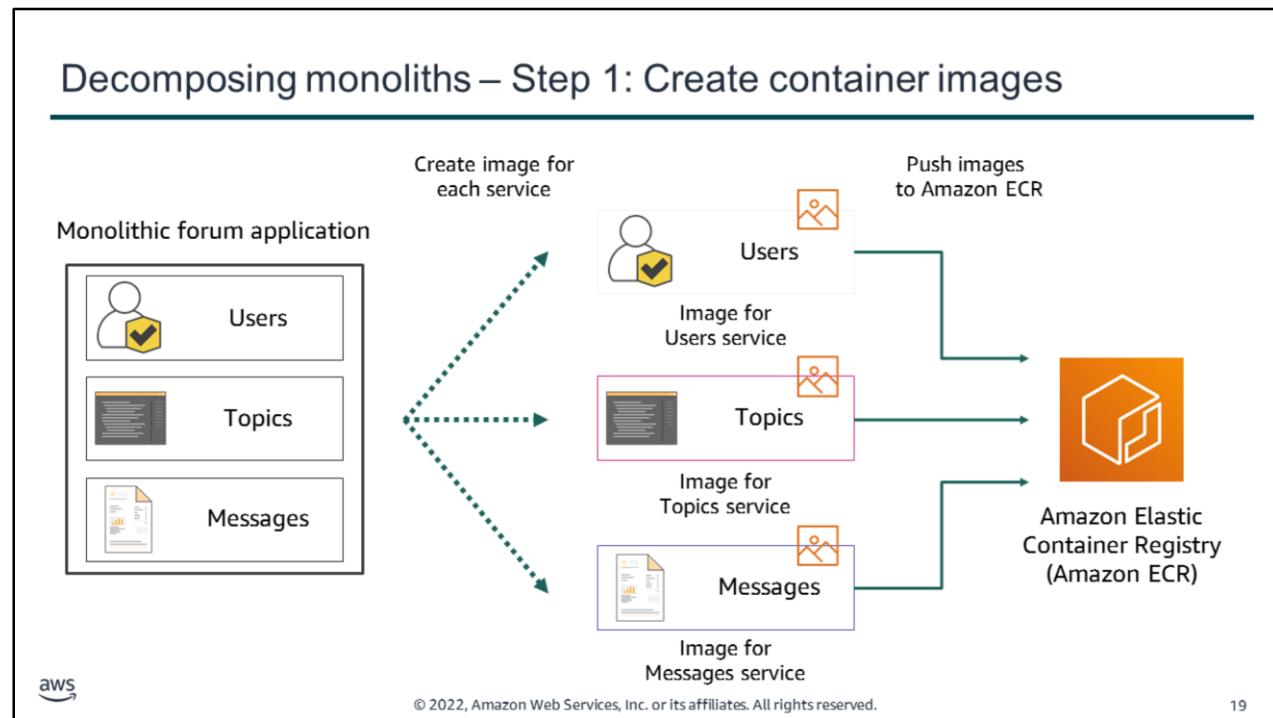
18

You can [create an Auto Scaling group for an Amazon ECS cluster](#). The Auto Scaling group contains container instances that you can scale out (and in) by using Amazon CloudWatch alarms. If you configure your Auto Scaling group to remove container instances, any tasks that are running on the removed container instances are stopped. If your tasks are running as part of a service, Amazon ECS restarts those tasks on another instance if the required resources are available. Examples of such required resources include CPU, memory, ports. However, tasks that were started manually are not restarted automatically.

You can also take advantage of [Amazon ECS cluster auto scaling](#), which gives you more control over how you scale tasks in a cluster. It increases the speed and reliability of cluster scale-out. It gives you control over the amount of spare capacity that is maintained in your cluster, and automatically manages instance termination on scale-in.

With cluster auto scaling, you can configure Amazon ECS to scale your Auto Scaling group in and out automatically. Cluster auto scaling relies on capacity providers, which link your ECS cluster to the Auto Scaling groups that you want to use. Each Auto Scaling group is associated with a capacity provider, and each capacity provider has only one Auto Scaling group. However, many capacity providers can be associated with one ECS cluster. To scale the entire cluster automatically, each capacity provider manages the scaling of its associated Auto Scaling group.

For more information about cluster auto scaling, see the [Amazon ECS Cluster Auto Scaling AWS News Blog post](#).



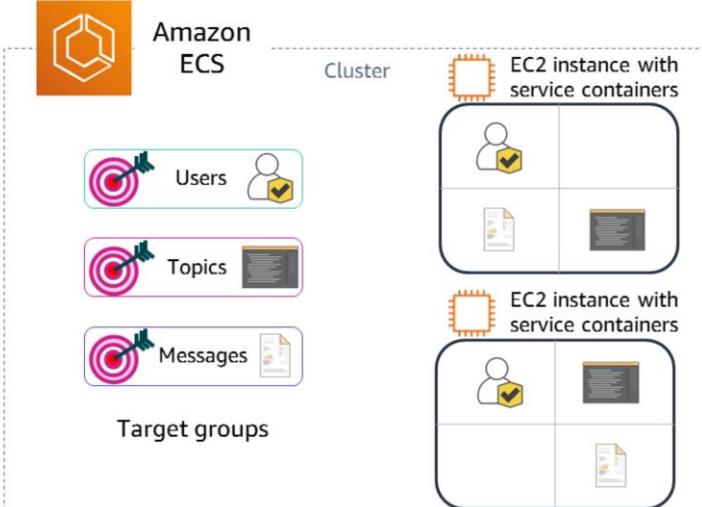
Again, consider the monolithic forum application that you saw earlier where the entire application runs as a single service. To rearchitect this application by using a microservice architecture, you can run each application process as a separate service within its own container. With a microservice architecture, the services can scale and be updated independently of the others.

To deploy the monolithic application as a microservice application, first build and tag an image for each service. Then, register the images with Amazon Elastic Container Registry (Amazon ECR).

Decomposing monoliths – Step 2: Create service task definition and target groups

Service Task Definition

- Launch type = [EC2 or Fargate]
- Name = [service-name]
- Image = [service ECR repo URL]:version
- CPU = [256]
- Memory = [256]
- Container port = [3000]
- Host port = [0]



The diagram illustrates the decomposition of a monolith into microservices using Amazon ECS. It shows a central 'Cluster' box containing three 'Target groups': 'Users', 'Topics', and 'Messages'. Each target group is associated with an 'EC2 instance with service containers' box, which contains icons representing a user profile, a document, and a terminal window. The 'Amazon ECS' logo is positioned above the cluster.

Service Target Group

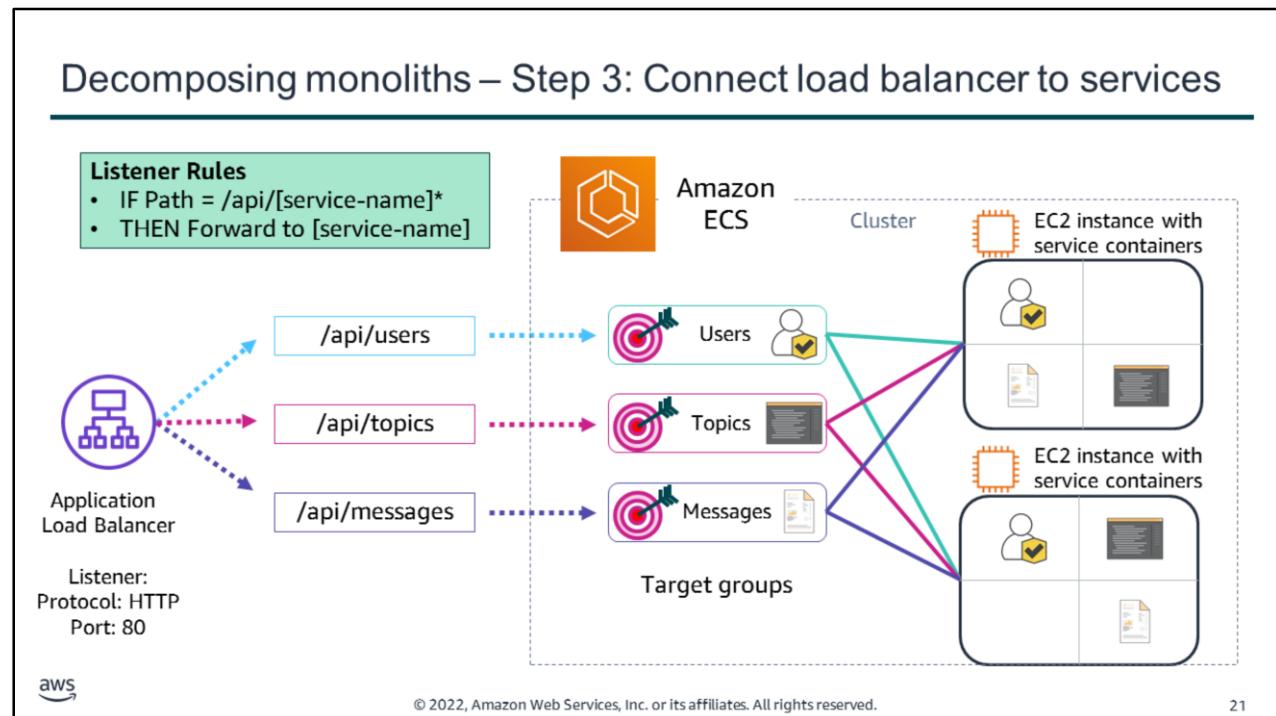
- Name = [service-name]
- Protocol = [HTTP]
- Port = [80]
- VPC = [vpc-name]

© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

20

Next, choose a launch type and create a new service for each piece of the original monolithic application. Amazon ECS deploys each service into its own container across an ECS cluster.

Then, create a target group for each service. The target group tracks the instances and ports of each container that is running for that service.



Finally, create an Application Load Balancer and configure listener rules to connect to the services. The listener checks for incoming connection requests to your load balancer and uses the rules to route traffic appropriately. In the example, the listener for the Application Load Balancer listens for HTTP service requests on Port 80 and routes them to the appropriate service.

Tools for building highly available microservice architectures



AWS Cloud Map

- Is a fully managed discovery service for cloud resources
- Can be used to define custom names for application resources
- Maintains updated location of dynamically changing resources, which increases application availability



AWS App Mesh

- Captures metrics, logs, and traces from all your microservices
- Enables you to export this data to Amazon CloudWatch, AWS X-Ray, and compatible AWS Partner Network (APN) Partner and community tools
- Enables you to control traffic flows between microservices to help ensure that services are highly available



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

22

AWS Cloud Map and AWS App Mesh are two tools that can help you build highly available microservice architectures.

[AWS Cloud Map](#) is a fully managed discovery service for cloud resources. You can use it to define custom names for your application resources (such as databases, queues, microservices, and other cloud resources). AWS Cloud Map maintains the updated location of these dynamically changing resources. This location maintenance increases the availability of your application because your web service always discovers the most up-to-date locations of its resources. You can add and register any resource with minimal manual intervention of mappings. AWS Cloud Map assists with service discovery, continuous integration, and health monitoring of your microservices and applications.

For more information about AWS Cloud Map, read this [AWS Open Source Blog post](#). To learn more about how you can use AWS Cloud Map to enable your containerized services to discover and connect with each other, read [AWS Fargate, Amazon EKS, and Amazon ECS now integrate with AWS Cloud Map](#).

When you create your task definitions, you can enable App Mesh integration. [AWS App Mesh](#) captures metrics, logs, and traces from all of your microservices. You can export this data to Amazon CloudWatch, AWS X-Ray, and compatible AWS Partner Network (APN) Partner and community tools for monitoring and tracing. AWS App Mesh also enables you to control how traffic flows between your microservices to make sure that every service is highly available during deployments, after failures, and as your application scales.

App Mesh enables you to configure microservices to connect directly to each other via a proxy instead of requiring code within the application or by using a load balancer. App Mesh uses Envoy, an open source service-mesh proxy, which is deployed alongside your microservice containers.

For more information about AWS Cloud Map and AWS App Mesh, see this [AWS YouTube video](#).

AWS Fargate



- Is a **fully managed** container service
- Works with **Amazon Elastic Container Service** (Amazon ECS) and **Amazon Elastic Kubernetes Service** (Amazon EKS)
- Provisions, manages, and scales your container clusters
- Manages runtime environment
- Provides automatic scaling



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

23

In this section, you have learned that Amazon ECS offers two launch types: EC2 and Fargate.

[AWS Fargate](#) is a fully managed container service that works with both Amazon ECS and Amazon Elastic Kubernetes Service (Amazon EKS). It enables you to run containers without needing to manage servers or clusters. With AWS Fargate, you no longer need to provision, configure, and scale clusters of virtual machines to run containers. As a result, you don't need to choose server types, decide when to scale your clusters, or optimize cluster packing. AWS Fargate reduces the need for you to interact with or think about servers or clusters. Fargate enables you to focus on designing and building your applications instead of managing the infrastructure that runs them.

Section 3 key takeaways



- [Amazon ECS](#) is a highly scalable, high-performance container management service. It supports Docker containers and enables you to easily run applications on a managed cluster of Amazon EC2 instances.
- [Cluster auto scaling](#) gives you more control over how you scale tasks in a cluster.
- [AWS Cloud Map](#) enables you to define custom names for your application resources. It maintains the updated location of these dynamically changing resources.
- [AWS App Mesh](#) is a service mesh that provides application-level networking. It enables your services to communicate easily with each other across multiple types of compute infrastructure.
- [AWS Fargate](#) is a fully managed container service that enables you to run containers without needing to manage servers or clusters.

© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

24

Some key takeaways from this section of the module include:

- Amazon ECS is a highly scalable, high-performance container management service. It supports Docker containers and enables you to easily run applications on a managed cluster of Amazon EC2 instances.
- Cluster auto scaling gives you more control over how you scale tasks within a cluster.
- AWS Cloud Map enables you to define custom names for your application resources. It maintains the updated location of these dynamically changing resources.
- AWS App Mesh is a service mesh that provides application-level networking to make it easy for your services to communicate with each other across multiple types of compute infrastructure.
- AWS App Mesh is a service mesh that provides application-level networking. It enables your services to communicate easily with each other across multiple types of compute infrastructure.
- AWS Fargate is a fully managed container service that enables you to run containers without needing to manage servers or clusters.

Module 13 – Guided Lab 1: Breaking a Monolithic Node.js Application into Microservices

(Optional lab)



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

25

You might choose to complete Module 13 – Guided Lab 1: Breaking a Monolithic Node.js Application into Microservices. This lab is optional.

Guided lab 1: Tasks

1. Prepare the AWS Cloud9 development environment
2. Run a monolithic application on a basic Node.js server
3. Containerize the monolith for Amazon ECS
4. Deploy the monolith to Amazon ECS
5. Refactor the monolith into containerized microservices



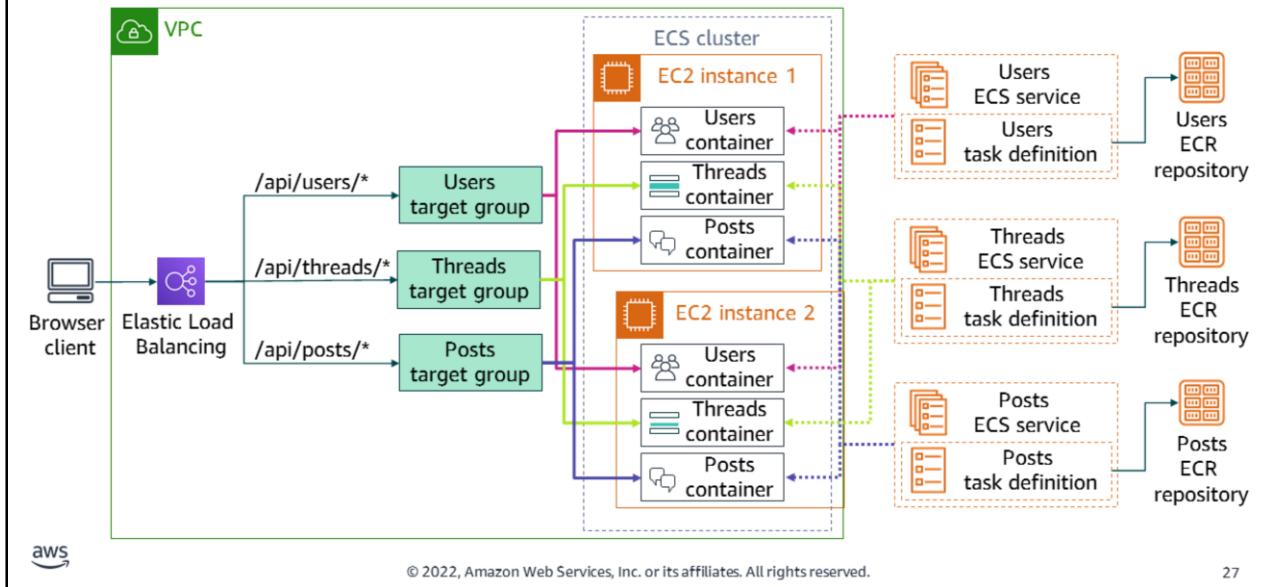
© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

26

In this guided lab, you will complete the following tasks:

1. Prepare the AWS Cloud9 development environment
2. Run a monolithic application on a basic Node.js server
3. Containerize the monolith for Amazon ECS
4. Deploy the monolith to Amazon ECS
5. Refactor the monolith into containerized microservices

Guided lab 1: Final product



27

The diagram summarizes what you will have built after you complete the lab.



A circular icon with a play button symbol and the text "≈ 3 hours".

Begin Module 13 –
Guided Lab 1: Breaking a
Monolithic Node.js
Application into
Microservices

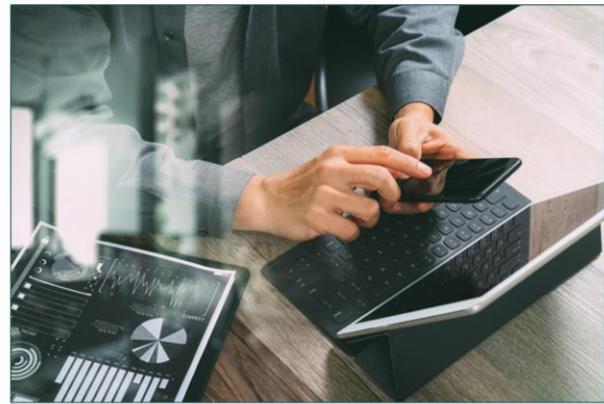
aws

© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

28

It is now time to start the optional guided lab.

Guided lab 1 debrief: Key takeaways



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

29

Your educator might choose to lead a conversation about the key takeaways from this guided lab after you have completed it.

Section 4: Introducing serverless architectures

Module 13: Building Microservices and Serverless Architectures



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

Introducing Section 4: Introducing serverless architectures.

What does serverless mean?

A way for you to build and run applications and services without thinking about servers



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

31

So far, you have learned that you can use Amazon ECS to build your microservice applications by using containers. Amazon ECS is a container orchestration service where you manage your application code, data source integrations, security configuration, updates, network configuration, firewall, and management tasks. You also learned that you can use the Fargate launch type to host your cluster on a *serverless* infrastructure that Amazon ECS manages.

But what does serverless mean?

Serverless is the native architecture of the cloud that enables you to shift more operational responsibilities to AWS, which can increase your agility and innovation. Serverless enables you to build and run applications and services without thinking about servers. Your application still runs on servers. However, AWS does all the server management tasks, such as server or cluster provisioning, patching, operating system maintenance, and capacity provisioning.

Tenets of serverless architectures

No infrastructure provisioning,
no management



Automatic scaling



Pay for value



Highly available and secure



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

32

The tenets that define serverless as an operational model include:

- No infrastructure to provision or manage (no servers to provision, operate, or patch)
- Automatically scales by unit of consumption (scales by unit of work or consumption rather than by server unit)
- *Pay-for-value* pricing model (you pay only for the duration that a resource runs, rather than by server unit)
- Built-in availability and fault tolerance (no need to architect for availability because it is built into the service)

For more information about what serverless is, see [this AWS website](#).

Benefits of serverless



Lower total cost of ownership



Focus on your application, not configuration



Build microservice applications

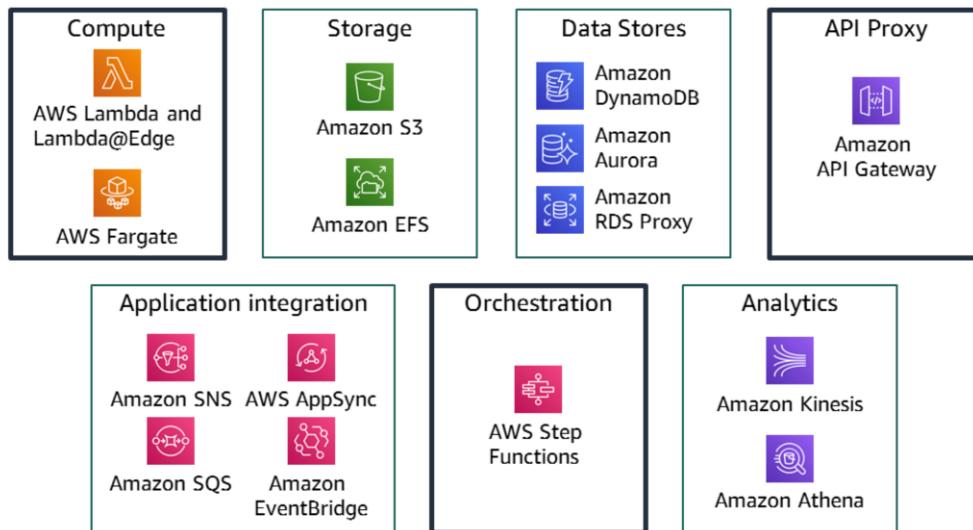


© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

33

Serverless enables you to build [modern applications](#) with increased agility and lower total cost of ownership (TCO). By using a serverless architecture, you can focus on your core product. You don't need to worry about managing and operating servers or runtimes, either in the cloud or on premises. This reduced overhead enables you to reclaim time and energy, which you can spend on developing products that scale and are reliable. Finally, serverless architectures enable you to build microservice applications.

AWS serverless offerings



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

34

AWS has many offerings that you can use to build serverless architectures on AWS. So far in this course, you have already learned about several of them.

The rest of this module focuses on how you can use AWS Lambda, Amazon API Gateway, and AWS Step Functions to build serverless architectures.

Section 4 key takeaways



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

35

- **Serverless computing** enables you to build and run applications and services without provisioning or managing servers
- Serverless architectures offer the following benefits –
 - Lower total cost of ownership (TCO)
 - You can focus on your application
 - You can use them to build microservice applications

Some key takeaways from this section of the module include:

- Serverless computing enables you to build and run applications and services without provisioning or managing servers
- Serverless architectures offer the following benefits –
 - Lower TCO
 - You can focus on your application
 - You can use them to build microservice applications

Section 5: Building serverless architectures with AWS Lambda

Module 13: Building Microservices and Serverless Architectures



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

Introducing Section 5: Building serverless architectures with AWS Lambda.

AWS Lambda



- Is a **fully managed** compute service
- Runs your code on a schedule or in **response to events** (for example, changes to an Amazon S3 bucket or an Amazon DynamoDB table)
- Supports Java, Go, PowerShell, Node.js, C#, Python, Ruby, and Runtime API
- Can run at edge locations closer to your users



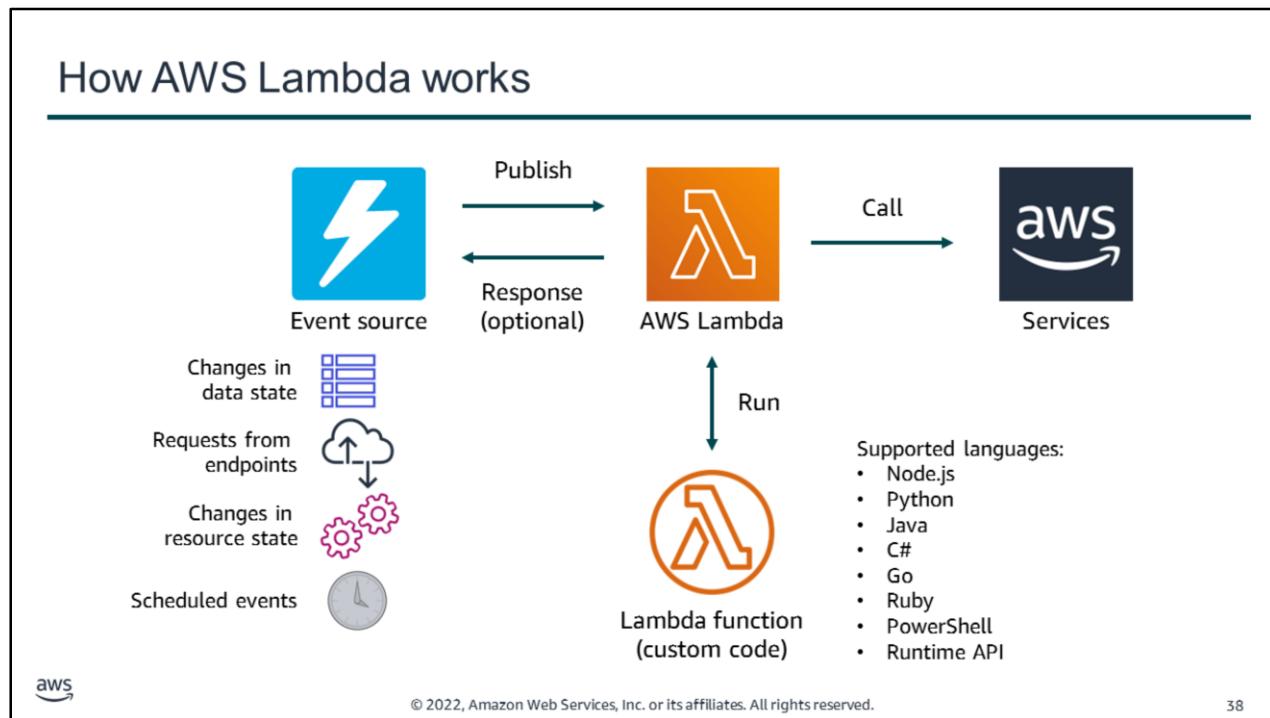
© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

37

[AWS Lambda](#) is a fully managed compute service that runs your code in response to events and automatically manages the underlying compute resources for you. Lambda runs your code on a high-availability compute infrastructure and performs all administration of the compute resources, including server and operating system maintenance, capacity provisioning, automatic scaling, code monitoring, and logging.

AWS Lambda natively supports Java, Go, PowerShell, Node.js, C#, Python, and Ruby code, and provides a Runtime API that enables you to use any additional programming languages to author your functions.

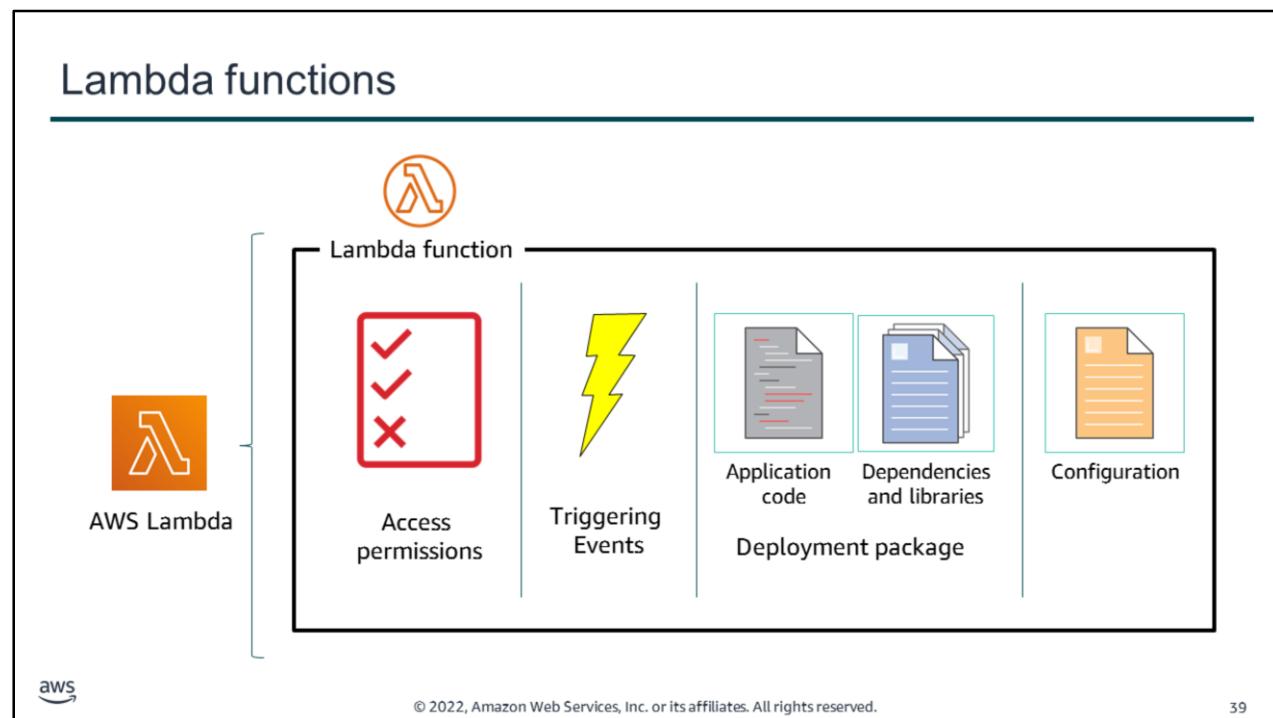
[Lambda@Edge](#) is a feature of Amazon CloudFront that enables you to run code closer to users of your application, which improves performance and reduces latency. Lambda@Edge runs your code in response to events that are generated by the Amazon CloudFront content delivery network (CDN). Lambda@Edge enables you to run Node.js and Python Lambda functions to customize content that Amazon CloudFront delivers. For information about how to add HTTP security response headers, read this [AWS Networking & Content Delivery Blog post](#).



AWS Lambda integrates with other AWS services to invoke Lambda functions. A *Lambda function* is custom code that you write in one of the languages that Lambda supports. You can configure triggers to invoke a function in response to resource lifecycle events, respond to incoming HTTP requests, consume events from a queue, or run on a schedule.

An *event source* is the entity that publishes the event to Lambda. Your Lambda function processes the event, and Lambda runs your Lambda function on your behalf.

Lambda functions are *stateless*, which means that they have no affinity to the underlying infrastructure. Lambda can rapidly launch as many copies of the function as needed to scale to the rate of incoming events.



When you create a Lambda function, you define the permissions for the function and specify which events trigger the function. You also create a deployment package that includes your application code and any dependencies and libraries that are needed to run your code. Finally, you configure runtime parameters such as memory, time out, and concurrency. When your function is invoked, Lambda will run an environment based on the runtime and configuration options that you selected.

For more information about how AWS Lambda runs, see the [AWS Documentation](#).

Anatomy of a Lambda function

Handler()

Function to be run upon invocation

Event object

Data sent during Lambda function invocation

Context object

Methods available to interact with runtime information (request ID, log group, more)

```
import json

def lambda_handler(event, context):
    # TODO implement
    return {
        'statusCode': 200,
        'body': json.dumps('Hello World')
    }
```



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

40

When a Lambda function is invoked, the code begins running at the handler. The handler is a specific code method or function that you create and include in your package. You specify the handler when you create a Lambda function. Each supported language has its own requirements for how a function handler can be defined and referenced within the package. After the handler is successfully invoked inside your Lambda function, the runtime environment belongs to the code you wrote.

The handler always takes two objects: the event object and the context object.

The event object provides information about the event that triggered the Lambda function. This event might be a pre-defined object that an AWS service generates, or a custom user-defined object in the form of a serializable string. An example of such a string might be a plain old Java object (POJO) or a JSON stream.

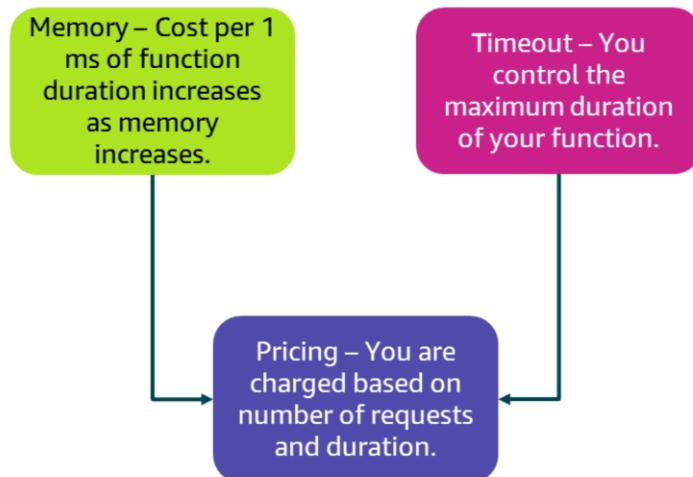
The contents of the event object include all the data and metadata that your Lambda function needs to drive its logic. The contents and structure of the event object vary, depending on which event source created it. For example, an event that is created by API Gateway contains details that are related to the HTTPS request that was made by the API client—such as path, query string, and request body. However, an event that is created by Amazon includes details about the bucket and the new object.

The context object is generated by AWS and provides metadata about the runtime environment. The context object enables your function code to interact with the Lambda runtime environment. The contents and structure of the context object vary based on the language runtime that your Lambda function uses.

However, at a minimum, the context object contains:

- `awsRequestId` – This property is used to track specific invocations of a Lambda function (important for error reporting or when contacting AWS Support)
- `logStreamName` – The CloudWatch log stream that your log statements will be sent to
- `getRemainingTimeInMillis()` – This method returns the number of milliseconds that remain before the running of your function times out

Lambda function configuration and billing



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

41

Memory and timeout are configurations that determine how your Lambda function performs. These configurations affect your billing. With AWS Lambda, you are charged based on the number of requests for your functions (the total number of requests across all your functions) and the duration (the time it takes for your code to run). The price depends on the amount of memory you allocate to your function.

Memory – You specify the amount of memory you want to allocate to your Lambda function. Lambda then allocates CPU power that is proportional to the memory. Lambda is priced so that the cost per 1 ms of function duration increases as the memory configuration increases. For example, say that you have a Lambda function with 256 MB of memory, and that it runs for 110 milliseconds. This function will cost twice as much as a Lambda function with 128 MB of memory that runs for the same time.

Timeout – You can control the maximum duration of your function by using the timeout configuration. You can set the timeout value for a function to any value up to 15 minutes. When the specified timeout is reached, AWS Lambda stops the running of your Lambda function. Using a timeout can prevent higher costs that come from long-running functions. You must find the right balance between not letting the function run too long and being able to finish under normal circumstances.

Follow these best practices:

- Test the performance of your Lambda function to make sure that you choose the optimum memory size configuration. You can view the memory usage for your function in Amazon CloudWatch Logs.

- Load-test your Lambda function to analyze how long your function runs and determine the best timeout value. This is important when your Lambda function makes network calls to resources that might not be able to handle the scaling of Lambda functions.

See the following resources for information about:

- [AWS Lambda limits](#)
- [AWS Lambda pricing](#)

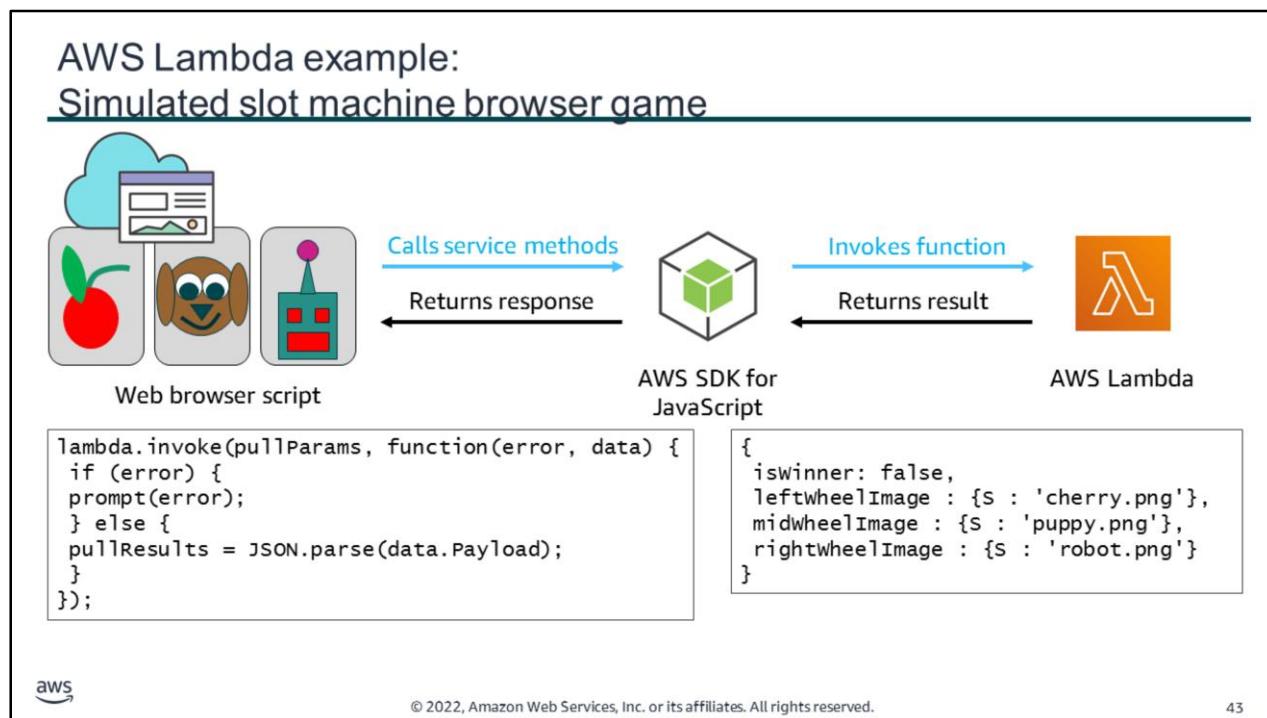
Demonstration: Creating an AWS Lambda function



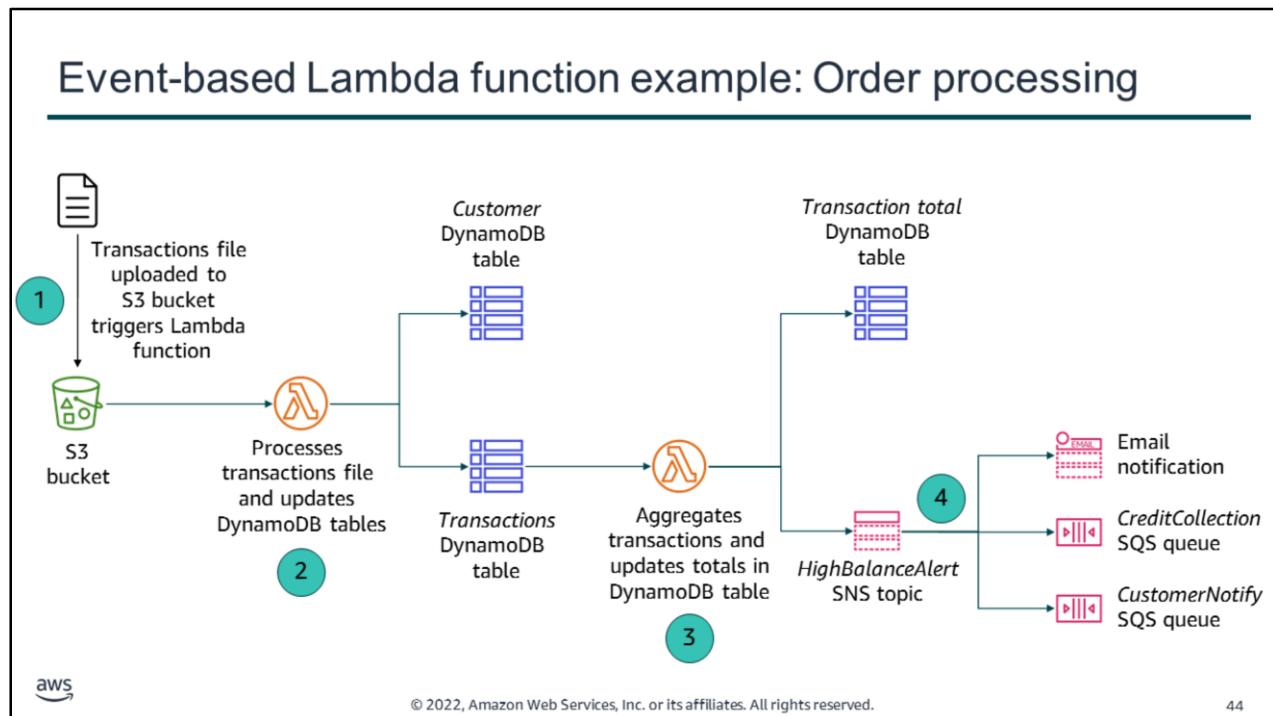
© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

42

Now, the educator might choose to demonstrate how to create an AWS Lambda function.



You can create Lambda functions to perform various tasks. This example uses a browser-based game that simulates a slot machine. The game invokes a Lambda function that generates the random results of each slot pull. The function returns those results as the file names of images that are used to display the result. The images are stored in an Amazon S3 bucket that is configured to function as a static web host for the HTML, CSS, and other assets that are needed to present the application experience.



This example shows how Lambda can be used in a solution for order processing.

In this architecture:

1. A customer uploads a transactions file to an S3 bucket, which triggers the running of a Lambda function.
2. A Lambda function processes the transactions file and updates the *Customer* and *Transactions* DynamoDB tables.
3. Changes to the *Transactions* DynamoDB table trigger a second Lambda function to aggregate the transactions and update the totals in the *Transaction total* DynamoDB table. It also pushes a message to the *HighBalanceAlert* SNS topic.
4. The *HighBalanceAlert* SNS topic sends an email notification to the customer, and updates the *CreditCollection* and *CustomerNotify* SQS queues for payment processing.

Lambda layers



- Enable functions to share code easily – You can upload a layer one time and reference it in any function
- Promote separation of responsibilities – Developers can iterate faster on writing business logic
- Enable you to keep your deployment packages small
- Limits –
 - Up to five layers
 - 250 MB



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

45

When you build serverless applications, it is common to have code that is shared across Lambda functions. It can be custom code that two or more functions use, or a standard library that you add to simplify the implementation of your business logic.

Previously, you packaged and deployed this shared code together with all the functions that used it. Now, you can configure your Lambda function to include additional code and content as layers. A layer is a .zip archive that contains libraries, a custom runtime, or other dependencies.

With Lambda layers, functions can share code. Developers use layers to upload code one time and reuse it multiple times. With layers, you can use libraries in your function without needing to include them in your deployment package.

Sharing code this way can help promote the separation of responsibilities. One person can be responsible for managing the core library. Another person can be responsible for using and building on top of the library code to build application logic.

Layers enable you to keep your deployment package small, which makes development easier.

A function can use up to five layers at a time. The total unzipped size of the function and all layers can't exceed the unzipped deployment package size limit of 250 MB.

For more information about layers, see [AWS Lambda Layers](#).

Demonstration: Using AWS Lambda with Amazon S3



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

46

Now, the educator might choose to demonstrate how to configure an Amazon S3 event to trigger a Lambda function.

Section 5 key takeaways



- Lambda is a serverless compute service that provides built-in fault tolerance and automatic scaling
- A Lambda function is custom code that you write that processes events
- A Lambda function is invoked by a handler, which takes an event object and context object as parameters
- An event source is an AWS service or developer-created application that triggers a Lambda function to run
- Lambda layers enable functions to share code and keep deployment packages small

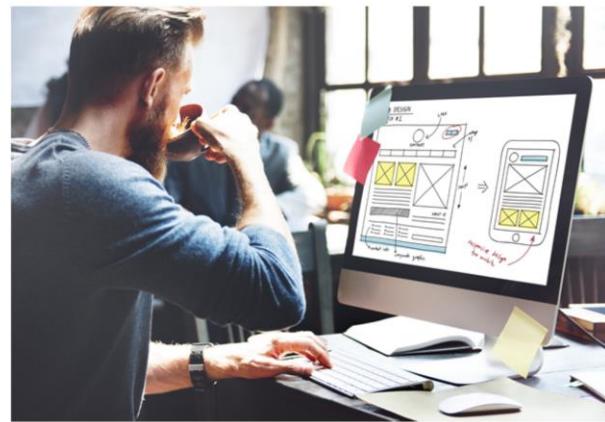
© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

51

Some key takeaways from this section of the module include:

- Lambda is a serverless compute service that provides built-in fault tolerance and automatic scaling.
- A Lambda function is custom code that you write that processes events.
- A Lambda function is invoked by a handler, which takes an event object and context object as parameters.
- An event source is an AWS service or developer-created application that triggers a Lambda function to run.
- Lambda layers enable functions to share code and keep deployment packages small.

Module 13 – Guided Lab 2: Implementing a Serverless Architecture on AWS



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

52

You will now complete Module 13 – Guided Lab 2: Implementing a Serverless Architecture on AWS.

Guided lab 2: Tasks

1. Create a Lambda function to load data
2. Configure an Amazon S3 event
3. Test the loading process
4. Configure notifications
5. Create a Lambda function to send notifications
6. Test the system



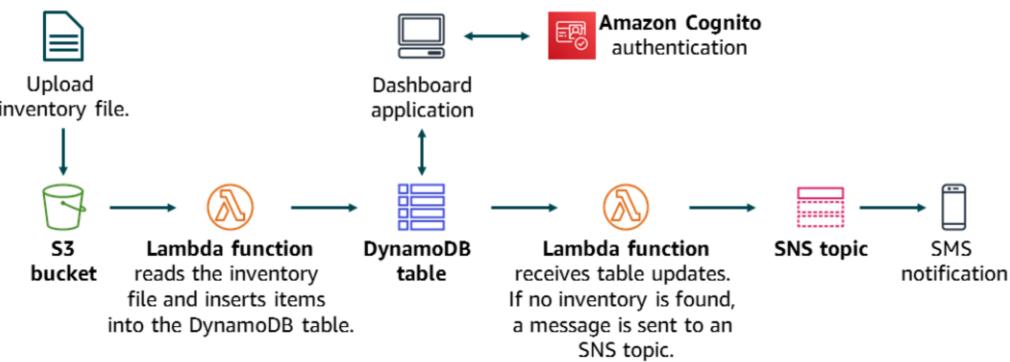
© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

53

In this guided lab, you will complete the following tasks:

1. Create a Lambda function to load data
2. Configure an Amazon S3 event
3. Test the loading process
4. Configure notifications
5. Create a Lambda function to send notifications
6. Test the system

Guided lab 2: Final product



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

54

The diagram summarizes what you will have built after you complete the lab.



A timer icon with the text "~ 40 minutes" indicating the duration of the lab.

Begin Module 13 –
Guided Lab 2:
Implementing a
Serverless Architecture on
AWS

aws

© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

55

It is now time to start the guided lab.

Guided lab 2 debrief: Key takeaways



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

56

Your educator might choose to lead a conversation about the key takeaways from this guided lab after you have completed it.

Section 6: Extending serverless architectures with Amazon API Gateway

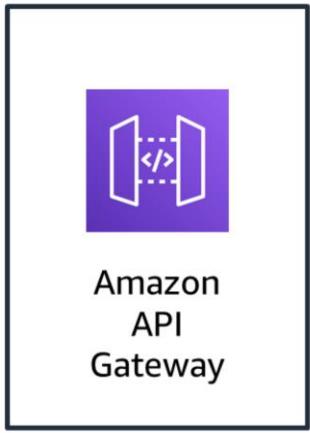
Module 13: Building Microservices and Serverless Architectures



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

Introducing Section 6: Extending serverless architectures with Amazon API Gateway.

Amazon API Gateway



- Enables you to create, publish, maintain, monitor, and secure APIs that act as entry points to backend resources for your applications
- Handles up to hundreds of thousands of concurrent API calls
- Can handle workloads that run on –
 - Amazon EC2
 - Lambda
 - Any web application
 - Real-time communication applications
- Can host and use multiple versions and stages of your APIs



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

58

Amazon API Gateway is a fully managed service that enables you to create, publish, maintain, monitor, and secure APIs at any scale. You can use it to create Representational State Transfer (RESTful) and WebSocket APIs that act as an entry point for applications so they can access backend resources. Applications can then access data, business logic, or functionality from your backend services. Such services include applications that run on Amazon EC2, code that runs on Lambda, any web application, or real-time communication applications.

API Gateway handles all the tasks that are involved in accepting and processing up to hundreds of thousands of concurrent API calls. Such calls might include traffic management, authorization and access control, monitoring, and API version management. API Gateway has no minimum fees or startup costs. You pay only for the API calls you receive and the amount of data that is transferred out. With the API Gateway tiered-pricing model, you can reduce your cost as your API usage scales.

You can use API Gateway to host multiple versions and stages of your APIs.

Amazon API Gateway security



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

59

When you make your APIs publicly available, you are exposed to attackers that try to exploit your services. With Amazon API Gateway, you can protect your APIs in several ways.

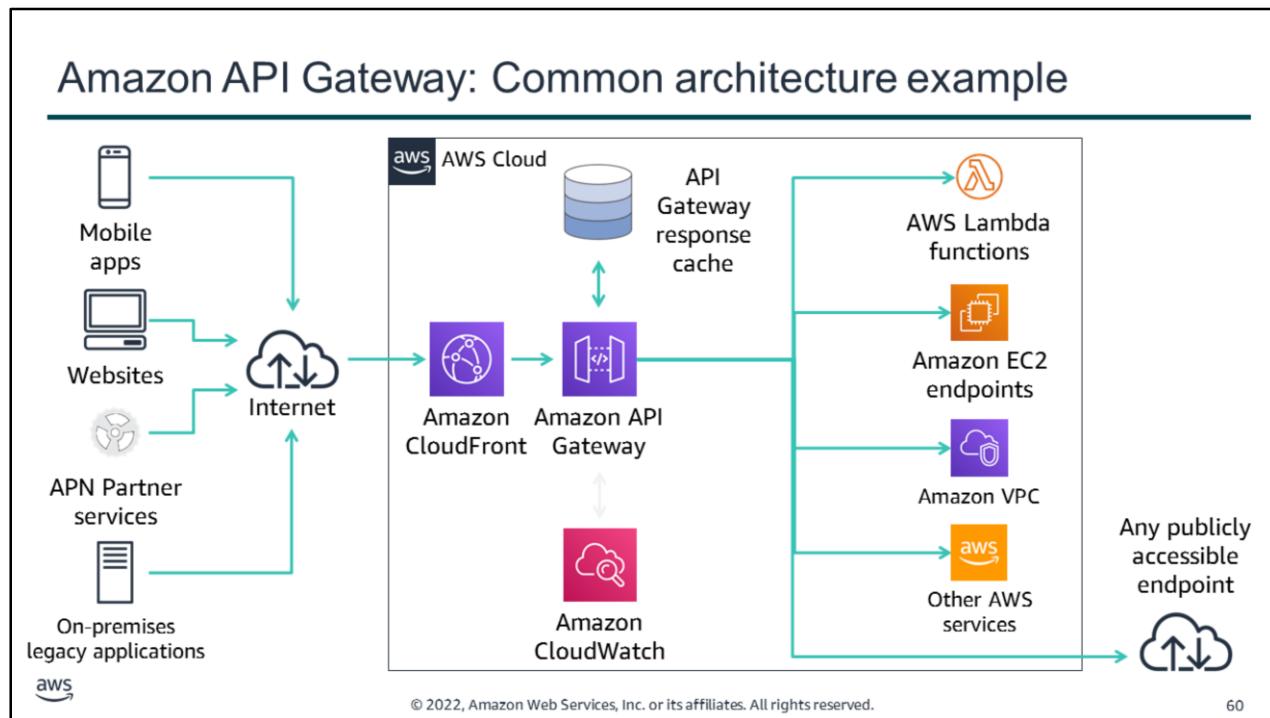
With Amazon API Gateway, you can optionally set your API methods to require authorization. When you set up a method to require authorization, you can use AWS Signature Version 4 or Lambda authorizers to support your own bearer token authentication strategy. AWS Signature Version 4 is the process to add authentication information to AWS requests sent through HTTP. For security, most requests to AWS must be signed with an access key, which consists of an access key ID and secret access key. You use these AWS credentials to sign requests to your service and authorize access, like other AWS services. You can retrieve temporary credentials that are associated with a role in your AWS account by using Amazon Cognito. A Lambda authorizer is a Lambda function that authorizes access to APIs by using a bearer token authentication strategy like OAuth.

You can also apply a resource policy to an API to restrict access to a specific Amazon VPC or VPC endpoint. You can give an Amazon VPC or VPC endpoint from a different account access to the private API by using a resource policy.

Amazon API Gateway supports throttling settings for each method or route in your APIs. You can set a standard rate limit and a burst rate limit per second for each method in your REST APIs and each route in WebSocket APIs.

Additionally, you can use AWS WAF to secure your API Gateway APIs. [AWS WAF](#) is a web application firewall that helps protect your web applications from common web exploits that could affect availability, compromise security, or consume excessive resources.

For more information about how to secure your APIs with Amazon API Gateway, see the [Security & Authorization section](#) of Amazon API Gateway FAQs.



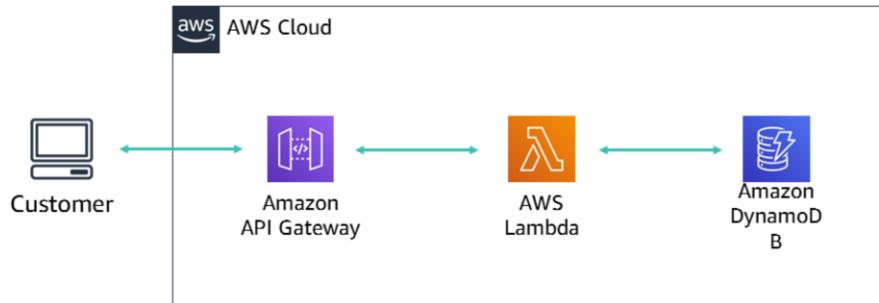
You can use Amazon API Gateway to provide the API layer for your applications. Here is an example of a common architecture with Amazon API Gateway.

In this example, front-end client applications and application services send traffic to API Gateway over the internet. Often, Amazon CloudFront is used to cache static content. API Gateway abstracts and exposes APIs that can call various backend applications. These applications include Lambda functions, Docker containers running on EC2 instances, virtual private clouds (VPCs), or any publicly accessible endpoint. If necessary, API Gateway can cache the response. Finally, all the API calls can be monitored with Amazon CloudWatch.

You can use API Gateway with other AWS managed services to build serverless backends for your applications. For example, API Gateway can proxy requests to Lambda functions that run your code and generate responses. You can even create APIs that proxy requests to other AWS services, such as Amazon Simple Storage Service (Amazon S3), without having to write any code. You can get production-scale backends running more quickly because you have less code to write, and you also offloaded operational burden to the AWS services.

For an example of how to use API Gateway with AWS Lambda, see this [AWS blog post](#).

Example: RESTful microservices



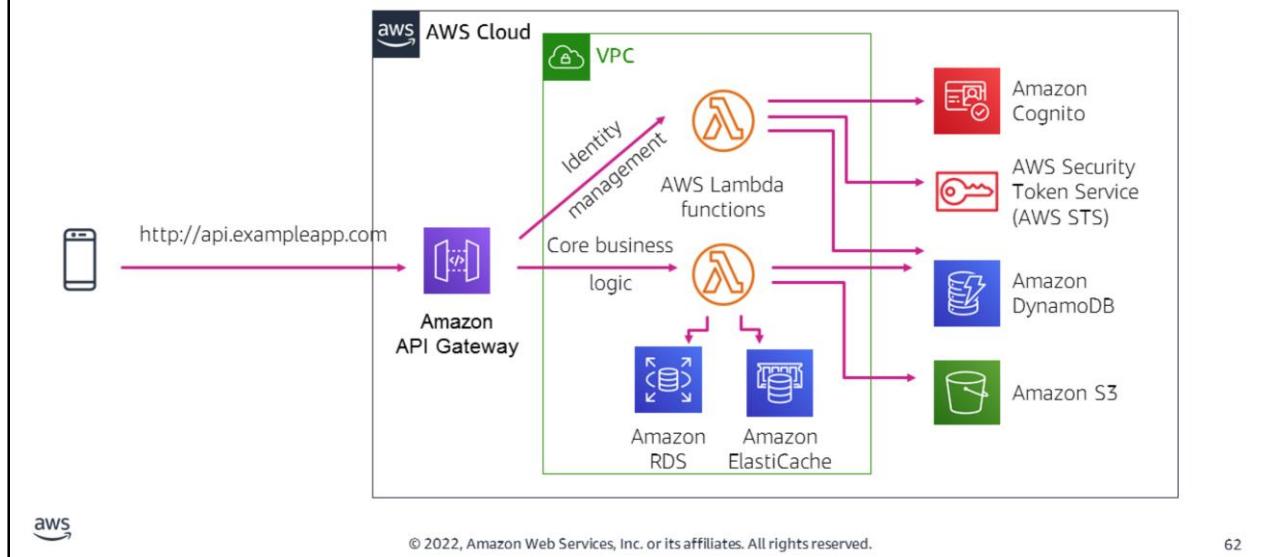
© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

61

Amazon API Gateway is deeply integrated with AWS Lambda. This example shows how the two services can be used together in an architecture for a RESTful microservice application.

In this example, customers can use your microservice by making HTTP API calls. Amazon API Gateway hosts RESTful HTTP requests from and responses to your customers. In this scenario, API Gateway provides built-in authorization, throttling, security, fault tolerance, request-response mapping, and performance optimizations. AWS Lambda contains the business logic to process incoming API calls and uses DynamoDB as a persistent storage. Amazon DynamoDB persistently stores microservice data and scales based on demand. Because microservices are designed to do one thing well, a schemaless NoSQL data store is regularly incorporated.

Example: Serverless mobile backend



Consider yet another example of how Amazon API Gateway can be used with Lambda in a serverless architecture as the gateway to the backend for a mobile application. Mobile apps are expected to have a fast, consistent, and feature-rich user experience, and they often have a global footprint. Further, mobile user patterns are dynamic, with unpredictable peak usage. Mobile apps require a rich set of mobile services that work together seamlessly without sacrificing control and flexibility of the backend infrastructure.

In this example, a mobile app sends a request to Amazon API Gateway, which forwards the request to a Lambda function that calls Amazon Cognito and AWS Security Token Service (AWS STS) to manage identity. Amazon Cognito authenticates users of your mobile app through external identity providers (IdPs) that support Security Assertion Markup Language (SAML) or OpenID Connect, social IdPs (for example, Facebook, Twitter, Amazon), and custom IdPs. After the identity of the mobile user is confirmed, another Lambda function runs the core business logic of the app.

Section 6 key takeaways



- Amazon API Gateway is a fully managed service that enables you to create, publish, maintain, monitor, and secure APIs at any scale.
- Amazon API Gateway acts as an entry point to backend resources for your applications. It abstracts and exposes APIs that can call various backend applications. These applications include Lambda functions, Docker containers that run on EC2 instances, VPCs, or any publicly accessible endpoint.
- Amazon API Gateway is deeply integrated with Lambda.

© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

63

Some key takeaways from this section of the module include:

- Amazon API Gateway is a fully managed service that enables you to create, publish, maintain, monitor, and secure APIs at any scale.
- Amazon API Gateway acts as an entry point to backend resources for your applications. It abstracts and exposes APIs that can call various backend applications. These applications include Lambda functions, Docker containers that run on EC2 instances, VPCs, or any publicly accessible endpoint.
- Amazon API Gateway is deeply integrated with Lambda.

Section 7: Orchestrating microservices with AWS Step Functions

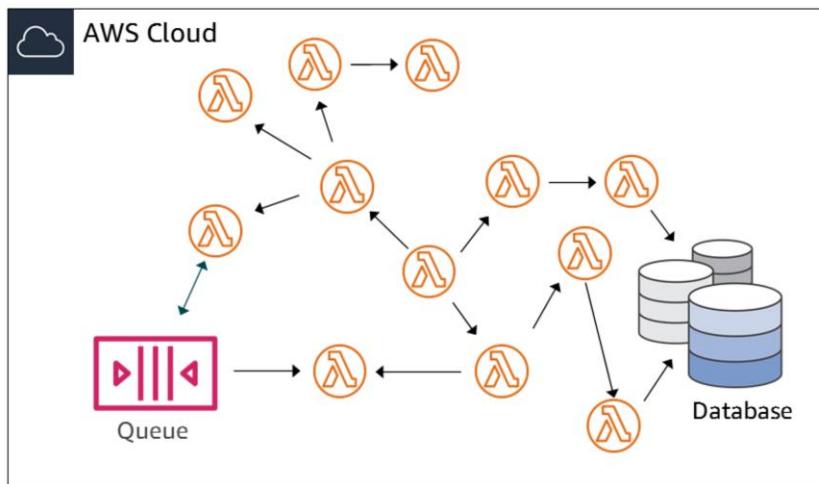
Module 13: Building Microservices and Serverless Architectures



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

Introducing Section 7: Orchestrating microservices with AWS Step Functions.

Challenges with microservice applications



As your application grows in size, the number of components increases, and many patterns of which tasks to run and in which order are possible. Consider a microservice application that is built by using Lambda functions, for example. You might want to invoke a Lambda function immediately after another function, and only if the first function runs successfully. You might want two functions to be invoked in parallel and then feed the combined results into a third function. Or you might want to choose which of two functions is invoked based on the output of another function.

Function invocation can result in an error for several reasons. Your code might raise an exception, time out, or run out of memory. The runtime that runs your code might encounter an error and stop. When an error occurs, your code might have run completely, partially, or not at all. In most cases, the client or service that invokes your function retries if it encounters an error, so your code must be able to process the same event repeatedly without unwanted effects. If your function manages resources or writes to a database, you must handle cases where the same request is made several times.

You need a way to coordinate the components of your application. This coordination layer must be able to scale automatically in response to changing workloads, and handle errors and timeouts. It must also maintain the state of your application while it is running, such as tracking which step is currently running and storing data that is moving between the steps of your workflow. These features help you build and operate your applications. You also want visibility

into your application so that you can troubleshoot errors and track performance.

AWS Step Functions



- Coordinates microservices by using visual workflows
- Enables you to step through the functions of your application
- Automatically triggers and tracks each step
- Provides simple error catching and logging if a step fails

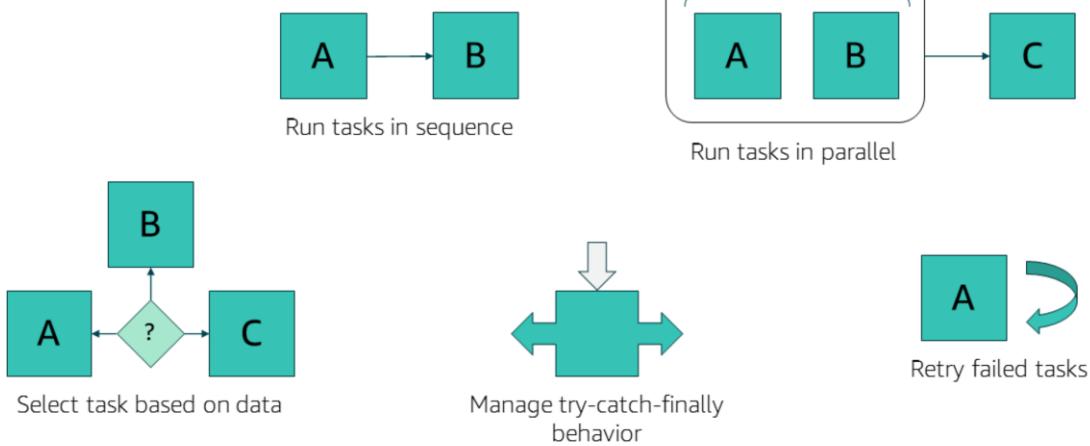


© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

66

[AWS Step Functions](#) is a web service that enables you to coordinate components of distributed applications and microservices by using visual workflows. Step Functions provides a reliable way to coordinate components and step through the functions of your application. Step Functions offers a graphical console so that you can visualize the components of your application as a series of steps. It automatically triggers and tracks each step, and it also retries when there are errors, so your application runs in order and as expected. Step Functions logs the state of each step, so you can diagnose and debug problems quickly.

Workflow coordination



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

67

AWS Step Functions manages the logic of your application for you. It implements basic primitives, such as running tasks sequentially or in parallel, branching, and timeouts. This technique removes extra code that might be repeated in your microservices and functions. AWS Step Functions automatically handles errors and exceptions with built-in try-catch and retry, whether the task takes seconds or months to complete. You can automatically retry failed or timed-out tasks. You can respond differently to different types of errors and recover gracefully by falling back to designated cleanup and recovery code.

State machines



A **state machine** is a collection of states that can do work.

Vending machine

Waiting for transaction

Soda selection

Vend soda



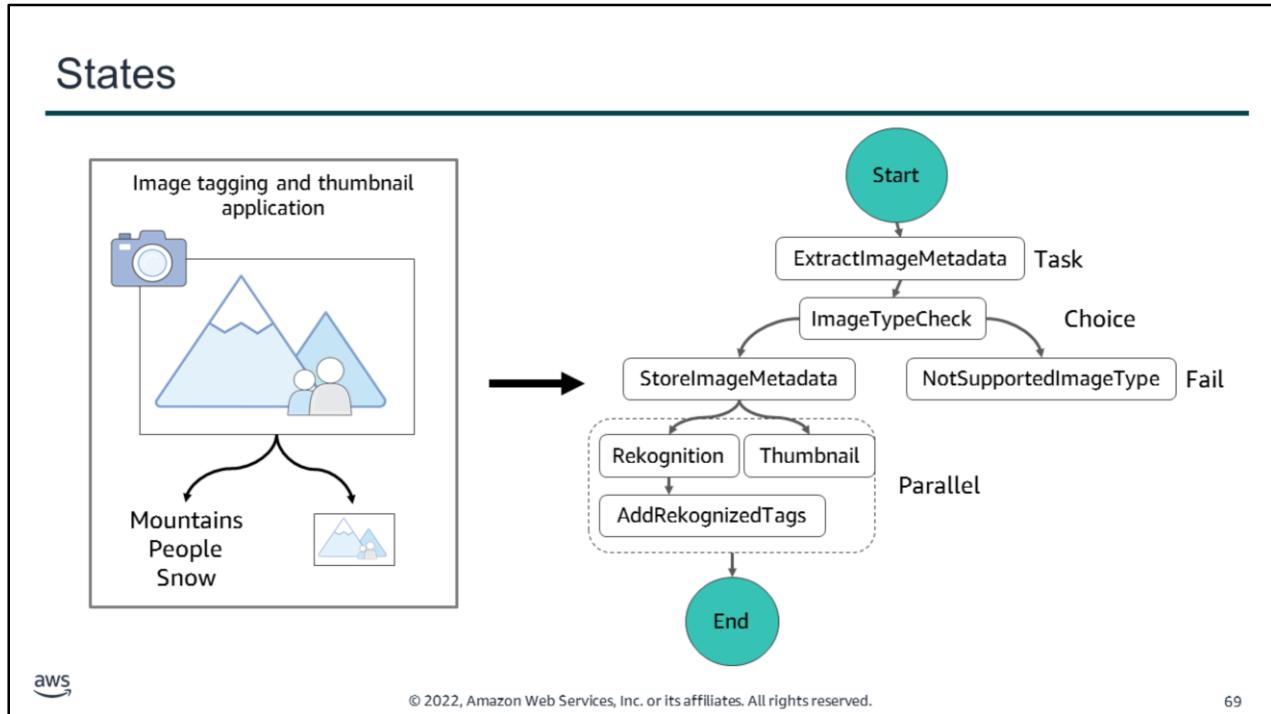
© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

68

AWS Step Functions enables you to create and automate your own state machines within the AWS environment by using the JSON-based Amazon States Language. This language contains a structure that is made of various states, tasks, choices, error handling and more.

A *state machine* is collection of states that can do work.

A common example of a state machine is a soda vending machine. The machine starts in the operating state (waiting for a transaction), and then moves to soda selection when money is added. Next, it enters a vending state, where soda is deployed to the customer. After completion, the state returns back to operating.



A finite state machine can express an algorithm as a number of states, their relationships, and their input and output. *States* are elements in your state machine. Individual states can make decisions based on their input, perform actions, and pass output to other states.

State types

Task	A single unit of work performed by a state machine
Choice	Adds branching logic to a state machine
Fail	Stops a running state machine and marks it as a failure
Succeed	Stops a running state machine successfully
Pass	Passes its input to its output, without performing work
Wait	Delays from continuing for a specified time
Parallel	Creates parallel branches to run in your state machine
Map	Dynamically iterates steps



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

70

States can perform various functions in your state machine:

- Do some work in your state machine (a *Task* state)
- Make a choice between branches of state machines to run (a *Choice* state)
- Stop an running state machine with a failure or success (a *Fail* state or *Succeed* state)
- Pass its input to its output or inject some fixed data (a *Pass* state)
- Provide a delay for a certain amount of time or until a specified time and date (a *Wait* state)
- Begin parallel branches to run in your state machine (a *Parallel* state)
- Dynamically iterate steps (a *Map* state)

For more information about state types, see [States](#) in the AWS Documentation.

Amazon States Language

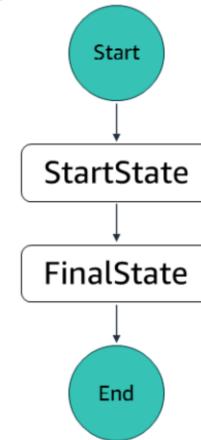
Define workflow in JSON by using the Amazon States Language

```
{  
  "Comment": "An example of the ASL.",  
  "StartAt": "StartState",  
  "States": {  
    "StartState": {  
      "Type": "Task",  
      "Resource": "arn:aws:lambda:us-east...",  
      "Next": "FinalState"  
    }  
    "FinalState": {  
      "Type": "Task",  
      "Resource": "arn:aws:lambda:us-east...",  
      "End": true  
    }  
  }  
}
```



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

Visualize workflow in the Step Functions console

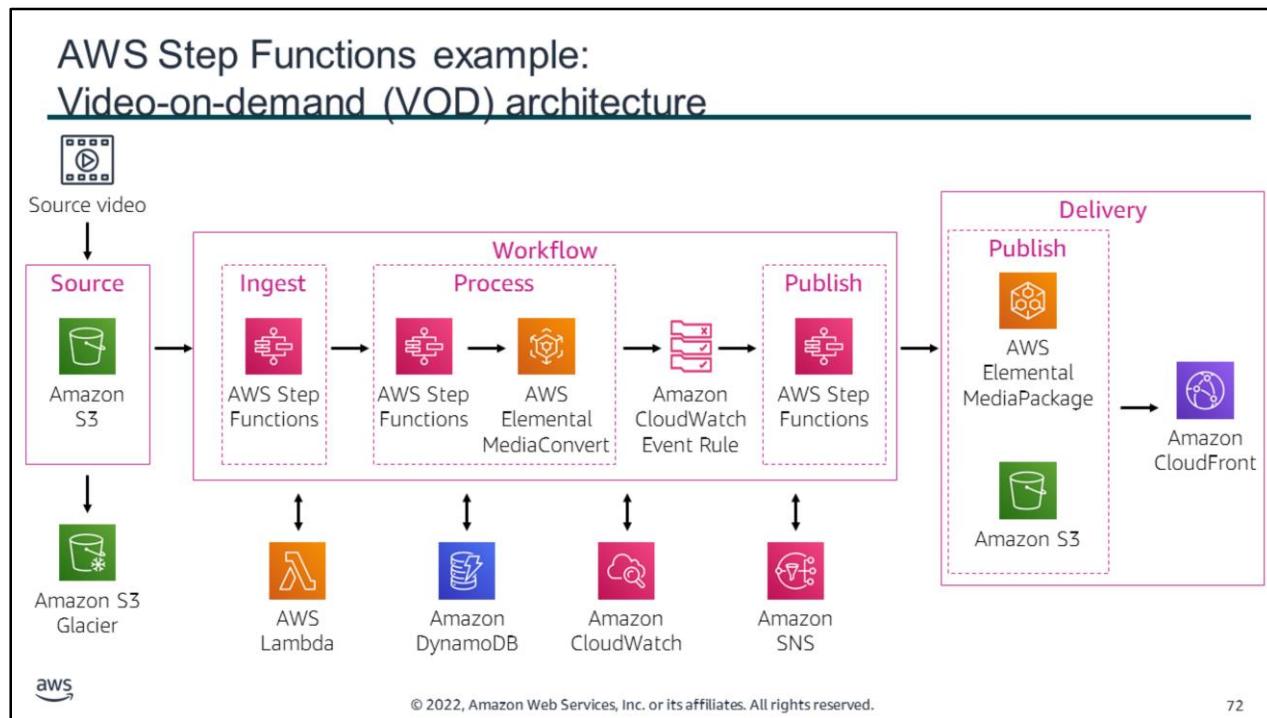


71

You define state machines by using the Amazon States Language. The Amazon States Language is a JSON-based, structured language. AWS Step Functions then represents the JSON structure in a real-time graphical view. In this way, you can visualize your state machine directly in the Step Functions console.

Here, you see an example of a state machine with two task state types.

For more information about the Amazon States Language, see the [AWS Documentation](#).



This architecture diagram shows an example use case for AWS Step Functions in a video-on-demand (VOD) solution. Another key component in this architecture is AWS Elemental MediaConvert, which is a file-based video transcoding service with broadcast-grade features. It enables you to create VOD content for broadcast and multiscreen delivery at scale.

In this solution, source videos and metadata files are ingested and processed for playback on a wide range of devices.

- AWS Step Functions creates **Ingest**, **Process**, and **Publish** step functions.
- AWS Lambda functions perform the work of each step and process error messages.
- Amazon S3 buckets store source and destination media files.
- Amazon CloudWatch is used for logging.
- Amazon CloudWatch event rules for AWS Elemental MediaConvert notifications.
- An Amazon DynamoDB table stores data captured through the workflow.
- Amazon SNS topics send encoding, publishing, and error notifications.
- The transcoded media files are stored for on-demand delivery to users through Amazon CloudFront.

For more information about this architecture, see [Architecture Overview](#).

Section 7 key takeaways



- AWS Step Functions is a web service that enables you to coordinate components of distributed applications and microservices by using visual workflows
- AWS Step Functions enables you to [create and automate your own state machines](#) within the AWS environment
- AWS Step Functions [manages the logic of your application for you](#), and it implements basic [primitives](#), such as sequential or parallel branches, and timeouts
- You define state machines by using the [Amazon States Language](#)

© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

73

Some key takeaways from this section of the module include:

- AWS Step Functions is a web service that enables you to coordinate components of distributed applications and microservices by using visual workflows
- AWS Step Functions enables you to create and automate your own state machines within the AWS environment
- AWS Step Functions manages the logic of your application for you, and it implements basic primitives, such as sequential or parallel branches, and timeouts
- You define state machines by using the Amazon States Language

Module 13 – Challenge Lab: Implementing a Serverless Architecture for the Café



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

74

You will now complete Module 13 – Challenge Lab: Implementing a Serverless Architecture for the Café.

The business need: A serverless environment



Sofía and Nikhil want to further decouple the architecture and move the cron job into a managed, serverless environment that will scale well and reduce costs.



Frank and Martha want to get daily email reports about all the orders that were placed on the website. Olivia advises Sofía and Nikhil that non-business-critical reporting tasks should be kept separate from the production web server instance.



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

75

Frank and Martha want to get daily reports via email about all the orders that were placed on the website. Frank wants to anticipate demand so he can bake the correct number of desserts going forward (reducing waste). Martha wants to identify any patterns in the café's business (analytics). Currently, Sofía has set up a cron job on the web server instance that sends these daily order report email messages to Frank and Martha. However, the cron job is resource-intensive and reduces the performance of the web server.

Olivia advises Sofía and Nikhil that non-business-critical reporting tasks should be kept separate. Sofía and Nikhil want to further decouple the architecture and move the cron job into a managed, serverless environment that will scale well and reduce costs.

Challenge lab: Tasks

1. Downloading the source code
2. Creating the *DataExtractor* Lambda function in the VPC
3. Creating the *salesAnalysisReport* Lambda function
4. Creating an SNS topic
5. Creating an email subscription to the SNS topic
6. Testing the *salesAnalysisReport* Lambda function
7. Setting up an Amazon EventBridge event to trigger the Lambda function each day



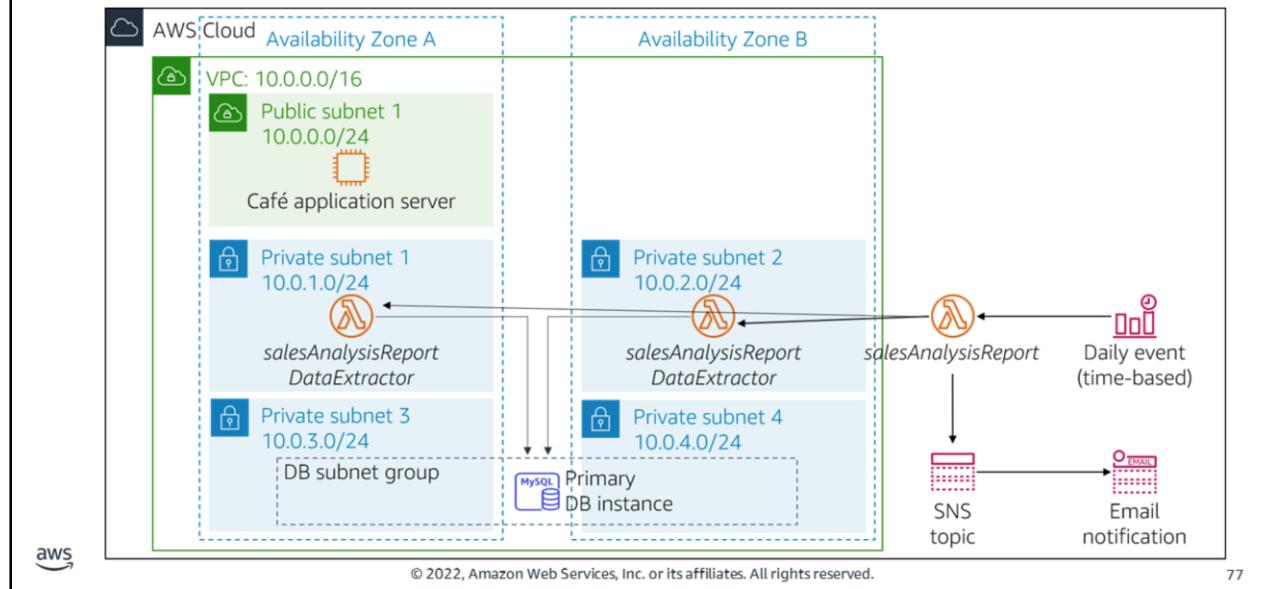
© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

76

In this challenge lab, you will complete the following tasks:

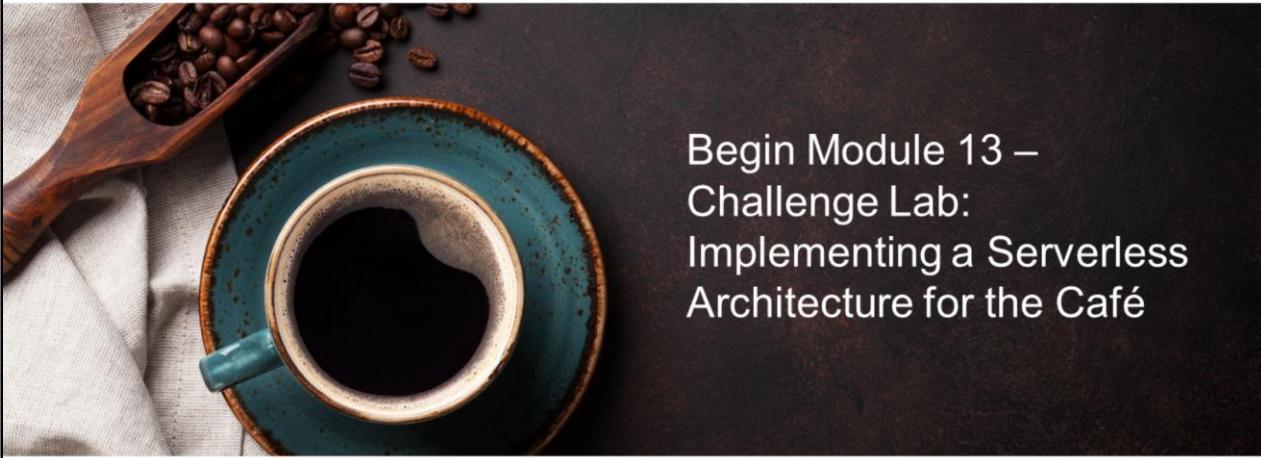
1. Downloading the source code
2. Creating the *DataExtractor* Lambda function in the VPC
3. Creating the *salesAnalysisReport* Lambda function
4. Creating an SNS topic
5. Creating an email subscription to the SNS topic
6. Testing the *salesAnalysisReport* Lambda function
7. Setting up an Amazon EventBridge event to trigger the Lambda function each day

Challenge lab: Final product



The diagram summarizes what you will have built after you complete the lab.

For accessibility: Diagram of three-tier architecture with cafe application server in public subnet 1, sales analysis report data extractor Lambda function in private subnets 1 and 2, and primary DB instance in DB subnet group that spans private subnets 3 and 4. Daily event triggers the sales analysis report Lambda function, which triggers data extractor Lambda function, and also an email notification to be sent from SNS topic. **End of accessibility description.**



A timer icon with the text "~ 90 minutes" indicating the duration of the challenge lab.

**Begin Module 13 –
Challenge Lab:
Implementing a Serverless
Architecture for the Café**

aws

© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

78

It is now time to start the challenge lab.

Challenge lab debrief: Key takeaways



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

79

Your educator might choose to lead a conversation about the key takeaways from this challenge lab after you have completed it.

Module wrap-up

Module 13: Building Microservices and Serverless Architectures



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

It's now time to review the module and wrap up with a knowledge check and discussion of a practice certification exam question.

Module summary

In summary, in this module, you learned how to:

- Indicate the characteristics of microservices
- Refactor a monolithic application into microservices and use Amazon ECS to deploy the containerized microservices
- Explain serverless architecture
- Implement a serverless architecture with AWS Lambda
- Describe a common architecture for Amazon API Gateway
- Describe the types of workflows that AWS Step Functions supports



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

81

In summary, in this module, you learned how to:

- Indicate the characteristics of microservices
- Refactor a monolithic application into microservices and use Amazon ECS to deploy the containerized microservices
- Explain serverless architecture
- Implement a serverless architecture with AWS Lambda
- Describe a common architecture for Amazon API Gateway
- Describe the types of workflows that AWS Step Functions supports

Complete the knowledge check



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

82

It is now time to complete the knowledge check for this module.



Sample exam question

An organization hosts 10 microservices, each in an Auto Scaling group behind individual Classic Load Balancers. Each EC2 instance is running at optimal load.

Which of the following actions would enable the organization to reduce costs without impacting performance?

Choice Response

- | | |
|---|--|
| A | Reduce the number of EC2 instances behind each Classic Load Balancer. |
| B | Change instance types in the Auto Scaling group launch configuration. |
| C | Change the maximum size but leave the desired capacity of the Auto Scaling groups. |
| D | Replace the Classic Load Balancers with a single Application Load Balancer. |

© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

83

Look at the answer choices and rule them out based on the keywords.

Sample exam question answer



An organization hosts 10 microservices, each in an Auto Scaling group behind individual Classic Load Balancers. Each EC2 instance is running at optimal load.

Which of the following actions would enable the organization to reduce costs without impacting performance?

The correct answer is D.

The keywords in the question are reduce costs without impacting performance and replace the Classic Load Balancers with a single Application Load Balancer.

© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

84

The following are the keywords to recognize: reduce costs without impacting performance and replace the Classic Load Balancers with a single Application Load Balancer.

The correct answer is D: “Replace the Classic Load Balancers with a single Application Load Balancer.” By process of elimination, choice D is correct. You can also use a single Application Load Balancer to route requests to all the services for your application instead of using a single Classic Load Balancer for each microservice.

Incorrect answers:

- Choice A can be eliminated—Because the EC2 instances are running at optimal load, they will become overloaded if you reduce their number.
- Choice B can be eliminated—You will not reduce costs if you choose a larger instance size. Alternatively, if you decrease the instance size, the EC2 instances will become overloaded because they are already running at optimal load.
- Choice C can also be eliminated—Costs will remain the same or increase if you increase the maximum size and keep the desired capacity.

Additional resources

- [Break a Monolith Application into Microservices](#) project
- [Serverless Architectures with AWS Lambda](#) whitepaper
- [AWS Serverless Multi-Tier Architectures with Amazon API Gateway and AWS Lambda](#) whitepaper
- [AWS Well-Architected Framework: Serverless Application Lens](#) whitepaper
- [Creating and Using Lambda Functions](#) tutorial



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

85

If you want to learn more about the topics covered in this module, you might find the following additional resources helpful:

- [Break a Monolith Application into Microservices](#) project
- [Serverless Architectures with AWS Lambda](#) whitepaper
- [AWS Serverless Multi-Tier Architectures with Amazon API Gateway and AWS Lambda](#) whitepaper
- [AWS Well-Architected Framework: Serverless Application Lens](#) whitepaper
- [Creating and Using Lambda Functions](#) tutorial



Thank you

Corrections, feedback, or other questions?
Contact us at <https://support.aws.amazon.com/#/contacts/aws-academy>.
All trademarks are the property of their owners.

© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

86

Thank you for completing this module.



**AWS Academy Cloud Architecting
Module 14 Student Guide
Version 2.0.8**

200-ACACAD-20-EN-SG

© 2022 Amazon Web Services, Inc. or its affiliates. All rights reserved.

This work may not be reproduced or redistributed, in whole or in part, without prior written permission from Amazon Web Services, Inc. Commercial copying, lending, or selling is prohibited.

All trademarks are the property of their owners.

Contents

Module 14: Planning for Disaster	4
----------------------------------	---



Module 14: Planning for Disaster

AWS Academy Cloud Architecting

© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

Welcome to Module 14: Planning for Disaster.

Module overview

Sections

1. Architectural need
2. Disaster planning strategies
3. Disaster recovery patterns

Lab

- Guided Lab: Hybrid Storage and Data Migration with AWS Storage Gateway File Gateway



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

2

This module includes the following sections:

1. Architectural need
2. Disaster planning strategies
3. Disaster recovery patterns

The module also includes a guided lab, where you will enable Amazon S3 cross-Region replication. You will configure a file gateway and mount the file share on an Amazon Elastic Compute Cloud (Amazon EC2) instance.

Finally, you will be asked to complete a knowledge check to test your understanding of key concepts that are covered in this module.

Module objectives

At the end of this module, you should be able to:

- Identify strategies for disaster planning
- Define recovery point objective (RPO) and recovery time objective (RTO)
- Describe four common patterns for backup and disaster recovery and how to implement them
- Use AWS Storage Gateway for on-premises-to-cloud backup solutions



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

3

At the end of this module, you should be able to:

- Identify strategies for disaster planning
- Define recovery point objective (RPO) and Recovery Time Objective (RTO)
- Describe four common patterns for backup and disaster recovery and how to implement them
- Use AWS Storage Gateway for on-premises-to-cloud backup solutions

Section 1: Architectural need

Module 14: Planning for Disaster



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

Introducing Section 1: Architectural need.

Café business requirement

If the café's infrastructure ever becomes unavailable, the staff must be able to get their applications running again within an amount of time that is acceptable to the business. They need an architecture that supports their disaster recovery plans while also optimizing for cost.



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

5

By now, the café has implemented several applications that run on AWS. They are also storing a significant amount of business-critical data in the AWS Cloud. Sofía realizes that if the café's infrastructure ever becomes unavailable, they must be able to get their applications running and accessible in an amount of time that's acceptable to the business. Currently, the café's staff hasn't developed any comprehensive disaster recovery plans.

Sofía raised this concern with Frank and Martha. They all agreed that it's important to put backup and disaster recovery plans into place. Their objective is to implement an architecture that supports their disaster recovery time objectives, while it also optimizes for cost. They also agreed that as their revenue grows, they will be able to afford a solution that supports a shorter recovery time objective.

In this module, you will learn about key AWS service features that support data backup and disaster recovery. With an understanding of these features, you should be able to help the café meet this essential business requirement.

Section 2: Disaster planning strategies

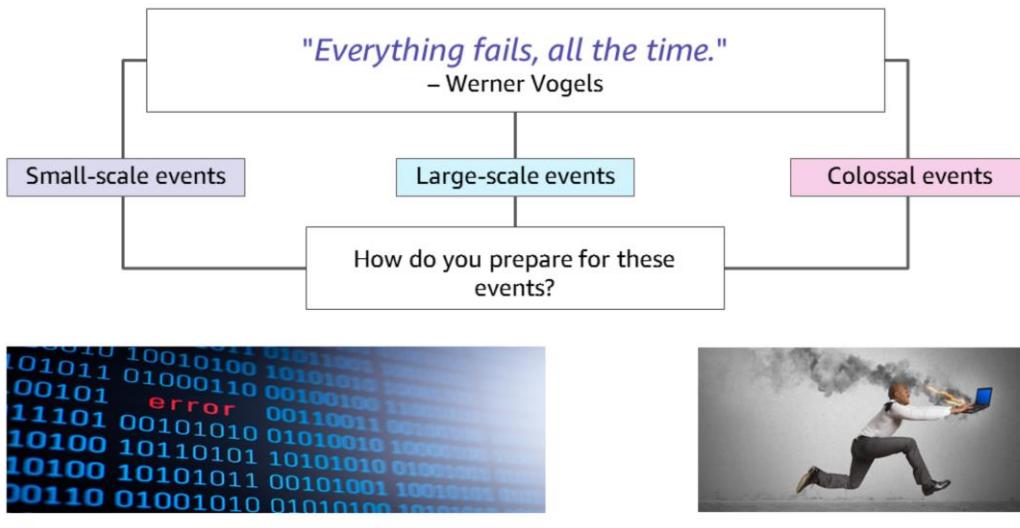
Module 14: Planning for Disaster



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

Introducing Section 2: Disaster planning strategies.

Planning for failures



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

7

The Chief Technology Officer (CTO) of AWS, Werner Vogels, has famously stated on more than one occasion that, "Everything fails, all the time." His pronouncement has continued to influence cloud computing architectural design for many years, because it speaks to a truism.

Failure should not be thought of as an unlikely aberration. Instead, it should be assumed that failures, both large and small, can—and will—occur. How do you prepare for these events?

A failure can be categorized as one of three types:

- A *small-scale event* – For example, a single server stopped responding or went offline
- A *large-scale event* – In this case, multiple resources were affected, perhaps even across Availability Zones within a Region
- A *colossal scale event* – In this case, the failure is widespread, and it affects a large number of users and systems

To minimize the impact of a disaster, organizations must invest time and resources to plan and prepare, to train employees, and to document and update processes. The amount of investment for disaster planning for a particular system can vary dramatically, depending on the cost of a potential outage.

Avoiding and planning for disaster

High availability

- Minimize how often your applications and data become unavailable

Backup

- Make sure that your data is safe in case of disaster

Disaster recovery (DR)

- Recover your data and get your applications back online after a disaster



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

8

You can work to avoid and plan for disaster in three ways:

- High availability* provides redundancy and fault tolerance. A system is highly available when it can withstand failure of an individual or multiple components (for example, hard disks, servers, or network connectivity). Production systems typically have defined uptime requirements.
- Backup* is critical to protecting data and ensuring business continuity. However, it can be a challenge to implement. The pace at which data is generated is growing exponentially. Meanwhile, the density and durability of local disks are not experiencing the same growth rate. Even so, it is essential to keep your critical data backed up, in case of disaster.
- Disaster recovery (DR)* is about preparing for and recovering from a disaster. A *disaster* is any event that has a negative impact on a company's business continuity or finances. Such events include hardware or software failure, a network outage, a power outage, or physical damage to a building (like fire or flooding). The cause can be human error, or some other significant event. Disaster recovery is a set of policies and procedures that enable the recovery or continuation of vital technology infrastructure and systems after any disaster.

Selected AWS Well-Architected Framework design principles

Operational Excellence pillar

- Anticipate failure
- Refine operational procedures frequently

Reliability pillar

- Test recovery procedures
- Automatically recover from failure



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

9

Consider some design principles that relate to the topic of disaster recovery.

The *Operational Excellence* pillar of the AWS Well-Architected Framework states the importance of *anticipating failure*. It recommends that you perform pre-mortem exercises to identify potential sources of failure so that they can be removed or mitigated. You must test your failure scenarios and validate your understanding of their impact. The AWS Well-Architected Framework also describes the benefits of refining your operational procedures frequently so that you can look for opportunities to improve them. Then, as you evolve your workload, you can evolve your procedures accordingly.

The *Reliability* pillar describes the importance of designing your systems. You must be able to recover from infrastructure or service disruptions, and mitigate disruptions such as misconfigurations or transient network issues.

One of the design principles that it mentions is to *test recovery procedures*. Test how your system fails, and validate your recovery procedures. You can use automation to simulate different failures or to recreate scenarios that led to previous failures. This testing exposes failure pathways that you can test and fix before a real failure scenario. It reduces the risk of components that have not been tested before they fail.

Another principle of design is to *automatically recover from failure*. By monitoring a system for

key performance indicators (KPIs), you can trigger automation when a threshold is breached. These KPIs should be a measure of business value, not the technical aspects of how the service operates. Your automation could provide notifications and tracking of failures, and for automated recovery processes that work around or repair the failure.

Recovery point objective (RPO)

Recovery point objective (RPO) is the maximum acceptable amount of data loss, measured in time.

How often must your data be backed up?

Example RPO: *The business can recover from losing (at most) the last 8 hours of data.*



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

10

Organizations of all sizes, large and small, often have a Business Continuity Plan (BCP). A typical part of the BCP is to provide for IT Service Continuity, including IT disaster recovery planning.

One of the most important measures of a disaster recovery plan is to define your *recovery point objective (RPO)*. To calculate RPO, first determine how much data loss is acceptable, according to your BCP. Then, figure out how quickly that data loss might occur, as a time measurement.

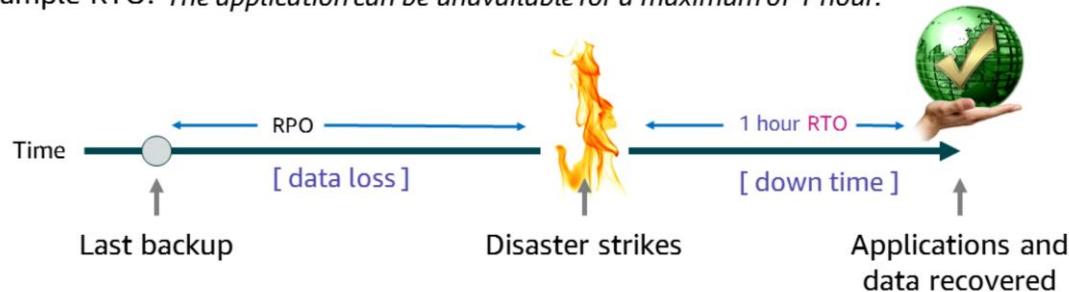
For example, suppose you determine that the data that your application generates is important but not critical, so that losing 800 records would be acceptable. You further calculate that even during peak times, no more than 100 records are created in an hour. In this scenario, you decide that an RPO of 8 hours is sufficient to meet your needs. If you then implement a disaster recovery plan that meets this RPO, you are sure to do data backups at least every 8 hours. Then, if a disaster occurs at 22:00, the system should be able to recover all data that was in the system before 14:00 PM.

Recovery time objective (RTO)

Recovery time objective (RTO) is the maximum acceptable amount of time after disaster strikes that a business process can remain out of commission.

How quickly must your applications and data be recovered?

Example RTO: *The application can be unavailable for a maximum of 1 hour.*



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

11

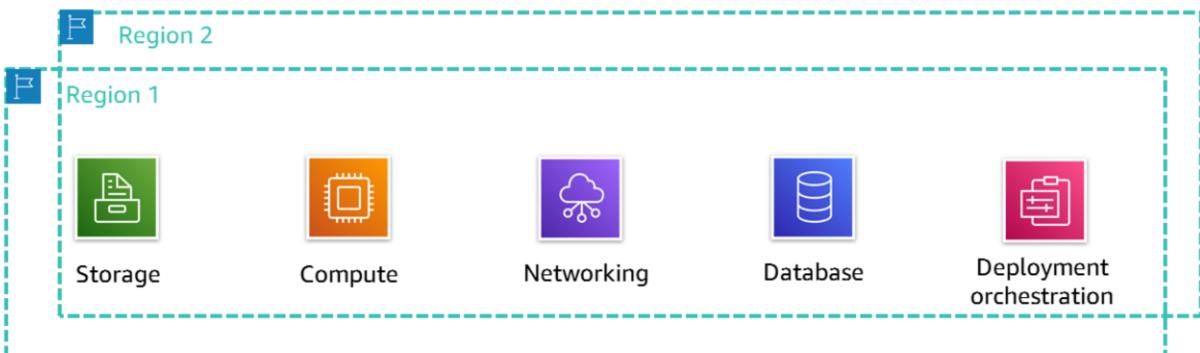
Another important measure of a disaster recovery plan is to define the *recovery time objective* (RTO). RTO is the time that it takes *after* a disruption to restore your applications and recover your data. To continue the previous example, suppose a disaster occurs at 22:00 and the RTO is 1 hour. In that scenario, the DR process should restore the business process to the acceptable service level by 23:00.

A company typically decides on acceptable RPO and RTO, and it bases its decision on the financial impact to the business when systems are unavailable. The company determines financial impact by considering many factors. These factors include loss of business and damage to its reputation because of downtime and the lack of systems availability.

IT organizations then plan solutions to provide cost-effective system recovery. The solutions are based on the RPO within the timeline and the service level that the RTO establishes.

Plan for disaster recovery

Be intentional about where your data is stored and where your applications run.



The most robust DR plans span more than one Region.



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

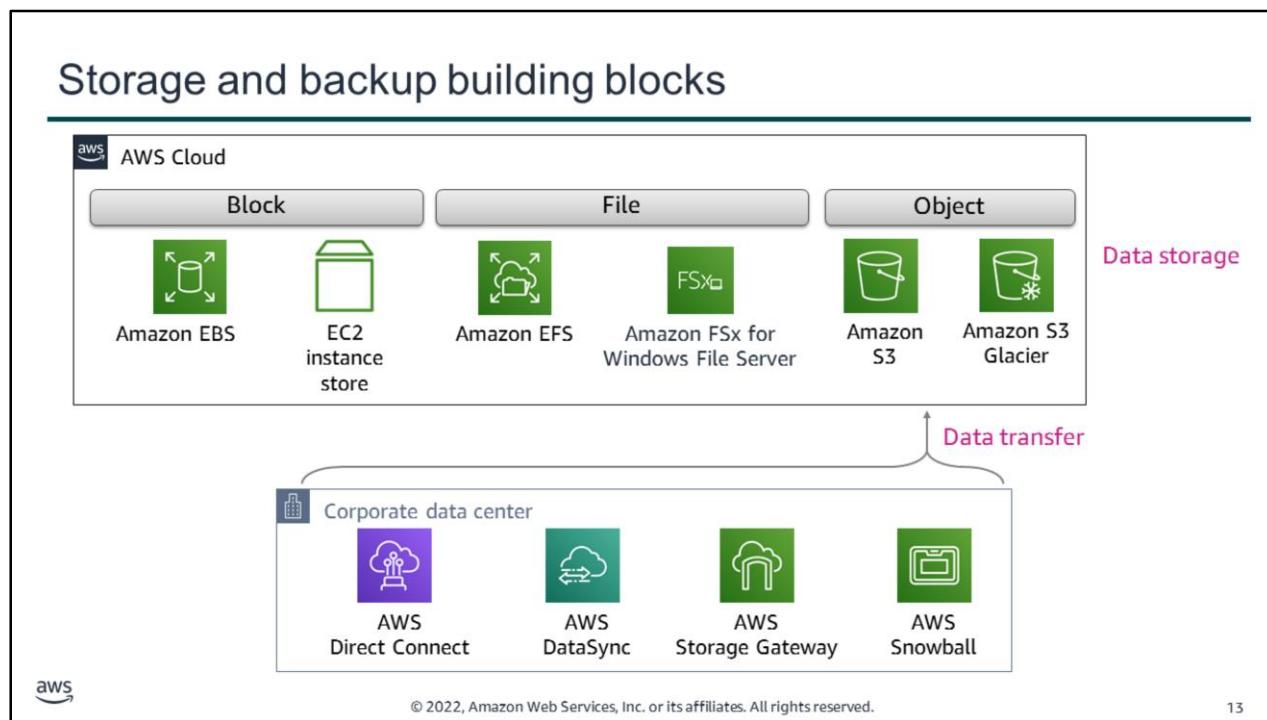
12

To properly scope your disaster recovery planning, you must look holistically at your use of AWS. Most organizations use a combination of services that can be broadly categorized as encompassing these five service categories:

- Storage
- Compute
- Networking
- Databases
- Deployment orchestration services

If a disaster occurs, your RPO and RTO will guide your backup-and-restore plans and procedures across each of these service areas. They will also likely affect your production deployment architecture.

It is also important to keep in mind that, although it's unlikely for a Region to be unavailable, it is within the realm of possibility. If some large-scale event affects a Region—for instance, a meteor strike—would your data still be available? Would your applications still be accessible? AWS provides multiple Regions around the world. Thus, you can choose the most appropriate location for your disaster recovery site, in addition to the site where your system is fully deployed.



The following services are referenced in the diagram:

- Amazon Elastic Block Store (Amazon EBS)
- Amazon Elastic Compute Cloud (Amazon EC2)
- Amazon Elastic File System (Amazon EFS)
- Amazon Simple Storage Service (Amazon S3)
- Amazon Simple Storage Service Glacier (Amazon S3 Glacier)

To start your disaster planning in detail, look at the data storage layer (postponing the discussion of database layer for the moment).

Your AWS Cloud storage can consist of a combination of block storage, file system storage, and object storage. Meanwhile, your organization might also use AWS services that connect the on-premises data center to the AWS Cloud.

In the next few slides, you will learn about high-level best practices for each of these three areas.

One service that you might not be familiar with is *AWS DataSync*. AWS DataSync provides movement of large amounts of data online between on-premises storage and Amazon S3, Amazon EFS, or Amazon FSx for Windows File Server. It supports scripted copy jobs and scheduled data transfers from on-premises Network File Systems (NFS) and Server Message Block (SMB) storage. It can also optionally use AWS Direct Connect links.

Best practice: S3 Cross-Region Replication

The diagram shows the AWS Cloud interface with various storage options under the 'Object' tab: Amazon EBS, EC2 instance store, Amazon EFS, Amazon FSx for Windows File Server, Amazon S3, and Amazon S3 Glacier. Below this, a dotted arrow points from a 'Source S3 bucket (replication configured)' in Region A to a 'Destination S3 bucket' in Region B, illustrating the process of cross-Region replication.

- Most S3 storage classes replicate data across Availability Zones within a single Region
- Configure S3 cross-Region replication for higher-level data security
 - Automatically, asynchronously replicates objects created after you add the replication configuration
 - Can also help meet compliance requirements and reduce latency for users who are accessing objects

© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved. 14

For many organizations, the bulk of their data that is stored on AWS is in Amazon S3, which provides object storage.

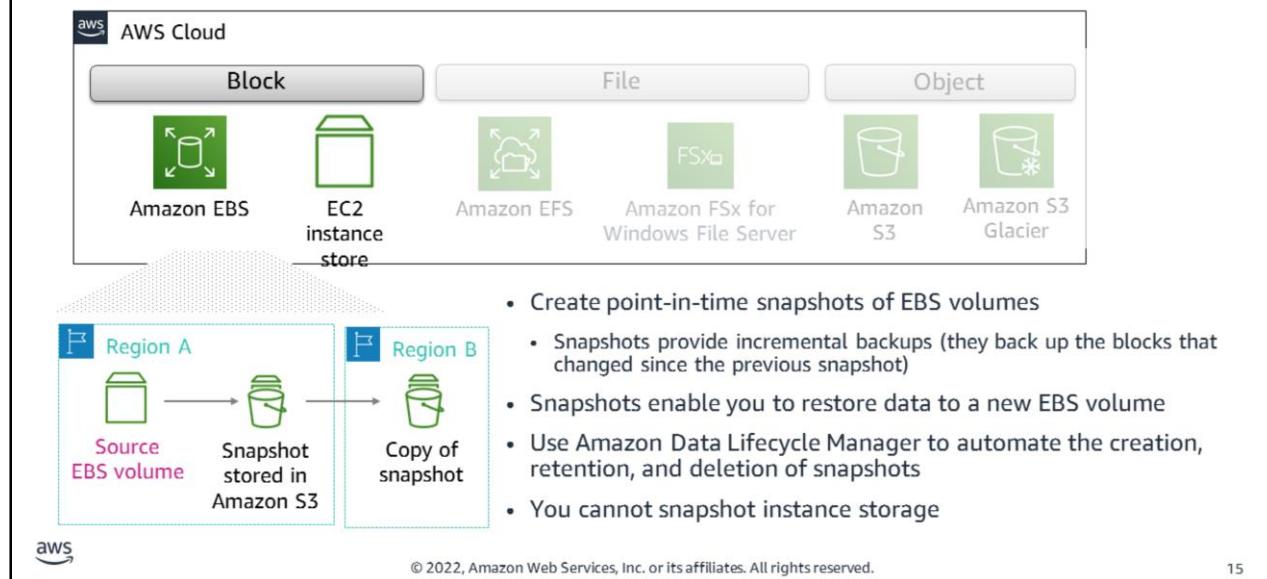
Recall that S3 buckets exist in a specific AWS Region. You choose the Region when you create the bucket. Amazon S3 provides 11 9s (99.99999999 percent) of durability for S3 Standard, S3 Standard-IA, S3 One Zone-IA, and Amazon S3 Glacier storage classes. Amazon S3 Standard, S3 Standard-IA, and Amazon S3 Glacier are all designed to sustain data if an entire Amazon S3 Availability Zone loss occurs. They provide this stability by automatically storing your objects across a minimum of three Availability Zones, each separated miles apart, across a *single AWS Region*.

For critical applications and data scenarios where you want a higher level of data security, it is a best practice to configure S3 cross-Region replication. To enable the replication, you add a replication configuration to your *source bucket*. The minimum configuration must indicate the destination bucket where you want Amazon S3 to replicate all objects, or a subset of all objects. It must also include an AWS Identity and Access Management (IAM) role that grants Amazon S3 permissions to copy the objects to the destination bucket.

Copied objects retain their metadata. The *destination bucket* can belong to another storage class. For example, the contents of an S3 Standard bucket might be replicated to an Amazon S3 Glacier bucket. You can assign different ownership to the objects in the destination bucket. You can also

use S3 Replication Time Control (S3 RTC) to replicate your data across different Regions in a predictable time frame. S3 RTC replicates four 9s (99.99 percent) of new objects stored in Amazon S3 within 15 minutes (backed by a service-level agreement).

Best practice: EBS volume snapshots



Regarding to block storage, you can back up the data that is on EBS volumes to Amazon S3 by taking point-in-time *snapshots*. Snapshots are *incremental* backups, which means that it saves only the blocks on the device that have changed since your most recent snapshot. This architecture minimizes the time that is required to create the snapshot, and it saves on storage costs by not duplicating data.

Each snapshot contains all the information that is needed to restore your data (from the moment when the snapshot was taken) to a new EBS volume. When you create an EBS volume that is based on a snapshot, the new volume begins as an exact replica of the original volume. This original volume was used to create the snapshot. The replicated volume loads data in the background so that you can begin to use it immediately. If you access data that has not been loaded yet, the volume immediately downloads the requested data from Amazon S3. Then, it continues to load the rest of the volume's data in the background.

Amazon EBS volumes provide off-instance storage that persists independently from the life of an instance, and is replicated across multiple servers in an Availability Zone. Volumes prevent the loss of data from the failure of any single component. After you create a snapshot, it finishes copying to Amazon S3 (when the snapshot status is completed). Then, you can copy it from one AWS Region to another, or within the same Region.

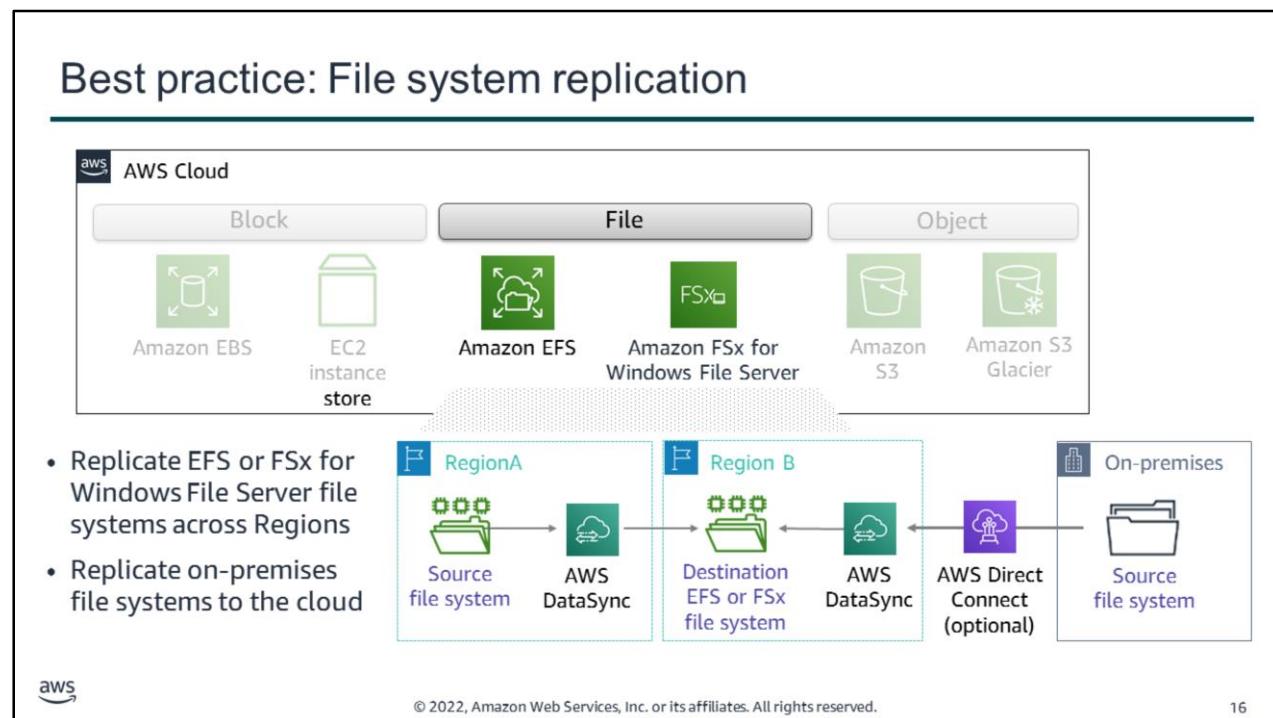
You can *use Amazon Data Lifecycle Manager to automate the creation, retention, and deletion of*

snapshots that back up your EBS volumes. Automating snapshot management helps you to:

- Protect valuable data by enforcing a regular backup schedule
- Retain backups as required by auditors or internal compliance
- Reduce storage costs by deleting outdated backups

You cannot create snapshots of EC2 instance store volumes. However, if you must back up data from an instance store, you can create a new EBS volume and format it. Then, mount the new volume to the EC2 instance guest OS and copy the data on your instance store volume to the EBS volume.

Recall that *instance store* volumes provide temporary block-level storage that works well for information that changes frequently, such as buffers, caches, and scratch data. You might find that you must back up data from an instance store. If so, you might want to rethink why you are storing that data on an instance store volume in the first place.



It is also a best practice to replicate your file storage.

AWS *DataSync* makes data move faster between two EFS or Amazon FSx Windows File Server file systems, or between on-premises storage and AWS file storage. You can use DataSync to transfer datasets over DX or the internet. Use the service for one-time data migrations or ongoing workflows for data protection and recovery.

You can learn more about how to use AWS *Backup* to manage EBS volume backups and to automate backups of EFS file systems. See the [Scheduling automated backups using Amazon EFS and AWS Backup](#) blog for details.

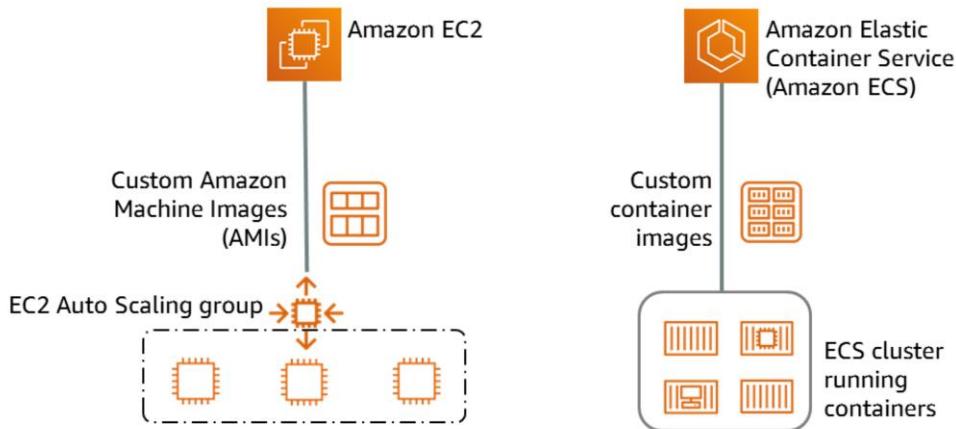
FSx for Windows File Server takes daily automatic backups of your file systems, and it enables you to take more backups at any point. Amazon FSx stores the backups in Amazon S3. The daily backup window is a 30-minute window that you specify when you create a file system. The daily backup retention period that is specified for your file system determines the number of days that your daily automatic backups are kept. (This number is 7 days by default.)

Like most Amazon S3 storage classes replicate data across Availability Zones, so do Amazon EFS and FSx for Windows File Server file systems. Your disaster recovery requirements might specify that you need a multi-region recovery solution. In that case, it is a best practice to replicate your Amazon EFS and FSx for Windows File Server file systems to a second Region. You can use AWS

DataSync to get this replication. To simplify file transfer between two EFS file systems by using DataSync, you can use the [AWS DataSync In-Cloud QuickStart and Scheduler](#).

Compute capacity should be quickly recoverable

Obtain and boot new server instances or containers within minutes.



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

17

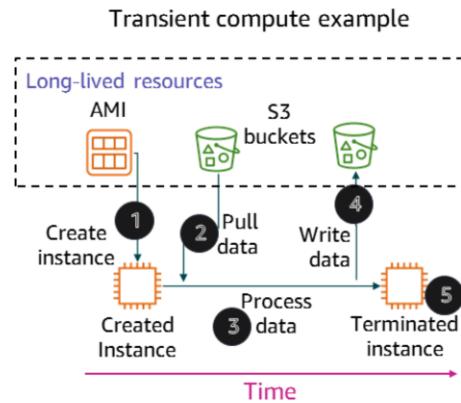
In the context of DR, it's critical that you can rapidly create virtual machines that you control. By launching instances in separate Availability Zones, you can protect your applications from the failure of a single location.

You can arrange for automatic recovery of an EC2 instance when a system status check of the underlying hardware fails. The instance is rebooted (on new hardware, if necessary)—but it retains its instance ID, IP addresses, EBS volume attachments, and other configuration details. For a complete recovery, make sure that the instance is configured to automatically start up any services or applications as part of its initialization process.

Amazon Machine Images (AMIs) are preconfigured with operating systems, and some preconfigured AMIs might also include application stacks. You can also configure your own custom AMIs. In the context of DR, AWS recommends that you configure and identify your own AMIs so that they launch as part of your recovery procedure. Such AMIs should be preconfigured with your operating system of choice, in addition to the appropriate pieces of the application stack.

Strategies for compute disaster recovery

- Use the Amazon EC2 [snapshot](#) capability for backups
 - Snapshots can be performed manually, or scheduled (for example, by using AWS Lambda)
- Use system or instance level system backups infrequently and as a last resort
 - Drives up the cost of storage that is used quickly
 - [Prefer automated rebuild](#) from configuration or code repositories instead
- Cross-region AMI copies
- Cross-region snapshot copies
- Consider [transient](#) compute architectures
 - Store essential data off the instance



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

18

For disaster recovery of compute resources, you will probably want to use the Amazon EC2 snapshot capability. Snapshots can be performed manually, or they can be scheduled.

Although you can create system or instance-level system backups, extensive use of this approach increases your storage costs. A better approach is to configure an automated rebuild process, where your source code is stored in a repository.

You might want to replicate Amazon S3 across Regions, and you probably also want to replicate your most critical AMIs and snapshots across Regions.

Finally, consider architecting your use of compute resources to store essential data off of the instances. As you see in the example, your data can be stored in an S3 bucket. When you must do data processing, you can launch one or more EC2 instances from a custom AMI that is preconfigured with application software. As soon as the instance is started, it can pull the needed data from the S3 bucket and process the data. Then, it can write the output data back to Amazon S3 (perhaps to another S3 bucket). After the instance completes its compute tasks, the instance can be terminated. Such an architecture—when it can still meet your business needs—makes it easier to design your disaster recovery strategy. It also can save on costs, because servers that are not in constant use can be terminated and then later re-created when needed.

Networking: Design for resilience, recovery



Amazon Route 53

- Traffic distribution
- Failover



Elastic Load Balancing

- Load balancing
- Health checks and failover



Amazon Virtual Private Cloud (Amazon VPC)

Extend your existing on-premises network topology to the cloud



AWS Direct Connect

Fast, consistent replication and backups of large on-premises environments to the cloud



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

19

When you work to recover from a disaster, it's likely that you must modify network settings to fail your system over to another site. AWS offers several services and features that enable you to manage and modify network settings, a few of which are highlighted next.

Amazon Route 53 provides load balancing and network routing capabilities that enable you to distribute network traffic. It also provides the ability to fail over between multiple endpoints and even to a static website that is hosted in Amazon S3.

The *Elastic Load Balancing* service automatically distributes incoming application traffic across multiple EC2 instances. It enables you to achieve fault tolerance in your applications by providing the load-balancing capacity that is needed in response to incoming application traffic. You can pre-allocate a load balancer so that its Domain Name System (DNS) name is already known, which can simplify implementation of your DR plan.

You can use *Amazon Virtual Private Cloud (Amazon VPC)* to extend an existing on-premises network topology to the cloud. This extension can be especially appropriate when you recover enterprise applications that might be hosted on an internal network.

Finally, *AWS Direct Connect* simplifies the setup of a dedicated network connection from an on-premises data center to AWS. Using DX can reduce network costs, increase bandwidth throughput, and provide a more consistent network experience than internet-based connections.

Databases: Features that support recovery



Amazon Relational Database Service (Amazon RDS)

- Take snapshot data and save it in a separate Region
- Combine read replicas with Multi-AZ deployments to build a resilient disaster recovery strategy
- Retain automated backups



Amazon DynamoDB

- Back up entire tables in seconds
- Use point-in-time-recovery to continuously back up tables for up to 35 days
- Initiate backups with a single click in the console or a single application programming interface (API) call
- Use Global Tables to build a multi-region, multi-master database that provides fast local performance for massively scaled globally distributed applications



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

20

AWS provides many database services. Some key features of Amazon RDS and Amazon DynamoDB that are relevant to disaster recovery scenarios are explained next.

Consider using *Amazon RDS* in the DR *preparation phase* to store a copy of your critical data in a database that is already running. Then, use Amazon RDS in the DR *recovery phase* to run your production database.

If you implement a multi-region DR plan, Amazon RDS gives you the ability to store snapshot data that was captured from one Region to another Region. You can share a manual snapshot with up to 20 other AWS accounts.

Combining read replicas with Multi-AZ deployments enables you to build a resilient disaster recovery strategy and simplify your database engine upgrade process. By using Amazon RDS read replicas, you can create one or more read-only copies of your database instance. You can create these copies within the same AWS Region, or in a different AWS Region. Updates to the source database are then asynchronously copied to your read replicas. Read replicas can be promoted to become a standalone database instance, when needed.

Use *Amazon DynamoDB* in the preparation phase to copy data to DynamoDB in another Region or to Amazon S3. During the recovery phase of DR, you can scale up in minutes. DynamoDB global tables replicate your DynamoDB tables automatically across your choice of AWS Regions.

They resolve update conflicts and enable your applications to stay highly available, even in the unlikely event that an entire Region is isolated or affected by degradation.

Automation services: Quickly replicate or redeploy environments



AWS CloudFormation

- Use templates to quickly deploy collections of resources as needed
- Duplicate production environments in new Region or VPC in minutes



AWS Elastic Beanstalk

- Quickly redeploy your entire stack in only a few clicks



AWS OpsWorks

- Automatic host replacement
- Combine it with AWS CloudFormation in the recovery phase
- Provision a new stack that supports the defined RTO



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

21

When you use automation services, you can quickly replicate or redeploy environments.

AWS CloudFormation enables you to model and deploy your entire infrastructure in a text file. This template can become the single source of truth for your infrastructure. When you use AWS CloudFormation to manage your entire infrastructure, it also becomes a powerful tool in your disaster recovery planning toolkit. It enables you to duplicate complex production environments in minutes, for example, to a new Region or a new VPC.

AWS CloudFormation provisions your resources in a repeatable manner, which enables you to build and rebuild your infrastructure and applications. You are not required to perform manual actions or write custom scripts.

If you use *AWS Elastic Beanstalk* to host your applications, you can upload an updated application source bundle and deploy it to your AWS Elastic Beanstalk environment. Alternatively, you can redeploy a previously uploaded version of an application. You can also deploy a previously uploaded version of your application to any of its environments.

Finally, *AWS OpsWorks* is an application management service that makes it easy to deploy and operate applications of all types and sizes. You can define your environment as a series of layers, and configure each layer as a tier of your application. AWS OpsWorks has automatic host replacement, so if you have an instance failure, it is automatically replaced. You can use AWS

OpsWorks in the DR *preparation phase* to template your environment and combine it with AWS CloudFormation in the DR recovery phase.

Section 2 key takeaways



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

22

- To choose the correct *disaster recovery* strategy, first identify your recovery point objective (RPO) and recovery time objective (RTO)
- Use features such as *S3 Cross-Region Replication*, *EBS volume snapshots*, and *Amazon RDS snapshots* to protect data
- Use networking features (such as *Route 53 failover* and *Elastic Load Balancing*) to improve application availability
- Use automation services (such as *AWS CloudFormation*) as part of your DR strategy to *quickly deploy duplicate environments* when needed

Some key takeaways from this section of the module include:

- To choose the correct *disaster recovery* strategy, first identify your recovery point objective (RPO) and recovery time objective (RTO)
- Use features such as *S3 Cross-Region Replication*, *EBS volume snapshots*, and *RDS snapshots* to protect data
- Use networking features—such as *Route 53 failover* and *Elastic Load Balancing*—to improve application availability
- Use automation services—such as *AWS CloudFormation*—as part of your DR strategy to *quickly deploy duplicate environments* when necessary

Section 3: Disaster recovery patterns

Module 14: Planning for Disaster



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

Introducing Section 3: Disaster recovery patterns.

Common disaster recovery patterns on AWS

Four disaster recovery patterns

- Backup and restore
- Pilot light
- Warm standby
- Multi-site



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

24

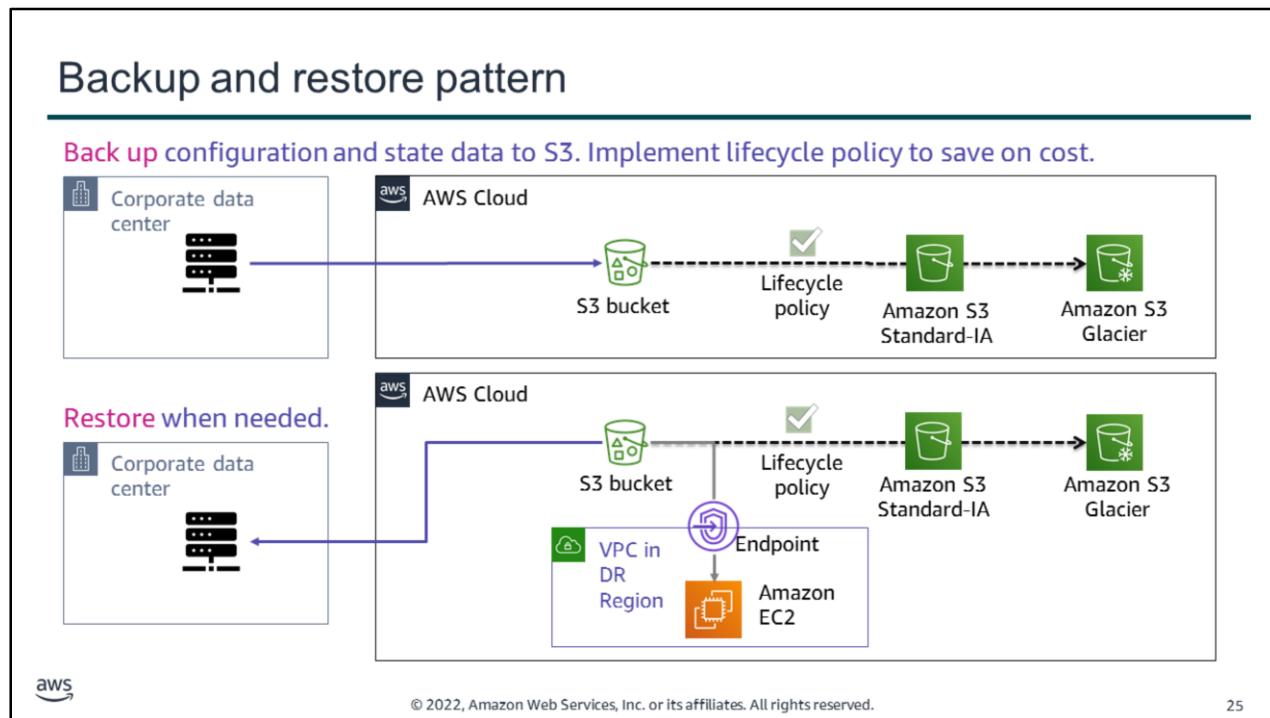
Each pattern is suited to a different combination of:

- Recovery point objective
- Recovery time objective
- Cost-effectiveness

Organizations often use these four common disaster recovery patterns:

- Backup and restore
- Pilot light
- Warm standby
- Multi-site

As you will discover in the details that follow, each pattern is well-suited to different requirements. Some of the patterns offer better cost-effectiveness. Others provide a faster RPO and faster RTO, but cost more to maintain.



The first disaster recovery approach is the *backup and restore pattern*.

In most traditional environments, data is backed up to tape and sent offsite regularly. If you use this method, it can take a long time to restore your system when a disaster occurs.

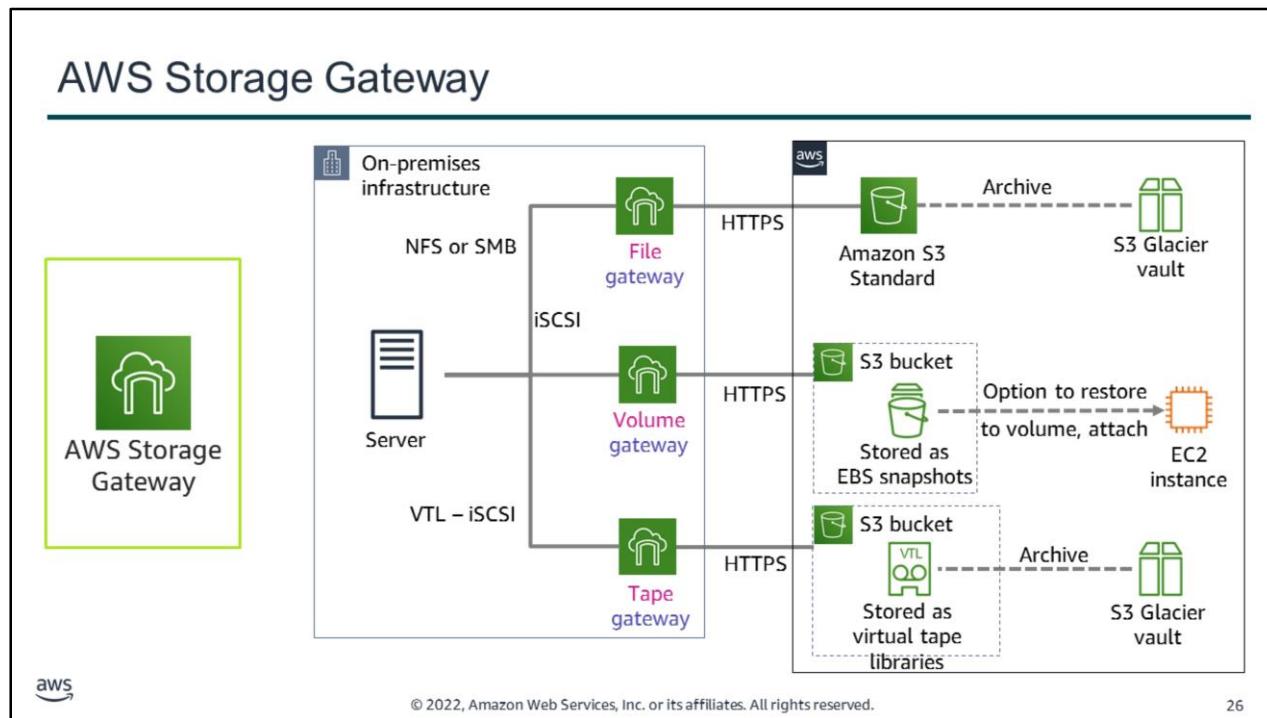
Amazon S3 provides a more easily accessible destination for backup data that might be needed quickly to perform a restore. Transferring data to and from Amazon S3 is typically done through the network, and is therefore accessible from any location.

In the example backup scenario, data is copied from the on-premises data center to Amazon S3. AWS DataSync or Amazon S3 Transfer Acceleration can optionally be used as part of this configuration to automate or increase the speed of data transfer. Then, an S3 lifecycle configuration that is applied to the bucket later moves the backup data to less-expensive Amazon S3 storage classes. The backup data moves to Amazon S3 Glacier or Amazon S3 Standard-IA, which saves on cost as the data ages and is not frequently accessed.

In the example restore scenario, the on-premises data might be temporarily or permanently lost. Then, the backup data can be downloaded from Amazon S3 back to the on-premises servers.

If your corporate data center remains offline, you can further ensure the ability to restore your data to your servers. You can have Amazon EC2 servers that are ready to go in a VPC in your

designated disaster recovery Region. This Region can connect to the S3 bucket that contains your backup application data. It can read that data, and perhaps temporarily host your applications while you work to restore your data center.



As part of the backup and restore pattern, you might find that it makes sense to use AWS Storage Gateway.

AWS Storage Gateway is a hybrid storage service that enables your on-premises applications to use AWS Cloud storage. You can use the service for backup and archiving, disaster recovery, cloud data processing, storage tiering, and migration.

Your applications connect to the service through a virtual machine or hardware gateway appliance by using standard storage protocols. These protocols include NFS, SMB, Virtual Tape Library (VTL), and Internet Small Computer System Interface (iSCSI). The gateway connects to AWS storage services—such as Amazon S3, Amazon S3 Glacier, and Amazon EBS—which provide storage for files, volumes, and virtual tapes. The service includes an optimized data transfer mechanism. It provides bandwidth management, automated network resilience, and efficient data transfer, in addition to a local cache for low-latency on-premises access to your most active data.

With a *file gateway*, you store and retrieve objects (by using the NFS or SMB protocol) in Amazon S3. You use a local cache for low-latency access to your most recently used data. When your files are transferred to Amazon S3, they are stored as objects and can be accessed through an NFS mount point.

The Storage Gateway *volume* interface presents your applications with block storage disk volumes that can be accessed by using the iSCSI protocol. Data on these volumes is backed up as point-in-time EBS snapshots, which enables you to access it through Amazon EC2, if needed.

The Storage Gateway *tape* interface presents the Storage Gateway to your existing backup application as a virtual tape library. This library consists of a virtual media changer and virtual tape drives. You can continue to use your existing backup applications while you write to a collection of virtual tapes. Each virtual tape is stored in Amazon S3. When you no longer require access to data on virtual tapes, your backup application archives it from the virtual tape library into Amazon S3 Glacier.

Backup and restore: Checklist

Preparation phase

- Create backups of current systems
- Store backups in Amazon S3
- Document procedure to restore from backups
- Know:
 - Which AMI to use, and build as needed
 - How to restore system from backups
 - How to route traffic to the new system
 - How to configure the deployment

In case of disaster

- Retrieve backups from Amazon S3
- Restore required infrastructure
 - EC2 instances from prepared AMIs
 - Elastic Load Balancing load balancers
 - AWS resources created by an AWS CloudFormation stack – automated deployment to restore or duplicate the environment
- Restore system from backup
- Route traffic to the new system
 - Adjust Domain Name System (DNS) records accordingly



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

27

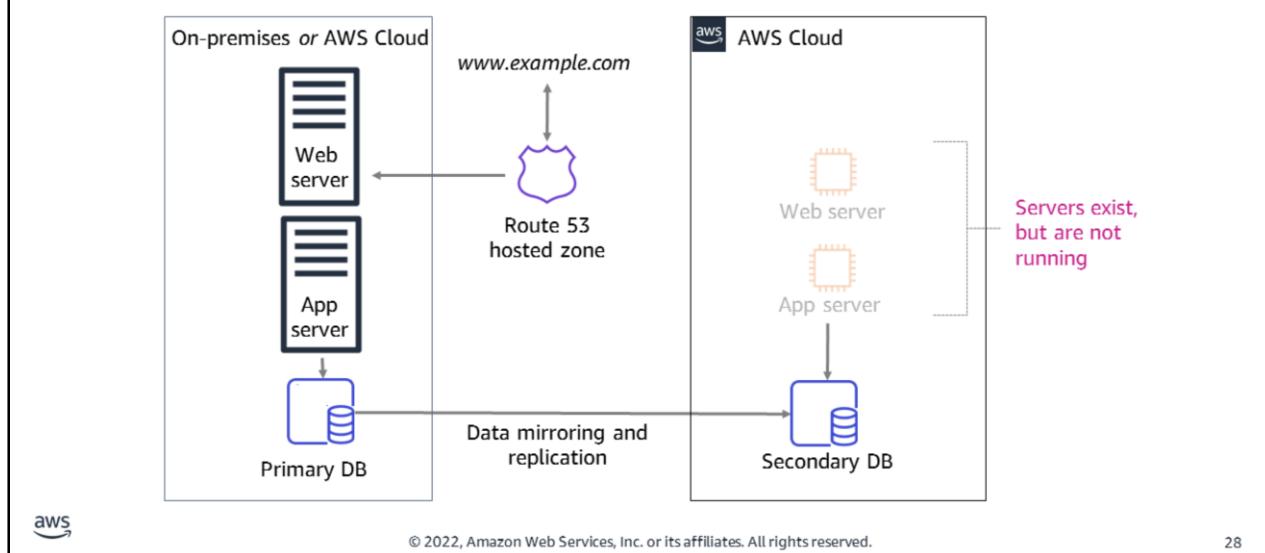
If you implement the backup and restore disaster recovery pattern, the key steps that you should complete during the *preparation phase* are:

- Create backups of current systems
- Store backups in Amazon S3
- Document the procedure to restore from backups

If you implement this pattern, the key steps to complete *in case of disaster* are:

- Retrieve backups from Amazon S3
- Start the required infrastructure
- Restore the system from backups
- Finally, route traffic to the new system

Pilot light pattern: Preparation phase



The second disaster recovery approach is the *pilot light pattern*.

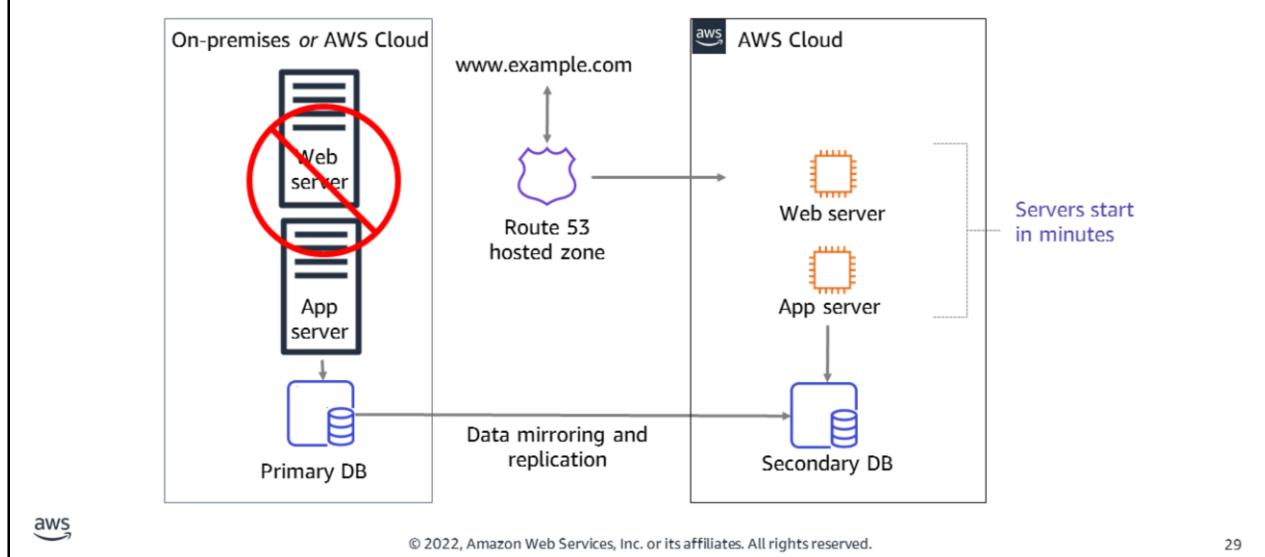
Pilot light describes a disaster recovery pattern where a *minimal* backup version of your environment is *always running*. The pilot light analogy comes from a gas heater: a small flame (or the pilot light) is always on, even when the heater is off. The pilot light can quickly ignite the entire furnace to heat a house. In the example pattern, the pilot light is the secondary database that is always running.

The pilot light scenario is similar to the backup-and-restore scenario. However, recovery time is typically faster because the core pieces of the system are already running and are continually kept up-to-date. When the time comes for recovery, you can rapidly provision a full production environment around the critical core.

Infrastructure elements for the pilot light itself typically include your database servers. This grouping is the critical core of the system (the pilot light). All other infrastructure pieces can quickly be provisioned around it to restore the complete system. To provision the rest of the infrastructure, you typically bundle preconfigured servers as AMIs that are ready to be started at a moment's notice. (Or they might be instances that are in a stopped state.) When recovery begins, these instances start quickly with their pre-defined role, which enables them to connect to the database.

This pattern is relatively inexpensive to implement. Regularly changing data must be replicated to the pilot light, the small core around which the full environment starts in the recovery phase. Your less frequently updated data, such as operating systems and applications, can be periodically updated and stored as AMIs.

Pilot light pattern: In case of disaster



Suppose that disaster strikes, and your primary application goes offline. In this case, you can quickly commission the compute resources to run the application or to orchestrate the failover to pilot light resources in AWS. In this example, the secondary database stores critical data. If there is a disaster, the new web server and app server start up and connect to the secondary database. Amazon Route 53 is configured to then route traffic to the new web server.

The primary environment can exist in an on-premises data center, or in another Region or Availability Zone on AWS. Either way, you can use the pilot light pattern to meet your recovery time objective (RTO).

Pilot light pattern: Checklist

Preparation phase

- Configure EC2 instances to replicate or mirror servers
- Ensure that all supporting custom software packages are available on AWS
- Create and maintain AMIs of key servers where fast recovery is needed
- Regularly run these servers, test them, and apply any software updates and configuration changes
- Consider automating the provisioning of AWS resources

In case of disaster

- Automatically bring up resources around the replicated core dataset
- Scale the system as needed to handle current production traffic
- Switch over to the new system
 - Adjust DNS records to point to AWS



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

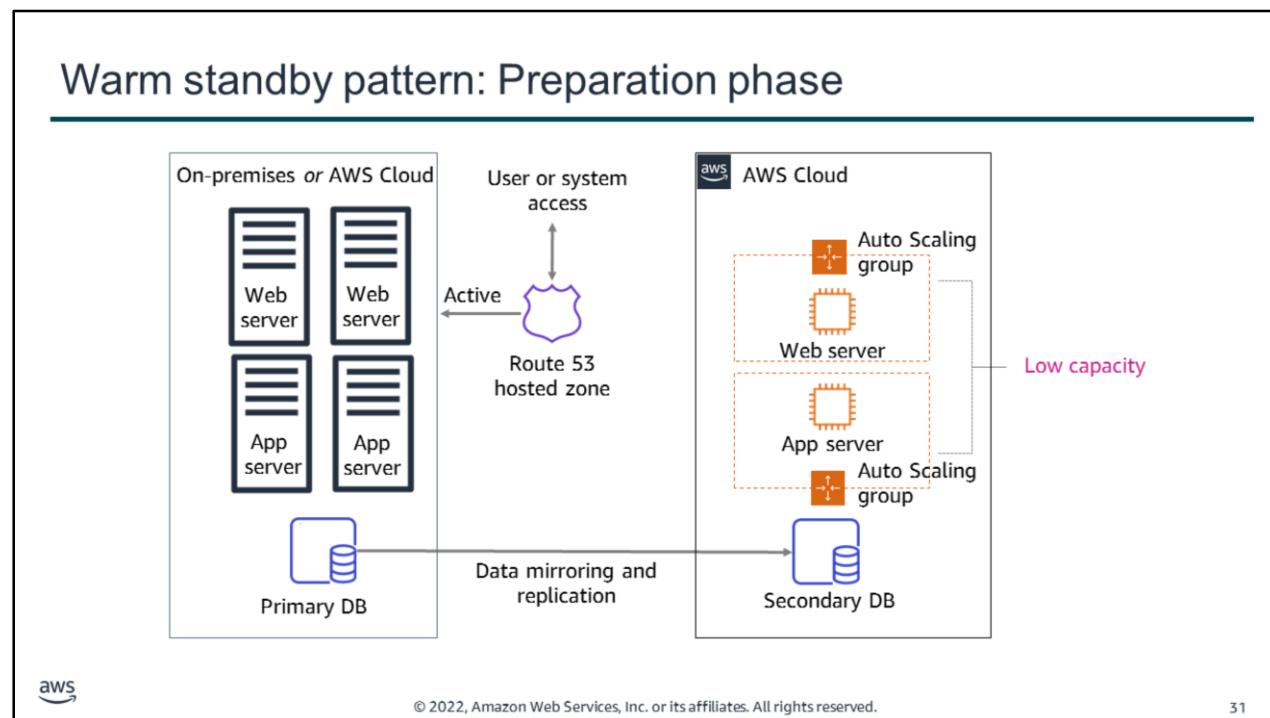
30

If you implement the pilot light disaster recovery pattern, the key steps that you should complete during the *preparation phase* are:

- Configure the EC2 instances
- Ensure that all of the supporting custom software packages are available
- Create and maintain essential AMIs where fast recovery is required
- Regularly run and test servers, and apply software updates and configuration updates
- Consider automating the provisioning of AWS resources

If you implement the pilot light pattern, the key steps to complete *in case of disaster* are:

- Automatically bring up resources around the replicated core dataset
- Scale the system as needed to handle current production traffic
- Switch over to the new system by adjusting the DNS records to point to the backup deployment

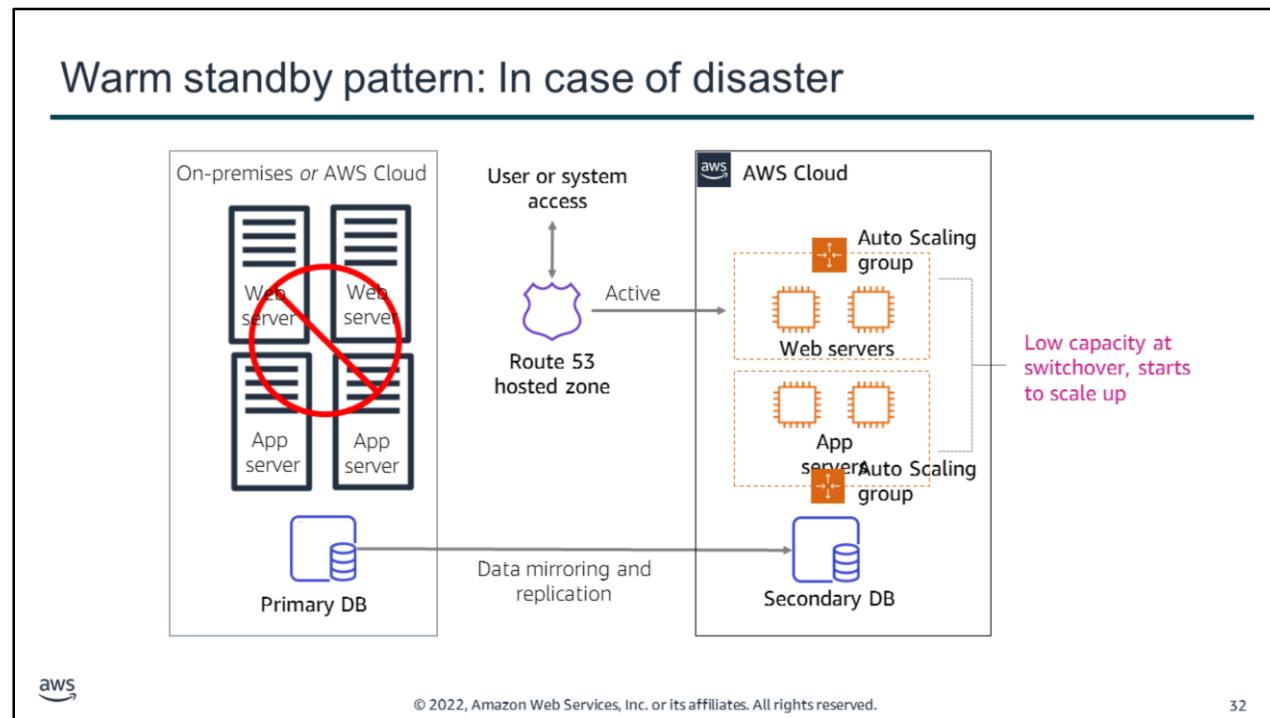


The third disaster recovery approach is the *warm standby pattern*.

The warm standby pattern is like the pilot light, but more resources are already running. The term *warm standby* describes a disaster recovery scenario where a scaled-down version of a fully functional environment is always running in the cloud. The 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 and have them always on.

These servers can be running on a minimum-sized fleet of EC2 instances with the smallest sizes possible. This solution is not yet scaled to take a full production load, but it is fully functional. Though it exists for DR purposes, you can also use it for non-production work, such as testing, quality assurance, and internal use.

In the example, two systems are running. The main system might be running in an on-premises data center or an AWS Region, and a low-capacity system is running on AWS. Use Amazon Route 53 to distribute requests between the main system and the backup system.



In a disaster, if the primary environment is unavailable, Amazon Route 53 switches over to the secondary system.

The secondary system can then quickly begin to scale up to handle the production load. You can produce this increase by adding more EC2 instances to the load balancer. Alternatively, you can resize the small capacity servers to run on larger EC2 instance types. Horizontal scaling (creating more EC2 instances) is preferred over vertical scaling (increasing the size of existing instances).

Warm standby pattern: Checklist

Preparation

- Similar to pilot light
- All necessary components running 24/7, but not scaled for production traffic
- Best practice: Continuous testing
 - Trickle a statistical subset of production traffic to the DR site

In case of disaster

- Immediately fail over most critical production load
 - Adjust DNS records to point to AWS
- (Automatically) Scale the system further to handle all production load



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

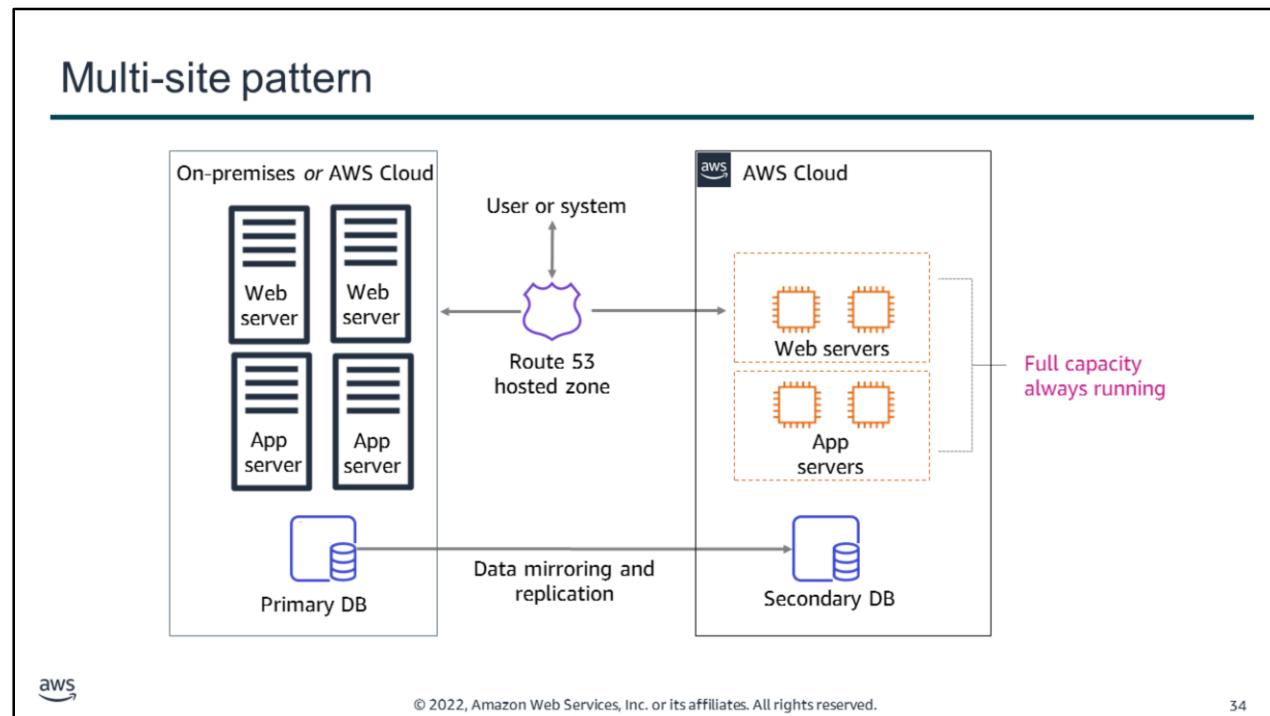
33

If you implement the warm standby disaster recovery pattern, the *preparation phase* is important. The key steps that you should complete during the preparation phase are similar to the steps that you complete for the pilot light pattern. The most notable difference is that all the necessary components should be left running 24/7, but not scaled for production traffic.

As a best practice, conduct continuous testing. You might also trickle a statistical subset of production traffic to the DR site. Thus, you can verify that it functions for users and systems as seamlessly as the primary system.

With the warm standby pattern, *in case of disaster*, the key steps to complete are:

- Immediately fail over the most critical production load
- Adjust DNS records to point to AWS
- (Automatically) Scale the system further to handle all production load



The fourth and final disaster recovery approach is the *multi-site pattern*. With this pattern, you have a fully functional system that runs in a second Region of AWS. It runs at the same time as the on-premises systems or the systems that run in a different AWS Region.

A multi-site solution runs in an *active-active configuration*. The data replication method that you employ is determined from the recovery point that you choose.

Because both sites can support the full production capacity, you might choose to use a DNS service that supports weighted routing. An example is Amazon Route 53, which routes production traffic to both sites that deliver the same application or service. In this scenario, a proportion of traffic goes to your infrastructure in AWS, and the remainder goes to your on-site infrastructure. (Or if the two environments exist in separate AWS Regions, the traffic is proportioned between these two Regions.)

In an on-site or primary AWS Region disaster situation, you can adjust the DNS weighting and send *all* the traffic to the second deployment. The capacity of the secondary deployment can then be rapidly increased to handle the full production load as needed. 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 already-running database services.

The cost of this scenario is determined from the volume of production traffic during normal operation. In the recovery phase, you pay for only what you use for the duration that the DR environment is needed at full scale. You can further reduce cost by purchasing Amazon EC2 Reserved Instances for your always-on AWS servers.

Multi-site: Checklist

Preparation

- Similar to warm standby
- Configured for full scaling in or scaling out for production load

In case of disaster

- Immediately fail over all production load



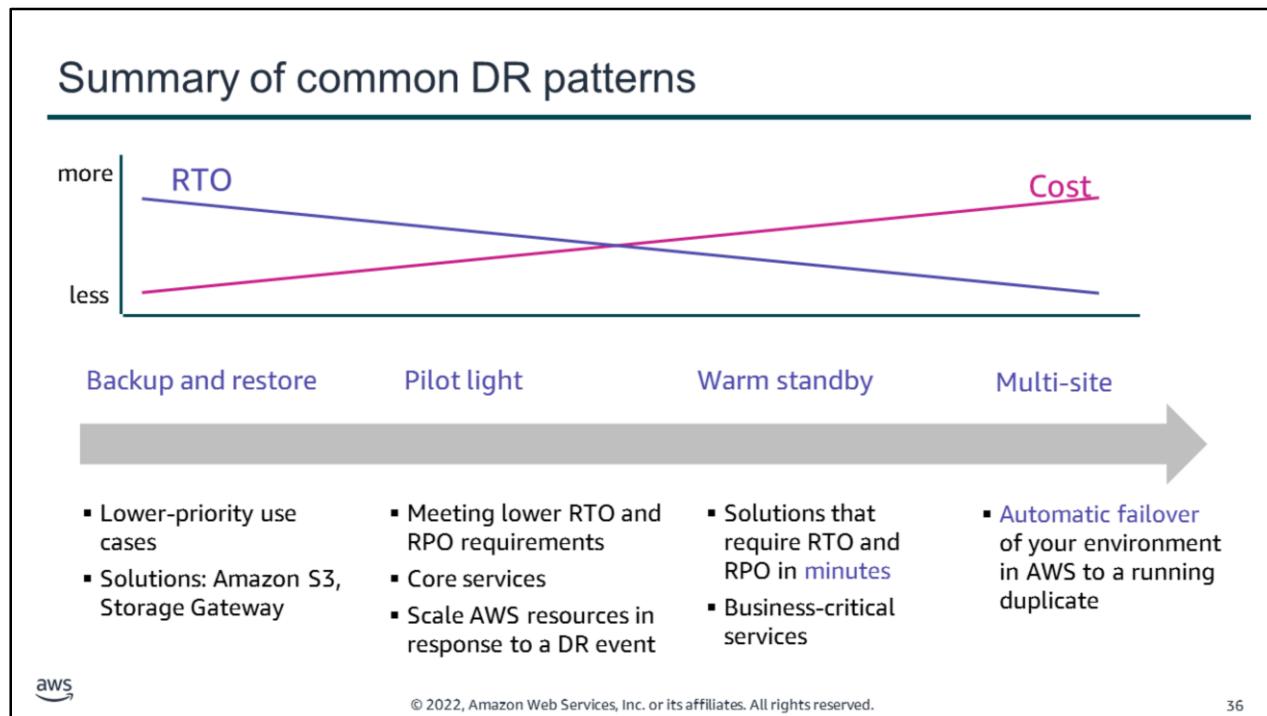
© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

35

If you are implementing the multi-site disaster recovery pattern, the key steps to complete during the *preparation phase* are similar to the warm standby pattern. You must configure the backup deployment for full scaling in and out of the production load. You should have the servers running and ready to receive traffic.

With the multi-site pattern, *in case of disaster*, you only need to complete one key step. That step is to immediately fail over all of the production load to the backup site.

The multi-site pattern potentially has the least downtime of all. However, it does have more costs that are associated with it, because more systems are running.



To summarize, each of the four DR patterns offers a different combination of benefits.

The diagram shows a spectrum for the four scenarios, arranged by how quickly a system can be available to users after a DR event.

The backup and restore pattern typically can be accomplished at the lowest cost, but it has a longer RTO. As a result, your systems are likely to be restored more slowly than with the other options.

The warm standby and multi-site patterns support a much faster RTO, but it is costly to have extra servers that are always running.

AWS enables you to cost-effectively operate each of these DR strategies. It's important to realize that these patterns are only examples of possible approaches, and variations and combinations of these patterns are possible. If your application runs on AWS, then you can use multiple Regions, and the same DR strategies still apply.

DR preparation: Best practices



Start simple



Check for software licensing issues



Practice Game Day exercises



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

37

Creating a comprehensive disaster recovery plan can be a complex undertaking. However, most organizations recognize—perhaps from past events—that it is worth the effort.

Even though it takes time to develop and implement a full plan, it should not stop you from taking some simple first steps. Start simple, and work your way up. For example, as a first step, create backups of data storage, databases, and critical servers. Then, work to incrementally improve RTO and RPO as a continuous effort.

Software licensing is an issue that can surface while you create backup sites. Look into the software licensing that you have to determine whether your current license contracts support your DR plans. Upgrade your licenses or adjust in other ways as necessary.

Finally, it is a best practice to consistently exercise your DR solution so you can ensure that it works as intended. Some suggested steps include:

- Practice Game Day exercises. These exercises test scenarios when critical systems go offline—or even entire regions. What if an entire fleet crashes?
- Ensure that backups, snapshots, and AMIs are being created, and that they can be used to successfully restore data.
- Monitor your monitoring system.

Test your response procedures to ensure they are effective and that teams are familiar with how

to put them into practice. Set up regular Game Days to test workload and team responses to simulated events.

Section 3 key takeaways



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

38

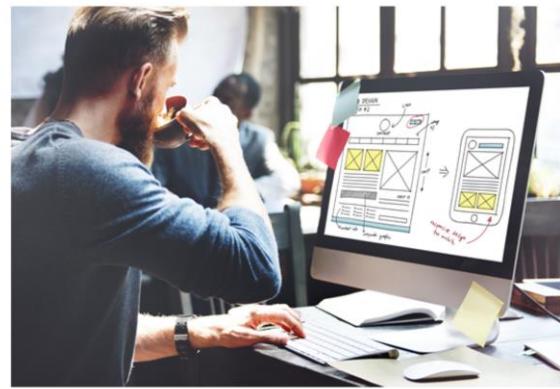
- Common [disaster recovery patterns](#) on AWS include backup and restore, pilot light, warm standby, and multi-site.
- [Backup and restore](#) is the most cost-effective approach. However, it has the highest RTO.
- [Multi-site](#) provides the fastest RTO. However, it costs the most because it provides a fully running production-ready duplicate.
- [AWS Storage Gateway](#) provides three interfaces—file gateway, volume gateway, and tape gateway—for data backup and recovery between on-premises and the AWS Cloud.

Some key takeaways from this section of the module include:

- Common *disaster recovery patterns* on AWS include backup and restore, pilot light, warm standby, and multi-site.
- *Backup and restore* is the most cost effective approach, but it has the highest RTO.
- *Multi-site* provides the fastest RTO, but it costs the most because it provides a fully running production-ready duplicate.
- *AWS Storage Gateway* provides three interfaces—file gateway, volume gateway, and tape gateway—for data backup and recovery between on-premises and the AWS Cloud.

Module 14 – Guided Lab:

Hybrid Storage and Data Migration with AWS Storage Gateway File Gateway



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

39

You will now complete Module 14 – Guided Lab: Hybrid Storage and Data Migration with AWS Storage Gateway File Gateway.

Guided lab: Tasks

1. Reviewing the lab architecture
2. Creating the primary and secondary S3 buckets
3. Enabling Cross-Region Replication
4. Configuring the file gateway and creating an NFS file share
5. Mounting the file share to the Linux instance and migrating the data
6. Verifying that the data is migrated



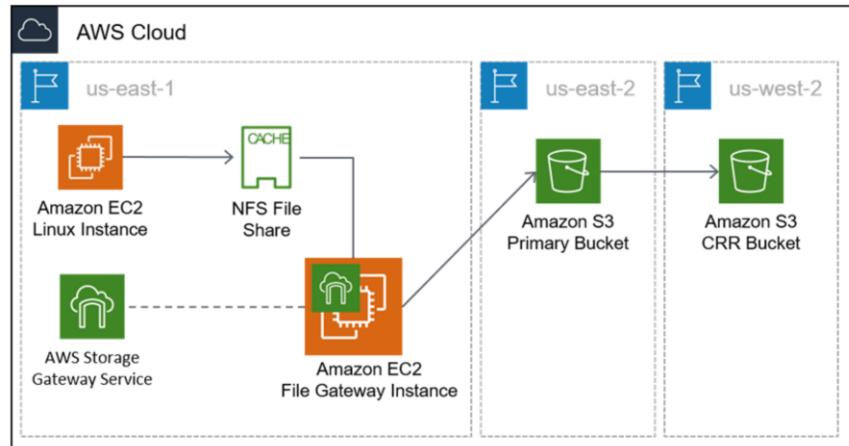
© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

40

In this guided lab, you will complete the following tasks:

1. Reviewing the lab architecture
2. Creating the primary and secondary S3 buckets
3. Enabling Cross-Region Replication
4. Configuring the file gateway and creating an NFS file share
5. Mounting the file share to the Linux instance and migrating the data
6. Verifying that the data is migrated

Guided lab: Final product



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

41

The diagram summarizes what you will have built after you complete the guided lab. You will have successfully migrated data to Amazon S3 by using the file gateway option in AWS Storage Gateway.

For accessibility: Diagram of NFS File Share cache processed by a File Gateway instance which loads data into an S3 bucket in another Region. Contents of the bucket are also copied to a third Region. **End of accessibility description.**



A timer icon with the text "~ 45 minutes" indicating the duration of the lab.

Begin Module 14 – Guided Lab: Hybrid Storage and Data Migration with AWS Storage Gateway File Gateway

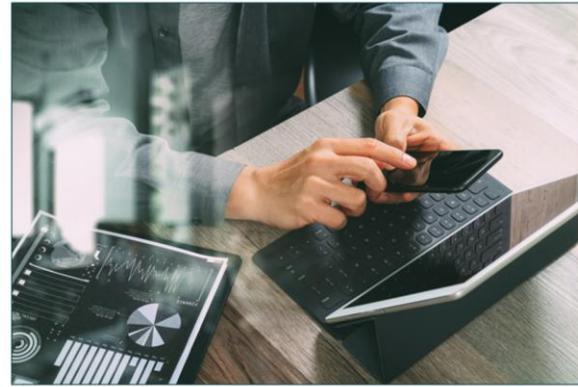
aws

© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

42

It is now time to start the guided lab.

Guided lab debrief: Key takeaways



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

43

Your educator might choose to lead a conversation about the key takeaways from this guided lab after you have completed it.

Module wrap-up

Module 14: Planning for Disaster



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

It's now time to review the module and wrap up with a knowledge check and discussion of a practice certification exam question.

Module summary

In summary, in this module, you learned how to:

- Identify strategies for disaster planning
- Define RPO and RTO
- Describe four common patterns for backup and disaster recovery and how to implement them
- Use AWS Storage Gateway for on-premises-to-cloud backup solutions



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

45

In summary, in this module, you learned how to:

- Identify strategies for disaster planning
- Define RPO and RTO
- Describe four common patterns for backup and disaster recovery and how to implement them
- Use AWS Storage Gateway for on-premises-to-cloud backup solutions

Complete the knowledge check



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

45

It is now time to complete the knowledge check for this module.

Sample exam question



Company salespeople upload their sales figures daily. A Solutions Architect needs a durable storage solution for these documents that also protects against users accidentally deleting important documents.

Which action will protect against unintended user actions?

Choice Response

- | | |
|---|---|
| A | Store data in an EBS volume and create snapshots once a week. |
| B | Store data in an S3 bucket and enable versioning. |
| C | Store data in two S3 buckets in different AWS Regions. |
| D | Store data on EC2 instance storage. |

© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

47

Look at the answer choices and rule them out based on the keywords.

Sample exam question answer



Company salespeople upload their sales figures daily. A Solutions Architect needs a durable storage solution for these documents that also protects against users accidentally deleting important documents.

Which action will protect against unintended user actions?

The correct answer is B.

The keywords in the question are “salespeople upload”, “durable storage solution”, and “protects against users accidentally deleting important documents.”

© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

48

The following are the keywords to recognize: **“salespeople upload”**. **“durable storage solution”**, and **“protects against users accidentally deleting important documents.”**

The correct answer is B: “Store data in an S3 bucket and enable versioning.” With this approach, if a versioned object is deleted, it can still be recovered by retrieving the final version.

Response A would lose any changes that were committed since the previous snapshot. Response C, storing the data in two S3 buckets, would provide slightly more protection than response A. However, a user might still delete the object from both buckets. Response D is not a good approach, because EC2 instance storage is ephemeral and should never be used for data that requires durability.

Additional resources

- [Amazon S3 Replication](#)
- [Amazon S3 Object Lifecycle Management](#)
- [Amazon EBS Snapshots](#)
- [Using AWS Lambda with Scheduled Events](#)
- [Backup & Restore resource center](#)
- [Disaster Recovery with AWS \(video\)](#)



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

49

If you want to learn more about the topics that are covered in this module, you might find the following resources helpful:

- [Amazon S3 Replication](#)
- [Amazon S3 Object Lifecycle Management](#)
- [Amazon EBS Snapshots](#)
- [Using AWS Lambda with Scheduled Events](#)
- [Backup & Restore resource center](#)
- [Disaster Recovery with AWS \(video\)](#)



Thank you

Corrections, feedback, or other questions?
Contact us at <https://support.aws.amazon.com/#/contacts/aws-academy>.
All trademarks are the property of their owners.

© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

50

Thank you for completing this module.



training and
certification

AWS Academy Cloud Architecting
Module 15 Student Guide
Version 2.0.12

200-ACACAD-20-EN-SG

© 2023, Amazon Web Services, Inc. or its affiliates. All rights reserved.

This work may not be reproduced or redistributed, in whole or in part,
without prior written permission from Amazon Web Services, Inc.
Commercial copying, lending, or selling is prohibited.

All trademarks are the property of their owners.

Contents

Module 15: Bridging to Certification

4



Module 15: Bridging to Certification

AWS Academy Cloud Architecting

© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

Welcome to Module 15: Bridging to Certification.

Module overview

Sections

1. Certification exam resources
2. Additional resources



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

2

This module includes the following sections:

1. Certification exam resources
2. Additional resources

Module objectives

At the end of this module, you should be able to:

- Identify how to prepare for the AWS Certified Solutions Architect - Associate exam
- Identify where to find resources



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

3

At the end of this module, you should be able to:

- Identify how to prepare for the AWS Certified Solutions Architect - Associate exam
- Identify where to find resources

Section 1: Certification exam resources

Module 15: Bridging to Certification



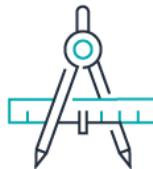
© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

Introducing Section 1: Certification exam resources.

Certification exam domain 1 objectives

Domain 1: Design Resilient Architectures (30%)

- 1.1 Design a **multi-tier** architecture solution
- 1.2 Design **highly available** and/or **fault-tolerant** architectures
- 1.3 Design **decoupling mechanisms** using AWS services
- 1.4 Choose appropriate **resilient storage**



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

5

Domain 1 of the AWS Certified Solutions Architect - Associate examination is *Design Resilient Architectures*. It is 30 percent of the exam and covers the following objectives:

- 1.1 Design a multi-tier architecture solution
- 1.2 Design highly available and/or fault-tolerant architectures
- 1.3 Design decoupling mechanisms using AWS services
- 1.4 Choose appropriate resilient storage

This information can be found in the [exam guide](#).

Certification exam domain 2 objectives

Domain 2: Design High-Performing Architectures (28%)

- 2.1 Identify elastic and scalable compute solutions for a workload
- 2.2 Select high-performing and scalable storage solutions for a workload
- 2.3 Select high-performing networking solutions for a workload
- 2.4 Choose high-performing database solutions for a workload



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

6

Domain 2 of the AWS Certified Solutions Architect - Associate examination is *Design High-Performing Architectures*. It is 28 percent of the exam and covers the following objectives:

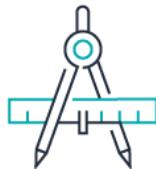
- 2.1 Identify elastic and scalable compute solutions for a workload
- 2.2 Select high-performing and scalable storage solutions for a workload
- 2.3 Select high-performing networking solutions for a workload
- 2.4 Choose high-performing database solutions for a workload

This information can be found in the [exam guide](#).

Certification exam domain 3 objectives

Domain 3: Design Secure Applications and Architectures (24%)

- 3.1 Design **secure access** to AWS resources
- 3.2 Design **secure application tiers**
- 3.3 Select appropriate **data security options**



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

7

Domain 3 of the AWS Certified Solutions Architect - Associate examination is *Design Secure Applications and Architectures*. It is 24 percent of the exam and covers the following objectives:

- 3.1 Design secure access to AWS resources
- 3.2 Design secure application tiers
- 3.3 Select appropriate data security options

This information can be found in the [exam guide](#).

Certification exam domain 4 objectives

Domain 4: Design Cost-Optimized Architectures (18%)

- 4.1 Identify **cost-effective storage** solutions
- 4.2 Identify **cost-effective compute and database** services
- 4.3 Design **cost-optimized network** architectures



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

8

Domain 4 of the AWS Certified Solutions Architect - Associate examination is *Design Cost-Optimized Architectures*. It is 18 percent of the exam and covers the following objectives:

- 4.1 Identify cost-effective storage solutions
- 4.2 Identify cost-effective compute and database services
- 4.3 Design cost-optimized network architectures

This information can be found in the [exam guide](#).

Exam guide and sample questions

aws training and certification

AWS Certified Solutions Architect – Associate (SAA-C02) Exam Guide

Introduction

The AWS Certified Solutions Architect - Associate (SAA-C02) examination is intended for individuals who perform in a solutions architect role. This exam validates an examinee's ability to effectively demonstrate knowledge of how to architect and deploy secure and robust applications on AWS technologies.

It validates an examinee's ability to:

- Define a solution using architectural design principles based on customer requirements.
- Provide implementation guidance based on best practices to an organization throughout the lifecycle of a project.

Recommended AWS Knowledge

- 1 year of hands-on experience designing available, cost-effective, fault-tolerant, and scalable distributed systems on AWS.
- Hands-on experience using compute, networking, storage, and database AWS services.
- Hands-on experience with AWS deployment and management services.
- Ability to identify and define technical requirements for an AWS-based application.
- Ability to identify which AWS services meet a given technical requirement.
- Knowledge of recommended best practices for building secure and reliable applications on the AWS platform.
- An understanding of the basic architectural principles of building in the AWS Cloud.
- An understanding of the AWS global infrastructure.
- An understanding of network technologies as they relate to AWS.
- An understanding of security features and tools that AWS provides and how they relate to traditional services.

Exam Guide



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

aws training and certification

AWS Certified Solutions Architect – Associate (SAA-C02) Sample Exam Questions

1) A customer relationship management (CRM) application runs on Amazon EC2 instances in multiple Availability Zones behind an Application Load Balancer.

If one of these instances fails, what occurs?

- The load balancer will stop sending requests to the failed instance.
- The load balancer will terminate the failed instance.
- The load balancer will automatically replace the failed instance.
- The load balancer will return 504 Gateway Timeout errors until the instance is replaced.

2) A company needs to perform asynchronous processing, and has Amazon SQS as part of a decoupled architecture. The company wants to ensure that the number of empty responses from polling requests are kept to a minimum.

What should a solutions architect do to ensure that empty responses are reduced?

- Increase the maximum message retention period for the queue.
- Increase the maximum receives for the receive poller for the queue.
- Increase the default visibility timeout for the queue.
- Increase the receive message wait time for the queue.

3) A company currently stores data for on-premises applications on local drives. The chief technology officer wants to reduce hardware costs by storing the data in Amazon S3 but does not want to make modifications to the applications. To minimize latency, frequently accessed data should be available locally.

What is a reliable and durable solution for a solutions architect to implement that will reduce the cost of local storage?

- Deploy an SFTP client on a local server and transfer data to Amazon S3 using AWS Transfer for SFTP.

Sample Exam Questions

9

Review the [Exam Guide](#), which contains the content outline and target audience for the certification exam.

Perform a self-assessment to identify your knowledge or skills gaps. AWS offers [Sample Exam Questions](#) for the AWS Certified Solutions Architect - Associate exam.

Exam readiness training

The screenshot shows the AWS Exam Readiness training page. It features three main sections:

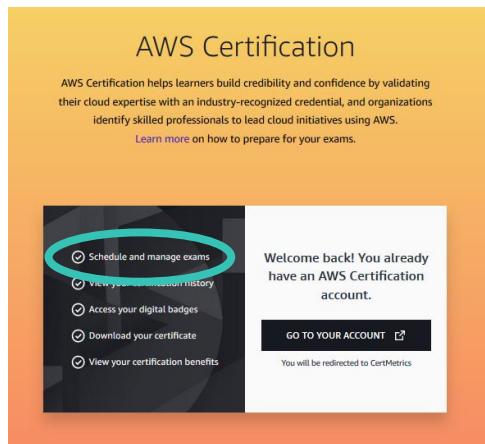
- Classroom course:** "Exam Readiness: AWS Certified Solutions Architect – Associate". Includes "ABOUT" and "SCHEDULE" links.
- Digital course:** "Exam Readiness: AWS Certified Solutions Architect – Associate (Digital)". Includes "ABOUT" and "MODULES" links.
- Intensive Workshop:** "Exam Readiness Intensive Workshop: AWS Certified Solutions Architect - Associate". Includes a "Get ready for your certification with a focused five-day workshop" callout.

Common buttons include "COURSE", "FIND A CLASS", "FREE COURSE", and "SIGN IN >". The AWS logo is at the bottom left, and copyright information is at the bottom right.

AWS also offers *Exam Readiness: AWS Certified Solutions Architect - Associate* training in [classroom](#) and [free digital training](#) formats. This training helps you prepare for the AWS certification exam. AWS develops the training so that the content is current with the newest best practices. The training teaches you how to interpret exam questions and allocate your study time. It also reviews sample exam questions in each topic area. It teaches you how to interpret the concepts that are tested so that you can more easily eliminate incorrect responses.

There is also a 5-day, instructor-led [Exam Readiness Intensive Workshop](#) that can help you prepare for the exam.

Practice exam



Register at [AWS Training](#)



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

11

When you are ready, you can register to take a practice exam at [AWS Training](#). The practice exam supplements the exam guide and helps you test your knowledge before you take the final exam.

AWS whitepapers and guides

AWS Whitepapers & Guides

Expand your knowledge of the cloud with AWS technical content authored by AWS and the AWS community, including technical whitepapers, technical guides, reference material, and reference architecture diagrams.

[AWS Whitepapers & Guides](#) website



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

12

It's a good idea to broaden your technical understanding by reading whitepapers and other technical content. AWS, independent analysts, and AWS Partner Network (APN) Partners create many of these helpful resources.

You can find whitepapers, technical guides, reference material, and architecture diagrams on the [AWS Whitepapers & Guides](#) website.

AWS Well-Architected Framework

Read the AWS Well-Architected Framework and Pillar whitepapers

AWS Well-Architected

Learn, measure, and build using architectural best practices

[AWS Well-Architected Framework website](#)



Framework



Security



Operational Excellence



Reliability



Performance Efficiency



Cost Optimization



Sustainability



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

13

To prepare for the AWS Certified Solutions Architect - Associate exam, focus on reading the whitepapers for the AWS Well-Architected Framework and each of the Pillars:

- [AWS Well-Architected Framework whitepaper](#)
- [Security Pillar whitepaper](#)
- [Operational Excellence Pillar whitepaper](#)
- [Reliability Pillar whitepaper](#)
- [Performance Efficiency Pillar whitepaper](#)
- [Cost Optimization Pillar whitepaper](#)
- [Sustainability Pillar whitepaper](#)

AWS frequently asked questions

FAQs

Select from the following list of Product and Technical FAQs. Browse through these FAQs to find answers to commonly raised questions. If you're a new customer of one of the services below, we encourage you to read through the relevant article.

Compute

[Amazon EC2 FAQ](#)

[Amazon EC2 Auto Scaling FAQ](#)

[Amazon EC2 Windows FAQ](#)

[Amazon EC2 Container Registry FAQ](#)

[Amazon EC2 Container Service FAQ](#)

[Amazon Lightsail FAQ](#)

[AWS Batch FAQ](#)

[AWS Elastic Beanstalk FAQ](#)

[AWS Fargate FAQ](#)

[AWS Lambda FAQ](#)

[AWS Serverless Application Repository FAQ](#)

[Amazon Elastic Load Balancing FAQ](#)

[VMware Cloud on AWS FAQ](#)

Storage

[Amazon EBS FAQ](#)

Machine Learning

[AWS SageMaker FAQ](#)

[AWS Comprehend FAQ](#)

[AWS Lex FAQ](#)

[AWS Polly FAQ](#)

[AWS Rekognition FAQ](#)

[Amazon Machine Learning FAQ](#)

[AWS Translate FAQ](#)

[AWS Transcribe FAQ](#)

[AWS DeepLens FAQ](#)

Analytics

[Amazon Athena FAQ](#)

[Amazon Elastic MapReduce FAQ](#)

[Amazon CloudSearch FAQ](#)

[Amazon Elasticsearch Service FAQ](#)

[Amazon Kinesis FAQ](#)

[Amazon QuickSight FAQ](#)

Review the FAQs for:

- **Amazon EC2**
- **Amazon S3**
- **Amazon VPC**
- **Amazon Route 53**
- **Amazon RDS**
- **Amazon SQS**

[AWS FAQs website](#)



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

14

Browse the [product and technical frequently asked questions \(FAQs\)](#) to familiarize yourself with commonly raised questions and issues.

Be sure to review the following FAQs:

- Amazon Elastic Compute Cloud (Amazon EC2)
- Amazon Simple Storage Service (Amazon S3)
- Amazon Virtual Private Cloud (Amazon VPC)
- Amazon Route 53
- Amazon Relational Database Service (Amazon RDS)
- Amazon Simple Queue Service (Amazon SQS)

Section 2: Additional resources

Module 15: Bridging to Certification



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

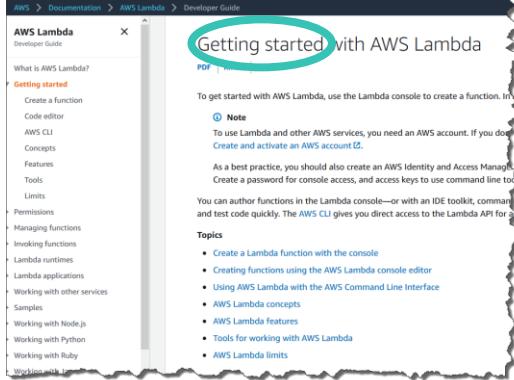
Introducing Section 2: Additional resources.

AWS Documentation

AWS Documentation

Find user guides, developer guides, API references, tutorials, and more.

[AWS Documentation website](#)



The screenshot shows the AWS Lambda Developer Guide. The left sidebar has a tree view with categories like 'Getting started', 'Code editor', 'AWS CLI', etc. The main content area is titled 'Getting started with AWS Lambda'. It includes a note about creating an AWS account, best practices for Lambda, and a list of topics such as 'Create a Lambda function with the console', 'Using AWS Lambda with the AWS Command Line Interface', and 'AWS Lambda limits'. A green oval highlights the title 'Getting started with AWS Lambda'.

aws

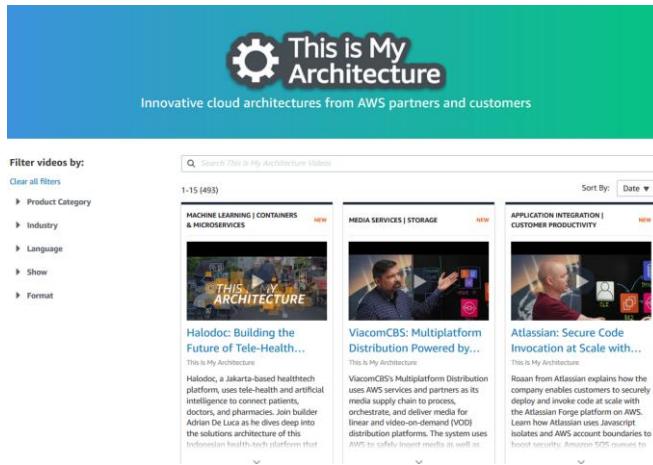
© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

16

You can find user guides, developer guides, API references, tutorials and projects, SDKs and toolkits, and various other general resources in the [AWS Documentation](#).

The AWS service developer guides contain a *Getting started* section with step-by-step tutorials that can help familiarize you with the services.

This is My Architecture video series



The screenshot shows the homepage of the "This is My Architecture" video series. At the top, there's a green banner with the title "This is My Architecture" and a subtitle "Innovative cloud architectures from AWS partners and customers". Below the banner is a search bar and a "Sort By" dropdown set to "Date". On the left, there's a sidebar with a "Filter videos by:" section containing links for "Clear all Filters", "Product Category", "Industry", "Language", "Show", and "Format". The main content area displays a grid of video thumbnails. Each thumbnail includes a category label (e.g., "MACHINE LEARNING | CONTAINERS & MICROSERVICES", "MEDIA SERVICES | STORAGE", "APPLICATION INTEGRATION | CUSTOMER PRODUCTIVITY"), a title, a brief description, and a small image. One thumbnail for "Halodoc: Building the Future of Tele-Health..." is visible.

This is My Architecture website

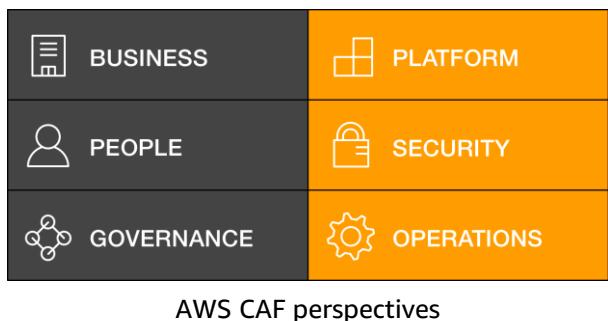
aws

© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

17

[This is My Architecture](#) is a video series that can help you learn about the innovative cloud architectures that APN Partners and customers have built.

AWS Cloud Adoption Framework



- Provides guidance and best practices to help organizations build a comprehensive approach to cloud computing
- Can be used across the organization and throughout the IT lifecycle to accelerate successful cloud adoption

[AWS Cloud Adoption Framework](#) website

- Is organized into six perspectives that consist of sets of capabilities



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

18

Each organization's cloud adoption journey is unique. However, for any organization to successfully migrate its IT portfolio to the cloud, three elements (people, process, and technology) must be in alignment. Business and technology leaders in an organization must understand the organization's current state, target state, and the transition necessary to achieve the target state. In this way, they can set goals and create processes for staff.

The AWS Cloud Adoption Framework ([AWS CAF](#)) provides guidance and best practices to help organizations identify gaps in skills and processes. It also helps organizations build a comprehensive approach to cloud computing—both across the organization and throughout the IT lifecycle—to accelerate successful cloud adoption.

At the highest level, the AWS CAF organizes guidance into six areas of focus, called *perspectives*. Perspectives span people, processes, and technology. Each perspective consists of a set of *capabilities*, which covers distinct responsibilities that functionally related stakeholders own or manage.

Capabilities within each perspective are used to identify which areas of an organization require attention. By identifying gaps, prescriptive work streams can be created that support a successful cloud journey.

There's no substitute for hands-on experience.



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

19

When you are preparing for the AWS Certification exam, it's important to remember that hands-on experience has no substitute. Before you take an AWS Certification exam, AWS recommends that you have hands-on experience with relevant AWS products and services. AWS offers various resources to help you gain hands-on experience with AWS services.

Self-Paced Labs

Self-Paced Labs

Get hands-on practice in a live AWS environment with AWS services and real-world cloud scenarios. Follow step-by-step instructions to learn a service, practice a use case, or prepare for AWS Certification.

[Take a Lab or Quest](#)

[Self-Paced Labs website](#)



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

20

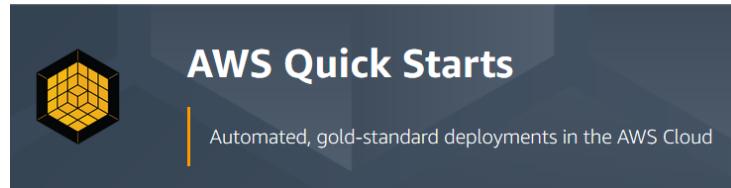
Self-paced labs are a way for you to get hands-on practice with AWS services and real world-cloud scenarios in a live AWS environment.

You can:

- Take an individual lab to get familiar with an AWS service.
- Follow a learning quest that will lead you through a sequence of labs so that you can learn how to work with related AWS services. When you complete a quest, you will earn a Quest Badge that you can display on your resume, website, or LinkedIn profile.

For more information about self-paced labs, see the [Self-Paced Labs](#) website.

AWS Quick Starts



[AWS Quick Starts website](#)



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

21

AWS solutions architects and partners build Quick Starts to help you deploy popular technologies on AWS. These technologies are based on AWS best practices for security and high availability. These accelerators reduce hundreds of manual procedures into a few steps, so you can build your production environment quickly and start to use it immediately.

Each Quick Start includes AWS CloudFormation templates that automate the deployment and a guide that discusses the architecture and provides step-by-step deployment instructions.

For more information, see [AWS Quick Starts](#).

Getting Started tutorials

The screenshot shows the AWS Getting Started Resource Center. At the top, there are navigation links: Getting Started Resource Center, Overview, Fundamentals, Learning Paths (which is highlighted with a green oval), Category Deep Dive, Hands-On Tutorials (also highlighted with a green oval), and Resources. Below the header, a banner says "Getting Started with AWS" and "Learn the fundamentals and start building on AWS now." On the left, there's a sidebar with a "Filter by" section containing "Category" (Containers, Compute, Databases, Dev Tools, Machine Learning, Mobile, Serverless, Storage, Account Management), "Cloud Level" (100, 200, 300), and "Content Type" (Hands-on, Video). A search bar is also present. The main content area displays four tutorial cards:

- STORAGE**: Host a Static Website (Hands-On, FREE TIER, 30 Minutes)
- COMPUTE**: Launch a Linux Virtual Machine (Hands-On, FREE TIER, 10 Minutes)
- MOBILE**: Deploy and Host a ReactJS App (Hands-On, FREE TIER, 10 Minutes)
- SERVERLESS**: Run a Serverless "Hello, World!" (Hands-On, FREE TIER, 10 Minutes)

At the bottom of the page, there's a link to "Getting Started with AWS website" and a copyright notice: "© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved." The AWS logo is also visible.

[Getting Started hands-on tutorials and learning paths](#) help you learn the fundamentals of AWS and start building on AWS right away.

Module wrap-up

Module 15: Bridging to Certification



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

It's now time to review the module.

Module summary

In summary, in this module, you learned how to:

- Identify how to prepare for the AWS Certified Solutions Architect - Associate exam
- Identify where to find resources



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

27

In summary, in this module, you learned how to:

- Identify how to prepare for the AWS Certified Solutions Architect - Associate exam
- Identify where to find resources

Thank you

Corrections, feedback, or other questions?
Contact us at <https://support.aws.amazon.com/#/contacts/aws-academy>.
All trademarks are the property of their owners.



© 2022, Amazon Web Services, Inc. or its affiliates. All rights reserved.

28

Thank you for completing this module.