

# Chapter 6

## Designing Resilient Architecture

This chapter covers the following topics:

- [Scalable and Resilient Architecture](#)
- [Application Integration Services](#)
- [Amazon API Gateway](#)
- [Automating AWS Infrastructure](#)
- [AWS Elastic Beanstalk](#)

This chapter covers content that's important to the following exam domain and task statement:

Domain 2: Design Resilient Architectures

Task Statement 1: Design scalable and loosely coupled architectures

The AWS Certified Solutions Architect – Associate (SAA-C03) exam requires that you understand the AWS services that assist in developing workloads and applications hosted in the AWS cloud. The exam does not expect that you're a developer; however, it does expect that you can help advise developers

and educate them on what services could be useful for creating stateless applications to run successfully in the AWS cloud.

Although the cloud can certainly host legacy monolithic applications, the SAA-C03 exam also tests your understanding of the purpose of AWS application integration services. Lifting and shifting an application from an on-premises location into AWS does work, but, in the long term, moving existing workloads to the cloud without re-architecting to take advantage of the features of AWS will not be a successful plan long-term.

This chapter begins by demystifying the terms *stateful* and *stateless* as they pertain to the AWS cloud. This chapter also looks at the Amazon API Gateway and automation strategies for deploying scalable architecture using AWS CloudFormation, AWS Service Catalog, and AWS Elastic Beanstalk.

## “Do I Know This Already?”

The “Do I Know This Already?” quiz allows you to assess whether you should read this entire chapter thoroughly or jump to the “Exam Preparation Tasks” section. If you doubt your answers to these questions or your own assessment of your knowledge of the topics, read the entire chapter. [Table 6-1](#) lists the major headings in this chapter and their corresponding “Do I Know This Already?” quiz questions. You can find the

answers in [Appendix A](#), “[Answers to the ‘Do I Know This Already?’ Quizzes and Q&A Sections.](#)”

**Table 6-1** “Do I Know This Already?” Section-to-Question Mapping

| Foundation Topics Section           | Questions |
|-------------------------------------|-----------|
| Scalable and Resilient Architecture | 1, 2      |
| Application Integration Services    | 3, 4      |
| Amazon API Gateway                  | 5, 6      |
| Automating AWS Infrastructure       | 7, 8      |
| AWS Elastic Beanstalk               | 9, 10     |

---

### Caution

The goal of self-assessment is to gauge your mastery of the topics in this chapter. If you do not know the answer to a question or are only partially sure of the answer, you should mark that question

as wrong for purposes of the self-assessment. Giving yourself credit for an answer you correctly guess skews your self-assessment results and might provide you with a false sense of security.

---

- 1.** Which of the following AWS services is stateful?
  1. Security groups
  2. AWS SQS
  3. Amazon Route 53
  4. Availability zone
  
- 2.** Which of the following terms would apply to a virtual server that saves data about each current client session?
  1. Stateless
  2. Stateful
  3. Primary
  4. Secondary
  
- 3.** Which AWS service could be useful for storing application state for processing?
  1. AWS SNS
  2. AWS SQS

3. Amazon S3 Glacier

4. Amazon EBS

**4.** Which AWS service is used to send notifications about AWS service changes to both humans and services?

1. Amazon SES

2. AWS SNS

3. Amazon Chime

4. Amazon Kinesis

**5.** Which AWS service can be used to host and execute custom functions?

1. Amazon EC2

2. AWS Lambda

3. Amazon IAM

4. AWS SQS

**6.** What is the commercial charging model for AWS Lambda?

1. Processing time

2. RAM/CPU and processing time

3. The number of functions executed per month per account

4. RAM and CPU allocated per function

**7.** What is the purpose of using AWS CloudFormation?

1. To recover from failures
2. To automate the building of AWS infrastructure components
3. To deploy applications with automation
4. To document manual tasks

**8.** What AWS CloudFormation component is used to advise on deployment changes to existing deployed infrastructure?

1. AWS CloudFormation template
2. Stack
3. Change set
4. JSON script

**9.** What two components are deployed using AWS Elastic Beanstalk?

1. Infrastructure and storage
2. Application and infrastructure
3. Compute and storage
4. Containers and instances

**10.** What term defines application updates that are applied to a new set of EC2 instances?

1. Rolling
2. Immutable
3. All at once
4. Blue/green

## Foundation Topics

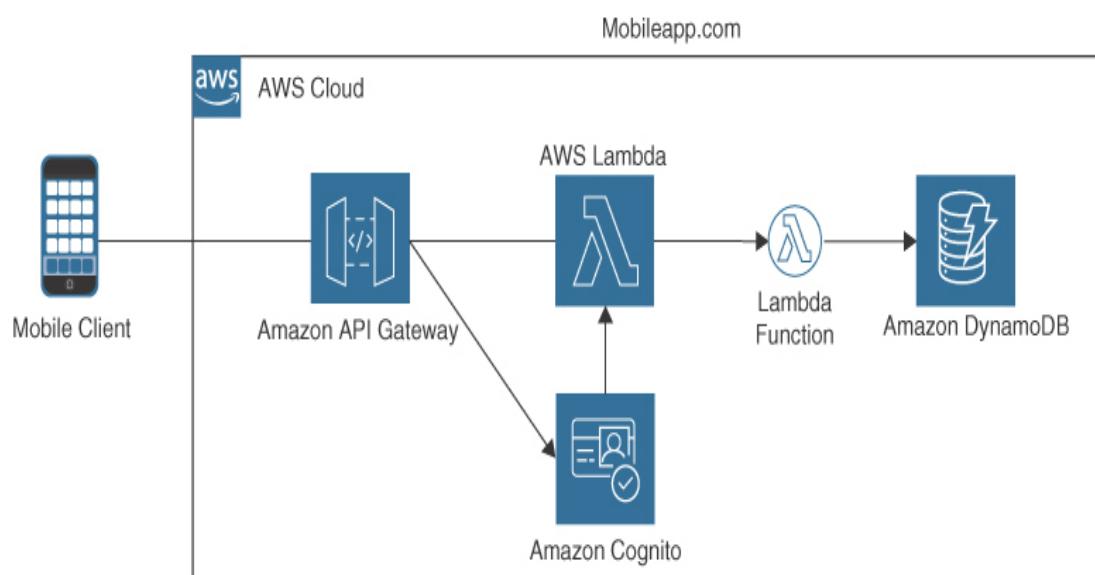
### Scalable and Resilient Architecture

Designing scalable and loosely coupled architecture allows organization workloads to automatically scale, meeting business and design requirements and increasing workload reliability.

Modern modular design can help create workloads constructed with many loosely coupled services integrated using a common set of APIs, creating a functional workload with micro-service architectures. Modular applications can be linked together using ***serverless*** services performing one or more specific tasks as an integrated workload service. The application shown in [Figure 6-1](#) utilizes several AWS serverless services:

- **Amazon API Gateway:** The AWS API Gateway is located between your client applications and back-end services and routes requests from clients to the appropriate back end

located at AWS, or on premises, returning information back to the client. API Gateway can also provide other useful features, such as authentication and caching of common API requests, to help improve the security and performance of the hosted API.



**Figure 6-1** Complex Application Broken into Smaller Parts

- **Amazon Cognito:** Amazon Cognito provides authentication, authorization, and user management for mobile applications helping developers build applications that securely store and manage user data. Amazon Cognito can authenticate end-user requests authorizing access to API Gateway endpoints.
- **AWS Lambda:** AWS Lambda executes custom functions in response to events within milliseconds of each request. The

underlying compute resources that execute each function are automatically managed in the background. Custom functions can be written and hosted by AWS Lambda implementing the business logic for an API hosted at Amazon API Gateway.

- **Amazon DynamoDB:** Amazon DynamoDB is a fully managed NoSQL database service that provides fast and predictable performance with seamless scalability. Amazon DynamoDB enables organizations to offload the administrative tasks of hardware provisioning, setup and configuration, replication, software patching, or cluster scaling. Amazon DynamoDB supports both document and key-value data models, enabling users to store and retrieve data for mobile, web, gaming, IoT, and applications that need low-latency access to its data.

## Scalable Delivery from Edge Locations

Including Amazon CloudFront and edge locations as part of your application solution architecture allows regional workloads to scale globally, and reliably. Leveraging a CloudFront distribution can also help minimize DDoS attacks by protecting application data stored in Amazon S3 buckets. AWS WAF filters can provide additional application layer protection to help protect web applications from common web exploits that could affect application availability, compromise security, or consume excessive resources. The following workload

designs will benefit from deploying Amazon CloudFront as a content delivery network (CDN):

- **Static website content delivery:** Use an Amazon S3 bucket to host a web application and store static web assets and content. For example, host a React application and deliver requested content quickly across hundreds of AWS edge locations using an Amazon CloudFront distribution.
- **Live streaming video:** CloudFront supports streaming of various types of media content, such as audio, video, and live events, using the HTTP Live Streaming (HLS) and Dynamic Adaptive Streaming over HTTP (DASH) protocols to any device caching media fragments at the edge location closest to the end user, ensuring delivery of live streaming fragments from origin. CloudFront streaming also integrates with other AWS services such as Amazon Elastic Transcoder and Amazon Kinesis Video Streams to provide solutions for streaming media content.
- **Encrypting system fields:** Use an HTTPS connection with field-level encryption to encrypt specific fields, ensuring applications are restricted from viewing sensitive data fields. AWS Field Level Encryption enables you to encrypt sensitive data fields in your objects, such as credit card numbers or personally identifiable information (PII), before they are cached in a CloudFront distribution at an edge location. AWS

Field Level Encryption ensures that the sensitive data is protected while in transit to and from CloudFront. Field Level Encryption is useful for e-commerce and financial services applications that handle large amounts of sensitive data.

- **Customize processing at edge locations:** AWS

Lambda@Edge is a feature of Amazon Web Services (AWS) that enables you to run serverless functions in response to CloudFront requests to website data records. AWS Lambda@Edge functions execute at edge locations, providing fast and reliable performance for requests and queries.

Technical details of Lambda@Edge include the following:

- Lambda@Edge functions are written in JavaScript using the Node.js runtime.
- Lambda@Edge can be triggered in response to four different types of CloudFront events: viewer request, viewer response, origin request, and origin response.
- Lambda@Edge is executed at the edge location, which provides faster response times.
- Lambda@Edge executed in the context of a specific CloudFront distribution can access information about request and response details, such as request headers and cookies.

## Stateful Versus Stateless Application Design

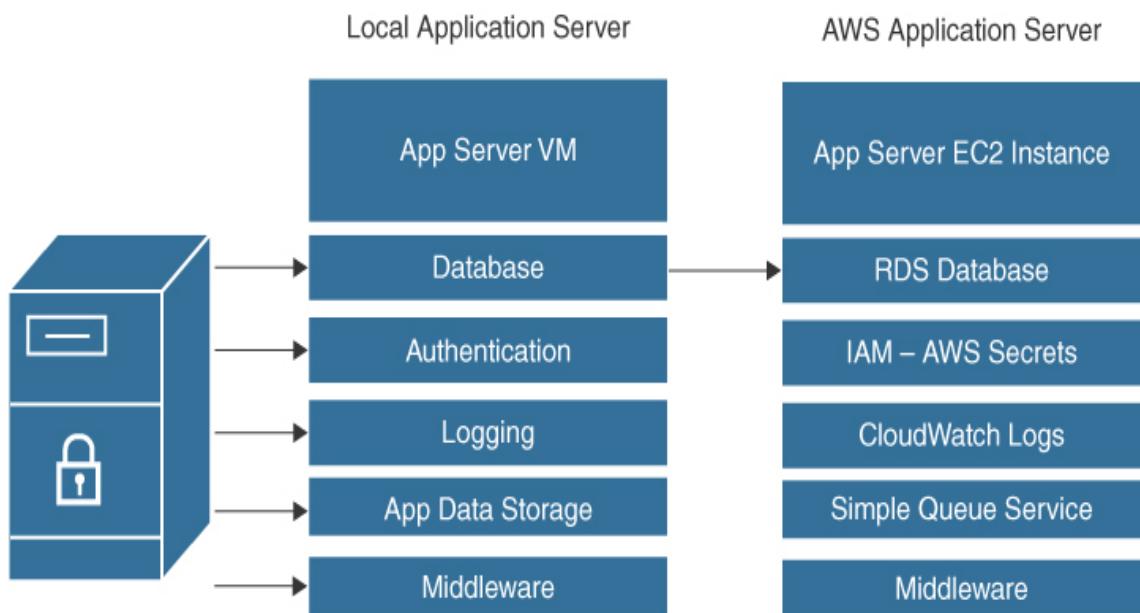


Before the popularity of the public cloud, the application stack was stored locally on one server that contained application data and the end user accounts. The application also maintained a data store, to store information about each user session. The local server also had local logging services, and middleware required to support and run the application. Application performance was limited by the size and speed of the physical hardware components (CPU, RAM, and hard drive). This design of the application stack was **stateful**—all required application components are running on one server in the customer's local data center.

A stateless application design utilizes a software architecture where the application does not maintain information about user accounts or sessions.

When older applications are moved to the cloud, migration tools lift and shift on-premises applications currently running on local servers to AWS. An application server rehosted at AWS will have local dependencies replaced with cloud services, as

illustrated in [Figure 6-2](#). Due to the application integration cloud services available at AWS, stateful features are easily replicated in the cloud using highly available cloud services.

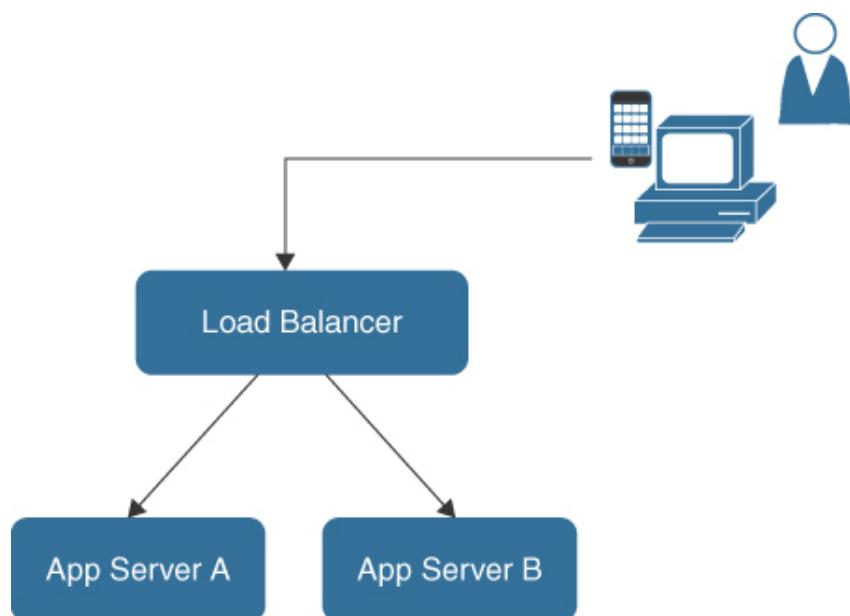


**Figure 6-2** Locally Hosted Application Server Versus AWS Hosted Server

When a URL is clicked for an application hosted at AWS, end-user requests are typically sent to a load balancer and onto a targeted web or application server, as illustrated in [Figure 6-3](#). At AWS, the load balancer service (Elastic Load Balancing) is part of a massive regional server farm composed of thousands of load balancers. If one load balancer instance fails, another one takes its place instantaneously. In addition, a load balancer is deployed to each availability zone. Each associated AWS cloud service provides resiliency, durability, failover,

performance, and high availability to each hosted workload design. There are several advantages to using a stateless design for your application:

- **Scalability:** Because stateless web and application servers do not maintain user, session, or application data state, they can be easily scaled horizontally by simply adding more instances to handle additional traffic.
- **Fault tolerance:** Stateless workloads can continue to operate even if an individual instance fails. This can help improve the overall reliability and availability of the application.
- **Flexibility:** Stateless applications that don't maintain user and application state information can be deployed on any number of servers.



**Figure 6-3** Adding a Load Balancer Adds High Availability to the Workload

## Changing User State Location



A server that stores user session information is defined as *stateful*, because of the single location for storing the two data states—application data and user session data.

A properly designed workload hosted at AWS uses cloud services to store both the user and application data records as **stateless** data in separate highly available and resilient locations:

- The user authentication credentials could be stored in an AWS Identity and Access Management (IAM) database, AWS Managed Active Directory AD, or using the IAM Identity Center or Amazon Cognito. User account authentication information is stateful—credentials remain the same until changed by the end user.
- The application state (orders and requests, for example) is stored in a messaging queue, such as Amazon Simple Queue Service (SQS), discussed later in this chapter.

- The application data is stored in a database solution, with a primary and standby database design such as Amazon RDS or Amazon DynamoDB.
- User session information can be stored redundantly and durably in an Amazon ElastiCache for Memcached or Amazon ElastiCache for Redis in-memory key-value store.

Reviewing the communication, when ordering from [Amazon.com](#), the essential data generated is a combination of stateful and stateless data (see [Table 6-2](#)). Stateless data is retained only as long as it is required; for example, for the duration of the authenticated user session, listening to online music, or playing online games. A typical online ordering process requires the following data:

- **Stateful data:** Data includes user account information, purchases, history, refunds, games played, high scores, music listened to, or music downloaded.
- **Stateless data:** Data includes user session information, such as information on browsing for products, browsing for games, reviewing account information, or searching for music.

Where does the application workload need to store the data? If data needs to be stored permanently, it should be stored in a

database. From time to time, records will be updated and changed, but SQL and NoSQL databases are a persistent data store.

Data stored for a short or a long period of time but not permanently can be defined as stateless. Stateless data is discarded after it has been used; for example, user session information or items in a shopping cart. Review [Table 6-2](#) for the various stateless options that could be used by a SaaS application.



**Table 6-2** Workload Data Choices

| Type of Data Stored | Stateful or Stateless | AWS Service   |
|---------------------|-----------------------|---|
| User account data   | Stateful              | AWS Managed Active Directory (AD), Amazon Cognito, IAM Identity Center, IAM users and roles |

| Type of Data Stored      | Type      | AWS Service  |
|--------------------------|-----------|--|
| Session information data | Stateless | Amazon DynamoDB,<br>Amazon ElastiCache for Redis, Amazon ElastiCache for Memcached |
| Load balancer            | Stateless | ELB, sticky sessions, cookies  |
| Database queries         | Stateful  | Amazon RDS database/read replicas, Amazon DynamoDB and DAX                         |
| Application state data   | Stateless | Amazon SQS, Amazon MQ  |
| Event notification data  | Stateless | Amazon Simple Notification Service (SNS), AWS EventBridge                          |

---

## Note

For the AWS Certified Solutions Architect – Associate (SAA-C03) exam, it is important to understand the distinction between stateless and stateful data in the context of stateless and stateful applications and how they are different.

---

## User Session Management

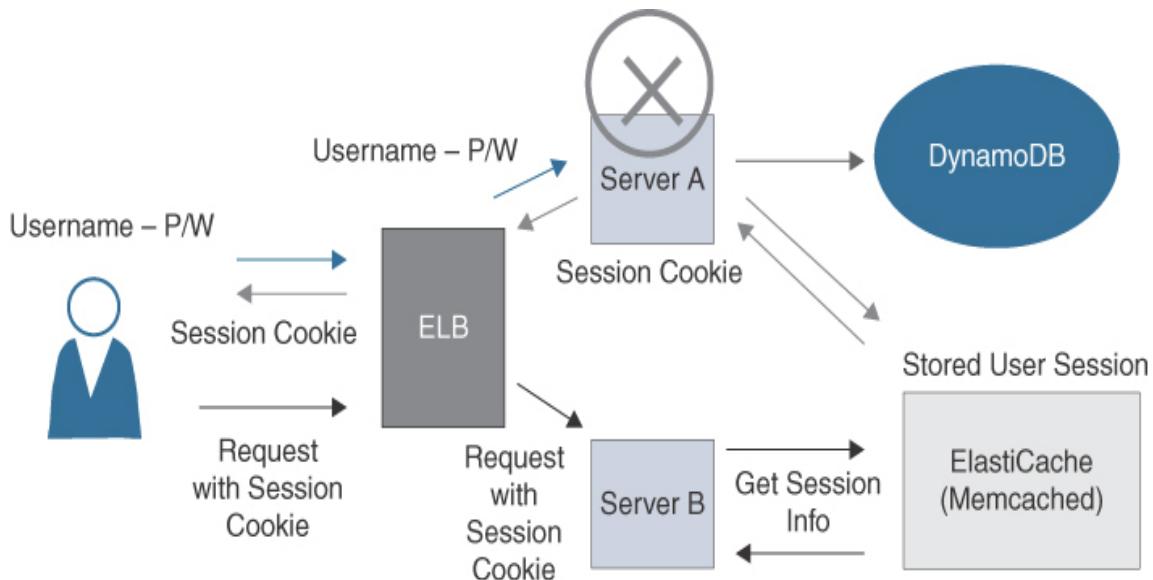


There are two common ways to manage AWS user sessions:

- **Sticky sessions**: When you deploy an application load balancer (ALB), you can enable sticky sessions by changing the default attributes on the target group of registered servers binding a user's session to a specific server. The term *sticky* means that once a user session is started with a specific application server, the session will continue with that application server until the user session has been completed. The drawback of sticky sessions is that if the application server fails, session information is lost. If an application creates its own session cookie, application-based stickiness

can be selected. If an application does not generate a session cookie, duration-based stickiness can be selected, generating a load balancer session cookie.

- **Distributed session management:** Another way to address shared data storage for user sessions is to use an in-memory key-value store hosted by Amazon ElastiCache deploying either ElastiCache for Redis or ElastiCache for Memcached caching the user session state. For a simple deployment with no redundancy, you could choose to employ ElastiCache for Memcached, but this solution provides no replication of the in-memory nodes. [Figure 6-4](#) shows the operation of a distributed cache with no built-in replication of the user cache. When an end user communicates with an application, the user session information is stored in an Amazon ElastiCache for Memcached cache. When Server A fails, the user session is continued on Server B because the user session information is stored in ElastiCache for Memcached instead of on an application server. For a redundant [\*\*\*distributed session\*\*\*](#) solution, you could deploy ElastiCache for Redis, which supports replicating user session information between multiple nodes across multiple availability zones (AZs), adding redundancy and durability to the cached user session information.



**Figure 6-4** An ElastiCache for Redis Distributed User Session Cache

### Note

The terms *stateless* and *stateful* have slightly different meanings when discussing network components such as security groups compared to network access control lists (NACLs). A security group is a firewall that controls each EC2 instance's incoming network traffic. Inbound rules control the incoming traffic, and outbound rules control the outgoing traffic. Any traffic allowed in by a security group is allowed back out. Security groups remember the incoming and outgoing network traffic flow state; they are stateful. In contrast, a NACL operates with a stateless mindset. An NACL is

a subnet firewall that either allows or denies incoming and outgoing requests at the subnet level. The NACL decision-making process about what traffic is allowed in or out is not dependent on what traffic was previously allowed (in or out). Incoming and outgoing traffic decisions are determined solely by the independent inbound and outbound allow and deny rules.

---

## Container Orchestration



Orchestration of container workloads at AWS can be carried out by an ever-increasing number of orchestration options, including Amazon Elastic Container Service (ECS) and the Amazon Elastic Kubernetes Service (EKS) control and data plane open-source deployments.

- **Amazon EKS:** Kubernetes control and data plane instances are located across three AZs, ensuring high availability. Amazon EKS can detect and replace unhealthy control plane instances.

- **Amazon ECS:** Docker containers are deployed across multiple AZs within an AWS region.
- **AWS Fargate:** Provision and manage containerized applications that are hosted on either Docker or Kubernetes deployments.

Other options for deploying and managing containerized workloads at AWS include AWS Copilot, AWS App Runner, AWS Lightsail, AWS App2Container (A2C), and AWS Elastic Beanstalk, as outlined in [Table 6-3](#).

**Table 6-3** Orchestration Options at AWS

|                                | AWS Service | Use Case                              | Details                     |
|--------------------------------|-------------|---------------------------------------|-----------------------------|
| <b>Container Orchestration</b> | Amazon ECS  | Docker applications or micro-services | Hybrid deployment and scale |

|                                | AWS Service | Use Case                              | Details                     |
|--------------------------------|-------------|---------------------------------------|-----------------------------|
| <b>Container Orchestration</b> | Amazon ECS  | Docker applications or micro-services | Hybrid deployment and scale |

| AWS Service                  | Use Case   | Details   |
|------------------------------|--|---|
| Amazon EKS                   | Manage Kubernetes container deployments (applications or micro-services) | Hybrid deployment and scale                             |
| <b>Compute Options</b>       | Fargate  | Manage ECS or EKS deployments                           |
| <b>Amazon EC2 instances</b>  |  | Manage container applications not infrastructure at AWS |
| Containers with full control | Bare-metal deployment  | AWS, AWS Outposts                                       |

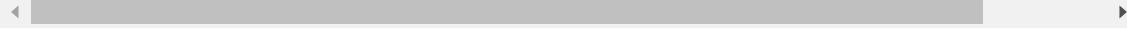
|   | AWS Service                                      | Use Case                               | Details                    |
|---|--|--|----------------------------|
| Container                               | AWS  | Manage                                 | CLI control                |
| Tools                                   | Copilot  | containerized applications             | deployment ECS and Fargate |
| Amazon Elastic Container Registry (ECR) |  |  |                            |
|   | Storage and deploy containers                    | AWS hosted repository                  |                            |
| AWS App Mesh                            |  |  |                            |
|   | Hybrid application-level networking service mesh | Use with Fargate, ECS, EKS, Kubernetes | AWS Output                 |

| AWS Service   | Use Case                             | Details  |
|---------------|--------------------------------------|--|
| AWS Cloud Map | AWS cloud resource discovery service | Registers AWS services (databases, queues, microservices) with custom names; resource location is checked and ensured the location is date |

|            |  |   |
|------------|--|---|
| AWS Lambda | Integrate with many AWS services with custom functions | Package Lambda functions into container application |
|------------|--|---|

| AWS Service         | Use Case   | Details   |
|---------------------|--|---|
| AWS App Runner      | Run containerized web apps at AWS                | Infrastructure and container orchestration fully hidden |
| Amazon ECS Anywhere | Run containers on customer-managed hardware      | AWS supported hybrid deployment                         |
| Amazon EKS Anywhere | Operate Kubernetes clusters on customer hardware | AWS supported hybrid deployment                         |

| AWS Service              | Use Case   | Details   |
|--------------------------|--|---|
| AWS Proton               | Self-service portal for platform team infrastructure deployment tool templates | Deploy infrastructure as-code to CloudFormation or Terraform tool templates |
| Amazon Elastic Beanstalk | Deploy the app and AWS infrastructure  | EC2 instances or Docker containers  |



## Migrating Applications to Containers

AWS App2 Container (A2C) transforms existing applications running on-premises VMs or EC2 instances into containers.

Applications supported include

- ASP.NET web apps running on Windows
- Java applications running on Linux JBoss and Apache Tomcat

A2C can create the containers using the following AWS services:

- ECS task definitions and Kubernetes deployment YAML files for integration with Amazon Elastic Container Registry, Elastic Container Service, and Elastic Kubernetes Service.
- CloudFormation templates to configure the required compute, network, and security infrastructure for the deployed containerized applications.
- Continuous integration/continuous delivery (CI/CD) pipelines for Amazon CodeBuild and CodeDeploy for building and deploying container builds.

## Resilient Storage Options



The default for all AWS storage services is built on failover and resiliency. Additional storage resiliency can be created using the respective features of each storage service or the AWS utilities listed in [Table 6-4](#).

**Table 6-4** Resilient Storage Options

| Storage Service | Resiliency | Additional Resiliency |
|-----------------|------------|-----------------------|
|                 |            |                       |

| Storage Service | Resiliency   | Additional Resiliency   |
|-----------------|--|---|
| Amazon EBS      | Stored within AZ   | EBS volumes copied across regions (AWS DataSync)<br>Snapshots copied across regions (Amazon Data Lifecycle Manager)<br>Multi-attach EBS volumes (io1) |
| Amazon EFS      | Multi-AZ deployment<br>Sync data across region/on-prem;<br>back up with AWS Backup | Transfer across AWS storage services (AWS DataSync), AWS Backup   |

| Storage Service                    | Resiliency                  | Additional Resiliency                               |
|------------------------------------|-----------------------------|---|
| Amazon FSx for Windows File Server | Multi-AZ deployment         | AWS DataSync, AWS Backup                            |
| Amazon RDS                         | Single AZ                   | Multi-AZ, cluster (three AZs), read replicas        |
| Amazon Aurora                      | Six copies across three AZs | Global Database, read replicas                      |
| Amazon DynamoDB                    | Six copies across three AZs | Global Tables, DynamoDB Accelerator (DAX)           |
| Amazon S3                          | Across three AZs minimum    | Single-region replication, cross-region replication |

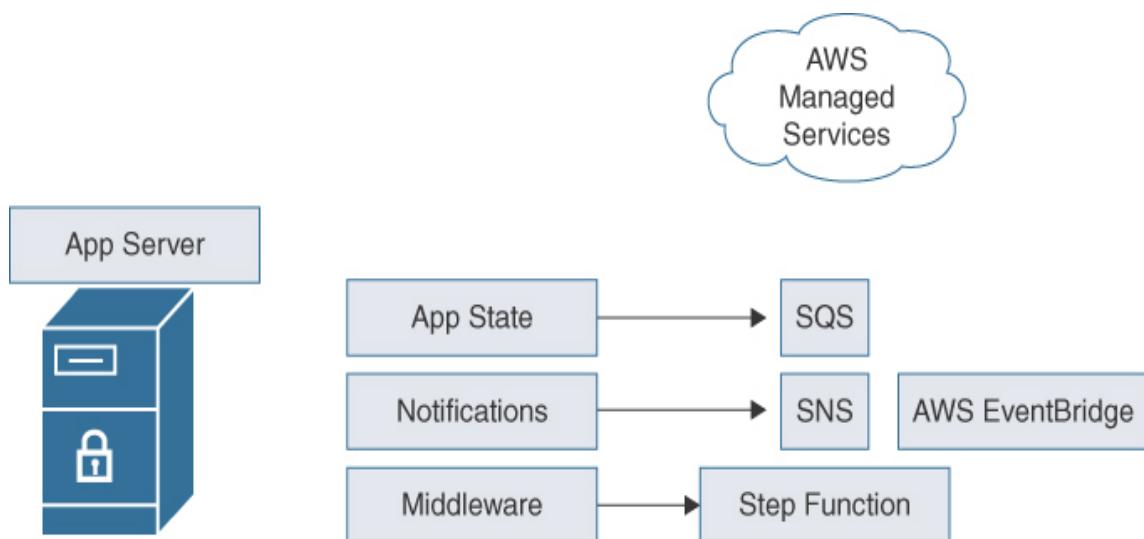
| Storage Service              | Resiliency  | Additional Resiliency                                 |
|------------------------------|---|---|
| Amazon Kinesis Data Streams  | Multi-AZ  | S3 storage  |
| Amazon Kinesis Data Firehose | S3, Amazon Redshift                                 | Analyze using Athena, EMR, and Redshift Spectrum      |
| Amazon Redshift              | Multi-AZ  | Continual backup to S3                                |
| Amazon SQS                   | Multi-AZ, 1 minute to 14 days retention of messages | Dead letter queue, backup to S3 using Lambda function |
| Amazon SNS                   | Multi-AZ, push notifications                        | DynamoDB  |

## Application Integration Services

The sections that follow cover the following application integration services:

- Amazon SNS
- Amazon SQS
- AWS Step Functions
- Amazon EventBridge

For the AWS Certified Solutions Architect – Associate (SAA-C03) exam, you need to understand the concepts of AWS application integration services (see [Figure 6-5](#)) that store stateless data for workloads.



**Figure 6-5** Data Store Options at AWS

## Amazon Simple Notification Service



Amazon Simple Notification Service (SNS) enables applications, end users, and devices to send, store, and receive notifications from different applications, services, and servers. SNS has four main integrated components:

- **Publishers**, applications, or AWS services send messages to access points called topics.
- **Messages** can be application-to-application messaging for the following subscribers: Kinesis Data Firehose delivery streams, Lambda functions, SQS queues, and HTTP/S endpoints.

Messages can also be application-to-person push notifications for mobile applications, phone numbers, and email addresses.

- **Topics** are the access points to which publishers send messages asynchronously.
- **Clients** subscribe to SNS topics to receive published messages.

Amazon SNS is integrated with Amazon CloudWatch. Utilizing SNS notifications as part of your workload design lets you decouple your application communications and react when changes occur to workloads or associated AWS services; for example, changes in a DynamoDB table being monitored by CloudWatch can trigger an alarm when data values increase or decrease.

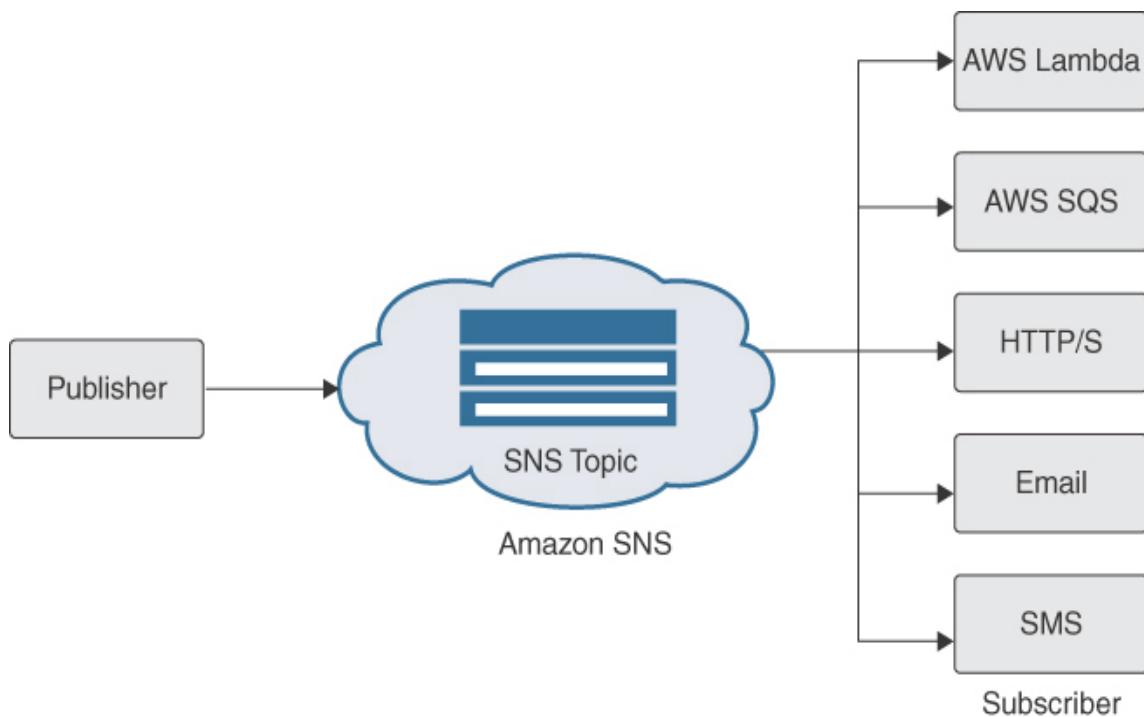
SNS messages are redundantly stored across multiple servers in each region. Many AWS cloud services can communicate with SNS topics as a publisher, sending messages and notifications to SNS topics using CloudWatch alarms, Amazon EventBridge, or Amazon Pinpoint.

To receive notifications from Amazon SNS, the appropriate service or end user subscribes to the desired SNS topic. Each SNS topic can have a choice of subscribers, as shown in [Figure 6-6](#), including these subscribers:

- **AWS Lambda:** An SNS notification can be linked to a function to carry out one or more tasks at AWS
- **Amazon SQS:** Queues can notify SNS topics that messages have been delivered
- **Amazon Kinesis Data Firehose:** Capture and upload data into Amazon S3, Amazon Redshift, or Elasticsearch data

stores for further analysis

- **HTTP/S endpoints:** Deliver SNS notifications to a specific URL
- **Email:** Email subscribers using push notifications



**Figure 6-6** SNS Publisher and Subscriber Options

Amazon SNS can be used to send **event notifications** when AWS service failures occur. Event updates and notifications concerning inventory changes or shipment status can also be immediately delivered to subscribers. For example, when you order a product online, many SNS notifications are executed in the background to complete the ordering process:

**Step 1.** You order a new set of dishes.

**Step 2.** A notification is issued to check for stock.

**Step 3.** After confirmation of inventory, the order is placed against your account.

**Step 4.** Your credit card information is checked.

**Step 5.** Taxes and relevant fees are added to the order.

**Step 6.** Shipping is calculated.

**Step 7.** An email is sent to your email account or device, thanking you for the order.

To use Amazon SNS, first create a topic and then associate the topic to a specific event type. [Figure 6-7](#) illustrates the creation of a standard notification topic.

**Key Topic**

**Details**

Type [Info](#)  
Topic type cannot be modified after topic is created

FIFO (first-in, first-out)

- Strictly-preserved message ordering
- Exactly-once message delivery
- High throughput, up to 300 publishes/second
- Subscription protocols: SQS

Standard

- Best-effort message ordering
- At-least once message delivery
- Highest throughput in publishes/second
- Subscription protocols: SQS, Lambda, HTTP, SMS, email, mobile application endpoints

Name  
  
Maximum 256 characters. Can include alphanumeric characters, hyphens (-) and underscores (\_).

Display name - *optional*  
To use this topic with SMS subscriptions, enter a display name. Only the first 10 characters are displayed in an SMS message. [Info](#)  
  
Maximum 100 characters, including hyphens (-) and underscores (\_).

**Figure 6-7** Creating a Notification Topic



## Amazon SNS Cheat Sheet

For the AWS Certified Solutions Architect – Associate (SAA-C03) exam, you need to understand the following critical aspects of SNS:

- Amazon SNS provides push-based deliveries of messages.

- Messages are application-to-application or application-to-person.
- Application-to-application message delivery choices are HTTP/HTTPS via email, SQS queue endpoints, Kinesis Data Firehose, and AWS Lambda.
- Application-to-person message delivery choices are SMS, email, and push notifications.
- JSON is the supported data type for messages.
- Amazon SNS messages are stored redundantly across multiple AZs.

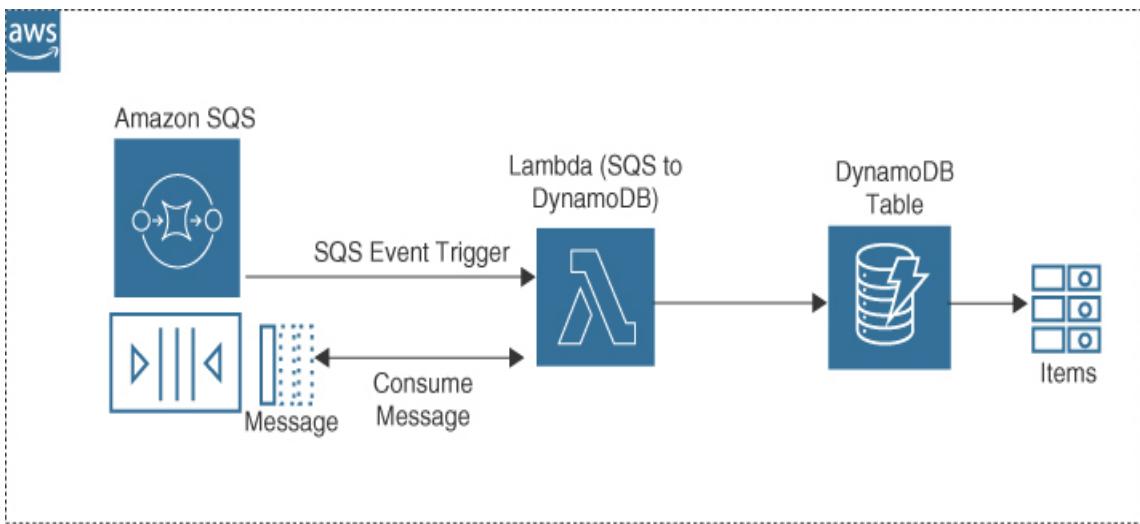
## Amazon Simple Queue Service

Amazon Simple Queue Service (SQS) is a fully managed message queue that allows developers to decouple communications between distributed workload components such as applications and microservices. Amazon SQS allows messages to be sent, stored, and received between applications and cloud services. Use cases include system-to-system messaging and request offloading. Messages can be stored and retrieved at any volume.

In Amazon SQS, a *message* is a discrete unit of data stored in SQS queues. SQS provides different options for sending and receiving messages, including the ability to send and receive

messages in batches, to specify the order in which messages are processed, and set the visibility timeout for them. SQS also supports the following features:

- Messages can be stored in SQS queues for up to 14 days; the default is 4 days.
- SQS messages can contain up to 256 KB of XML, JSON, and unformatted text.
- Messages larger than 256 KB can be sent to S3 buckets using the SQS Extended Client Library for Java.
- Applications can push messages into an SQS message queue, triggering a Lambda function that retrieves and stores the message into a storage service such as Amazon S3 or Amazon DynamoDB (see [Figure 6-8](#)).
- SQS queues can be used with applications hosted on EC2 instances and Elastic Container Services.
- All SQS message queues and messages are stored in a single AWS region across multiple AZs providing redundancy and failover.



**Figure 6-8** SQS and Triggered Lambda Function

## SQS Components

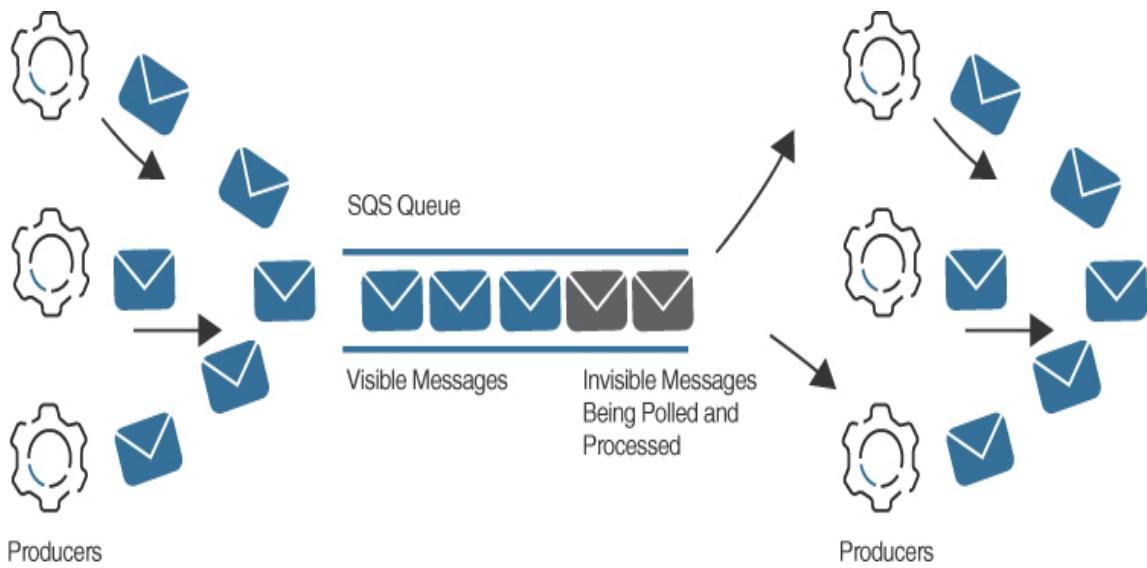


*Sender programs* (SQS-configured applications) send messages into a queue, and *receiver programs* (SQS-configured applications or services) retrieve messages from the queue for processing; each stateless program runs independently without any knowledge of each other. SQS messages are stored in SQS queues until they are processed by a receiving application or service. The SQS API enables developers to interact with SQS queues and messages using various programming languages and tools. The API provides a set of operations that can be used

to send, receive, and manage messages and queues. There are several main SQS components:

- **Standard queues:** This default queue type has the best-effort ordering of messages, and messages are delivered at least once. The order in which messages are received from a standard queue is not guaranteed to be the same as the order in which they were sent. Standard queues are suitable for a wide range of use cases, such as sending notifications, storing application state, or processing workloads in parallel.  
Standard queues are more cost-effective than FIFO queues.
- **FIFO (first-in, first-out) queues:** This queue type preserves the order in which messages are sent and received, and each message is delivered exactly once. FIFO queues should be used when the order of operations and events is important. Duplicate messages are not stored in a FIFO queue; instead, messages are grouped into distinct ordered bundles tagged with a message group ID. Messages are stored in the order in which they arrive and processed in order. FIFO queues are designed for use cases where message ordering and deduplication are important, such as financial transactions or communication between microservices. FIFO queues support a limited number of transactions per second and have higher costs compared to standard queues.

- **Message polling:** Applications can receive messages from an SQS queue using either the default short polling method or the long polling method. Communicating using a short polling method may return empty message responses. Long polling can help reduce the number of empty received message responses by allowing Amazon SQS to wait until messages are available in the queue before notifying a receiver program; the default polling wait time is 20 seconds for long polling.
- **Dead-letter queue (DLQ):** Messages can be stored in the optional dead-letter queue after the maximum number of processing attempts have completed successfully. Alarms linked to SNS notifications can be configured for any messages delivered to a DLQ. Messages can be moved from the DLQ back to the source queue for reprocessing.
- **Visibility timeout:** During the processing of each message, there is a period of time where a message being processed is not visible (see [Figure 6-9](#)). Once a message is successfully processed, a deletion request removes the processed message from the message queue.



**Figure 6-9** Visibility Timeouts

Amazon SQS can be used with the following AWS services:

- **Amazon DynamoDB:** You can use SQS to transfer messages to DynamoDB by using a Lambda function.
- **EC2 instances:** You can scale an Auto Scaling group out or in when messages in the SQS queue increase.
- **Amazon ECS:** Worker tasks within a container execute a script that polls for SQS messages and processes them as necessary.
- **Amazon RDS:** A lightweight daemon connects to an SQS queue and delivers messages to a SQL database.
- **AWS Lambda function:** SQS queues can be configured to trigger a Lambda function.

---

## Note

Amazon MQ supports several different messaging protocols, including JMS, NMS, and MQTT, and can be used to integrate different applications, services, and systems without having to rewrite the messaging code for migrated applications.

---

## Amazon SQS Cheat Sheet



For the AWS Certified Solutions Architect – Associate (SAA-C03) exam, you need to understand the following critical aspects of SQS:

- SQS uses pull-based polling.
- The visibility timeout is the amount of time a message is unavailable after a message is being processed.
- If a message cannot be processed within the visibility timeout period, the message becomes available in the queue for processing.
- Each message is deleted after it is processed successfully within the visibility timeout period.

- The maximum visibility timeout period is 12 hours.
- Queues can be either standard or FIFO.
- FIFO preserves the exact order in which messages are sent and received.
- The maximum size of SQS messages is 256 KB.
- Queues can be encrypted with server-side encryption (SSE) using keys managed by AWS Key Management Service (KMS).
- Notifications can be published to an SNS topic with multiple subscribers to SQS queues from different AWS accounts.

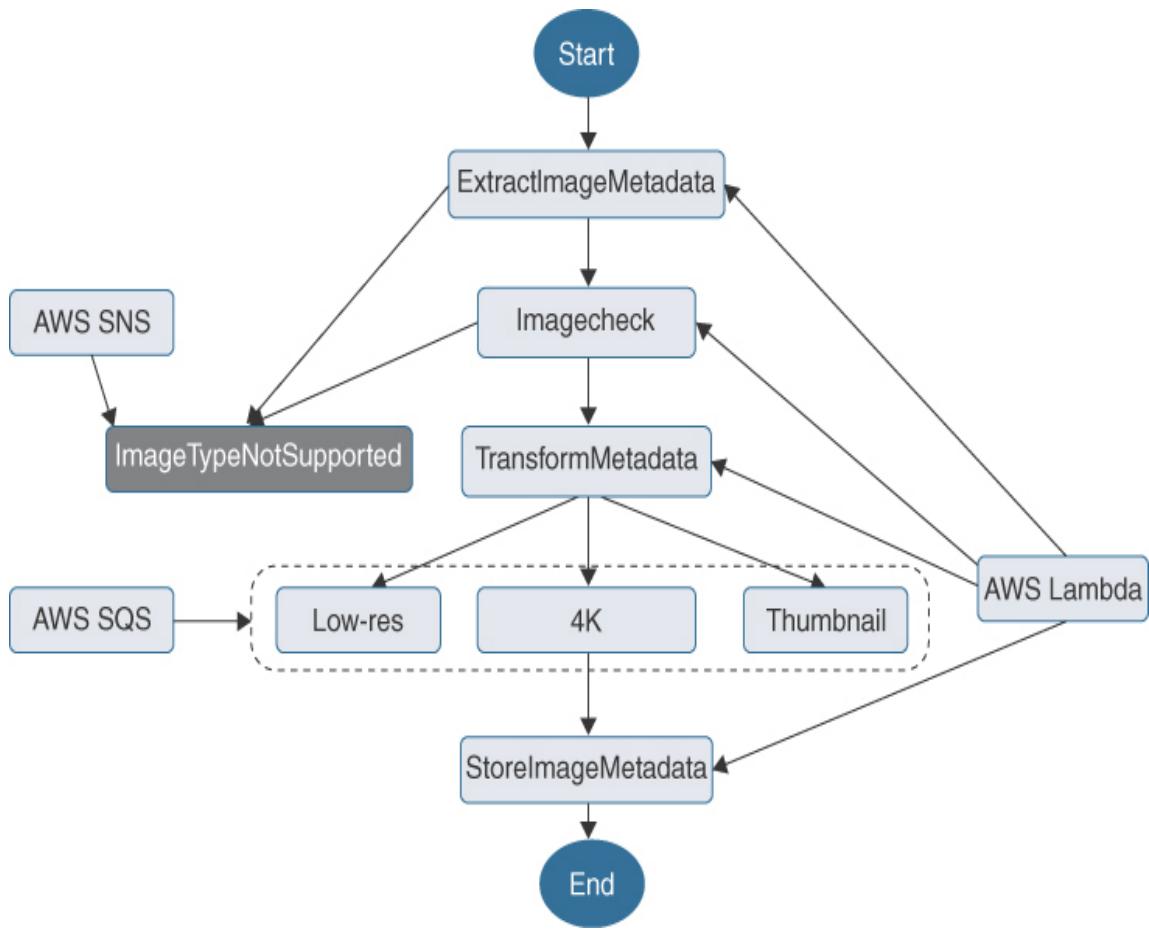
## AWS Step Functions

Step Functions provides a way to model and automate complex, multi-step processes and applications using a graphical workflow editor. AWS Step Functions is based on the concepts of tasks and state machines, and can track, monitor, and manage the execution of workflows. AWS Step Functions integrates with many other AWS services, including Amazon EC2, Amazon ECS, AWS Lambda, Amazon SQS, and Amazon SNS, and can be used to build scalable, reliable, and efficient applications and microservices. Step Functions enables you to orchestrate numerous Lambda functions and multiple AWS services into a serverless application process. Developers can create a *state machine* with multiple states. Each workflow has checkpoints that maintain each workflow throughout each

defined stage. The output of each stage in the workflow is the starting point of the next stage. Defined steps within each state function execute in the precise order defined by the logic. At each stage, decisions are made based on the supplied input parameters, actions are performed, and the output is passed to the next stage, as shown in [Figure 6-10](#).

AWS Step Functions has built-in application-level controls, including try/catch and retry and rollback capabilities, to help automatically deal with errors and exceptions. The following workflows are possible to design with AWS Step Functions:

- Consolidating data from multiple databases into a unified report
- Fulfilling orders or tracking inventory processes
- Implementing a user registration process
- Implementing a custom workflow



**Figure 6-10** Step Function Logic: Create a Workflow Using AWS SQS, AWS SNS, and AWS Lambda

Activity and Service tasks can be used with state machine step functions:

- **Activity tasks:** Allow you to assign a specific step in your defined workflow to external software code. The external function polls the function associated with any required work requests; when required, it performs its work and returns the results. The activity task can run on any

application that can make an HTTP connection from its location, such as an EC2 instance hosted at AWS, a mobile device, or an on-premises server.

- **Service tasks:** Allow you to connect steps in your defined workflow to a supported AWS service. In contrast to an activity task, a service task pushes requests to other AWS services; the service performs its action, reports to the workflow once the action is completed, and moves to the next step. Examples of service tasks include the following:
  - Running an Amazon ECS or Fargate task hosted in a VPC
  - Submitting an AWS Batch job and waiting for completion
  - Retrieving or placing a new item into an Amazon DynamoDB table

Step Functions integrate SNS notifications and SQS queues within a logical processing framework. The building blocks of Step Functions automate the relationship between SNS and SQS. SNS notifications can report when your workflow has completed. Workflow failure could trigger additional communications to developers indicating the problem and relevant error messages.

AWS Step Functions state machines are defined using JSON as the declarative language. You can create activity tasks by using

any AWS SDK that supports code written in Node.js, Python, Go, or C#. There are two choices for creating workflows:

- **Express workflows:** Use express workflows for workloads with high event rates of more than 100,000 per second and short durations of less than 5 minutes.
- **Standard workflows:** Use standard workflows for long-running, durable, and auditable workflows, such as machine learning models, generating reports, and the processing of credit cards. Standard workflows guarantee one execution of each workflow step, with a maximum duration of 1 year. Developers can inspect a processed workflow during and after the workflow execution has been completed.

[Table 6-5](#) compares the functions and use cases of SNS, SQS, and Step Functions.



**Table 6-5** SNS, SQS, and Step Functions

| Service | Function | Use Case |
|---------|----------|----------|
|         |          |          |

| Service        | Function                                 | Use Case  |
|----------------|--|---|
| SNS            | Sending notifications                    | Alert when services or applications have issues |
| Step Functions | Creating AWS service processing workflow | Visual workflow                                 |
| SQS            | Messaging queue                          | Decouple applications from application state    |

## Amazon EventBridge

Another AWS service that can react to changes in AWS services, AWS applications, or SaaS applications is Amazon EventBridge. The selected source event can be linked in real time to an AWS target, such as AWS Lambda, Simple Notification Service, and Kinesis Data Firehose (see [Figure 6-11](#)). EventBridge automatically ingests, filters, and sends each event to the selected target. Each event includes the source of the event, the

timestamp, and the AWS region. Events can be filtered by creating rules that match an incoming event to an event bus, routing the event to a specific target for processing.

EventBridge rules can customize events before they are delivered to a target by adding a custom response.

### Define rule detail Info

**Rule detail**

Name  
  
Maximum of 64 characters consisting of numbers, lower/upper case letters, .,-\_,~.

Description - optional

Event bus Info  
Select the event bus this rule applies to, either the default event bus or a custom or partner event bus.

Enable the rule on the selected event bus

Rule type Info

Rule with an event pattern  
A rule that runs when an event matches the defined event pattern. EventBridge sends the event to the specified target.

Schedule  
A rule that runs on a schedule

**Figure 6-11** Amazon EventBridge Setup

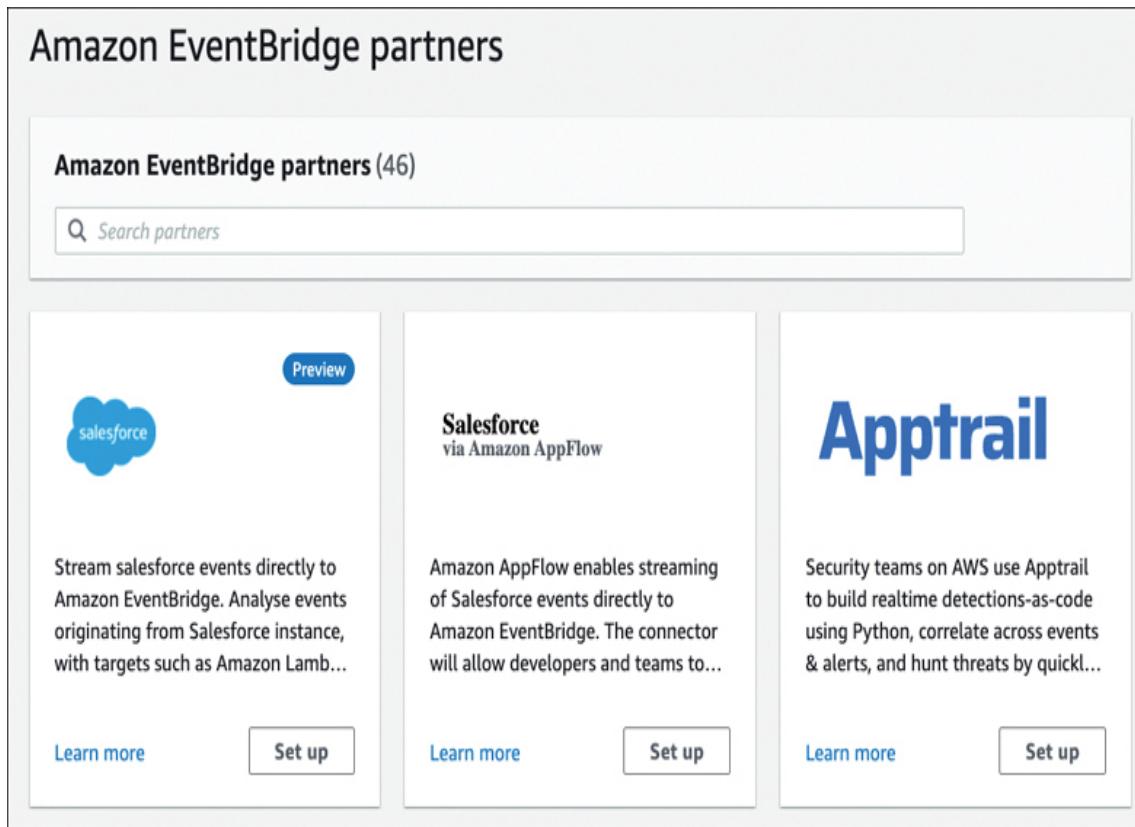
CloudWatch rules and events have been spun off into Amazon EventBridge. Both Amazon EventBridge and Amazon

CloudWatch use the same API and CloudWatch service infrastructure. However, Amazon EventBridge is recommended for ingesting data from SaaS applications due to Amazon EventBridge and the built-in SaaS integration with many popular SaaS applications.

Features of Amazon EventBridge include



- **Global endpoints:** Destinations for an Amazon EventBridge events can be replicated across primary and secondary regions for multi-region deployments.
- **API destinations:** Events can also be sent to on-premises SaaS applications for controlling application throughput and authentication.
- **Replay events:** Reprocess past events for analyzing application errors that have been fixed.
- **SaaS integration:** AWS EventBridge responds to events generated by well-known SaaS applications, including Shopify, Salesforce, SignalFx, and Zendesk (see [Figure 6-12](#)).
- **Event filtering:** Rules can match incoming events and route them to a specific target or AWS service for processing.
- **Targets:** Targets can be a single or multiple AWS accounts.



**Figure 6-12** SaaS Integration with EventBridge

## Amazon API Gateway

API Gateway is a service that makes it easy for developers to create, publish, maintain, and secure APIs that enable applications to access data and business logic from various AWS services, as well as from external sources. API Gateway provides features such as authentication and authorization, rate limiting, and caching, to help ensure that APIs are secure and scalable. It also provides tools for monitoring and analyzing API usage and performance. For a mobile application

hosted on a smartphone, the backend API or APIs for the application could be hosted at API Gateway. An **application programming interface (API)** is a set of rules and protocols for building software applications. Let's expand the definition of API a bit more:

- The *A*, for *application*, could be a custom function, the entire app, or something in between.
- The *P* is related to the type of *programming* language or platform that created the API.
- The *I* is for *interface*, and API Gateway interfaces with HTTP/REST APIs and/or WebSocket APIs. Java API types can direct HTTP requests across the private AWS network. Hosted APIs are exposed publicly using HTTPS endpoints.

APIs are commonly made available by third-party companies for use with mobile and web applications. One example is the API for Google Maps. When you book a hotel room using a mobile app, the app is likely using the Google API to call Google Maps with a location request. APIs can be thought of as software plug-ins that allow integration from one system to another. The Google API is the public frontend that communicates with the backend Google Maps application.

---

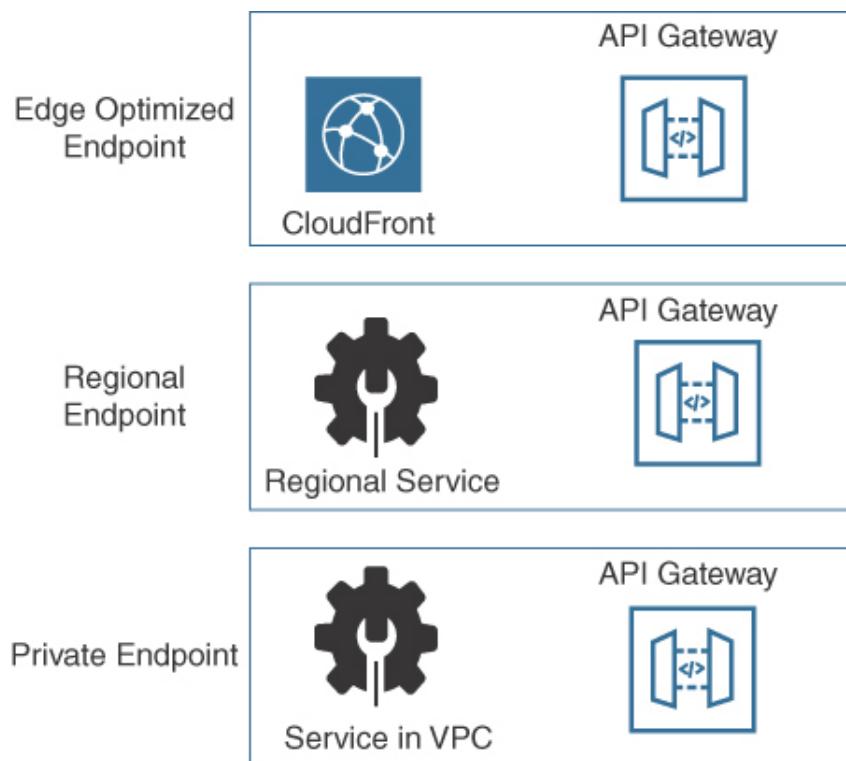
Note

For an older example, think of an EXE file, which is matched up with a library of DLLs. The library file contains a number of functions that, if called by the EXE file, would execute to carry out a task. If the EXE were a word processor like Word.exe, the associated DLLs would contain the code for calling the spell check routine or other features.

---

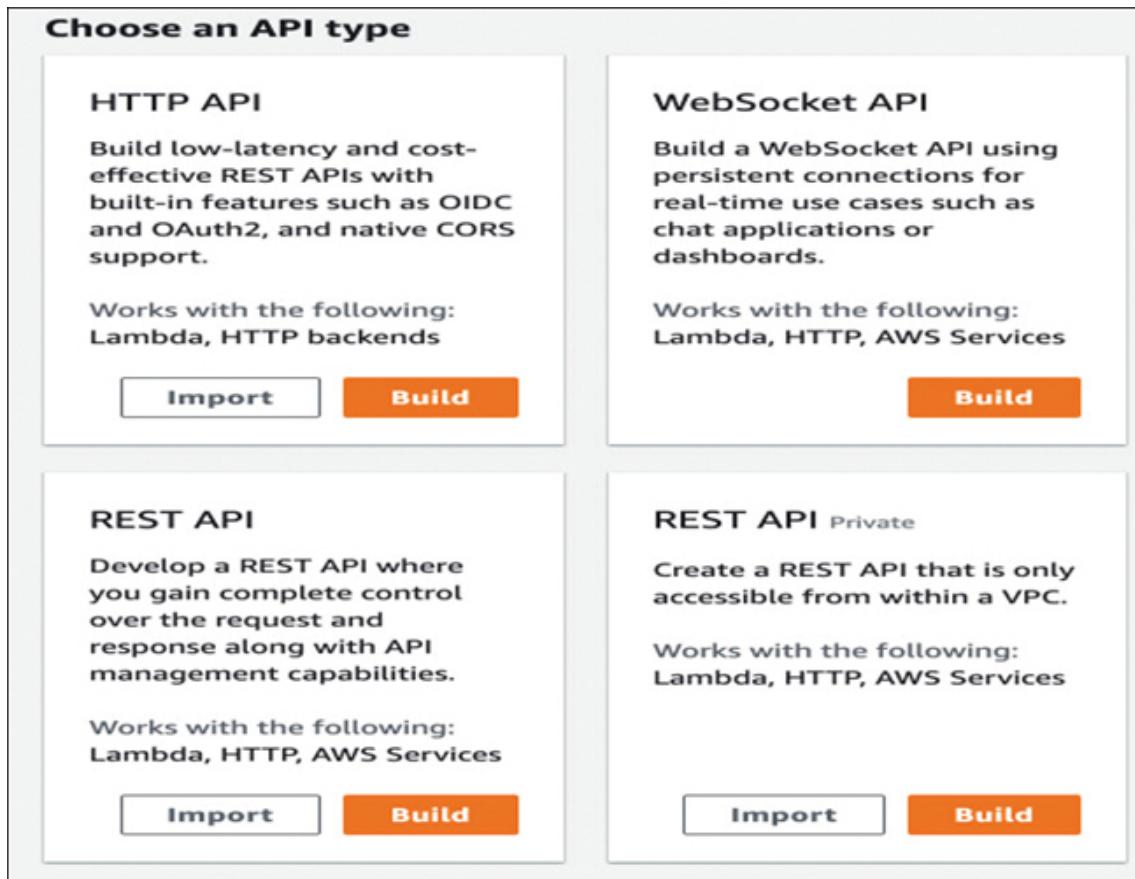
If you're programming applications to be hosted at AWS, you should consider hosting your applications' APIs using Amazon API Gateway. Think of API Gateway as the front door that, with authentication, allows entry to the AWS cloud where the selected AWS service resides. Several methods are available for communicating with API Gateway, including public communications through an edge location via Amazon CloudFront, a ***regional endpoint*** from a specific AWS region, or from a service or micro-service running on an EC2 instance hosted in a VPC using a private interface endpoint for communicating with the API Gateway service (see [Figure 6-13](#)).





**Figure 6-13** API Gateway Communication Options

API Gateway supports HTTP, REST, and WebSocket APIs (see [Figure 6-14](#)).



**Figure 6-14** Choosing the API Protocol to Use

---

### Note

APIs hosted at API Gateway can also call custom AWS Lambda functions, EC2 instances in your AWS account, and HTTP endpoints that access Elastic Beanstalk deployments.

---

**Key  
Topic**

Amazon API Gateway has the following features:

- **Security:** API Gateway supports AWS IAM and Amazon Cognito for authorizing API access.
- **Traffic throttling:** It is possible to cache API responses for incoming requests; cached responses can be answered for an API call with the same query. The number of requests each API can receive can be defined, with usage plans defining an API's allowed level of traffic.
- **Multi-version support:** Multiple API versions can be hosted by API Gateway.
- **Metering:** Metering allows you to throttle and control desired access levels to a hosted API.
- **Authorized access:** When an AWS API is called, API Gateway checks whether authentication is required before the task that the API has requested is carried out. Authentication options are an AWS Lambda authorizer or Amazon Cognito user pool (see [Figure 6-15](#)). API Gateway calls the selected authorizer, passing the incoming authorization token for verification. An Amazon Cognito user pool can be configured for authenticating the mobile application using various

methods, including single sign-on (SSO), OAuth, or an email address to access the backend application components.

## Authorizers

Authorizers enable you to control access to your APIs using Amazon Cognito User Pools or a Lambda function.

[+ Create New Authorizer](#)

### Create Authorizer

**Name \***  
Seminar\_Registration

**Type \* ⓘ**  
 Lambda  Cognito

**Cognito User Pool \* ⓘ**  
us-east-1  Corporate\_Mobile

**Token Source \* ⓘ**  
 **Token Validation ⓘ**  
 Please say that again

[Create](#) [Cancel](#)

**Figure 6-15** Selecting an Authorizer for the API Gateway

---

### Note

API Gateway can use client-side SSL certificates to verify that all requests made to your backend resources were sent by API Gateway.

---

## API Gateway Cheat Sheet



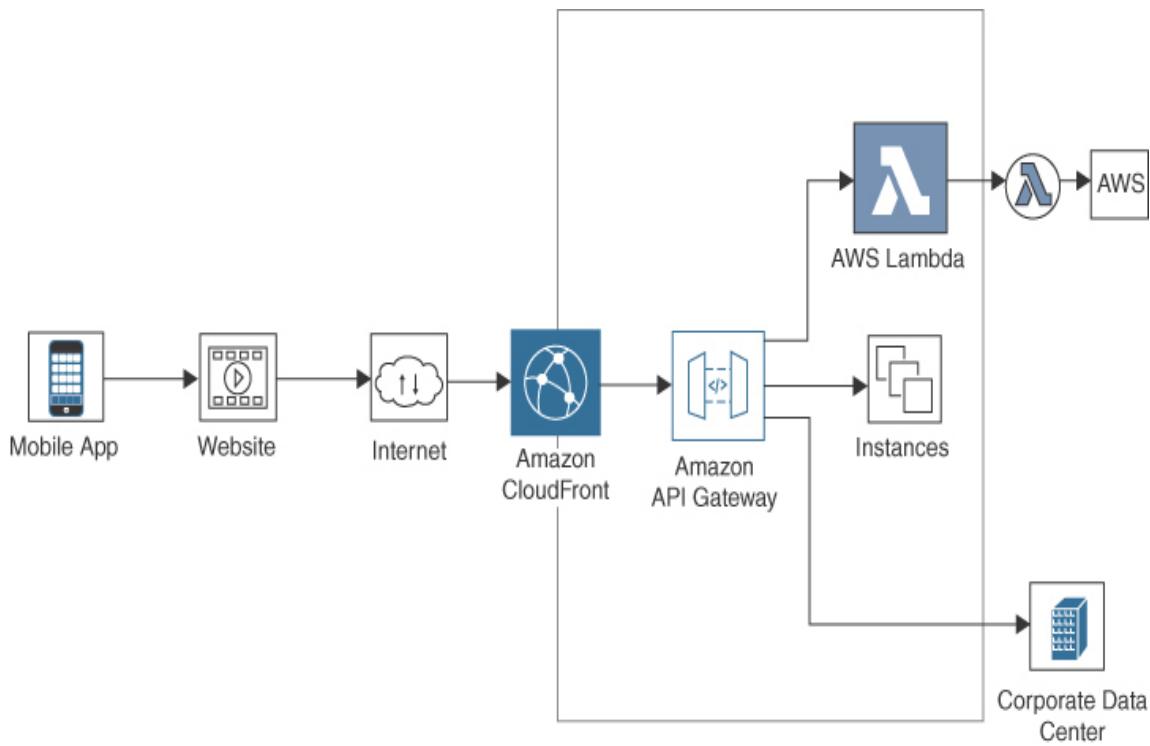
For the AWS Certified Solutions Architect – Associate (SAA-C03) exam, you need to understand the following critical aspects of APIs:

- With API Gateway, developers can publish, maintain, and secure APIs at any scale.
- API Gateway can process hundreds of thousands of concurrent API calls.
- API Gateway works together with Lambda to create application-facing serverless infrastructure.
- Amazon CloudFront distributions can be used as a public endpoint for API Gateway requests.
- Edge-optimized APIs can be used for clients in different geographic locations. API requests are routed to the closest edge location.
- Regional API endpoints can be used by clients located in the same AWS region.
- Private API endpoints can be accessed from a VPC using an interface VPC endpoint.
- API Gateway can scale to any traffic level required.

- API Gateway logs track performance metrics for the backend, including API calls, latency, and error rates.
- API Gateway only charges when your hosted APIs are called. Charges are based on the amount of data that is transferred outbound.
- API requests can be throttled to prevent overloading your backend services.

## Building a Serverless Web App

AWS Lambda and Amazon API Gateway can be used together to create a serverless application, such as an event website that allows users to register for a corporate function. For example, a simple web-based interface could allow users to register for a corporate function after registering as an attendee. [Figure 6-16](#) illustrates this scenario, and the following sections describe the development process in more detail.



**Figure 6-16** A Serverless Corporate Application

### Step 1: Create a Static Website

The first step in building the serverless web app just described is to create a website that can be hosted in an Amazon S3 bucket (see [Figure 6-17](#)). Because the website will be hosted in an S3 bucket, it must be a static website with no dynamic assets. After you configure the S3 bucket for website hosting, all the HTML, Cascading Style Sheets (CSS), images, and web server files can be uploaded and stored.

### Step 2: User Authentication

Create an Amazon Cognito user pool for the users registering for the conference (see [Figure 6-18](#)). Configure Cognito to send users who register on the conference website a standard confirmation email, including a verification code to confirm their identity. The corporate users will use their corporate email addresses to register as new users on the website.

**Static website hosting**  
Use this bucket to host a website or redirect requests. [Learn more](#)

Static website hosting  
 Disable  
 Enable

Hosting type  
 Host a static website  
    Use the bucket endpoint as the web address. [Learn more](#)  
 Redirect requests for an object  
    Redirect requests to another bucket or domain. [Learn more](#)

**Index document**  
Specify the home or default page of the website.

**Error document**  
This is returned when an error occurs.

**Figure 6-17** Using an S3 Bucket for Static Website Hosting

## Attribute verification and user account confirmation

Choose between Cognito-assisted and self-managed user attribute verification and account confirmation. Only verified attributes can be used for sign-in, account recovery, and MFA. A user account must be confirmed either by attribute verification, or user pool administrator confirmation, before a user is allowed to sign in.

### Cognito-assisted verification and confirmation [Info](#)

#### Allow Cognito to automatically send messages to verify and confirm - Recommended

Cognito sends a verification message with a code that the user must enter. For new users, this will verify the attribute and confirm their account. When this feature is not enabled, administrative API operations and Lambda triggers verify and confirm users.

#### Attributes to verify [Info](#)

Choose the user contact attribute that Cognito will send a verification message to. Recipient message and data rates apply when you use SMS.

##### Send SMS message, verify phone number

Verify with SMS to allow users to use their phone number for sign-in, MFA, and account recovery. SMS messages are charged separately by Amazon SNS.

##### Send email message, verify email address

Verify with email to allow users to use their email address for sign-in, MFA, and account recovery. Email messages are charged separately by Amazon SES.

##### Send SMS message if phone number is available, otherwise send email message

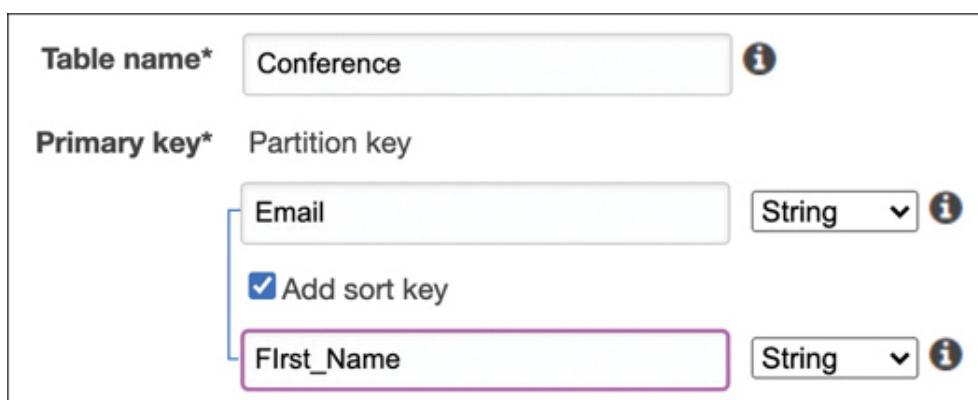
You must build custom code when you want to verify both email and phone numbers at user account creation.

**Figure 6-18** Creating an Authentication Pool Using Cognito

After the users have successfully signed in to the website, a JavaScript function communicates with Cognito, authenticating them using the Secure Remote Password (SRP) protocol and returning a web token that will be used to identify users as they request access to the conference.

## Step 3: Create the Serverless Backend Components

Create the Lambda function that registers users to the conference and sends them their attendance code. All registration requests are stored in a DynamoDB table (see [Figure 6-19](#)).



**Figure 6-19** Creating a DynamoDB Table

#### Step 4: Set Up the API Gateway

The hosted API allows registered users to register for the conference. Registered users have already been approved through registration and verification by being a member of the Cognito user pool. Each registration request invokes the Lambda function, which is securely called from the user's browser to carry out the registration as a RESTful API call to API Gateway (see [Figure 6-20](#)).

Create new API

In Amazon API Gateway, a REST API refers to a collection of resources and methods that can be invoked through HTTPS endpoints.

New API    Clone from existing API    Import from Swagger or Open API 3    Example API

### Settings

Choose a friendly name and description for your API.

|               |                         |
|---------------|-------------------------|
| API name*     | Conference              |
| Description   | Registration            |
| Endpoint Type | Regional <span>ⓘ</span> |

Please say that again

**Figure 6-20** Registering the RESTful API with the API Gateway

Representational State Transfer (REST) is a key authentication component of the AWS cloud, and RESTful APIs are the most common AWS API format. JavaScript runs in the background on user devices communicating with the publicly exposed API hosted by API Gateway carrying out each RESTful request. REST uses the following HTTP verbs to describe the type of each request:

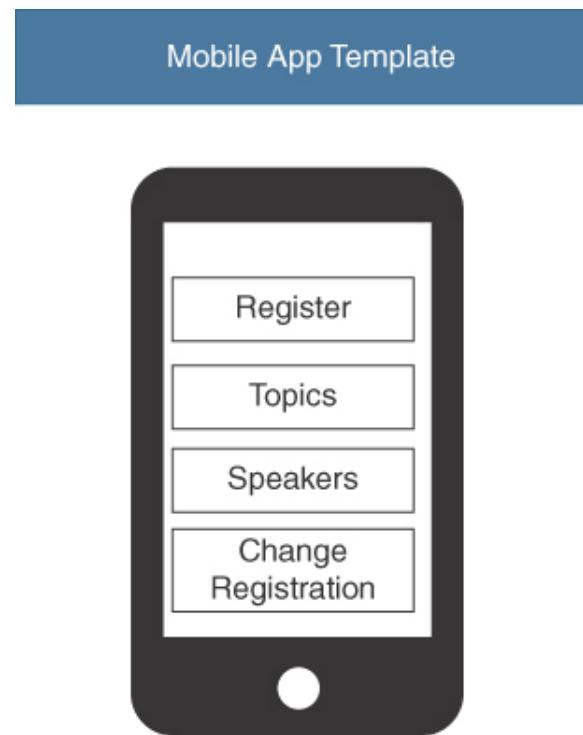
- **GET:** Request a record
- **PUT:** Update a record
- **POST:** Create a record
- **DELETE:** Delete a record

A user who types a URL into a browser sends a **GET** request. Submitting a request for the conference is a **POST** request.

RESTful communication is stateless, meaning that all the information needed to process a RESTful request is self-contained within the actual request; the server doesn't need additional information to process each request. The beauty of this design is that only Lambda is required to host the required functions for the application's logic and the application request that the user carries out.

### **Step 5: Register for the Conference**

The user sees none of the infrastructures described to this point; instead, they use the user interface to register for the conference, shown in [Figure 6-21](#).



**Figure 6-21** The Mobile Application on the User's Phone

## Automating AWS Infrastructure

*Automating infrastructure* refers to the use of tools and processes to manage and provision the underlying hardware and software infrastructure of an organization in a repeatable and efficient manner. This can include tasks such as provisioning and configuring servers, storage, and networking resources, deploying and updating applications and services, and managing and scaling infrastructure based on demand. Automating infrastructure can help organizations save time and reduce the potential for errors, and it enables them to

quickly and easily deploy new services and applications. Various tools and technologies are available for automating infrastructure, including infrastructure as code (IaC) tools, configuration management tools, and orchestration and deployment tools.

The one characteristic of AWS that stands above all others is the level of integrated automation used to deploy, manage, and recover AWS services. There is not a single AWS service offered that is not heavily automated for deployment and in its overall operation and management. When you order a virtual private network (VPN), it's created and available in seconds. If you order an Elastic Compute Cloud (EC2) instance through the AWS Management Console or the AWS command-line interface (CLI) tools, it's created and available in minutes. Automated processes provide the just-in-time response you want when you order cloud services.

AWS services are being changed, enhanced, and updated 24/7, with features and changes appearing daily. The AWS cloud is deployed and maintained using a combination of developer agility and automated processes.

AWS wasn't always so automated and controlled. In the early days, Amazon was a burgeoning online e-commerce bookseller.

In 2006, organizations could order a virtual machine from a cloud provider and wait several days for an email that the requested service was ready to go. Over time, Amazon developed rules for developers and the development process at AWS. One of the most important rules developers follow is this: “Each underlying service that supported the Amazon store must be managed using a core set of shared APIs available to all developers built and maintained on the common core of AWS compute and storage resources.”

Amazon developers built and continue to build its hosting environment using mandated internal processes, which can be described as a mixture of the following:

- **ITIL:** Information Technology Infrastructure Library (ITIL) is a framework of best practices for delivering IT services.
- **Scrum:** Scrum is a framework in which a development team works together to manage product development.
- **Agile:** Agile is a software development cycle in which developers plan, design, develop, test, and evaluate as a team with open communications.
- **DevOps:** DevOps is a continuation of the Agile framework that features full collaboration among the development and operations teams.

Many AWS developers work together effectively, making hundreds of changes to the AWS cloud every minute. In addition, all AWS services are being monitored, scaled, rebuilt, and logged through completely automated processes and services available to all organizations. Amazon avoids manual processes, and your long-term goal should be for your company to do the same.

Automation services will always manage your resources more effectively than you can manually. When resources are created using the AWS Management Console or the AWS CLI, the background automated processes finish the deployment and management of all AWS cloud resources.

There are several powerful tools in the AWS toolbox that can help you automate procedures:

- **AWS CloudFormation:** AWS CloudFormation enables developers to create, manage, and deploy infrastructure in the AWS cloud.
- **AWS Service Catalog:** AWS Service Catalog enables organizations to create, manage, and distribute a catalog of IT services that are approved for use on AWS built by CloudFormation templates.

- **AWS Elastic Beanstalk:** AWS Elastic Beanstalk enables developers to upload their application code and Elastic Beanstalk will automatically handle the deployment, scaling, and management of the underlying infrastructure.

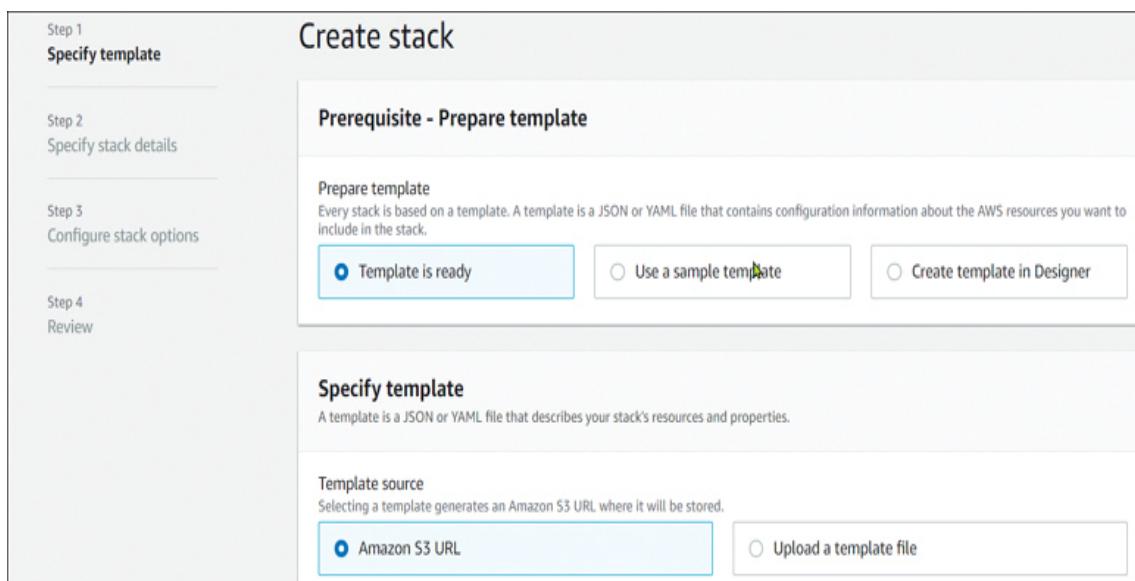
## AWS CloudFormation

Looking under the hood at any cloud service running at AWS, you'll find the deployment process largely driven by *JavaScript Object Notation (JSON)* scripts and AWS CloudFormation. AWS's extensive use of JSON is similar to Microsoft Azure's use of PowerShell. At AWS, JSON scripts are used internally for many tasks and processes; creating security policy with AWS IAM and working with AWS CloudFormation are two common examples.



AWS CloudFormation is an AWS-hosted orchestration engine that works with JSON and YAML templates to deploy AWS resources on demand and on predefined triggers (see [Figure 6-22](#)). AWS uses CloudFormation extensively, and AWS uses CloudFormation to manage the deployment of just about everything, including infrastructure workload deployments and updates. Each CloudFormation template declares the

infrastructure stack to be created, and the CloudFormation engine automatically deploys and links the needed resources together as described. Control variables can be added to each CloudFormation template to manage and control the precise order of the installation of resources. If mistakes are found in a CloudFormation script during deployment, all changes that have been carried out will be rolled back and reversed.



**Figure 6-22** The CloudFormation Console

Consider the following differences between the manual AWS deployment process and the automated CloudFormation process:

- **Time spent:** In the past, maintaining manual processes such as building computer systems and stacks provided job

security; these days, it's just not prudent to manually deploy production resources. Although every CloudFormation script takes time to craft, each CloudFormation deployment takes less time than a manual process; there are no wasted steps, and all steps are in the proper order. Over time, executing an automated process to build EC2 instances will save you hours or even weeks; the CloudFormation process runs in the background, allowing you to do something else.

CloudFormation can also perform updates and deletions for existing AWS resources and workloads.

- **Security issues:** Humans make mistakes, and mistakes made during manual changes can introduce huge security issues, particularly where the changes lack oversight.

CloudFormation templates can be secured and controlled for usage by specific IAM users and groups. Templates also carry out the same steps every time they are executed, preventing the fat-finger mistakes humans make. CloudFormation automation can also be locked down using a companion service called AWS Service Catalog, discussed later in this chapter, to ensure that only specific IAM users and groups can access and execute specific CloudFormation deployment tasks.

- **Documentation:** It is difficult to document manual processes if processes constantly change; CloudFormation templates

are readable, and when you get used to the format, they reveal themselves to be self-documenting. When you create a CloudFormation stack, you use a template that defines the collection of AWS resources to deploy. The template includes all the required information; names and settings of the resources and any dependencies between them. Later, when changes are made to the stack, such as adding or removing resources, CloudFormation automatically updates the stack using a change set and manages the underlying infrastructure changes accordingly, ensuring that your infrastructure is always in the desired state and that changes are made in a controlled and predictable manner.

- **Repeatability:** Repeating your manual steps not only is prone to error but also a waste of time and effort. With a CloudFormation template, you can deploy and redeploy the listed AWS resources in multiple environments, such as separate VPC development, staging, and production environments. Every time a CloudFormation template is executed, it repeats the same steps.

## CloudFormation Components



CloudFormation works with templates and change sets. A *CloudFormation template* is an AWS resource blueprint that can create a complete application stack or a simple stack, such as a VPC network complete with multiple subnets, Internet gateways, and NAT services, all automatically deployed and configured. You can also create a *change set template* to help visualize how proposed changes will affect AWS resources deployed by a CloudFormation template.

## CloudFormation Templates

Each CloudFormation template follows either JSON or YAML formatting standards. [Example 6-1](#) shows a CloudFormation template in JSON format deploying an EC2 instance, and [Example 6-2](#) displays the same information in YAML format. It's a matter of personal preference which format you use. When creating CloudFormation templates, you might find YAML easier to read.

### Example 6-1 CloudFormation Template in JSON Format

[Click here to view code image](#)

```
{  
  "AWSTemplateFormatVersion" : "2022-09-09",
```

```
"Description": "EC2 instance",
"Resources": {
    "EC2Instance" : {
        "Type" : "AWS::EC2::Instance",
        "Properties": {
            "ImageId" : "ami-0ff8a91497e77f667",
            "InstanceType" : "t1.micro"
        }
    }
}
}
```

## Example 6-2 CloudFormation Template in YAML Format

[Click here to view code image](#)

```
AWSTemplateFormatVersion: '2022-09-09'
Description: EC2 instance
Resources:
    EC2Instance:
        Type: AWS::EC2::Instance
        Properties:
            ImageId: ami-0ff8a91497e77f667
```

CloudFormation templates can have multiple sections, as shown in [Example 6-3](#). However, the only mandatory section is

Resources. Using the Metadata section for comments is highly recommended to ensure that templates can be understood when troubleshooting. As with any other template or script, the better the detailed comments within the script, the more usable a CloudFormation template is—for the author and other individuals.

### **Example 6-3** Valid Sections in a CloudFormation Template

[Click here to view code image](#)

```
"AWSTemplateFormatVersion": "version date",
"AWSTemplateFormatVersion": "2022-09-09"
<TemplateFormatVersion: Defines the current CF template>

"Description": "Here are the additional details about
and what it does",
<Description: Describes the template: must always follow
section>

"Metadata": {
    "Metadata" : {
        "Instances" : {"Description : "Details about the
        "Databases": {"Description: "Details about the dat
    }
},
}
```

```
<Metadata: Additional information about the resources  
by the template>
```

```
"Parameters": {  
    "InstanceTypeParameter" : {  
        "Type": "String",  
        "Default" : "t2.medium",  
        "AllowedValues" : ["t2.medium", "m5.large", "m5.xlarge"],  
        "Description" : "Enter t2.medium, m.5large, or  
        Default is t2.medium."  
    }  
}
```

```
<Parameters: Defines the AWS resource values allowed  
and used by your template>
```

```
"Mappings": {  
    "RegionMap" : [  
        "us-east-1" : { "HVM64" : "ami-0", "HVM32" : "ami-1" },  
        "us-west-1" : { "HVM64" : "ami-2", "HVM32" : "ami-3" },  
        "eu-west-1" : { "HVM64" : "ami-4", "HVM32" : "ami-5" },  
        "us-southeast-1" : { "HVM64" : "ami-6", "HVM32" : "ami-7" },  
        "us-northeast-1" : { "HVM64" : "ami-8", "HVM32" : "ami-9" }  
    ]  
}
```

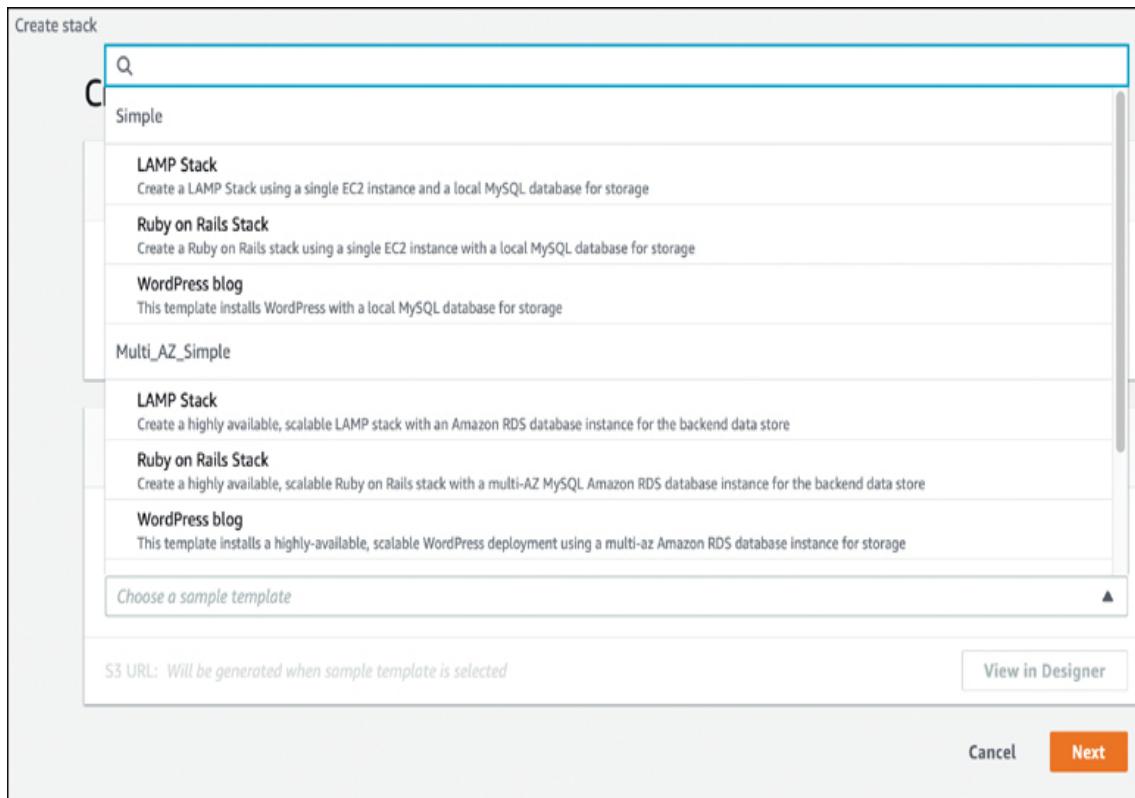
```
<Mappings: Defines conditional parameters defined by  
example, the AWS region and a set of AMI values to be  
used in the template>
```

```
"Conditions": {  
    "CreateTestResources": {"Fn::Equals" : [{"Ref" : "test"}]}  
},  
<Conditions: Defines dependencies between resources,  
order when resources are created or where resources are  
example, “test” deploys the stack in the test environ
```

## CloudFormation Stacks



AWS has many sample CloudFormation templates that you can download from online CloudFormation documentation, as shown in [Figure 6-23](#), and deploy at AWS. A CloudFormation stack can be as simple as a single VPC or as complex as a complete three-tier application stack with the required network high availability infrastructure and associated cloud services.



**Figure 6-23** AWS Sample Stacks and CloudFormation Templates

CloudFormation can be useful for deploying infrastructure at AWS, including the following areas:

- **Network:** Define a baseline template for developers to ensure that their VPC network deployment matches company policy.
- **Frontend infrastructure:** Deploy Internet gateways, associated route table entries, or load balancers into existing or new AWS network infrastructure.

- **Backend infrastructure:** Create database infrastructure, including primary and alternate database instances, subnet groups, and associated security groups.
- **Multi-tier application:** Using a multi-tier CloudFormation script allows you to handle failures or disasters by enabling you to rebuild a complete application stack with required network and infrastructure components or to launch the application stack in another AWS region.
- **Windows Server Active Directory:** Deploy Active Directory on a Microsoft Windows Server 2022 instance in a VPC for a custom Microsoft SQL Server deployment.
- **Demo applications:** Define an application stack for demonstrations, allowing the sales team or end users to quickly create a working AWS infrastructure environment.
- **AWS managed services:** Use CloudFormation templates to automate the setup of any AWS managed services. For example, you can enable and set up AWS Config or Amazon Inspector using a CloudFormation template.

---

#### Note

There are many standardized CloudFormation templates, called AWS Blueprints, available from AWS (see

<https://aws.amazon.com/solutions/implementations>

[/aws-blueprints/](#). AWS solution architects and trusted partners have built these templates to help you deploy complete solutions on AWS. These templates follow the current AWS best practices for security and high availability.

---

## Creating an EC2 Instance

[Example 6-4](#) provides a simple example that shows how to create an EC2 instance using a CloudFormation template. The template parameters are easily readable from top to bottom. Under Properties, the AMI ID, subnet ID, and EC2 instance type all must be present already in the AWS region where the template is executed; if not, the deployment will fail. The Ref statement is used in this template to attach the Elastic IP (EIP) address to the defined EC2 instance deployed and referenced under the resources listed as EC2 Machine.

---

### Note

Take a look at the AWS Quick Starts website <https://aws.amazon.com/quickstart/> to see CloudFormation deployment options.

---

## Example 6-4 CloudFormation Template for Creating an EC2 Instance

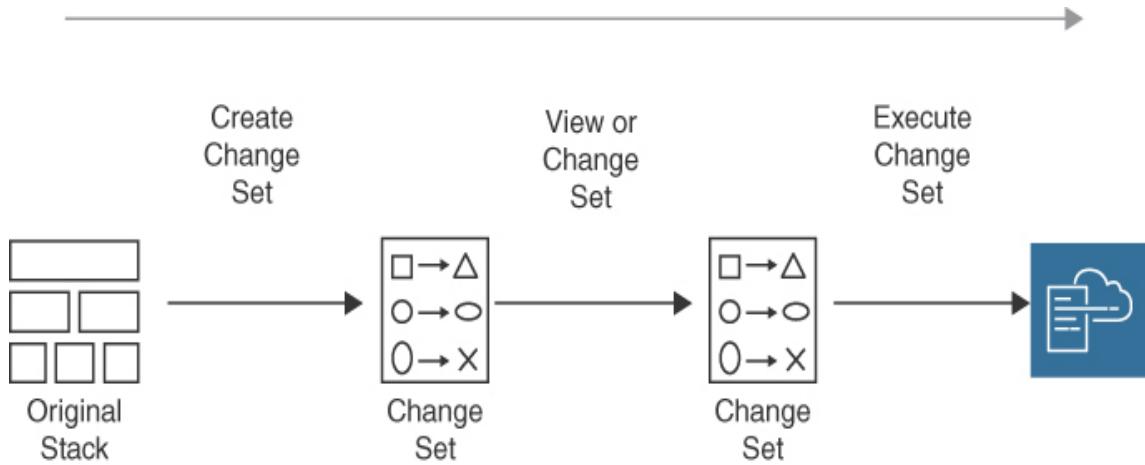
[Click here to view code image](#)

```
AWSTemplateFormatVersion: 2022-09-09
Description: EC2 Instance Template
"Resources": {
    "EC2Machine": {
        "Type": "AWS::EC2::Instance",
        "Properties": {
            "ImageId": "i-0ff407a7042afb0f0",
            "NetworkInterfaces": [{
                "DeviceIndex": "0",
                "DeleteOnTermination": "true",
                "SubnetId": "subnet-7c6dd651"
            }]
            "InstanceType": "t2.small"
        }
    }
},
"EIP": {
    "Type": "AWS::EC2::EIP",
    "Properties": {
        "Domain": "VPC"
    }
}
},
```

```
"VpcIPAssoc": {  
    "Type": "AWS::EC2::EIPAssociation",    "Properties": {  
        "InsanceID": {  
            "Ref": "EC2Machine"  
        },  
        "AllocationId": {  
            "Fn::GetAtt": ["EIP",  
                "AllocationId"]  
        }  
    }  
}
```

## Updating with Change Sets

A change set is a summary of changes that will be made to a stack in AWS CloudFormation. Change sets allow you to preview how your existing AWS resources will be modified when a deployed CloudFormation resource stack needs to be updated (see [Figure 6-24](#)). Select an original CloudFormation template to edit and input the desired set of changes. CloudFormation then analyzes your requested changes against the existing CloudFormation stack and produces a change set that you can review and approve or cancel.



**Figure 6-24** Using Change Sets with CloudFormation

Multiple change sets can be created for various comparison purposes. Once a change set is created, reviewed, and approved, CloudFormation updates your current resource stack. Several behaviors can be chosen when change set updates are approved:

- **Updates with No Interruption:** The resource is updated without disrupting the operation of the resource. For example, if you update any property on a CloudTrail resource, CloudFormation updates the trail without disruption.
- **Updates with Some Interruption:** The resource is updated with some interruption. For example, if you update certain properties on an EC2 instance resource, the instance might

have some interruption while CloudFormation and EC2 reconfigure changes to the instance.

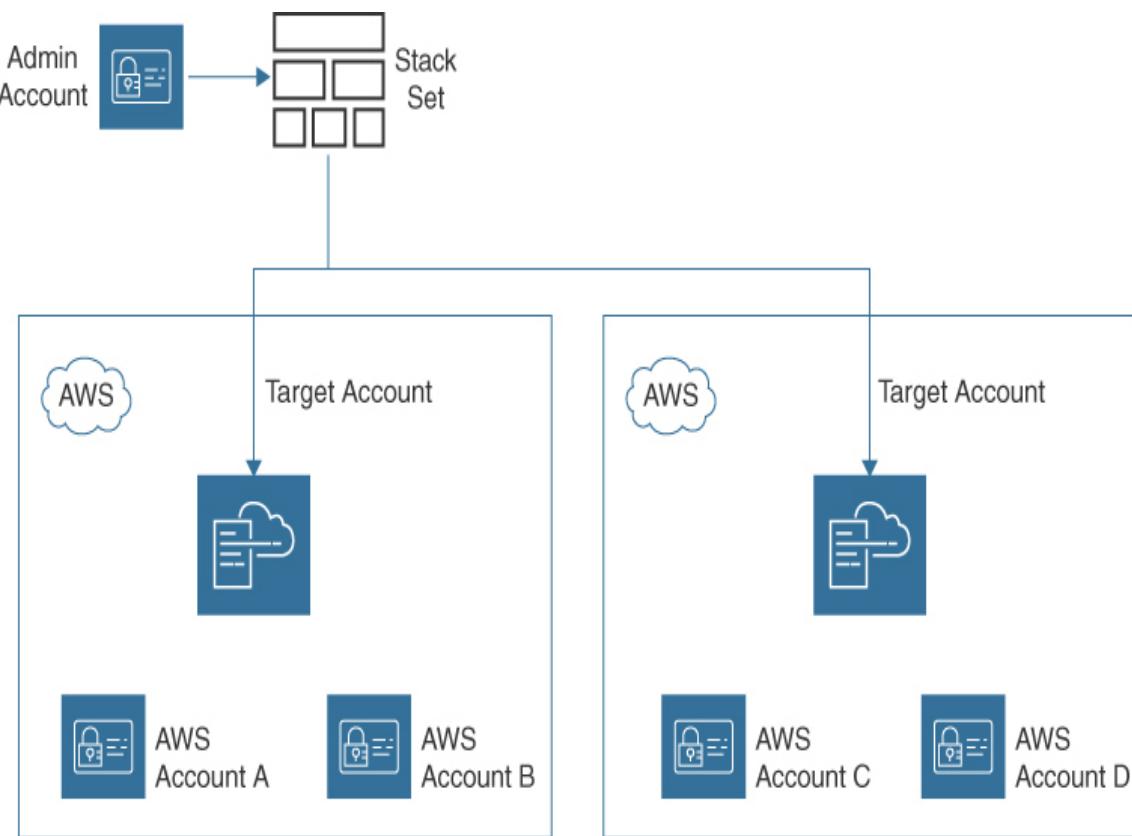
- **Replacement:** CloudFormation re-creates the resource during an update, which also generates a new physical ID. CloudFormation first creates the replacement resource, then changes references from other dependent resources to point to the replacement resource, and then the old resource is deleted. For example, if you update the Engine property of an Amazon RDS DBInstance resource type, CloudFormation creates a new resource and replaces the current DB instance resource with a new database instance.

## CloudFormation Stack Sets



A stack set enables you to create a single CloudFormation template to deploy, update, or delete AWS infrastructure across multiple AWS regions and AWS accounts. When a CloudFormation template is deploying infrastructure across multiple AWS accounts, as shown in [Figure 6-25](#), the AWS resources that the template references must be available in the AWS account and region. Region-specific resources must be copied to each AWS region where the CloudFormation template

is executed. For example, EC2 instances, EBS volumes, AMIs, and key pairs are always created in a specific AWS region. It is also important to review global resources such as S3 buckets created by the CloudFormation template to make sure there are no naming conflicts during creation, as global resources such as S3 bucket names must be unique across all AWS regions.



**Figure 6-25** A Stack Set with Two AWS Target Accounts

Once a stack set is updated, all instances of the stack created are also updated. All corresponding stack sets are deleted if a stack set is deleted.

A stack set is first created in a single AWS account. Before additional stack instances can be created from the primary stack set, trust relationships using IAM roles must be created between the initial AWS administrator account and the administrators in the AWS target accounts.

For testing purposes, one example available in the AWS CloudFormation console is a sample stack set that allows you to enable AWS Config across selected AWS regions or accounts.

## Third-Party Solutions

There are several third-party solutions, such as Chef, Puppet, Ansible, and TerraForm, for performing automated deployments of compute infrastructure. CloudFormation is not a replacement for these third-party products but can be a useful tool for building automated solutions for your AWS infrastructure if you don't use one of the third-party orchestration tools. AWS has a managed service called OpsWorks that comes in three flavors and might be useful to your deployments at AWS if your company currently uses Chef or Puppet:

- **AWS OpsWorks Stacks:** Manage applications and services hosted at AWS and on premises by running Chef recipes,

Bash, or PowerShell scripts.

- **AWS OpsWorks for Chef Automate:** Build a fully managed Chef Automate server that supports the latest versions of Chef server and Chef Automate, any community-based tools or cookbooks, and native Chef tools.
- **AWS OpsWorks for Puppet Enterprise:** A fully managed Puppet Enterprise environment that patches, updates, and backs up your existing Puppet environment and allows you to manage and administrate both Linux and Windows Server nodes hosted on EC2 instances and on premises.

## AWS Service Catalog

AWS Service Catalog enables organizations to create, manage, and distribute a catalog of approved IT services installed with CloudFormation templates (see [Figure 6-26](#)). Service Catalog can help organizations maintain control over their IT environments by providing a central repository for approved cloud services and ensuring that only authorized services are deployed. With AWS Service Catalog, administrators can create and manage catalogs of AWS and third-party services, making them available to users within the organization. Users can access the catalog and launch services while AWS Service Catalog ensures that they are launched in the correct AWS accounts and with the appropriate permissions. To control who gets to deploy

specific CloudFormation templates, AWS Service Catalog can be used to manage the distribution of CloudFormation templates portfolios to a single AWS account ID or an organizational unit contained within an AWS organization.

**Key Topic**

| Local portfolios (3)  |             |                                   |                    |  |              |                 |
|---|-------------|-----------------------------------|--------------------|--|--------------|-----------------|
| <input type="text"/> Search portfolios <span style="float: right;">Actions ▾</span> |             |                                   |                    |  |              |                 |
|   | Name ▾      | Created time ▾                    | Portfolio ID ▾     | ARN ▾  | Owner ▾      | Description ▾   |
| <input type="radio"/>   | Dev Group A | Sun, Aug 20, 2017, 3:40:51 PM EDT | port-qzjzxkg46mhtm | arn:aws:catalog:us-east-1:31385861:4000:portfolio/port-qzjzxkg46mhtm | mw           | RDS MySQL Stack |
| <input type="radio"/>   | Dev Group B | Tue, Feb 5, 2019, 6:46:14 PM EST  | port-dccemukxxtsgu | arn:aws:catalog:us-east-1:31385861:4000:portfolio/port-dccemukxxtsgu | Mark Wilkins | .Net Developers |

**Figure 6-26 Portfolios in Service Catalog**

When an approved product is selected, Service Catalog delivers a confirmation to CloudFormation, which executes the template and creates the product.

Each developer in an AWS account could be granted access to a Server Catalog portfolio of multiple approved products. Because products are built using common confirmation templates, any AWS infrastructure components, including EC2 instances and databases hosted privately in a VPC, can be deployed. Service Catalog also supports third-party products hosted in the AWS Marketplace, and software appliances are bundled with a CloudFormation template. VPC endpoints can access AWS Service Catalog and associated portfolios.

When you're creating a Service Catalog portfolio of products, you can use constraints with IAM roles to limit the level of administrative access to the resources contained in the stack being deployed (see [Figure 6-27](#)).

In addition, you can add rules that control any parameter values that developers enter during deployment. For example, you could mandate that specific subnets must be used for a stack deployment. You can also define rules that allow you to control which AWS account and region a product is allowed to launch.

Use CloudFormation and Service Catalog together to create a self-serve portal of portfolios and products.

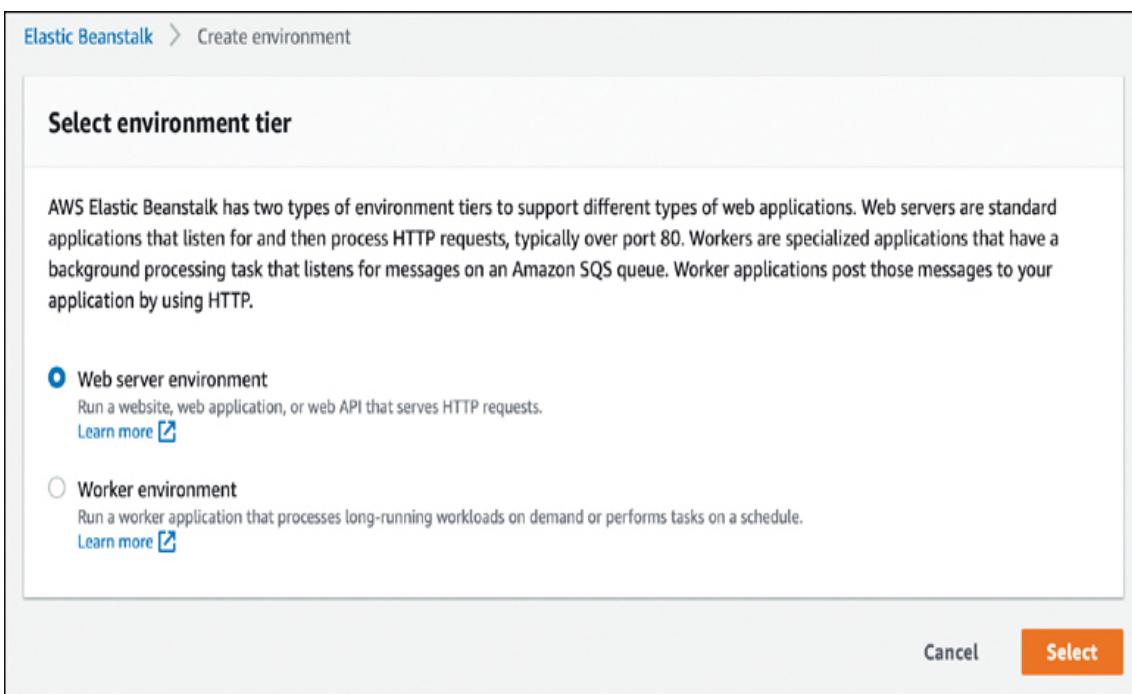
| Grant users access to the portfolio and allow them to view and launch its products. |  |       |
|---|--|-------|
| Groups  | Roles  | Users |
| <b>Groups (2/13)</b>  |  |       |
| <input type="text"/> <i>Search groups</i>   |  |       |
| Group name  | ARN  |       |
| <input type="checkbox"/> admin  | arn:aws:iam::313858614000:group/admin            |       |
| <input type="checkbox"/> admin-dev  | arn:aws:iam::313858614000:group/admin-dev        |       |
| <input type="checkbox"/> admincanada  | arn:aws:iam::313858614000:group/admincanada      |       |
| <input type="checkbox"/> auditors   | arn:aws:iam::313858614000:group/auditors         |       |
| <input checked="" type="checkbox"/> East_Coast_Admin                                | arn:aws:iam::313858614000:group/East_Coast_Admin |       |
| <input type="checkbox"/> managers   | arn:aws:iam::313858614000:group/managers         |       |

**Figure 6-27** IAM Group Constraints Controlled by Service Catalog

## AWS Elastic Beanstalk

When moving to the AWS cloud, developers typically have little time and budget but must develop a web application or migrate an existing web app into the AWS cloud while adhering to the company's compliance standards. The web application needs to be reliable, able to scale, and easy to update. In such situations, Elastic Beanstalk can be of some help.

Elastic Beanstalk, which has been around since 2011, was launched to help enable developers to easily deploy web applications hosted on AWS Linux and Windows EC2 instances in the AWS cloud. Elastic Beanstalk automates both application deployment, as shown in [Figure 6-28](#), and the infrastructure components required by the application, including single and multiple EC2 instances, load balancers, and EC2 Auto Scaling. Monitoring is carried out with CloudWatch metrics for monitoring the health of your application infrastructure. Elastic Beanstalk also integrates with AWS X-Ray, which can monitor and debug the internal operations of your hosted application.



**Figure 6-28** Elastic Beanstalk Creating Infrastructure

Elastic Beanstalk supports several development platforms, including Java (Apache HTTP or Tomcat) for PHP, Node.js (Nginx or Apache HTTP), Python (Apache HTTP), Ruby (Passenger), .NET (IIS), and the Go language. Elastic Beanstalk allows you to deploy different runtime environments across multiple technology stacks running on EC2 instances or Docker containers.

Developers can use Elastic Beanstalk to quickly deploy and test applications on defined infrastructure. After testing, the infrastructure can be quickly discarded at little cost. Keep in mind that Elastic Beanstalk is not a development environment (like Visual Studio). The application must be written before Elastic Beanstalk is useful. After an application has been written, create and upload a configuration file that details the infrastructure that needs to be built, and Elastic Beanstalk deploys the infrastructure and the application.

Elastic Beanstalk can help developers automate tasks and procedures previously carried out by administrators when applications were hosted in an on-premises data center. Elastic Beanstalk carries out the following tasks automatically:

- Provisions and configures EC2 instances, containers, and security groups using a CloudFormation template

- Configures an external database server environment
- Configures your load balancer and Auto Scaling
- Stores the application server's source code, associated logs, and artifacts in an S3 bucket
- Enables CloudWatch alarms that monitor the load of your application, triggering Auto Scaling for your infrastructure as necessary
- Routes access from the hosted application to a custom domain
- Performs blue/green deployments and immutable updates

Elastic Beanstalk is free of charge; you are charged only for the resources used for the deployment and hosting of your applications. The AWS resources that you use are provisioned within your AWS account, and you have full control of these resources. At any time, you can go into the Elastic Beanstalk configuration and make changes, as shown in [Figure 6-29](#).

| Configuration overview                                  |  | <input type="button" value="Cancel"/> | <input type="button" value="Review changes"/> | <input type="button" value="Apply configuration"/> |
|---|--|---------------------------------------|---|--|
|   |  | <input type="checkbox"/> Table View   |   |  |
| <input type="text"/> Search for an option name or value |  |                                       |   |  |
| Category  | Options  |                                       |   |  |
| Software  | Environment properties: GRADLE_HOME, JAVA_HOME, M2, M2_HOME, XRAY_ENABLED<br>Log streaming: disabled<br>Rotate logs: disabled<br>X-Ray daemon: enabled |                                       |   |  |
| Instances   | EC2 security groups:<br>IOPS: container default<br>Monitoring interval: 5 minute<br>Root volume type: container default<br>Size: container default     |                                       |   |  |
| Capacity  | AMI ID: ami-01b43d86b8d626b47<br>Environment type: single instance<br>Instance type: t1.micro<br>Scaling cooldown: 360 seconds<br>Time-based Scaling:  |                                       |   |  |

**Figure 6-29** Modifying the Capacity of the Elastic Beanstalk Application Infrastructure

Applications supported by Elastic Beanstalk include simple HTTPS web applications and applications with worker nodes that can subscribe to Amazon SQS queues to carry out more complex, longer-running processes.

After an application has been deployed by Elastic Beanstalk, you can automatically update the selected application platform environment by enabling managed platform updates, which can be deployed during a defined maintenance window. These updates include minor platform version updates and security patching but not major platform updates to the web services used; major updates must be initiated manually.

Database support for Elastic Beanstalk includes any application installed on an EC2 instance, RDS database options, and DynamoDB. A database can be provisioned by Elastic Beanstalk during launch or can be exposed to the application through environmental variables.

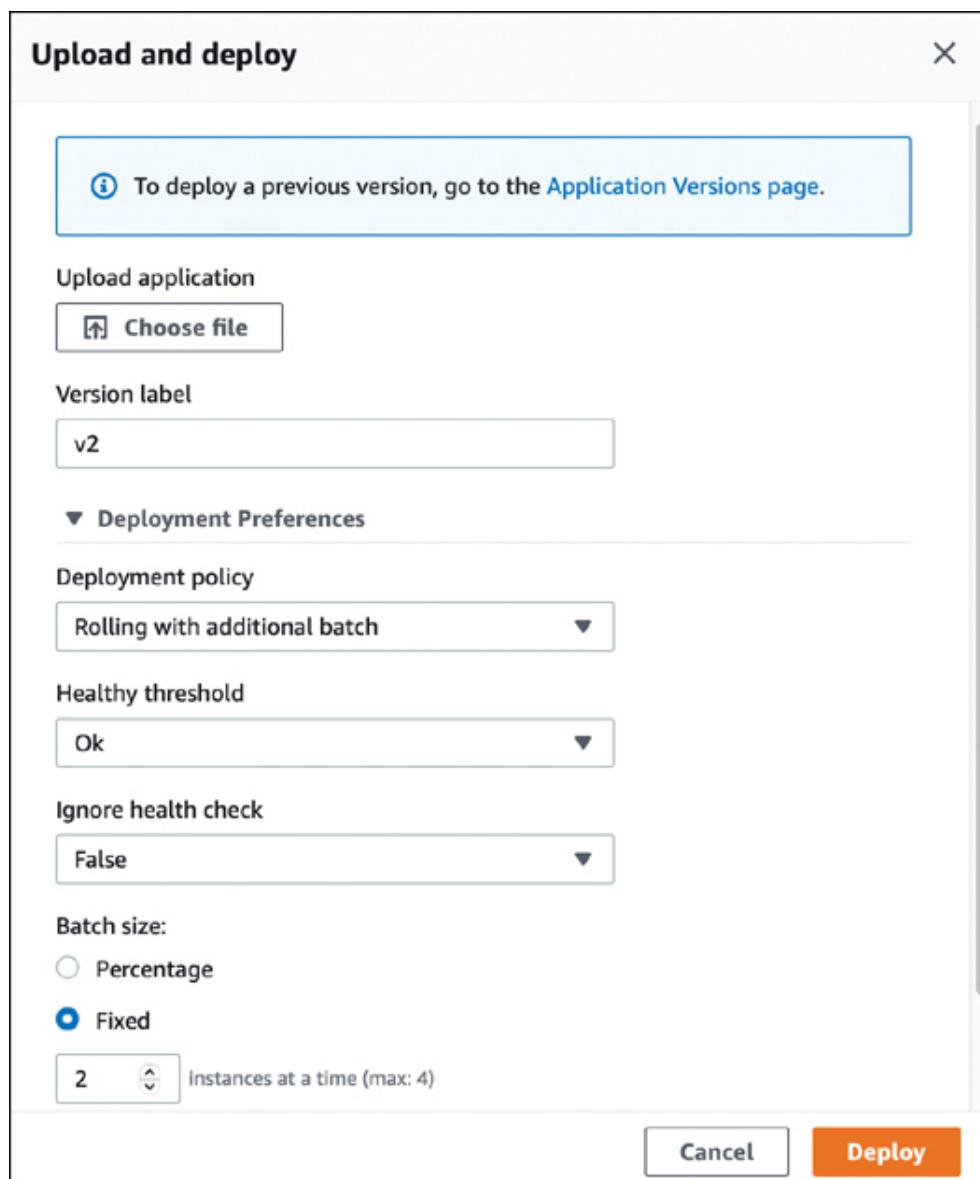
## Updating Elastic Beanstalk Applications



You can deploy new versions of the application to your Elastic Beanstalk environment, depending on the complexity of the application. During updates, Elastic Beanstalk archives the current application version in an S3 bucket. The methods available for updating Elastic Beanstalk applications include the following:

- **All at once:** The new application version is deployed to all EC2 instances simultaneously. The current application is unavailable while the deployment process is underway. To keep an older version of your application functioning until the new version is deployed, choose the immutable method or the blue/green update method.

- **Rolling:** The application is deployed in batches to a select number of EC2 instances defined in each batch configuration, as shown in [Figure 6-30](#). As the batches of EC2 instances are being updated, they are deregistered from the load balancer queue. When the update is successful and the instances pass load-balancing health checks, the instances are registered to the load-balancing target group again.
- **Immutable:** The application update is only installed on new EC2 instances contained in a second Auto Scaling group launched in your environment. Only after the new environment passes health checks is the old application version removed. The new application servers are made available all at once. Because new EC2 instances and Auto Scaling groups are being deployed, the immutable update process takes longer.



**Figure 6-30** Apply Rolling Updates to an Elastic Beanstalk Application

- **Blue/green:** The new version of the application is deployed to a separate environment. When the new environment is healthy, the CNAMEs of the two environments are swapped, so traffic is redirected to the new application version. In this

scenario, if a production database is to be used in the application design to maintain connectivity, the database must be installed separately from the Elastic Beanstalk deployment. Externally installed databases remain operational when the new Elastic Beanstalk application version is installed and swapped on the application's EC2 instances.

- **Traffic-splitting deployments:** Canary testing can also be included in your application deployment. Elastic Beanstalk can launch a set of new instances with a new version of the application; however, only a specific percentage of incoming traffic will initially be forwarded to the new application instances for a defined time. If the new application instances remain healthy during the evaluation period, Elastic Beanstalk then forwards traffic to the new instances and terminates the old instances. If the new instances don't pass their health checks, traffic is moved back to the current instances, and the new instances that were under evaluation are terminated, leaving the existing instances online with no service interruption.

## Exam Preparation Tasks

As mentioned in the section “[How to Use This Book](#)” in the Introduction, you have a couple of choices for exam

preparation: the exercises here, [Chapter 16](#), “[Final Preparation](#),” and the exam simulation questions in the Pearson Test Prep Software Online.

## Review All Key Topics

Review the most important topics in the chapter, noted with the Key Topic icon in the margin of the page. [Table 6-6](#) lists these key topics and the page number on which each is found.



**Table 6-6** [Chapter 6](#) Key Topics

| Key Topic Element         | Description                                  | Page Number |
|---------------------------|--|-------------|
| Section                   | Stateful Versus Stateless Application Design | 239         |
| Section                   | Changing User State Location                 | 241         |
| <a href="#">Table 6-2</a> | Workload Data Choices                        | 242         |

| Key Topic Element                 | Description                        | Page Number |
|-----------------------------------|------------------------------------|-------------|
| Section                           | User Session Management            | 243         |
| Section                           | Container Orchestration            | 244         |
| Section                           | Resilient Storage Options          | 246         |
| Section                           | Amazon Simple Notification Service | 248         |
| <a href="#"><u>Figure 6-7</u></a> | Creating a Notification Topic      | 250         |
| Section                           | Amazon SNS Cheat Sheet             | 250         |
| Section                           | SQS Components                     | 251         |
| Section                           | Amazon SQS Cheat Sheet             | 253         |
| <a href="#"><u>Table 6-5</u></a>  | SNS, SQS, and Step Functions       | 256         |
| List                              | Features of EventBridge            | 257         |

| Key Topic Element  | Description                                       | Page Number |
|--------------------|---|-------------|
| <u>Figure 6-13</u> | API Gateway Communication Options                 | 259         |
| List               | API Gateway features                              | 260         |
| Section            | API Gateway Cheat Sheet                           | 261         |
| Paragraph          | CloudFormation as AWS-hosted orchestration engine | 268         |
| Section            | CloudFormation Components                         | 269         |
| Section            | CloudFormation Stacks                             | 272         |
| Section            | CloudFormation Stack Sets                         | 276         |
| <u>Figure 6-26</u> | Portfolios in Service Catalog                     | 278         |

| Key Topic Element  | Description                               | Page Number |
|--------------------|---|-------------|
| <u>Figure 6-28</u> | Elastic Beanstalk Creating Infrastructure | 280         |
| Section            | Updating Elastic Beanstalk Applications   | 282         |

## Define Key Terms

Define the following key terms from this chapter and check your answers in the glossary:

serverless

stateful

stateless

sticky session

distributed session

event notification

[application programming interface \(API\)](#)

[regional endpoint](#)

## Q&A

The answers to these questions appear in [Appendix A](#). Use the Pearson Test Prep Software Online for more practice with exam format questions.

- 1.** What is the disadvantage of enabling sticky sessions?
- 2.** What is the advantage of using a central location to store user state information?
- 3.** What is the purpose of enabling notifications with Simple Notification Service?
- 4.** How can Simple Notification Service and Simple Queue Service work together?
- 5.** What is the advantage of using Step Functions?
- 6.** What is the advantage of using Lambda to respond to SNS notifications?

**7.** Why would you use Lambda to create serverless applications?

**8.** How can Lambda be used with API Gateway?

**9.** What service allows you to deploy an architectural solution for an application that you have already written?

# Chapter 7

## Designing Highly Available and Fault-Tolerant Architecture

This chapter covers the following topics:

- [High Availability and Fault Tolerance](#)
- [AWS Regions and Availability Zones](#)
- [Choosing an AWS Region](#)
- [Distributed Design Patterns](#)
- [Failover Strategies](#)
- [AWS Service Quotas](#)
- [Amazon Route 53](#)

This chapter covers content that's important to the following exam domain and task statement:

Domain 2: Design Resilient Architectures

Task Statement 2: Design highly available and/or fault-tolerant architectures

When discussing the proper design of infrastructure for supporting customer workloads at AWS, terms used repeatedly in the AWS Well-Architected Framework documentation when

describing proper workload design (covered in depth in [Chapter 2, “The AWS Well-Architected Framework”](#)) are high availability, fault tolerance, and resilience. This chapter is focused on achieving workload reliability designing with high availability creating fault-tolerant workloads based on the recommendations of the Well-Architected Framework’s Reliability pillar. To be reliable describes the capability of a workload to be able to recover successfully from an infrastructure or service disruption or network failure. Workload reliability also depends on the following well-architected framework pillars:

- **Operational Excellence pillar:** When problems occur, automation should be the solution when responding to failures, utilizing Amazon CloudWatch metrics and alarms, Amazon Simple Notification Service (SNS), AWS Lambda functions, and Amazon EventBridge rules and events.
- **Security pillar:** Workloads must be designed to restrict any harm to valuable data records or infrastructure. Options can include encryption at rest for all data records stored at AWS, AWS WAF filters for protecting workload or data, Elastic Load Balancing (ELB) providing high availability failover for web and application servers, Amazon Virtual Private Cloud (VPC) deployments across multiple availability zones, and deploying the Amazon Route 53 Resolver DNS Firewall,

creating rules that filter outbound DNS traffic from a virtual private cloud (VPC).

- **Performance Efficiency pillar:** Performance efficiency refers to how well a system or process can achieve its desired output utilizing the available resources (CPU, memory, and storage) to complete tasks quickly and efficiently. A system or process that is highly performance-efficient can lead to cost savings and improved user experience.
- **Cost Optimization pillar:** The cost optimization pillar focuses on identifying and implementing strategies to reduce costs and optimize spending on AWS resources. Cost optimization strategies include right-sizing resources, using the best-managed service, and leveraging pricing discounts for compute resources. Scaling compute resources out horizontally can reduce workload costs for web apps, containerized apps, and database compute engines.

*High availability* refers to hosted workloads being always available, regardless of the situation or circumstances that happen from time to time when running workloads in the cloud. In other words, a system with high availability can remain operational and accessible to end users even during failures or disruptions. The phrase “Everything fails” is a favorite saying of Amazon’s CTO, Werner Vogels. How can workloads automatically respond to failure? If one web server,

database, or storage array fails, there should be a backup web or database server or separate storage location available for automatic failover.

High availability ensures *fault tolerance*. Fault-tolerant cloud services and workloads are designed to continue operating when problems occur, without failing or experiencing a loss of data. Having multiple web servers or containers hosted in separate availability zones fronted by a highly available load-balancing service also deployed across multiple availability zones provides a high degree of fault tolerance, high availability, and resilience. A highly available workload environment typically has a defined level of uptime defined as a percentage, such as 99.9% uptime over a given year. Therefore, high availability refers to a workload that can operate for an extended period with minimal downtime. Adding fault tolerance services to a workload results in minimal service interruptions, with significantly higher operating costs as more resources have been utilized. Most organizations accept the higher operating costs of high availability and fault tolerance for mission-critical workloads hosted in the cloud, such as online stores or financial systems that must be available at all times.

*Resiliency* is the capability of a system or component to withstand external stresses such as sudden increases in load, attacks from malicious actors, or environmental factors such as extreme temperatures or natural disasters. A resilient system has been designed for high availability operation across multiple availability zones and can recover quickly from such stresses and continue to function without significant disruption. Resiliency is present because of high availability. All storage services available at AWS maintain multiple copies of stored data records stored across multiple physical locations in separate availability zones. Multiple web servers hosted in availability zones or across separate AWS regions provide high availability, fault tolerance, and a resiliency.

Each AWS Certified Solutions Architect – Associate (SAA-C03) exam question presents a current or proposed design that needs improvement. Always consider answers that present a solution that achieves high availability, fault tolerance, and resiliency.

Any workload running in the AWS cloud should be able to continue to operate when internal failures, latency, and unexpected problems occur. This chapter looks at the building blocks that AWS uses to build and host its cloud services and for customers to deploy their resilient workloads. By designing

with regions, availability zones, and global cloud services, you can ensure workloads enjoy high availability, reliability, resilience, and fault tolerance.

## “Do I Know This Already?”

The “Do I Know This Already?” quiz allows you to assess whether you should read this entire chapter thoroughly or jump to the “Exam Preparation Tasks” section. If you are in doubt about your answers to these questions or your own assessment of your knowledge of the topics, read the entire chapter. [Table 7-1](#) lists the major headings in this chapter and their corresponding “Do I Know This Already?” quiz questions. You can find the answers in [Appendix A, “Answers to the ‘Do I Know This Already?’ Quizzes and Q&A Sections.”](#)

**Table 7-1** “Do I Know This Already?” Section-to-Question Mapping

| Foundation Topics Section             | Questions |
|---------------------------------------|-----------|
| High Availability and Fault Tolerance | 1, 2      |
| AWS Regions and Availability Zones    | 3, 4      |

| Foundation Topics Section   | Questions |
|-----------------------------|-----------|
| Choosing an AWS Region      | 5, 6      |
| Distributed Design Patterns | 7, 8      |
| Failover Strategies         | 9, 10     |
| AWS Service Quotas          | 11, 12    |
| Amazon Route 53             | 13, 14    |

---

### Caution

The goal of self-assessment is to gauge your mastery of the topics in this chapter. If you do not know the answer to a question or are only partially sure of the answer, you should mark that question as wrong for purposes of the self-assessment.

Giving yourself credit for an answer you correctly guess skews your self-assessment results and might provide you with a false sense of security.

---

**1.** Which of the following definitions describes the concept of a highly available workload?

1. The workload remains available even when various issues occur to the infrastructure and to the managed services hosting the workload.
2. The workload is available only during business hours.
3. The workload remains available even under increased load.
4. The workload is hosted in a single availability zone.

**2.** Which listed AWS service provides increased workload availability?

1. ELB
2. An EC2 instance
3. Route table
4. Web Application Firewall (WAF)

**3.** Which of the following does Amazon use to store multiple copies of objects stored in an S3 bucket?

1. AWS Regions
2. Availability zones
3. Edge locations
4. Data centers

**4.** How do multiple availability zones help increase workload reliability in an AWS region?

1. By caching content for Amazon CloudFront
2. By storing backup copies of EBS volumes
3. Workloads are hosted across multiple subnets in different availability zones within an AWS region
4. By locating workload servers closer to the end users' physical locations

**5.** What is the name of the compliance program that defines the available cloud services that can be accessed by federal agencies operating at AWS?

1. HIPAA
2. FERPA
3. FISMA
4. ITAR

**6.** Where are existing compliance reports hosted and made available for AWS customers?

1. AWS Compliance website
2. AWS IAM
3. Artifact
4. AWS Config

**7.** What benefit is gained by deploying workloads across multiple availability zones?

1. Automatic scaling of resources on demand
2. High availability
3. Automatic backups
4. Lower operating costs

**8.** What does the term immutable mean?

1. Never changed or altered
2. Always changes
3. Rollback when failure occurs
4. Highly scalable

**9.** What is a benefit of a pilot light disaster recovery scenario?

1. Automatic failover
2. Required services are kept up to date
3. Disaster recovery site servers are smaller until needed
4. RTO is very small

**10.** What is a benefit of a warm standby disaster recovery scenario?

1. Disaster recovery site servers are smaller until needed.

2. Automatic failover occurs.
3. Disaster recovery site resources are online.
4. RTO is longer than for a pilot light DR deployment.

**11.** What utility reports current service quota levels?

1. AWS Inspector
2. AWS Config
3. AWS Control Tower
4. AWS Trusted Advisor

**12.** Once you have created an AWS account, how does Amazon control the resources you're allowed to order?

1. Allowable resources are based on the purchased support agreement.
2. AWS applies hard and soft limits.
3. Service quotas control resource amounts.
4. By AWS regional resources used.

**13.** What Route 53 DNS record provides redundancy for AWS services?

1. Alias records
2. Latency records
3. Geoproximity records

#### 4. A records

**14.** What AWS service provides DNS services for both private and public queries?

1. CloudFront
2. Route 53
3. Global Accelerator
4. Traffic policies

## Foundation Topics

### High Availability and Fault Tolerance

Successful organizations achieve and maintain highly available and fault-tolerant workloads by deploying compute, database, and storage services based on the recommendations of the Well-Architected Framework. Decisions are based on taking into account the availability, reliability, and resilience required by each workload's business needs and requirements.

Application security is a key component of workload availability and reliability. If a workload gets hacked and is not available, it's not very reliable. Ultimately, the terms high availability, fault tolerance, resilience, and reliability describe a properly designed workload.

Let's explore the meaning of these terms further in the context of hosting and operating workloads at AWS. In the context of AWS, the term *workload* refers to an application and the associated AWS cloud services. When you host a workload at AWS, there are a number of moving parts to consider within each workload that you design and deploy. Cloud hosted applications use what is sometimes called a managed service. A *managed service* is an AWS service that is built, maintained, and patched by AWS; every single cloud service offered by AWS has some level of AWS management. For example, Amazon CloudWatch is the monitoring service that is embedded into many AWS infrastructure services such as Amazon EC2, ELB, Amazon EBS, Amazon S3, and Amazon RDS, enabling customers to monitor each cloud service using metrics that can alert you when a cloud service is either working or not working as expected.

The Amazon mantra they follow when building the cloud services that we use and depend on is to design with security at all layers, ensuring each service is as reliable and dependable and as responsive and as fast as it can be performance-wise, without sacrificing any security or reliability requirements. This is the mindset you must assume when answering AWS Certified Solutions Architect – Associate (SAA-C03) exam questions, proving to Amazon that you're a competent cloud

architect. Amazon does not expect you to have every single technical answer for each AWS service, but it does expect you to display a level of technical common sense and be able to implement Amazon's best practices for designing workloads that meet the stated business objectives and requirements and take into account security, reliability, performance, and cost.

## High Availability in the Cloud



High availability becomes a little more complicated when there are numerous cloud services involved behind the scenes for each hosted workload. ***Availability*** refers to the proportion of time that a system, service, or resource is able to function and be accessed as a percentage, indicating the amount of time that a system, service, or resource is expected to be operational and accessible to end users. As previously mentioned, availability is usually defined as a percentage of uptime ranging from 99.9% to 100% over a defined period, typically per year. [Table 7-2](#) lists some common examples of high availability and the maximum potential unavailability over a calendar year.

**Table 7-2** Availability

| Availability (%) | Potential Unavailability per Year | Use Case                           |
|------------------|-----------------------------------|------------------------------------|
| 99%              | 3 days, 15 hours                  | Batch processing, data transfer    |
| 99.9%            | 8 hours, 45 minutes               | Project management                 |
| 99.95%           | 4 hours, 22 minutes               | Point of sale (POS), online retail |
| 99.99%           | 52 minutes                        | Video delivery                     |
| 99.999%          | 5 minutes                         | ATM transactions                   |

AWS publishes service-level agreements (SLA) for many of its cloud services, including Amazon Elastic Compute Cloud (EC2), Elastic Block Store (EBS), and Elastic Container Service (ECS) with a stated availability of 99.99%, which means that the total

downtime expected per year utilizing any of these services is 52 minutes per service, per year. However, there is no exact time when downtime will even actually occur. Perhaps your workloads hosted at AWS won't have any downtime!

Having a hosted workload that is expected to be always available—when the workload is created out of numerous independent cloud services—adds additional complexity to each workload design. Service-level agreements define the individual AWS service availability, not a workload's availability. Customers shouldn't host mission-critical workloads on a single EC2 instance; instead, multiple EC2 instances are typically deployed and hosted in separate availability zones within each AWS region, and in many cases across multiple AWS Regions.

The virtual hard drives created for boot drives and data volumes are EBS storage volumes. Multiple copies of each EBS volume are automatically created, providing an added measure of high availability, fault tolerance, and reliability. EBS volumes are designed to be highly reliable, with an average annual failure rate of less than 0.1%; EBS volumes are expected to experience an outage or failure less than once per year on average. The EBS service provides volume durability using data

replication and backup within the availability zone where each volume is created, increasing the reliability of each EBS volume.

Snapshots are point-in-time backups of EBS volumes. EBS snapshots can be used to recover data in the event of data loss, corruption, or accidental deletion, or to create new EBS volumes with the same data as the original volume. EBS snapshots can also be used to migrate data from one AWS region to another, or to create a new EBS volume in a different Availability Zone or region for disaster recovery purposes.

EBS volumes are used when deploying Amazon RDS databases, which are designed to be highly reliable, with an average annual failure rate of less than 0.1%. RDS recommends Multi-AZ deployments, which automatically create one or more secondary copies of the database in multiple availability zones, providing additional protection against outages and data loss.

## Reliability



Workload ***reliability*** is measured in terms of the availability of a system or service, and its capability to handle failures and

disruptions without impacting the overall quality of the application. Organizations operating workloads at AWS need to design their workloads based on their own expectations and business requirements for each hosted workload. How much workload downtime is acceptable to your organization? How many failures can occur before the entire workload cannot be trusted to operate at an acceptable level?

The reliability of each workload can be affected by several factors, such as the design of the system, the quality of the cloud services used, and the level of maintenance and support. It is important to carefully evaluate the reliability of a workload before using it in critical applications, and to implement appropriate procedures to ensure that the workload remains reliable over time. When designing for reliability, consider the acceptable trade-off between the desired service level based on your business requirements and needs and the true cost of maintaining desired workload reliability.

You might be wondering why the cost has entered the discussion of reliability. If unlimited funds are available, workloads can be designed to rarely, if ever, fail. Having 1,000 workload servers hosted across six different availability zones will certainly guarantee success—but probably isn't affordable for most organizations. Thankfully, the AWS cloud is a pretty

reliable entity on its own. However, it is each customer's responsibility to design their hosted workloads for the required availability and reliability.

Other concerns that can affect reliability are operational changes such as updates or patching that are carried out on your workload on a daily, weekly, and yearly cadence. Are these changes performed using automation and playbooks? Are you using multiple AWS VPCs to host separate test, development, and production environments? Well-architected designs must also include workload security and performance efficiencies. If workload security is lax, allowing harm to the workload or to the essential data records, overall reliability will be reduced. In the area of performance efficiencies, are you paying too much to ensure compute resources have high availability and stability 24-7, or should you automatically scale up the compute capacity as required? To achieve workload reliability, each of the six pillars of the Well-Architected Framework needs to be fully considered. As a reminder, [Chapter 2](#) explores the AWS Well-Architected Framework design concepts in more detail.

The next section looks at how Amazon has designed its cloud services to be as available and reliable as possible—and how customers can use their online cloud services to design highly available, fault-tolerant, and reliable workloads.

## AWS Regions and Availability Zones

It is important to understand where Amazon cloud services are located, why they are located in multiple physical locations, and how this design helps organizations in designing hosted workloads for high availability, reliability, and durability.

Amazon cloud services are hosted in *regions* in populous areas of the world. Each AWS **region** is located in a specific geographic area of the world. Each region has, at a minimum, two availability zones. Each availability zone has a minimum of one data center for hosting customer workloads. Availability zones are spaced apart by a minimum of 10–15 miles and are powered by different electrical grids and external support services such as ISPs and third-party services. Additional data centers within each AWS region and availability zone locations host various AWS-managed cloud services for storage, databases, and managed services used by workloads. Each AWS region is completely isolated from other regions; problems that arise within one region remain localized to that one region and should not affect any other AWS region you also may be operating in.

Each customer with a standard AWS account can choose to work in any AWS region except for the GovCloud regions and several regions in China that require a special AWS GovCloud

(US) account. [Figure 7-1](#) shows the areas of the world where AWS regions are currently located.



**Figure 7-1** AWS Regions and Geographic Locations

Currently, there are at least 30 AWS regions worldwide. There may be additional regions online by the time you read this book because AWS is continuously expanding its cloud resources throughout the world.

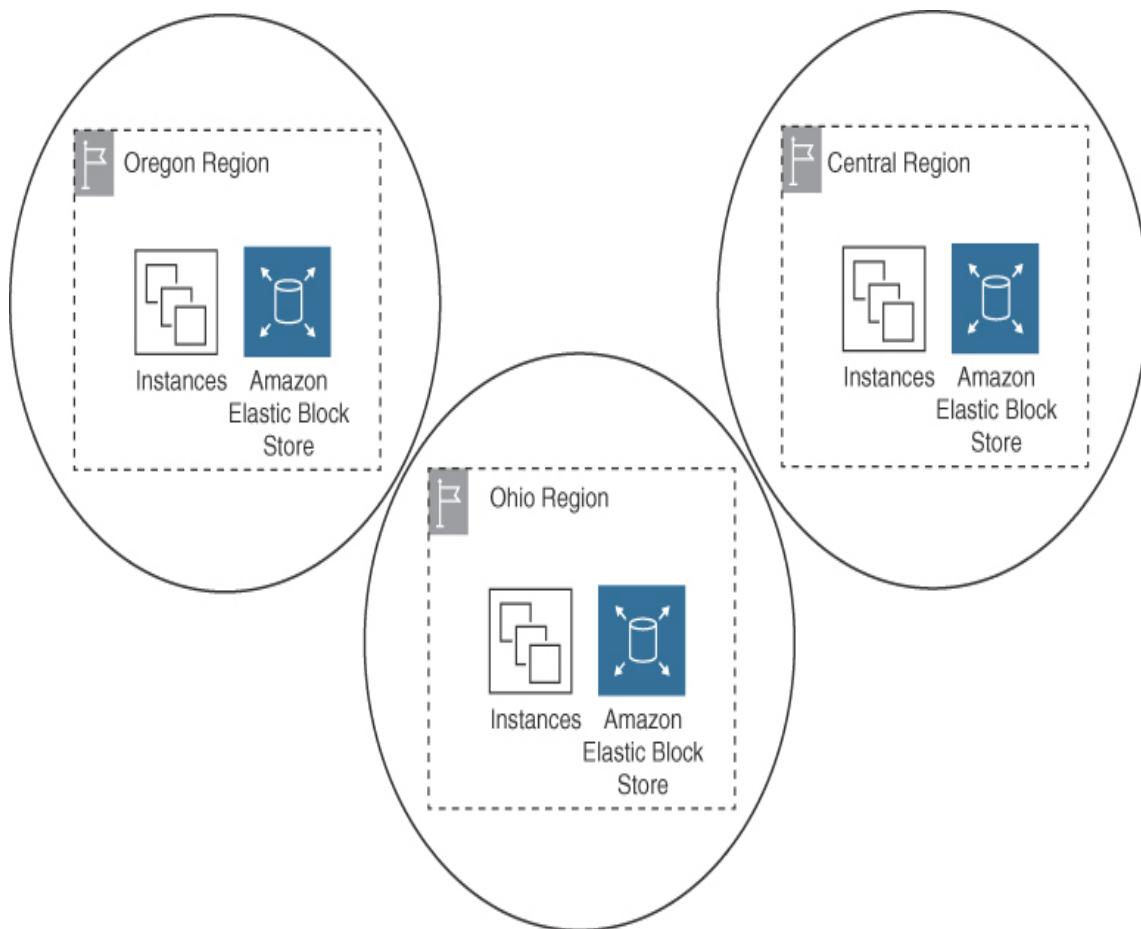
---

Note

For information on current regions and availability zones, see [https://aws.amazon.com/about-aws/global-infrastructure/regions\\_az/](https://aws.amazon.com/about-aws/global-infrastructure/regions_az/).

---

Each AWS region is initially physically and securely separated from other regions, as shown in [Figure 7-2](#).



**Figure 7-2** AWS Regions Starting Off as Isolated Entities

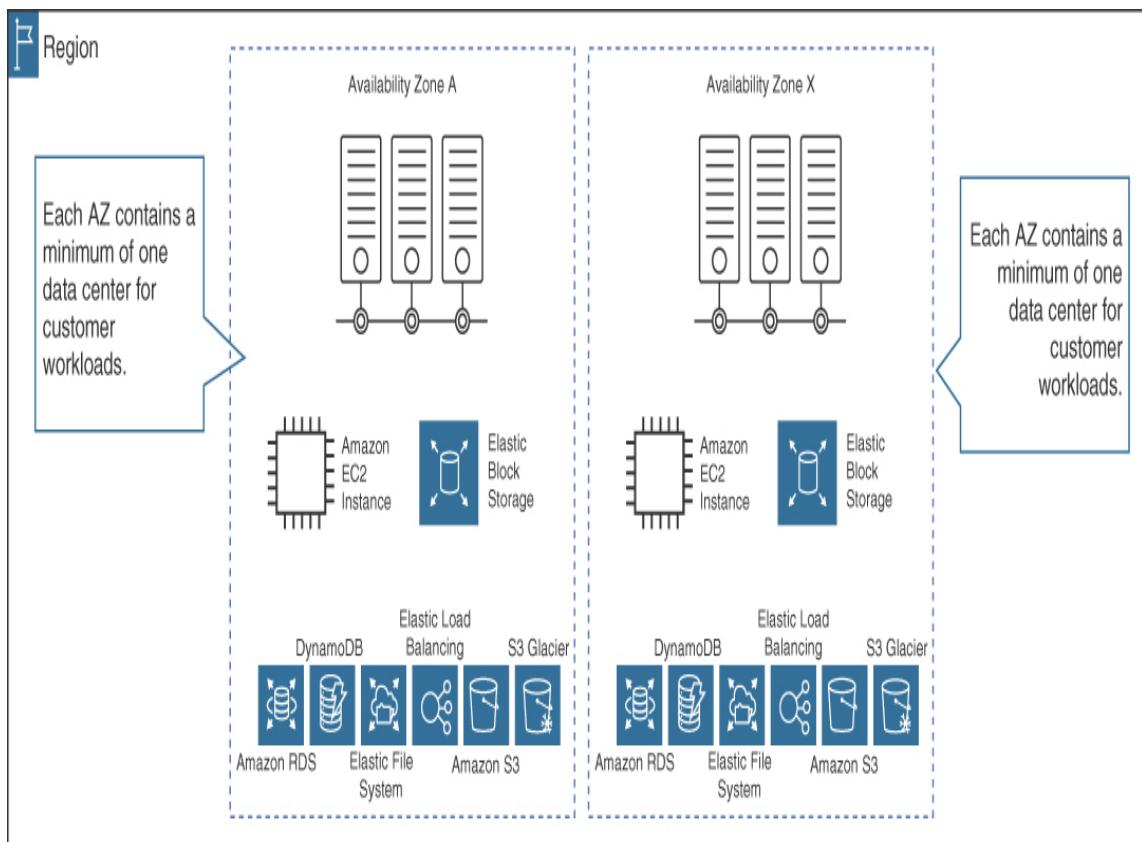
Isolating regions enables AWS to guarantee a level of operational compliance and data sovereignty. AWS recognizes that the industry that you work in might have strict rules and regulations regarding how your user data is managed and controlled. If your data is stored within a particular AWS region, that data will never leave its selected region unless you move it or specifically direct AWS to do so.

As mentioned, within each region, AWS offers a variety of cloud services that are designed not only to support customers' needs for storage, load balancing, DNS services, managed database services, and more but also to be highly available and fault tolerant with the ability to fail over to other service locations within the same AWS region when disaster strikes.

Various numbers of separate regional data centers support the offered AWS cloud resources and services; it's handy to visualize a sampling of strategically placed regional cloud services linked together with high-speed private networking across each availability zone and AWS region (see [Figure 7-3](#)). (Availability zones are discussed in the next section.) The AWS cloud services are designed for high availability, fault tolerance, and resilience. The failure of customer workloads located within each region does not affect and derail the operation of the AWS managed services because AWS managed services are

located in their own regional data centers. For example, Amazon's S3 storage and Elastic Load Balancing (ELB) services are specifically designed to integrate with customer workloads within each AZ; however, each of these services is located in its own regional data center cluster.

### Key Topic



**Figure 7-3** Visualizing Availability Zones and Regional Services

Depending on the scope and requirements of an organization's workload design, the AWS region that is chosen must meet all the required design needs and compliance requirements; there may be no need to operate in multiple AWS regions. For other use cases, you might have business requirements to host workloads spanning multiple AWS regions; for example, the use case for a public-facing software as a service (SaaS) workload requiring database synchronization across multiple regions to protect against a regional failure of the database service.

Amazon DynamoDB provides high availability and horizontal scaling of regional tables that can also be synchronized across multiple regions. Amazon Aurora is MySQL and PostgreSQL compatible, storing data records in clustered shared storage deployed across single or multiple AWS regions. A regional solution could be Amazon RDS MySQL, deployed using a cluster of primary and two standby databases utilizing multiple database instances across availability zones. AWS Route 53 DNS services and custom traffic policies can also be used to geographically load balance workloads that are hosted in different AWS regions.

---

#### Note

You might not know much about databases at this point—and that is okay. For this discussion, you just need to be able to appreciate that each of the potential database options just mentioned provides a much higher level of availability and reliability than a single database running on a single virtual server with multiple virtual hard drives. Databases are covered in detail in [Chapter 10](#), “[Determining High-Performing Database Solutions](#).” For now, we are concerned with the concept of high availability, fault tolerance, and reliability and the various cloud services provided by AWS that you might choose to use. Using multiple AWS regions can provide a very large measure of high availability, fault tolerance, and reliability for hosted workloads.

---

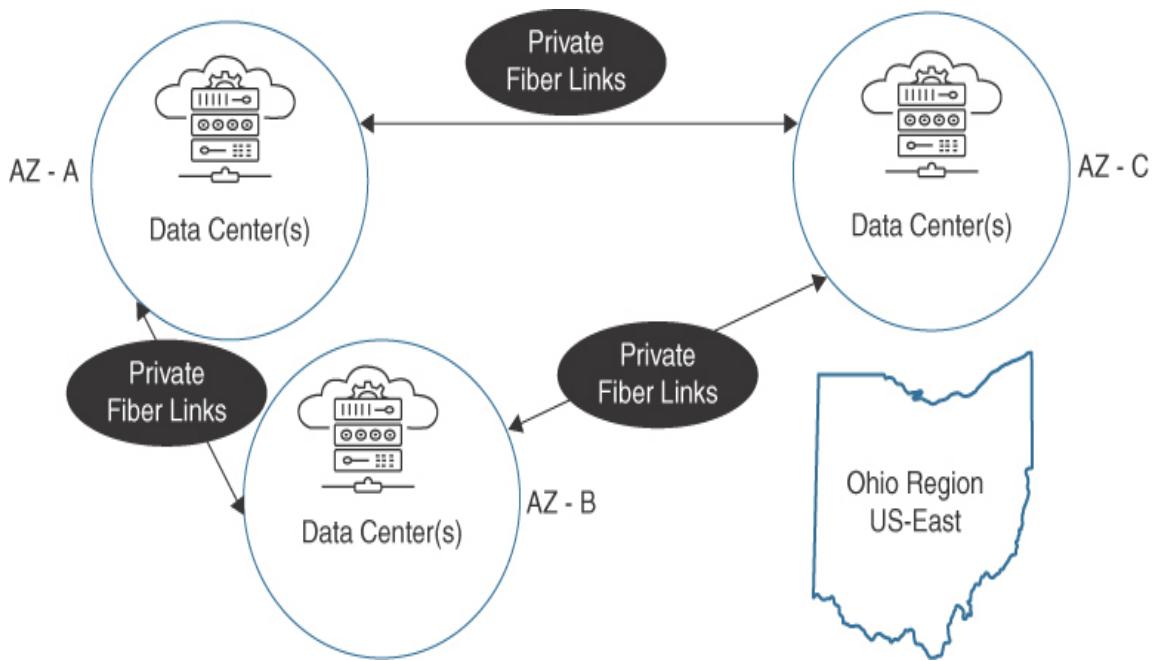
## Availability Zones



AWS provides availability zones for reliability and high availability for the cloud services offered within each region

and for your hosted workloads. **Availability zones (AZs)** are physical locations within a region where data centers are located. Each availability zone has at a minimum one data center dedicated to customer workloads. Each AZ is designed to be isolated from the others, with its own power, cooling, and networking infrastructure. This allows for the creation of highly available and fault-tolerant applications, because data can be stored and accessed from multiple availability zones, providing protection against outages or other disruptions in any single location. Most AWS regions have three AZs or more.

Each AZ is linked to the other AZs in the same region through private dedicated, redundant, low-latency fiber network connections that Amazon owns (see [Figure 7-4](#)). As an example of network speed, traffic between EC2 instances and S3 storage can utilize up to 100 Gbps of bandwidth to VPC endpoints and public IPs within the same region.



**Figure 7-4 Availability Zones**

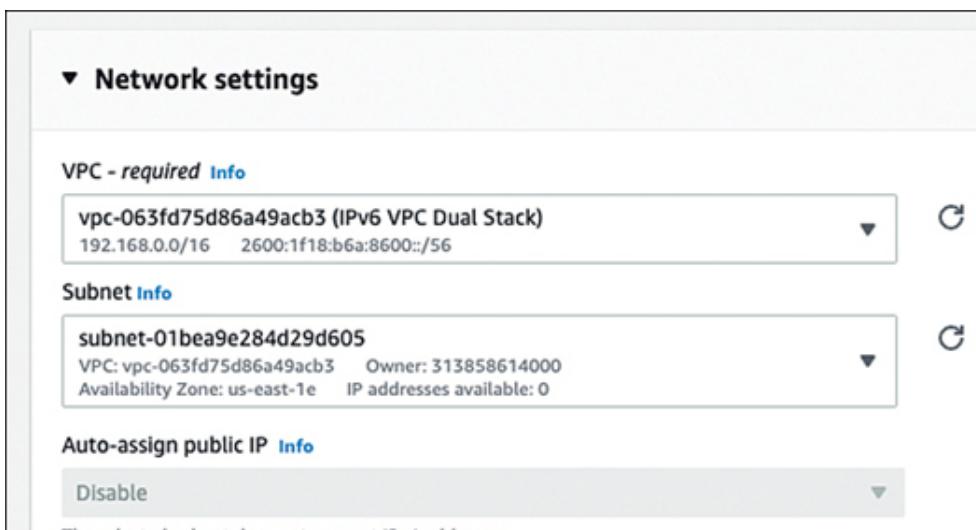
To offer cloud services that are designed to adhere to the many compliance programs that AWS supports, including HIPAA and PDI DSS, Amazon owns the cloud service infrastructure.

Designing a workload to operate across multiple availability zones is one of the major concepts to understand for the AWS Certified Solutions Architect – Associate (SAA-C03) exam.

Amazon S3 storage, DynamoDB, and Amazon Aurora deployments operate across three AZs.

AWS does not disclose whether an AZ has a single or multiple data centers; that's private information. It's important to keep in mind that the discussion of data centers within the AZ refers

to the data centers dedicated to customer workloads. Amazon does not officially disclose infrastructure numbers; however, in its largest region, North Virginia, us-east-1, there are many data centers dedicated for customer workloads. When a customer selects a particular AZ to host servers or containers, the exact data center is not specifically identified, just the AZ, as shown in [Figure 7-5](#). It is important to reiterate that no two AZs share the same single data center. Each data center in each AZ is a separate entity that is isolated from other data centers within the same and different AZs.



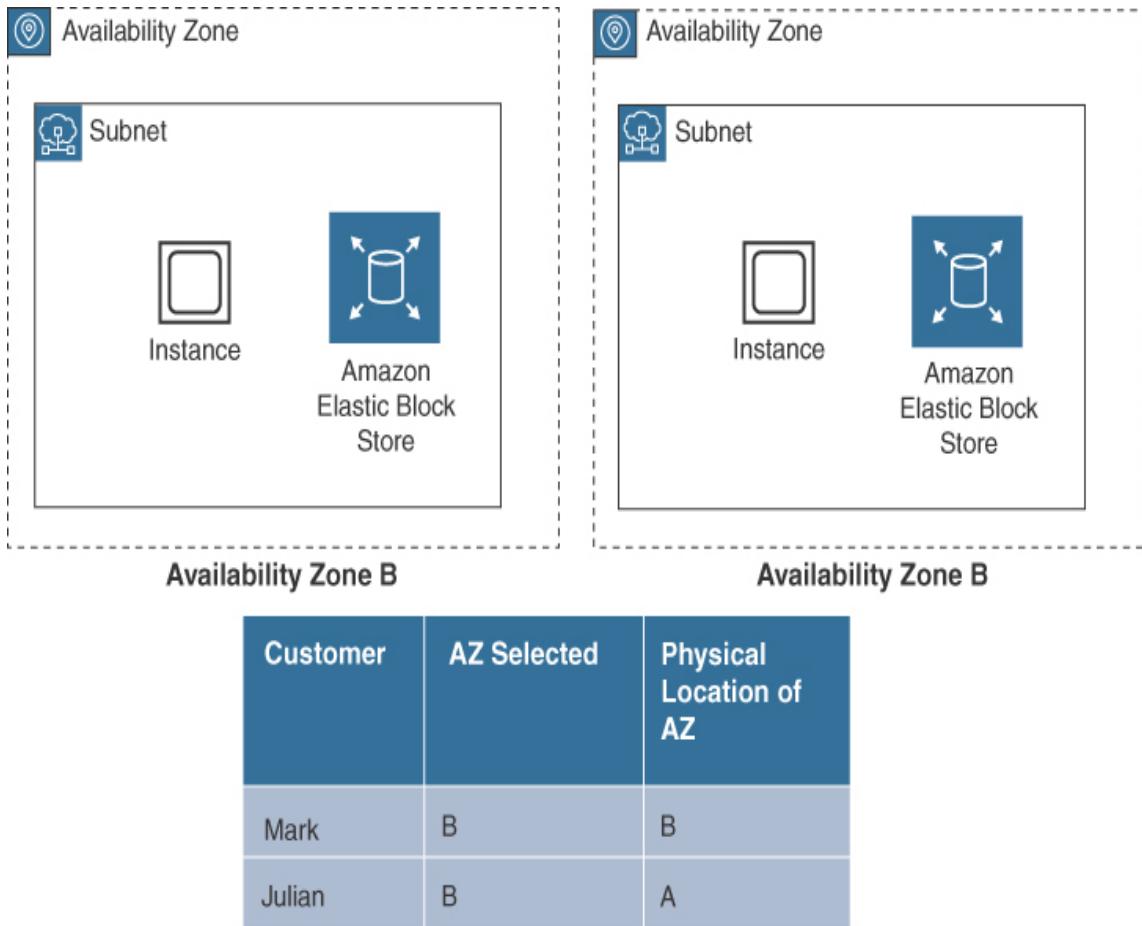
**Figure 7-5 Availability Zone Selection**

Throughout its tenure as a public cloud provider, AWS has had very limited failures of availability zones and regions. We need to keep perspective on AWS failures when they are reported by

the media. Remember there are many data centers within each region that host the regional cloud services offered by AWS; there are also the many data centers that are provided for customer workloads. Separating cloud services and customer workloads into separate data centers ensures a very high level of reliability and uptime across the entire AWS cloud.

## **Availability Zone Distribution**

It is also important to understand that AWS carries out balancing and distribution of customer resources that are hosted in availability zones. For example, if Mark logs in to his AWS account, selects the US-East Northern Virginia region, and creates a subnet in Availability Zone A, the physical data center location of Mark's subnet is probably not the same as the subnet for another Amazon customer, Julian, who has also selected the US-East Northern Virginia region creating a subnet in Availability Zone A. Both customers have created a subnet in Availability Zone A, but their subnets are most likely not in the same physical AZ. They are most likely in different physical data centers, as shown in [Figure 7-6](#). What's important is the use of multiple availability zones within a region; the exact physical location of each AZ is not public information.



**Figure 7-6** AZ Balancing and Distribution of Resources

Latency between AZs within an AWS region is also not an issue; average latency between AZs within a region is around a few milliseconds. Certainly, if you're concerned about network speeds, you can perform latency testing from one EC2 instance hosted in one AZ to an EC2 instance located in another AZ by using ping (local routing is enabled within a virtual private cloud). You could also deploy iPerf3 – SpeedTest Server on Ubuntu, available in the AWS Marketplace, which can

determine the maximum achievable bandwidth across the AWS private network. You can also use the AWS Latency Monitoring tool at <https://www.cloudping.co/grid> to review the current latencies between AWS regions.

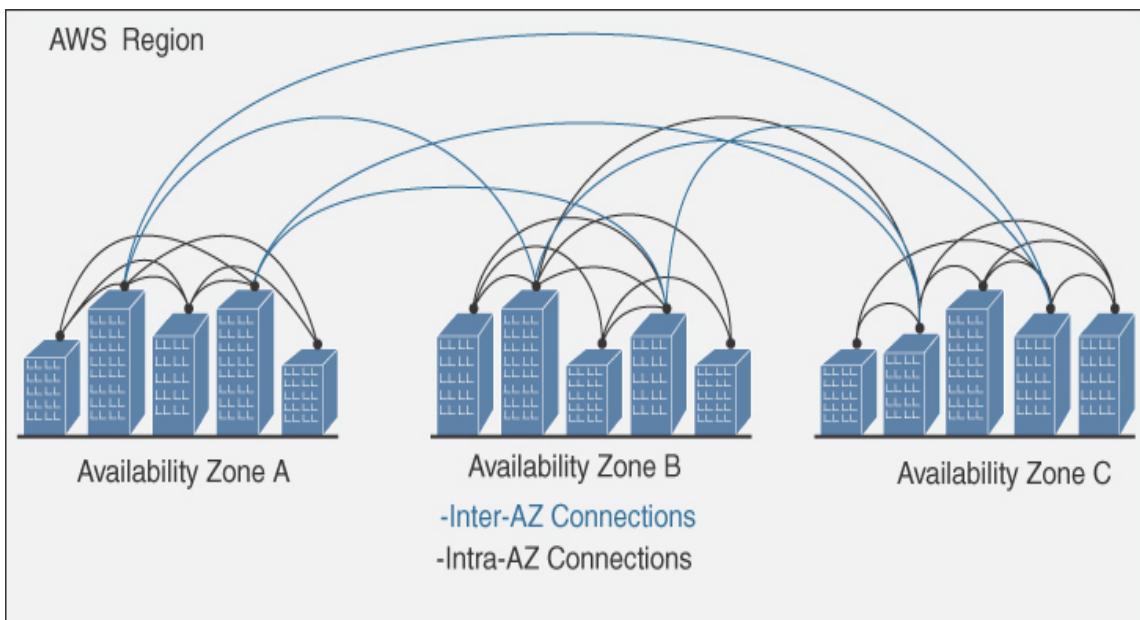
---

#### Note

Each AZ within each AWS region has a moniker based on the region it is hosted within as well as a letter code, such as us-west-1.

---

Within a single AZ, the data center private network connections are defined as intra-AZ connections, as they are local to the AZ, as shown in [Figure 7-7](#). The wiring between the AZs is defined as inter-AZ connections, with additional private links connecting the regions. The primary reasons for using AZs in your infrastructure design are high availability with automatic workload failover and primary/standby/read replica database replication.



**Figure 7-7** Private Network Wiring for AWS Regions and AZs

Customers that operate across a minimum of two AZs with independent redundant components, such as EC2 instances and multi-AZ RDS database deployments with an availability of 99.9%, will have an effective 99.9999% availability.

AWS services defined as global services are designed to operate outside of AWS regions located at the perimeter of the AWS cloud in *edge locations*. Edge services include the DNS service Amazon Route 53, Amazon CloudFront, AWS's CDN, and the AWS Web Application Firewall. You can find more about the edge services in [Chapter 11, “High-Performing and Scalable Networking Architecture.”](#)

---

### Note

All data flowing across AWS regions across the AWS global network is automatically encrypted at the physical layer before it leaves AWS facilities. All network traffic between AZs is also encrypted.

---



## Planning Network Topology

When deploying workloads at AWS, designing with multiple availability zones to provide high availability, fault tolerance, and reliability is an important concept to understand. The key infrastructure services offered by AWS are all designed to be deployed across multiple AZs, providing each customer with the ability to design their workload infrastructure with high availability, fault tolerance, and reliability with networking (Amazon VPC), load balancing services (ELB), virtual servers or containers (Amazon EC2 instances, Amazon EKS), Amazon RDS databases, Amazon DynamoDB, monitoring (Amazon CloudWatch), and Auto Scaling (designed to scale your compute resources based on demand).

Workloads can also be hosted across multiple AWS regions or be part of a hybrid design with existing on-premises data centers. Amazon Virtual Private Cloud (VPC) allows you to provision a private, isolated network utilizing multiple AZs within a single region. VPC endpoints allow private connections from VPC workloads hosted on subnets to AWS services across the AWS private network, allowing private network communication. Resiliency for private network connections at AWS, and from on-premises locations to AWS VPCs and AWS cloud services, can also be provided by the following cloud services:

- Using AWS Direct Connect (DX), a dedicated high-speed connection can connect an on-premises data center to the AWS cloud.
- Having redundant DX connections from multiple separate data centers or adding a second DX connection can provide additional high availability connections.
- Using AWS VPN connections can provide dedicated VPN connections at 1.25 Gbps throughput per VPN tunnel.
- Deploying AWS Marketplace appliances with high availability across multiple AZs on redundant EC2 instances.

High-availability endpoints for public network connections from end users to public-facing workloads and AWS resources

are provided by the following:

- **Amazon CloudFront:** Delivers data through over 300 points of presence (PoPs) to edge location caches and regional edge caches.
- **AWS Global Accelerator:** Optimizes workload traffic requests to multiple edge location endpoints.
- **Amazon API Gateway:** Operates as a reverse proxy accepting API calls to hosted APIs; API Gateway is located between the end-use requests and the AWS backend services.
- **Elastic Load Balancing (ELB) service:** Provides load balancing across availability zones and integrates with EC2 Auto Scaling to provide high availability and self-healing infrastructure. Incoming traffic requests can be filtered using AWS WAF.
- **Amazon Route 53 (highly available DNS service):** Routes user requests to workloads and services running at AWS, edge locations, multi-region workload deployments, and outside of AWS to other external on-premises locations or other public clouds. The following are high-availability routing choices:
  - **Latency routing policy:** Routes end users to the AWS region that has the lowest latency
  - **Failover routing policy:** Routes end users to resources that are available in an active-passive or active-active

design across multiple regions

- **Geolocation routing policy:** Routes traffic based on the location of the end user
  - **Geoproximity routing policy:** Routes traffic based on the location of your resources
  - **Multivalue answer routing:** Includes health checks of resources, improving high availability
- 

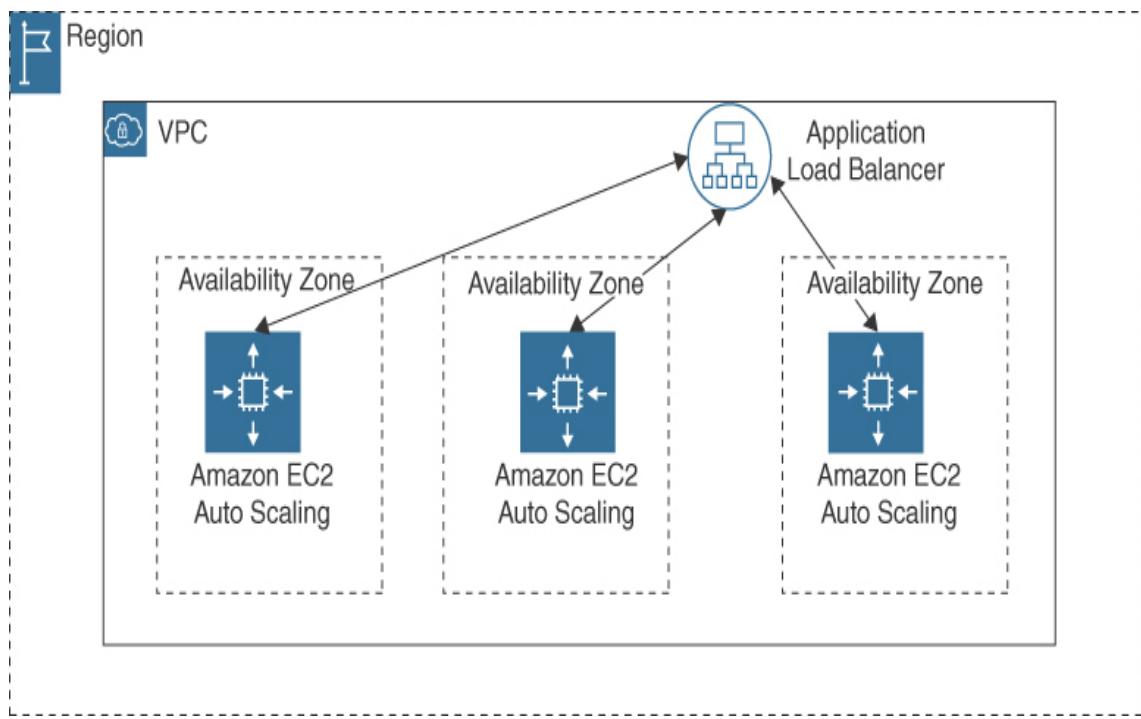
#### Note

As you've noticed, a number of Amazon services start with the word *Elastic* (for example, Elastic Compute Cloud and Elastic Block Store). A service that is defined as *elastic* allows the resource to change size and power after the initial creation. For example, an EC2 instance can be changed to increase resource size without having to start over, and EBS volumes' size and speed can also be changed whenever the need arises.

---

A few years ago, Amazon had issues in Australia. The media announced that “the Amazon cloud was down.” This was true to a degree because an AZ in the Asia Pacific region had issues; however, not all the AZs within the Asia Pacific region had issues. If an organization’s workload or website was hosted in

the unavailable AZ, and the workload or website was not designed for automatic failover to another AZ, as shown in [Figure 7-8](#), then workloads were potentially not available for a period of time.



**Figure 7-8** Failover Possibilities of Availability Zones for Workloads

Is a sudden lack of availability of a customer's hosted workload Amazon's fault? This is a fair question to consider; however, it most likely is the customer's fault for not utilizing proper cloud design standards for its hosted workloads. In fact, a customer that fails to design its workloads to properly fail over across multiple AZs violates the customer's SLA with AWS. If workload

issues are due to an Amazon failure beyond the customer's control and the customer has designed its workload following best practices, the customer will get a credit on its bill for the downtime. The customer also needs to prove that the workload was down by providing relevant network traces. In some situations, a credit is automatically provided because an outage is obviously AWS's problem.

---

### Note

Hosting your workload in a single AZ is accepting a single point of failure.

---

## Local Zones



Even with the rapid expansion of regional cloud resources carried out by AWS over the last few years, there are still many businesses specializing in real-time gaming, streaming, video productions, or virtual reality that haven't moved to the AWS public cloud because the speed of accessing cloud services is not fast enough for their needs. Local data residency requirements for sectors such as healthcare, financial services,

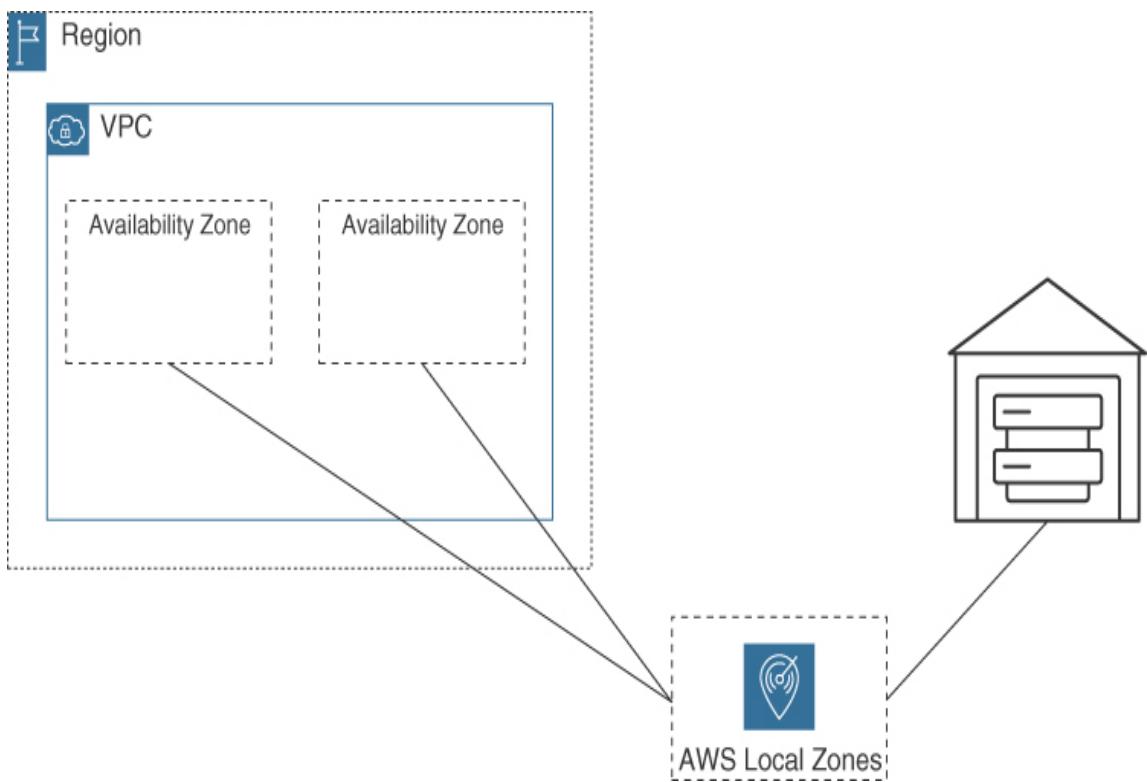
and governments have also made moving to the public cloud an issue for some sectors. To solve these issues, Amazon has rolled out **Local Zones** to over 30 cities in 27 countries, as shown in **Figure 7-9**. Local Zones are deployments of AWS infrastructure in a single data center, including compute, storage, and database services, resulting in single-digit millisecond latency speeds for customers located in these locations. Local Zones for these use cases may help by

- Deploying workload stacks closer to the location of the business and end users that require low latency and high performance
- Migrating on-premises workloads to a nearby Local Zone and maintain low-latency requirements for hybrid designs
- Meeting data residency requirements for compliance with country or business regulations



**Figure 7-9** Local Zones Available in Many AWS Regions

A Local Zone contains cloud resources without any built-in redundancy; however, each Local Zone can be associated with an AWS region by linking the Local Zone with an organization's existing region's AWS VPC network infrastructure by including Local Zone subnets, as shown in [Figure 7-10](#), providing additional redundancy and failover through connections to other AZs in the AWS region.



**Figure 7-10** Local Zone Architecture

## Wavelength Zones

As the 5G protocol becomes more commonplace across the world, the infrastructure that hosts 5G workloads for mobile applications is moving closer to the end user's handheld 5G smartphone or device. Workloads deployed in Wavelength Zones are moving to the edge of the cloud, as shown in [Figure 7-11](#), and closer to the end user. Amazon is installing AWS compute and storage services in the telecommunications provider's data center. The workloads are developed and hosted in the telecommunications provider's own data center

on AWS infrastructure, shortening the end users' distance to the 5G workload, resulting in ultra-low latency and increased performance.

The screenshot shows the AWS Wavelength Zones interface. At the top, it displays the region as "US East (Verizon) / us-east-1-wl1" with an "Opted in" status. There are two buttons: "Edit" and "Manage". Below this, a section titled "Wavelength Zones" lists ten locations, each with its name and city:

- us-east-1-wl1-atl-wlz-1 (use1-wl1-atl-wlz1) - Atlanta
- us-east-1-wl1-bna-wlz-1 (use1-wl1-bna-wlz1) - Nashville
- us-east-1-wl1-bos-wlz-1 (use1-wl1-bos-wlz1) - Boston
- us-east-1-wl1-chi-wlz-1 (use1-wl1-chi-wlz1) - Chicago
- us-east-1-wl1-clt-wlz-1 (use1-wl1-clt-wlz1) - Charlotte
- us-east-1-wl1-dfw-wlz-1 (use1-wl1-dfw-wlz1) - Dallas
- us-east-1-wl1-dtw-wlz-1 (use1-wl1-dtw-wlz1) - Detroit
- us-east-1-wl1-iah-wlz-1 (use1-wl1-iah-wlz1) - Houston
- us-east-1-wl1-mia-wlz-1 (use1-wl1-mia-wlz1) - Miami
- us-east-1-wl1-msp-wlz-1 (use1-wl1-msp-wlz1) - Minneapolis
- us-east-1-wl1-nyc-wlz-1 (use1-wl1-nyc-wlz1) - New York
- us-east-1-wl1-tpa-wlz-1 (use1-wl1(tpa-wlz1) - Tampa
- us-east-1-wl1-was-wlz-1 (use1-wl1-was-wlz1) - Washington DC

**Figure 7-11** Wavelength Zones Move the Workload Closer to the End User

## AWS Services Use Cases

**Key Topic**

It is not expected or even possible that you will understand the complete details of every AWS service covered by the AWS Certified Solutions Architect – Associate (SAA-C03) exam. It is, however, important to have a conceptual use case in your mind for each AWS service that could be the focus of an exam question. When an AWS service is described in the cloud services FAQ, a use case is provided. For example, the service Amazon Comprehend is defined as “a natural language processing service that uses machine learning to find meaning and insights in text.” The suggested use cases for Amazon Comprehend are to analyze the “text” for key phrases and customer sentiment analysis. Developers looking to deploy natural learning language processing in workloads hosted at AWS can use this service. For the SAA-C03 exam, that’s the amount of detail to know about Amazon Comprehend.

For many AWS cloud services, a short description is all you need to know for the SAA-C03 exam. Other cloud services covered by the exam require additional knowledge of deployment and design details. The knowledge level being tested of an AWS cloud architect is a competent understanding of the cloud services covered by the exam. During your preparation for the SAA-C03 exam, it is highly recommended to review the FAQs for the highly available (HA) and fault-tolerant (FT) services listed in [Table 7-3](#), noting the use cases.

---

## Note

AWS FAQs can be found at  
<https://aws.amazon.com/faqs/>.

---

**Table 7-3** AWS Services Details

| AWS Service | What is it?     | FT and HA Use Case                    | Deployment   |
|-------------|-----------------|---------------------------------------|--|
| Amazon EC2  | Virtual servers | Multiple AZs with ELB, EC2 Auto Scale | Requires Amazon Machine Image (AMI), VPC, security group |

|             |                   |                            |                |
|-------------|-------------------|----------------------------|----------------|
| AWS Service | What is it?       | FT and HA Use Case         | Deployment     |
| Amazon      | Docker containers | A cluster of EC2 instances | Task Statement |
| ECS         |                   | across AZs                 |                |

|            |                     |                 |                         |
|------------|---------------------|-----------------|-------------------------|
| Amazon EBS | Virtual hard drives | Multiple copies | Attach to EC2 instances |
|------------|---------------------|-----------------|-------------------------|

|                  |                             |                           |  |
|------------------|-----------------------------|---------------------------|--|
| AWS Service      | What is it?                 | FT and HA Use Case Across | Deployment   |
| EC2 Auto Scaling | Scale compute automatically | multiple AZs with ELB     | Scheduled, step, simple, target tracking, Predictive scaling |
| AWS Auto Scaling | Compute auto scaling        | Across multiple AZs       | Predictive scaling   |

|                   |             |                    |                        |
|-------------------|-------------|--------------------|------------------------|
| AWS Service       | What is it? | FT and HA Use Case | Deployment             |
| Amazon CloudFront | CDN         | Origin Shield      | Edge locations,        |
|                   |             |                    | Regional edge caches,  |
|                   |             |                    | HTTPS,                 |
|                   |             |                    | Field-Level Encryption |

|                                 |                              |   |  |
|---------------------------------|------------------------------|---|--|
| Application Load Balancer (ALB) | HTTPS load balancing service | EC2 instance and Docker container failover, health checks | Supports operation across multiple AZs |
|---------------------------------|------------------------------|---|--|

|                     |                            |  |  |
|---------------------|----------------------------|--|--|
| AWS Service Network | What is it?                | FT and HA Use Case Workload                        | Deployment                             |
| Load Balancer (NLB) | TCP load balancing service | failover, health checks, TLS end-to-end encryption | Supports operation across multiple AZs |

## Choosing an AWS Region

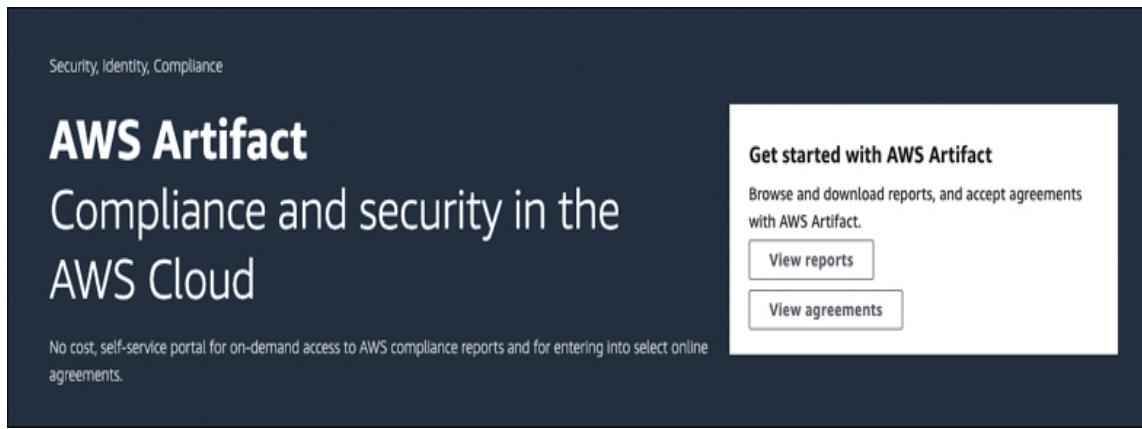
Reasons to choose a specific AWS region depend on four interlinked conditions:

- **Compliance rules:** Where is your organization allowed to operate?
- **Latency issues:** How far away are your customers from the desired cloud service?
- **Services offered:** In what region is the cloud service that you require offered?
- **Pricing:** Are infrastructure costs the main driver for migrating to the cloud?

To make the decision about which AWS region to operate in, evaluate each of these conditions thoroughly. The following sections look at each of these conditions, starting with the most important condition: the compliance rules and regulations that your organization must follow.

## Compliance Rules

Before your company begins hosting any workloads in the AWS cloud, it needs to analyze any rules and regulations that it is required to follow in its day-to-day business practices. For example, your organization must adhere to ISO 27001 standards. AWS maintains its data centers, networks, and shared infrastructure in accordance with a suite of ISO certifications that mandate a strict level of compliance with security management, information security controls, and best practices while operating in the public cloud. The security standards in this example are likely very close to AWS's own standards because AWS holds certification for compliance with the current ISO/IEC 27001, 27017, and 27018 certifications. The [\*\*AWS Artifact\*\*](#) utility is located in the AWS Management console and allows AWS customers to review the compliance standards supported by AWS, as shown in [Figure 7-12](#).



**Figure 7-12** Artifact Utility

Third-party ISO auditors audit the AWS cloud on a rigid schedule to ensure that AWS maintains and upholds its overall operations to the current ISO/IEC security management standards. Other third-party compliance auditors also ensure that AWS lives up to the many other compliance standards and assurance programs that AWS is also aligned with. When an organization signs on as an AWS customer, it has access to the compliance reports regarding the standards, certifications, and attestations that Amazon has achieved and maintains. Most compliance reports are available to all AWS customers upon request; for others, you must sign a nondisclosure agreement (NDA) with AWS.

The steps to get started with reviewing the available compliance and attestation reports at AWS are as follows:

**Step 1.** Sign into your AWS account with root user account credentials.

**Step 2.** In the AWS Management Console, search for Artifact. The available security and compliance documents are listed for all AWS customers.

**Step 3.** Choose your compliance program and click Get This Artifact.

**Step 4.** Download the selected document and review the services that are defined as being in scope, shown in [Figure 7-13](#).

| Reports (71)   |                                      |                                 |  | <a href="#">Copy link</a>           | <a href="#">Download report</a> |
|--|--------------------------------------|---------------------------------|--|-------------------------------------|---------------------------------|
|  |                                      |                                 |  | <input type="text"/> Search reports | < 1 2 > ⌂                       |
| Title  | Reporting period                     | Category                        | Description  |                                     |                                 |
| Outsourced Service Provider Audit Report (OSPAR)                             | April 1, 2019 to March 31, 2020      | Certifications and Attestations | on all Singapore banks' outsourced service providers ("OSPs") who provide material outsourced services and/or have access to the financial institution clients' information. This audit report covers the period from 1 April 2019 to 31 March 2020. The current OSPAR report was issued on 7th July 2020 and it will remain valid for 12 months from the date of issue. |                                     |                                 |
| PCI 3DS Attestation of Compliance (AOC) and Responsibility Summary - Current | October 15, 2020 to October 14, 2021 | Certifications and Attestations | This is the most recent AWS PCI 3DS assessment package dated October 15, 2020. An external Qualified Security Assessor Company (QSAC), Coalfire Systems Inc. has validated that AWS has successfully completed PCI 3DS Core Security Standard v1.0 assessment and were found to be compliant.  |                                     |                                 |

**Figure 7-13** Reviewing Audit Reports Using AWS Artifact

## **Understanding Compliance Rules at AWS: Use Case**

To begin to understand the compliance rules and regulations that AWS supports, follow these steps:

**Step 1.** Visit the AWS Compliance Programs website at <https://aws.amazon.com/compliance/programs/>.

**Step 2.** Review the IT standards that Amazon currently complies with.

**Step 3.** Select a standard and review the following details:

1. What level of certification does AWS hold?
2. What services are in the scope of the selected compliance program?
3. What does AWS's attestation of compliance cover?
4. What controls is the customer responsible for?
5. Do your organization's compliance needs match successfully with what AWS currently offers?

A detailed review of the scope of the IT standards that your organization must adhere to is the best first step to follow when analyzing what is possible in the AWS cloud. If you find that an AWS service you were thinking of using is not yet supported by

your compliance rules, it's certainly better to know that before starting.

You might prefer to visit an AWS data center and perform your own auditing, but that is not possible. Random visitors to a data center adhering to a high level of compliance are just not allowed. Even Jeff Bezos would have trouble getting into an AWS data center unless he was cleared to perform data center-specific tasks.

For example, suppose you work for a company that wants to analyze AWS for its level of Payment Card Industry Data Security Standard (PCI DSS) compliance. Your company is designing a workload for customers to pay their bills online. AWS will be hosting the SaaS workload, and you are tasked with reviewing and understanding both your company's responsibilities and the responsibilities of AWS. The compute services for the SaaS workload are to be hosted on the shared EC2 infrastructure at AWS, if that's allowed. The workload must adhere to the rules of PCI DSS compliance in terms of the storage, processing, and transmission of the credit card information. Several vice presidents are sure that PCI compliance is not allowed in the AWS cloud, or, if it is allowed, bare-metal servers must be used. To get the most up-to-date answer about PCI DSS, follow these steps:

**Step 1.** From AWS Artifact in the AWS Management Console, download the PCI DSS Attestation of Compliance (AoC) and Responsibility Summary.

**Step 2.** Read the summary report and review the services that are within the scope of PCI DSS and determine whether the standard virtual environment for computing and storage at AWS is PCI compliant for both Amazon's and your responsibilities.

**Step 3.** Visit <https://aws.amazon.com/compliance/programs/> scroll down, and click PCI DSS Level 1.

**Step 4.** Under PCI DSS Resources on the right side of the page, select the “Compliance Guide: PCI DSS 3.2.1 on AWS” and review requirements for Amazon, requirements for the cloud service provider (CSP), and the responsibilities of the client, as shown in [Table 7-4](#).

**Table 7-4** PCI Checklist for Compliance

| PCI DSS Requirement | Details | Customer Responsibilities | AWS Response |
|---------------------|---------|---------------------------|--------------|
|                     |         |                           |              |

| PCI DSS Requirement                                       | Details                   | Customer Responsibilities                              | AWS Response                       |
|---|---------------------------|--|------------------------------------|
| Install and maintain firewalls to protect cardholder data | Shared responsibility     | Firewalls on instances, firewalls at the subnet levels | Firewalls in cloud between clients |
| Don't use vendor defaults for passwords and security      | Customer's responsibility | Workloads  | Network security                   |
| Protect stored cardholder data                            | Customer's responsibility | Encrypted data   | Secure arrays                      |

| PCI DSS Requirement  | Details                   | Customer Responsibilities | AWS Response      |
|--|---------------------------|---------------------------|-------------------|
| Encrypt transmission of cardholder data across public networks | Customer's responsibility | Encrypted transmissions   | Supplier security |
| Use and update antivirus software                              | Customer's responsibility | Third-party software      | N/A               |
| Develop and maintain secure systems                            | Shared responsibility     | EC2 instance              | Hyper security    |

| PCI DSS Requirement                                   | Details                   | Customer Responsibilities        | AWS Responsibility    |
|---|---------------------------|----------------------------------|-----------------------|
| Develop and maintain secure workloads                 | Customer's responsibility | On the instance                  | Hyper security        |
| Restrict access to cardholder data                    | Customer's responsibility | Access controls                  | Physical security     |
| Assign unique IDs to each person with computer access | Customer's responsibility | Access controls, password policy | Strong authentication |

| PCI DSS Requirement   | Details                 | Customer Responsibilities      | AWS Responsibility                           |
|---|-------------------------|--------------------------------|--|
| Restrict physical access to cardholder data                           | Amazon's responsibility | N/A                            | Physical                                     |
| Track and monitor all access to network resources and cardholder data | Shared responsibility   | Customer's virtual environment | Physical infrastructure and hybrid           |
| Test security systems and processes regularly                         | Shared responsibility   | Customer's virtual environment | Intrusion detection system prevention system |

| PCI DSS Requirement  | Details                 | Customer Responsibilities  | AWS Response          |
|--|-------------------------|----------------------------|-----------------------|
| Define security policies for information security            | Shared responsibility   | Customer's security policy | ISO security policies |
| Additional PCI DSS requirements for shared hosting providers | Amazon's responsibility | N/A                        | PCI DS                |

## AWS Compliance Standards

AWS supports several compliance programs for a variety of businesses running in regulated industries, including financial services, healthcare, and the U.S. government (see [Table 7-5](#)). AWS compliance programs support global standards and countries grouped by Americas, Asia Pacific and Europe, Middle

East and Africa. For further details look here:  
<https://aws.amazon.com/compliance/programs/>.

**Table 7-5** North American Compliance Frameworks

| Compliance Framework                              | Details   |
|---|---|
| CJIS Criminal Justice Information Services        | Workloads for state and federal law enforcement agencies at AWS       |
| Family Educational Rights and Privacy Act (FERPA) | Educational agencies and institutional storage of data records at AWS |

| Compliance Framework                                       | Details  |
|--|--|
| Federal Financial Institutions Examination Council (FFIEC) | Rules for federal financial institutions on the use and security of AWS services at AWS  |
| Federal Information Security Modernization Act (FISMA)     | Security authorizations for government agencies using systems hosted at AWS, adhering to NIST 800-37 standards in AWS GovCloud |
| GxP  | Rules and guidelines for food and medical products data hosted at AWS  |
| HIPAA  | Rules for processing, storing, and transmitting protected health information at AWS  |

| Compliance Framework                             | Details  |
|--|--|
| International Traffic in Arms Regulations (ITAR) | Compliance with ITAR in AWS GovCloud   |
| Motion Picture Association of America (MPAA)     | Rules for securely storing, processing, and delivering protected media and content   |
| NIST SP 800-53                                   | Security controls applied to U.S. federal information systems to ensure confidentiality, integrity, and availability (CIA) |

| Compliance Framework  | Details   |
|---|---|
| Voluntary Product Accessibility Template (VPAT)/Section 508 | Rules for developing electronic and information technology for people with disabilities |

In addition, AWS supports some well-known global compliance programs. The SOC 2 audit is a good place to start in reviewing available security controls at AWS

(<https://aws.amazon.com/compliance/soc-faqs/>). All current compliance certifications that AWS is aligned with are audited and assessed on a regular schedule using independent third-party auditors.

## HIPAA

If your business needs to comply with the 1996 Health Insurance Portability and Accountability Act (HIPAA), you must provide protections for what is defined as protected health

information (PHI). Each healthcare provider—defined as the “covered entity” using AWS services to architect and host its workloads—is solely responsible for complying with the HIPAA rules and regulations. In HIPAA terminology, Amazon is defined as having a business associate role. HIPAA is applicable in the United States. Depending on where you live, there may be other compliance standards that are relevant to your country. Make sure to review the compliance standards and compliance certifications that AWS currently supports by visiting

<https://aws.amazon.com/compliance/programs/>.

Since 2013, Amazon has provided a signed contract called a Business Associate Addendum (BAA). In this contract, Amazon promises to safeguard the stored healthcare information properly and lays out the rules and responsibilities that AWS is undertaking, including a list of the services and tasks that AWS will carry out on the customer’s behalf. Each customer who enters a BAA with AWS must use only the defined HIPAA-eligible AWS services defined in the BAA.

Many common services available at AWS are now allowed by HIPAA regulations. However, to be sure, check the current AWS compliance documentation because certifications and regulations are constantly in flux. For example, the AWS Systems Manager Console and Resource Groups are currently

not in scope for HIPAA. However, the VPC, encrypted EBS volumes, EC2 instances, and S3 storage are all supported under the HIPAA BAA; each customer decides how encryption is enabled for each service. A great place to start learning about compliance is by reviewing the AWS Services in Scope by Compliance Program (see <https://aws.amazon.com/compliance/services-in-scope/>).

## NIST

Your business might align its compliance rules with the National Institute of Standards and Technology (NIST). In 2011, NIST presented preliminary documentation on what the public cloud industry was doing up to that point. The NIST definitions have morphed into a set of standards that many organizations, including the U.S. government, must achieve and maintain (<https://nvlpubs.nist.gov/nistpubs/legacy/sp/nistspecialpublication800-145.pdf>).

There is a wealth of NIST documentation to review at <https://www.nist.gov> that deals with cloud security, virtualization, operations, and many other areas of the public cloud. Even if you are not bound by a compliance standard, reviewing and following NIST recommendations can help you in developing your organization's required level of security in

the AWS cloud. Visit this link for a summary of the features, benefits, risks, and recommendations for cloud computing for government agencies: <https://www.nist.gov/publications/cloud-computing-review-features-benefits-and-risks-and-recommendations-secure-efficient>.

AWS is compliant with the NIST 800-37 security controls, which allow U.S. government agencies to achieve and sustain compliance with FISMA. FISMA was passed into law in 2002 (and at the time stood for Federal Information Security Management Act) and required federal agencies to follow a set of standards when implementing their information security programs. FISMA focuses on properly managing and securing federal data records.

AWS has also obtained Federal Risk and Authorization Management Program (**FedRAMP**) authorization to operate its GovCloud regions. The U.S. federal government is using AWS to deliver some cloud services, and AWS must adhere to and demonstrate compliance with the FedRAMP standard, which is why the AWS GovCloud (US East/West) regions exist. The GovCloud region us-gov-west-1 is categorized as a high-impact level (Level 5); for comparison, public us-east-1 and us-west-2 regions are categorized as a moderate impact level (Level 2). Consider the following definitions:

- **NIST 800-53 Rev. 5:** This publication provides security guidance for security controls, including access control, auditing and accountability, configuration management, risk assessment, and incident response (<https://csrc.nist.gov/publications/detail/sp/800-53/rev-5/archive/2020-09-23>).
- **FedRAMP:** A defined methodology for security assessments, authorization, and continuous monitoring of cloud services and products.
- **High impact level:** Unclassified national security data.
- **Moderate impact level:** Publicly releasable data.
- **FISMA:** All data should be classified based on its sensitivity and automatically encrypted. If you work in the U.S. federal government or any state government, FISMA applies to you, and you must meet these requirements:
  - Maintain an up-to-date inventory of all information systems deployed
  - Categorize each information system's level of risk to the appropriate FISMA risk level
  - Create and maintain for your organization a security plan that details security policies and controls
  - Carry out risk assessments against your business processes, security risks at the organizational level, and risks at the information systems level

- Conduct annual security audits

## AWS GovCloud

AWS GovCloud is an isolated region that has been designed to match the regulatory compliance requirements of the U.S. government, federal and state agencies, and anybody else who works with the U.S. government (see [Figure 7-14](#)). It's important to note that all AWS regions and services have the same level of security. The differences with GovCloud follow:

- Vetted U.S. citizens manage the operations of GovCloud.
- AWS GovCloud is accessible only to U.S. citizens or green card root account holders.
- GovCloud is physically and logically isolated from all other AWS regions.
- GovCloud has provisional authorization for Department of Defense SRG Impact Level 2 (publicly releasable data), Impact Level 4 (unclassified sensitive data), and Impact Level 5 (unclassified national security data) workloads.
- GovCloud supports HIPAA, CJIS, and ITAR regulations.
- Certain AWS services, especially newly introduced services, will not be in scope for the GovCloud regions. New AWS services are reviewed for inclusion in GovCloud at both the moderate and the high impact levels.



**Figure 7-14** Authenticating to the GovCloud Region

## Latency Concerns

### Key Topic

Network latency, typically measured in milliseconds, tells you how much time a packet takes to travel from its source location to its destination. The interface between your end users and a hosted AWS workload is typically a browser or app interface that communicates with the hosted workload or website hosted

at AWS. Latency in this example is how much time packets take to travel from your device to AWS and back to your location. Workloads and services hosted in the cloud will be influenced by latency. Connecting to workloads across the Internet is unpredictable because Internet speeds are unreliable. If any workload at AWS needs to query a database located in the corporate data center, latency can be an issue. Slow connection problems can be solved with faster private connection speeds such as AWS Direct Connect; however, the workload that you need to connect to may be available only across the Internet.

Because most, if not all, workload requests require some form of data transfer, using a content distribution network such as AWS CloudFront to cache data records or web pages can greatly help in speeding up workload access. For example:

- If you're accessing any workload hosted at AWS from a device such as a phone, computer, tablet, or watch, you will connect to your workload most likely across the Internet. In the background, Amazon's DNS service, Route 53, helps direct you to the public endpoint of the hosted workload at AWS.
- If you're planning to move workloads from your corporate data centers to the AWS cloud and are planning to operate in a hybrid model (with resources remaining in the corporate data center and some resources hosted in the cloud), the

location of your data centers might influence the AWS region chosen and the types of network connections to use.

- For a public-facing SaaS workload, companies typically choose a region closest to where customers are located. Deploying Amazon's content delivery network (CDN), CloudFront, will help remediate latency concerns for customers located globally.

Latency can also be solved with the following AWS solutions:

- **Connecting to a hosted workload in another geographic location:** Connection speeds across the Internet to AWS are slower the farther away your user location is. AWS Global Accelerator can route your user traffic to a hosted workload at AWS using the closest edge location speeding up access.
- **Transferring data records to an S3 bucket:** Amazon S3 Transfer Acceleration speeds up content transfers to Amazon S3 storage location using Amazon CloudFront edge as the ingress entry point onto the AWS private network, speeding up the transfer of files over long distances between the end user and the S3 bucket location.
- **Replicating MySQL DB clusters:** Read replicas are read-only copies of the source database records. Read replicas can be geographically located in specific AWS regions to speed up queries for end users located in different geographical

locations. Since read-replica updates are performed with asynchronous replication, replicas can contain stale information from time to time until replication has completed. Another solution is to deploy Amazon Aurora across multiple AWS regions. Replication from the primary DB cluster to all secondary read-only clusters is handled by the Aurora storage layer instead of the database engine, resulting in replicated changes in less than one second.

## Services Offered in Each AWS Region

Not all AWS services are available in all regions; it can take a long time for a recently introduced cloud service to become available in every AWS region. *Generally available* is a term that AWS uses to indicate that a specific AWS service is operating in most, if not all, AWS regions. The core infrastructure services including EC2 instances, RDS, ELB, S3 buckets, virtual private networking, and many others are available in every AWS region.

Each new AWS service is typically offered in a *preview mode*, where the service is initially available to a limited number of users that sign up to test the new service. Some services can take years to change from preview status to full online status. (For example, Amazon Elastic File System [EFS] was in preview

mode for quite a long time before it became generally available across all AWS regions.)

The cloud services being offered in each AWS region might dictate what regions you choose. Whereas core compute, networking, and storage services are available everywhere, newer AWS services take time to become generally available in all regions. As a general rule, the newer the service, the longer it will take for the service to be deployed in all regions.

Your compliance rules and regulations may also determine your service availability. For example, if you are bound by FedRAMP rules and regulations, there will typically be several services and management tools that are not approved for use at this time—and that perhaps never will be.

## **Calculating Costs**

It is important to understand costs when dealing with a cloud provider. The AWS mantra is to pay for what you use. AWS uses a consumption-based model; as you use a higher volume of AWS services, you pay more—but on a sliding scale. Some costs are bundled into the prices of services; for example, there's no charge to order an AWS VPC and add subnets. However, there are charges when EC2 instances are deployed, and there are

additional costs for replicating compute instance traffic across availability zones. Adding additional EC2 when running workloads designed for high availability and failover will also increase overall costs.

AWS pricing depends on the region and AZs that you have selected to operate in; you can pay a little or a lot more for the same service in comparison to other AWS regions. An organization might have a limited choice of regions if compliance rules for your industry dictate where you can operate in the cloud. For example, choosing the AWS Canada Central region ca-central-1 means you are using AWS resources and AZs located in Canada, and depending on the currency exchange rate, pricing in Canada might be less than what you would pay in your home country. Is your company allowed to operate in Canada? During the planning and design phases of moving to the AWS cloud, it is important to carefully consider the compliance rules that your company may have to follow.

The biggest and least expensive AWS region is the us-east-1 (Northern Virginia) region, offering the largest number of availability zones and AWS services. Another region with comparable pricing and service availability to Northern Virginia is us-east-2 (Ohio). For the EU or São Paulo regions, you can expect pricing to be a few hundred percent more!

Don't forget that the monthly costs that customers pay for their workload infrastructure will increase as the requirements for high availability and improved reliability increase. Customers can greatly reduce their EC2 instance pricing by ordering Reserved Instances and saving on other compute costs with Savings Plans. ([Chapters 12, 13, 14](#), and [15](#) provide detailed information about managing cloud costs.)

## Distributed Design Patterns

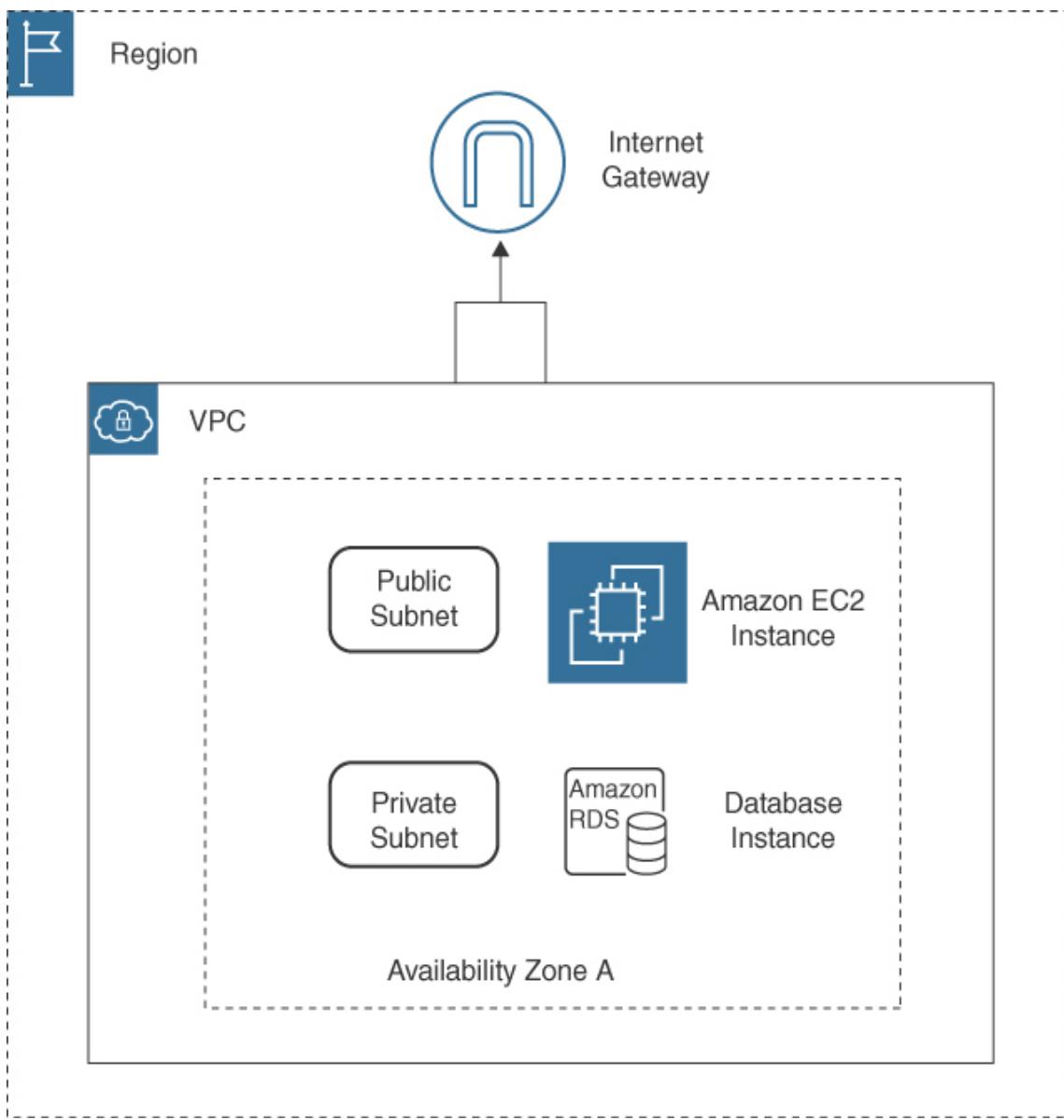
Successful workloads hosted at AWS have high availability and fault tolerance baked into the underlying design. As you prepare for the AWS Certified Solutions Architect – Associate (SAA-C03) exam, you need to understand these concepts conceptually in conjunction with the services that can provide a higher level of redundancy and availability. After all, if a workload is designed to be highly available, it will have a higher level of redundancy and will continue to function at an acceptable working level when issues occur.

The same is true with a fault-tolerant design—if your workload is hosted on infrastructure that has been designed with tolerance for issues such as latency, performance, and durability issues, your workload will be able to operate most of the time successfully.

## Designing for High Availability and Fault Tolerance



Imagine that your initial foray into the AWS cloud is to host a monolithic workload at AWS. Although this is possible using a “lift and shift” cloud migration tool that moves the virtual server from the on-premises data center to the AWS cloud, there will eventually be issues with availability. The single EC2 instance will eventually fail or need to be taken down for maintenance. A failure of the EC2 instance, as shown in [Figure 7-15](#), impacts all clients. There are also design issues related to the workload state and data records stored on a single EC2 instance. The old-style monolithic workload will not function well on premises nor in the cloud.

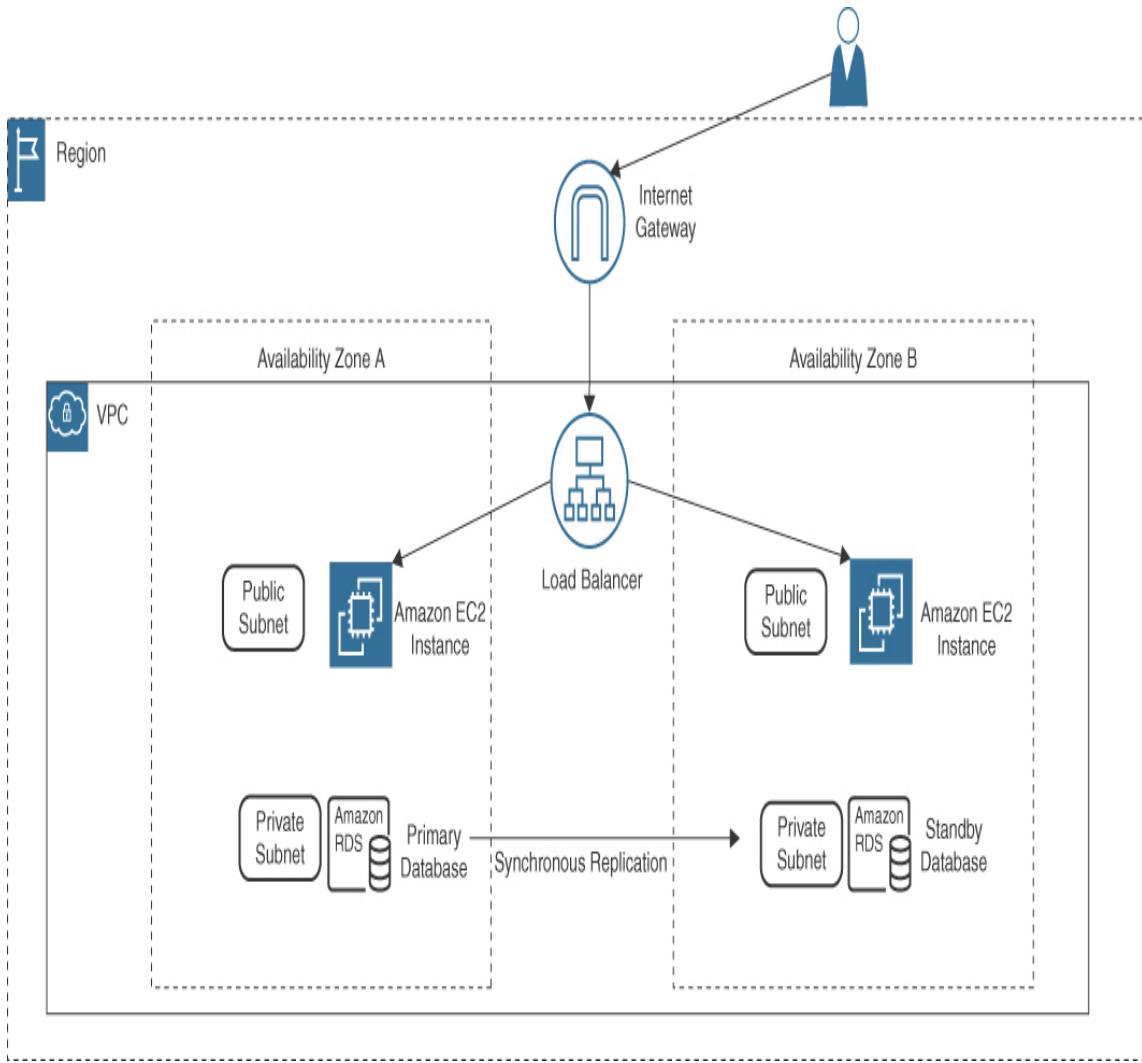


**Figure 7-15** Starting AWS Infrastructure

To add high availability to this design, as a first step you could split the monolithic workload into a web tier with multiple EC2 instances hosted across multiple availability zones receiving workload requests from a load balancer, as shown in [Figure 7-](#)

**16**, providing high availability and additional redundancy. Therefore, a database tier is also added, hosting the data records on at least two database servers. The primary database server replicates all changes to the secondary database using synchronous replication. The secondary database must communicate to the primary database that the data transferred has been successfully received and written before additional records will be sent.

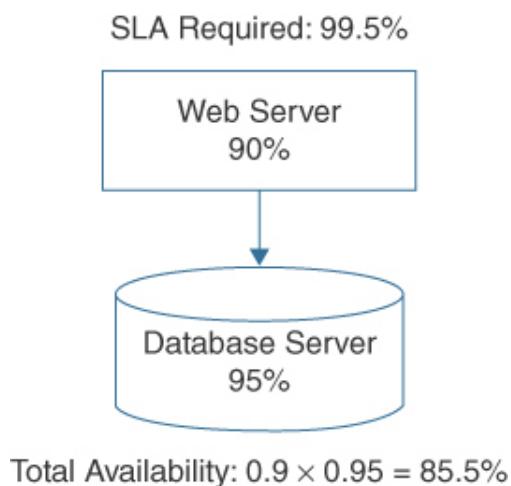




**Figure 7-16** Hosting Across Multiple AZs to Increase Availability and Fault Tolerance

AWS data centers and AZs are built for deploying high-availability designs connected together with high-speed/low-latency links and are located far enough away from each other that a single natural disaster won't affect all of the AZs in each AWS region at the same time.

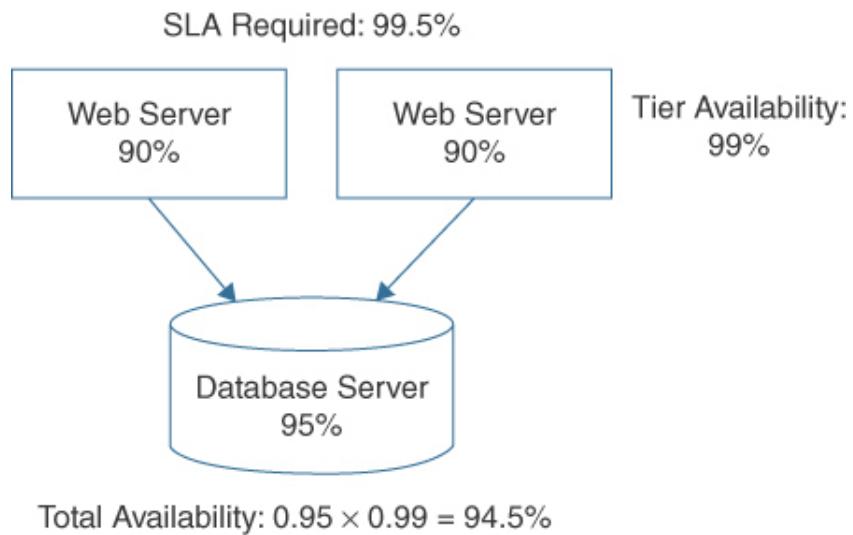
If the mandate for this workload is that it should be down for no more than 4.5 hours in any given year, this equates to 99.5% **uptime**. Internal reviews of the current infrastructure show that over the past year, the web server has 90% availability and the database server has 95% availability, resulting in total availability of 85.5%, as shown in [Figure 7-17](#).



**Figure 7-17** Availability Calculation for a Simple Hosted Workload

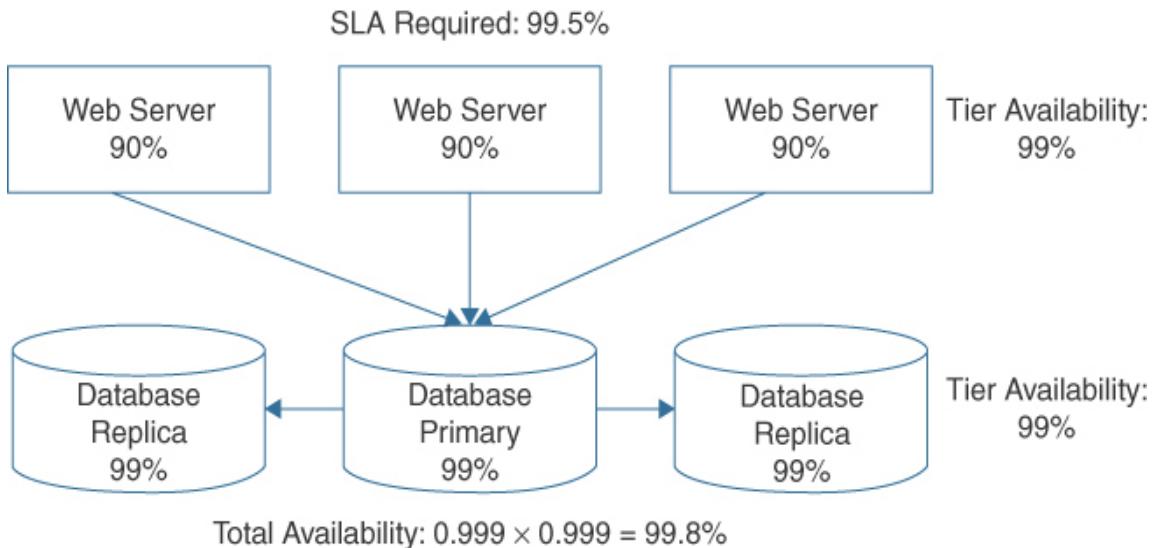
This amount of uptime and availability obviously needs to be increased to meet the availability mandate of the business requirements. Adding an additional web server to the design increases the overall availability to 94.5%, as shown in [Figure 7-18](#).

**Key Topic**



**Figure 7-18** Increasing Workload Availability by Adding Compute Workload

The design now has some additional high availability and built-in fault tolerance for the web tier, so if one server fails, the workload will still be available. Next, the database tier availability needs to be increased. Adding two replica database servers to the existing database infrastructure in addition to adding a third web server to the web server tier results in both tiers achieving a total availability of 99.8%, exceeding the **SLA** goal, as shown in [Figure 7-19](#).



**Figure 7-19 Adding Availability to Both Tiers**

## Removing Single Points of Failure

Eliminating as many single points of failure as possible in your workload design will greatly increase workload high availability and fault tolerance. A *single point of failure* is a critical component in a system that, if it fails, will cause the entire system to fail. Single points of failure are a major concern in system design because they can lead to significant disruptions and downtime. To avoid single points of failure, design with redundant components or backup systems in place, ensuring that the application remains operational even if a critical component fails. Take some time to review [Table 7-6](#), which discusses possible mitigation plans for single points of failure.

**Key Topic****Table 7-6** Avoiding Single Points of Failure

| Possible Single Point of Failure | Mitigation Plan | Reason   |
|----------------------------------|-----------------|--|
| On-premises DNS                  | Route 53 DNS    | Anycast DNS services are hosted across all AWS regions and edge locations.<br>Health check and traffic patterns support workload failover. |

### Possible Single

| Point of Failure          | Mitigation Plan                       | Reason  |
|---------------------------|---------------------------------------|---|
| Third-party load balancer | Elastic Load Balancing (ELB) services | ELB load-balancers are deployed per AZ. Elastic IP addresses provide fast failover. |

|               |                  |  |
|---------------|------------------|--|
| Web workloads | ELB/Auto Scaling | Scale workload compute resources automatically up and down to meet demand. |
|---------------|------------------|--|

## Possible Single

| Point of Failure     | Mitigation Plan                                | Reason  |
|----------------------|--|---|
| RDS database servers | Redundant data nodes (primary/standby/cluster) | Synchronize replication between primary and multiple standby nodes provides a minimum of three or more copies of data |

### Possible Single

| Point of Failure | Mitigation Plan                  | Reason   |
|------------------|----------------------------------|--|
| EBS data volumes | Snapshots and retention schedule | Automatical copy snapshots across regions for additional redundancy using Lifecycle Manager. |

### Authentication problems

Redundant authentication nodes

Multiple Managed AD domain controllers provide alternate authentication options.

| Possible Single Point of Failure |                                       |  |
|----------------------------------|---------------------------------------|--|
| Point of Failure                 | Mitigation Plan                       | Reason   |
| Data center failure              | Multiple availability zones           | Each region has multiple AZs providing high-availability and failover design options.          |
| Regional disaster                | Multi-region deployment with Route 53 | Route 53 routing policy provides geographic redundancy for workloads hosted across AWS regions |

[Table 7-7](#) lists AWS services that can be improved with high availability, fault tolerance, and redundancy.

**Key Topic****Table 7-7** Planning for High Availability, Fault Tolerance, and Redundancy

| AWS Service   | High Availability               | Fault Tolerance             |
|---------------|---------------------------------|-----------------------------|
| EC2 instance  | Additional EC2 instance         | Multiple availability zones |
| EBS volume    | Cluster design                  | Snapshots                   |
| Load balancer | Multiple AZs                    | Elastic IP addresses        |
| Containers    | Elastic Container Service (ECS) | Fargate management          |

| AWS Service               | High Availability                              | Fault Tolerance                              |
|---------------------------|--|--|
| RDS deployment            | Multiple AZs                                   | Synchronous replication                      |
| Custom EC2 database       | Multiple AZs and replicas                      | Asynchronous/synchronous replication options |
| Aurora (MySQL/PostgreSQL) | Six copies of data replicated across three AZs | Multiple writers                             |

|                  |  |                  |
|------------------|--|------------------|
| AWS Service      | High Availability                              | Fault Tolerance  |
| DynamoDB (NoSQL) | Six copies of data replicated across three AZs | Multiple writers |

|          |               |                  |
|----------|---------------|------------------|
| Route 53 | Health checks | Failover routing |
|----------|---------------|------------------|

|           |                         |          |
|-----------|-------------------------|----------|
| S3 bucket | Same-region replication | Built-in |
|-----------|-------------------------|----------|

## Immutable Infrastructure

Immutable architecture is a design in which select components of a workload, such as a web or application server, are replaced rather than modified or updated. When changes need to be made, the old components are discarded and new, updated components are put in their place. This approach has several advantages over mutable architecture, in which resources are

modified in place. Because immutable architectures use new resources, they are not affected by changes that might have been made to the old resources. Because older components are discarded rather than modified, it is easier to revert to a previous version of the system if necessary. The term **immutable**, when applied to the AWS cloud, defines resources deployed to a production environment by replacing the existing version of the EC2 instance or image (AMI) with a new, updated version that should not change when they are deployed. For example, an EC2 instance, after being deployed, may get security updates from the Internet. Because of the updates there is no guarantee that the installed dependencies have not changed since the last deployment if immutable deployments are not the norm. With immutable servers, any changes result in new servers being deployed instead of updating the servers currently in production. Changes are not allowed in immutable infrastructure; therefore, the state of an immutable server is always known.

Creating an immutable server starts with an image called a golden image or golden AMI. Starting with the base image, all unnecessary software packages and tools are removed. Security patches and approved configurations are applied to harden the image. After the image is tested and approved, it is locked and ready for deployment in the production environment.

When workloads are deployed using immutable infrastructure, canary deployments should be used to minimize the risk of failure when new server versions are deployed into production. *Canary deployments* are a technique for releasing software updates, and the new version of the software is initially made available to a small subset of users. This enables the update to be tested in a production environment before it is made available to the larger user base. If the software update performs as expected and no issues are discovered, it is gradually rolled out to additional users. However, if any problems are found, the update can be quickly rolled back without affecting the entire user base.

Immutable deployments have no upgrades, because upgrades have been replaced with new deployments (see [Figure 7-20](#)).

Administrative access is not required to production servers preventing undocumented manual changes and hotfixes. All servers (web, application, database) that are part of the same workload are version controlled.

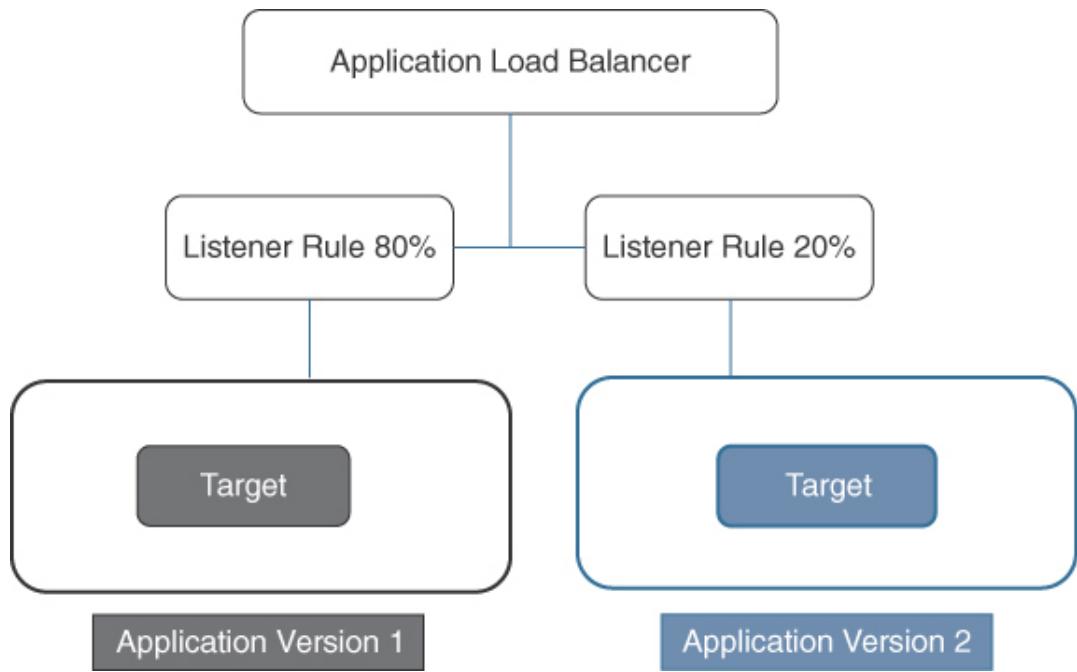
### Key Topic



**Figure 7-20** Deploying Immutable Servers

Immutable infrastructure is supported by the following services at AWS:

- AWS CloudFormation can test deployed stacks to determine if any changes, called drift, have occurred on the deployed stack resources when compared to the initial stack template.
- AWS Elastic Beanstalk supports immutable deployments of EC2 instances and Docker containers.
- Amazon EC2 Auto Scaling supports instance refresh that enables automatic deployments of new workload versions using Auto Scaling Groups (ASG).
- ELB Application and Network Load Balancers supports A/B and canary deployments by routing requests to different weighted target groups based on listener rules, as shown in [Figure 7-21](#). Developers can control the distribution of traffic to multiple versions of their workload using two target groups with weights of 80 and 20; the load balancer will route 80 percent of the traffic to the first target group and 20 percent to the second target group.



**Figure 7-21** Weighted ALB Target Groups

- Multiple VPCs can be created for dev, test, and production environments.
- Amazon EC2 Image Builder, in conjunction with AWS CodePipeline, supports the building, testing, and deployment of virtual machine and container images for immutable deployments.

---

#### Note

AWS X-Ray can be used to analyze workload behavior using end-to-end tracing identifying performance issues and errors for individual workload requests to workloads running on EC2

instances and Amazon ECS, AWS Lambda, Kinesis, DynamoDB, and AWS Elastic Beanstalk.

---

## Storage Options and Characteristics

Storage services also utilize availability zones for their replicated data center storage locations. Each storage service listed in [Table 7-8](#) can operate in a single AZ with redundancy and high availability or across multiple AZs within the chosen AWS region of deployment.



**Table 7-8** Storage Service Characteristics

| Storage    | Durability  | Replication                                       |
|------------|---|---|
| Amazon EBS | Multiple storage volumes, snapshots stored in Amazon S3 | Replicated within each AZ across multiple servers |
| Amazon FSx | Multiple storage volumes, automatic                     | Single or Multi-AZ deployments                    |

## backup to Amazon S3

|            |   |                                |
|------------|---|--------------------------------|
| Amazon EFS | Multiple storage volumes, automatic backup to Amazon S3 | Single or Multi-AZ deployments |
|------------|---|--------------------------------|

|                    |                      |  |
|--------------------|----------------------|--|
| Amazon S3 Standard | Eleven 9s durability | Across three AZs or more depending on AWS Region |
|--------------------|----------------------|--|

|                    |                      |                          |
|--------------------|----------------------|--------------------------|
| Amazon S3 One Zone | Eleven 9s durability | Single availability zone |
|--------------------|----------------------|--------------------------|

|                                   |                      |                  |
|-----------------------------------|----------------------|------------------|
| Amazon S3 Glacier<br>Deep Archive | Eleven 9s durability | Across three AZs |
|-----------------------------------|----------------------|------------------|

|                 |                                 |   |
|-----------------|---------------------------------|---|
| Amazon DynamoDB | SSD volumes on multiple servers | Across three AZs, or multiple AWS regions |
|-----------------|---------------------------------|---|

|            |             |                               |
|------------|-------------|-------------------------------|
| Amazon RDS | EBS volumes | Across one, two, or three AZs |
|------------|-------------|-------------------------------|

|               |                                       |   |
|---------------|---------------------------------------|---|
| Amazon Aurora | SSD volumes<br>cluster shared storage | Across three AZs, or multiple AWS regions |
|---------------|---------------------------------------|---|

|            |                        |                  |
|------------|------------------------|------------------|
| AWS Backup | Eleven 9s availability | Across three AZs |
|------------|------------------------|------------------|

---

#### Note

Use the EBS lifecycle service to schedule EBS and remove unneeded snapshots. Use S3 versioning and lifecycle policies to purge old object versions when they are no longer required.

---



## Failover Strategies

Each customer must use the tools and services available at AWS to design their workload environment to be able to withstand failures when they occur. Workloads must be designed to be redundant and proper backups must be available to help restore the customer's workload when required. A disaster recovery (DR) strategy must be designed and implemented for workloads running at AWS.

High-availability designs deployed across multiple availability zones protect against disruptions in a single AZ, as the other AZs will continue to function, providing workload availability and no data loss. If failures must be prevented against larger potential disasters involving multiple AZs, backups of key services and data records can be stored in other regions, or multiple AWS regions can be used for active-passive or active-active deployments.

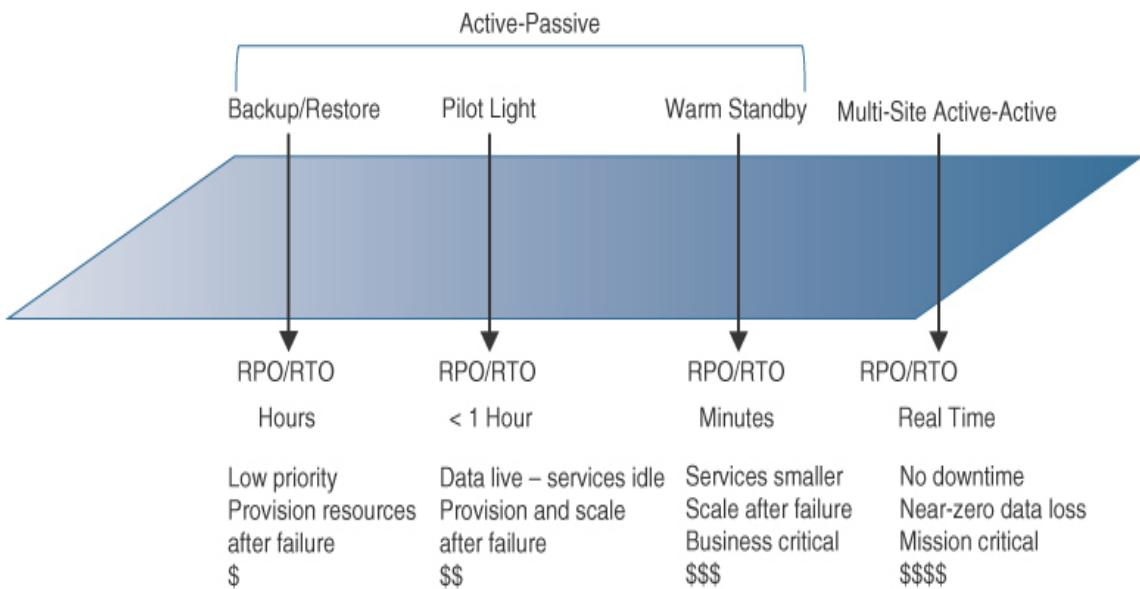
It's important to understand the published uptime figures for the AWS cloud. Published uptime figures are not guarantees; instead, they are what AWS strives for—and typically achieves. Route 53, AWS's DNS service, is designed for 100% uptime, but issues still may occur without warning.

Failures are going to happen. Disaster recovery procedures focus on carrying out the recovery steps after a disaster has

occurred. For each workload, acceptable recovery objectives must be defined. There are two generally agreed-upon metrics that define how a disaster recovery process should be designed: What is the recovery objective when a disaster occurs? What is the acceptable amount of data that can be lost when a disaster occurs?

- **Recovery point objective (RPO):** *RPO* is defined as the maximum acceptable amount of time since the last data recovery point or the acceptable loss of data. If your RPO is defined as 5 minutes, then the disaster recovery process needs to be designed to restore the data records to within 5 minutes of when the disaster occurred.
- **Recovery time objective (RTO):** *RTO* is defined as the maximum acceptable delay between a service interruption and a return to normal service. If RTO is set to 1 hour, for example, the workload should be up and functional within that 1-hour timeframe.

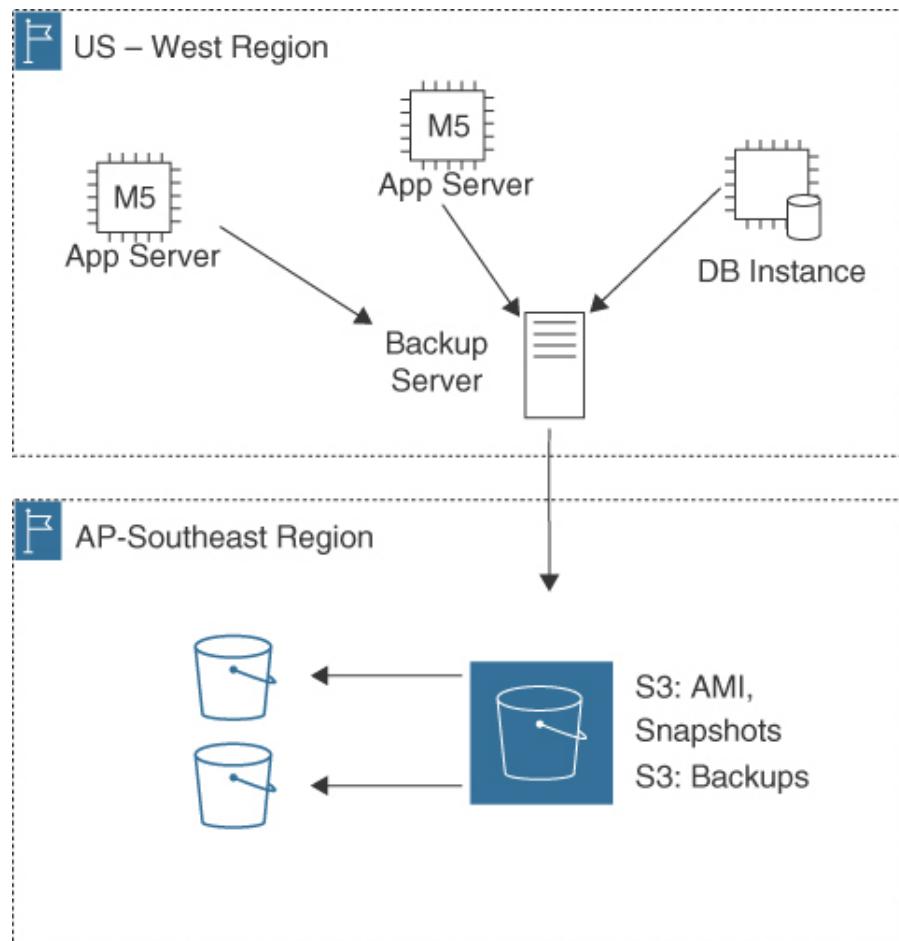
Customers need to define acceptable values for RPO and RTO, defining a recovery strategy that meets the recovery objectives of each workload. Strategies chosen include active-passive (backup and restore, pilot light, or warm standby), or multi-site active-active (see [Figure 7-22](#)).



**Figure 7-22** Disaster Strategies Compared

## Backup and Restore

Backup and restore strategies back up data and workloads into a DR location. The DR location could be a separate AWS region, as shown in [Figure 7-23](#), or an availability zone within a select AWS region. Using a Multi-AZ strategy within a single AWS region may be sufficient for some workloads; it will depend on the specific use case and business requirements. For an on-premises workload, a hybrid architecture could be deployed, allowing operation in both locations using AWS Direct Connect providing a dedicated high-speed connection.



**Figure 7-23** Multi-Region Backup and Restore

The following AWS services can help assist in creating a backup and restoration plan:

- **Amazon Machine Images:** Images can be manually copied to multiple regions for safekeeping. Amazon Data Lifecycle Manager and AWS EC2 Image Builder can automate the creation and retention of Amazon EBS-backed AMIs across multiple AWS regions and AWS accounts.

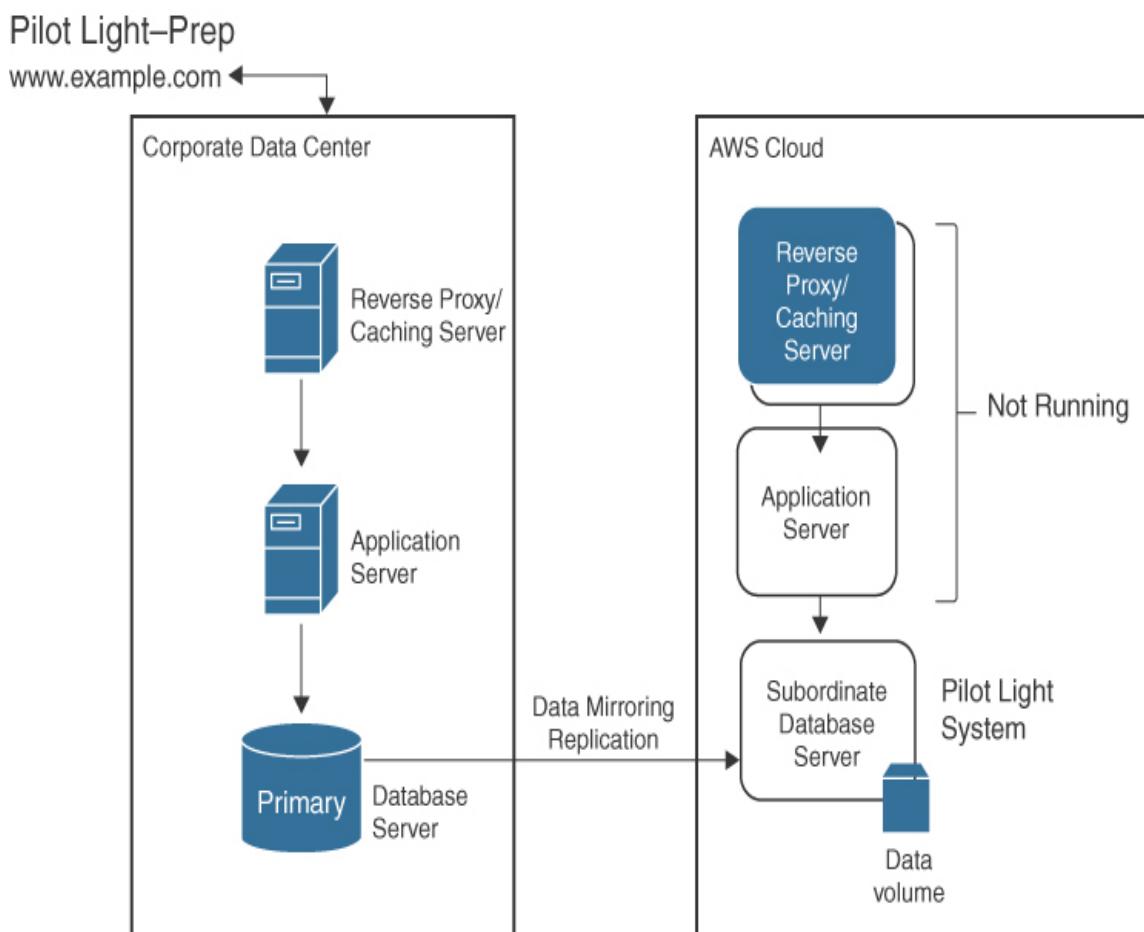
- **Amazon EBS Snapshots:** Snapshots can be manually copied to multiple regions for safekeeping. Amazon Data Lifecycle Manager can automatically copy snapshots on a schedule across up to three regions. The Data Lifecycle Manager also supports cross-account sharing, automatically sharing and copying snapshots across AWS accounts.
- **Amazon Outposts:** Outposts allows customers to deploy Amazon RDS database engines as a hybrid cloud database deploying RDS to an on-premises data center across a Direct Connect connection. Backups and snapshots are stored locally or in an AWS region.

## Pilot Light

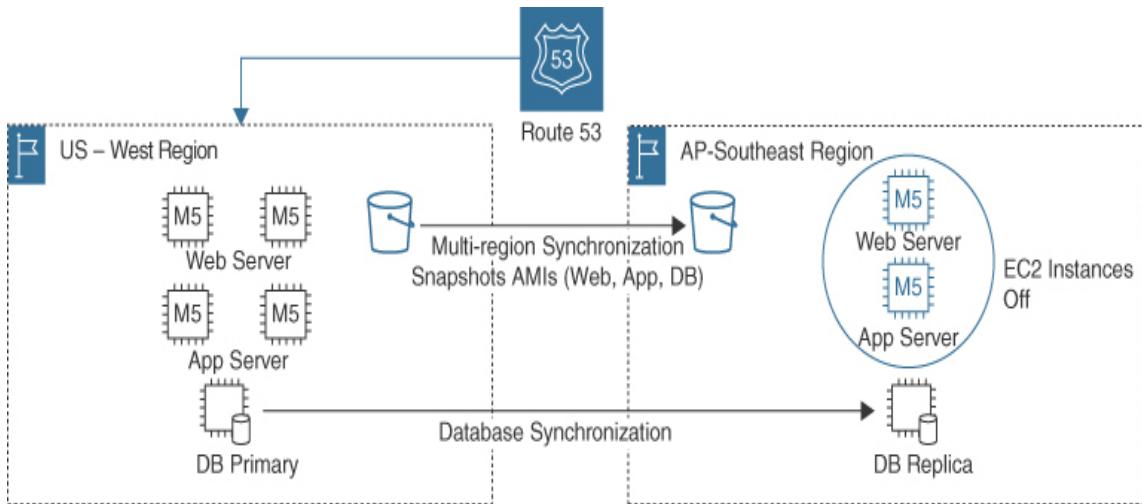
A disaster recovery pilot light is a disaster recovery strategy in which a minimal system is kept running at all times, ready to be expanded into a full-scale workload in the event of a disaster.

Pilot light deployments could have synchronized database records and idle compute services. For example, a pilot light disaster recovery strategy has a defined online primary site where web and primary database servers are operating and are fully operational. The primary site could be on premises, as shown in [Figure 7-24](#), an AWS availability zone or regional location, as shown in [Figure 7-25](#). The standby web and application servers are ready to launch but are not powered on.

The primary database server located in the primary site replicates data updates and changes to the standby database server. When planning the AWS location for your disaster recovery site, the compliance rules and regulations that your organization adheres to may dictate which region should be used. In addition, you might want the DR location to be as close as possible to your physical corporate location if latency is a concern.



**Figure 7-24** Pilot Light Setup

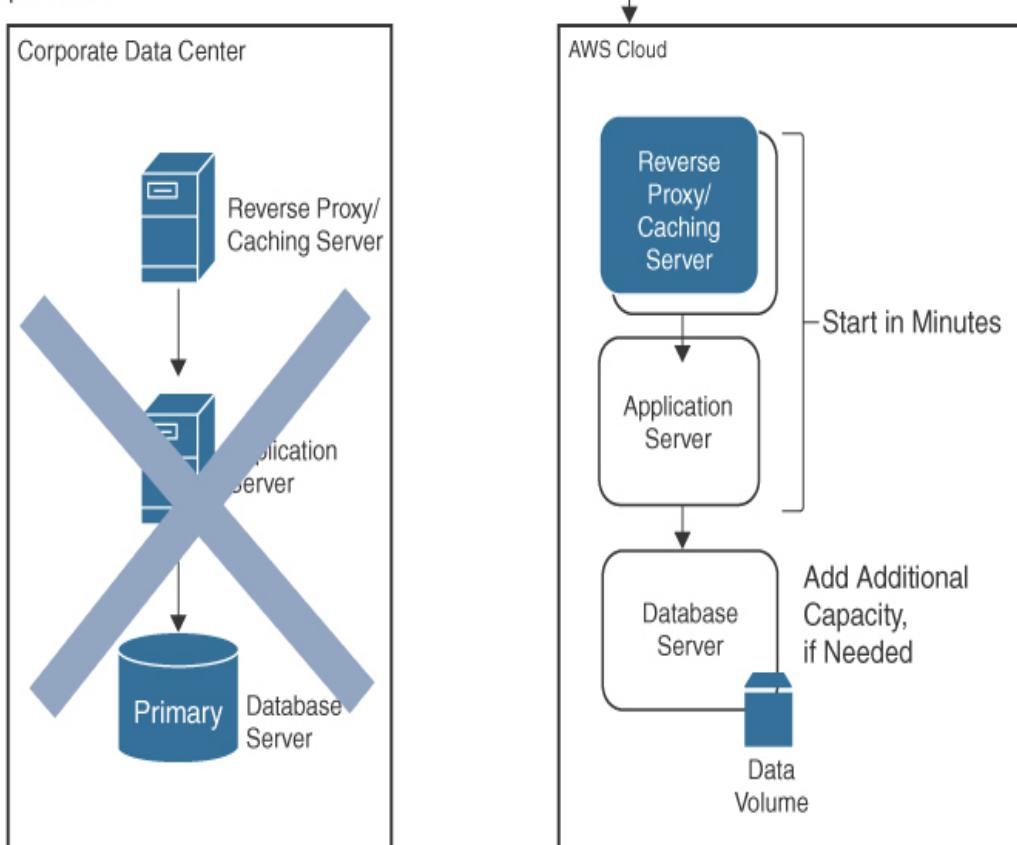


**Figure 7-25** Multi-Region Pilot Light Setup

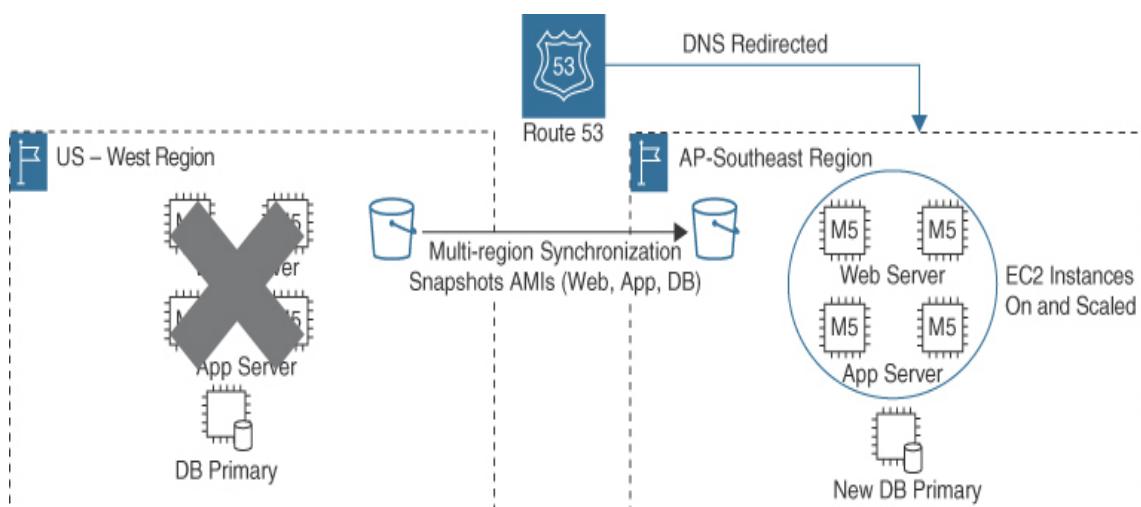
When a disaster occurs, the standby web and workload instances will need to be turned on or deployed using an Amazon Machine Images that were copied to the recovery site. The standby database server at the recovery site will need to be defined as the primary database server, and DNS services will have to redirect traffic to the AWS region disaster recovery site, as shown in the example for the on-premises pilot light response in [Figure 7-26](#), and the multi-region responses shown in [Figure 7-27](#). The RTO to execute a pilot light deployment is certainly faster than the backup and restoration scenario; there is no data loss, but there is also no access to the hosted workload at the recovery site until configuration and startup is complete.

## Pilot Light Recovery

www.example.com



**Figure 7-26** On Premises Pilot Light Response



### **Figure 7-27 Multi-region Pilot Light Response**

Another option for a faster pilot light DR deployment is to automate the configuration using CloudFormation templates, ensuring that DR infrastructure is built and ready to go as fast as possible.

AWS Backup can also assist you in designing a backup and restoration solution that is much more effective than the traditional DR design. AWS Backup can back up Amazon EC2 instances, Amazon EBS volumes, Amazon S3 buckets, RDS deployments, Amazon EFS, Amazon FSx for Windows File Server, Amazon DynamoDB tables, and VMware workloads on premises, on Amazon Outposts, and hosted in VMware Cloud on AWS.

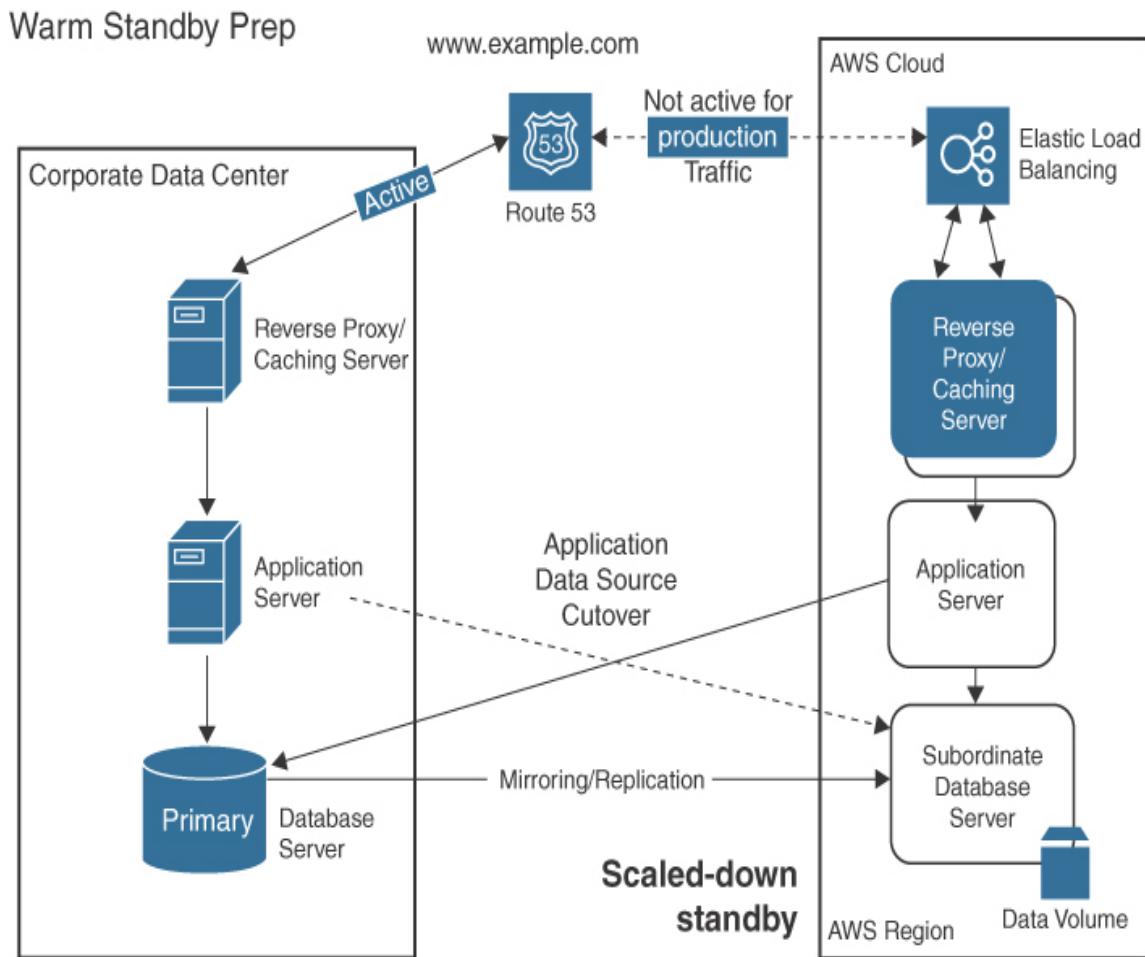
Many third-party backup vendors also have built-in native connectors to directly write to Amazon S3 buckets as the storage target. Backups can be uploaded and written to S3 storage using a public Internet connection or using a private AWS Direct Connect or AWS VPN connection.

### **Warm Standby**

A **warm standby** solution speeds up the recovery time because all the components in the warm standby location are already in

active operation—hence the term *warm*—but at a smaller scale of operation when compared to the primary site.

The secondary system in the standby location is maintained and updated in a similar manner to the system's primary location, but it is not actively in use. In the event of a disaster, the warm standby can be quickly activated and replace the primary, enabling the organization to maintain critical operations and minimize downtime. Warm standby is more comprehensive than a disaster recovery pilot light, because the secondary system location is fully functional and ready for use instead of being a minimal recovery solution that needs to be turned on and scaled. Standby web and workload servers are online and in operation, as shown in the example shown in [Figure 7-28](#). The database servers are also online and functional, and load balancing (ELB) and auto-scaling (EC2 Auto Scaling) have been deployed.

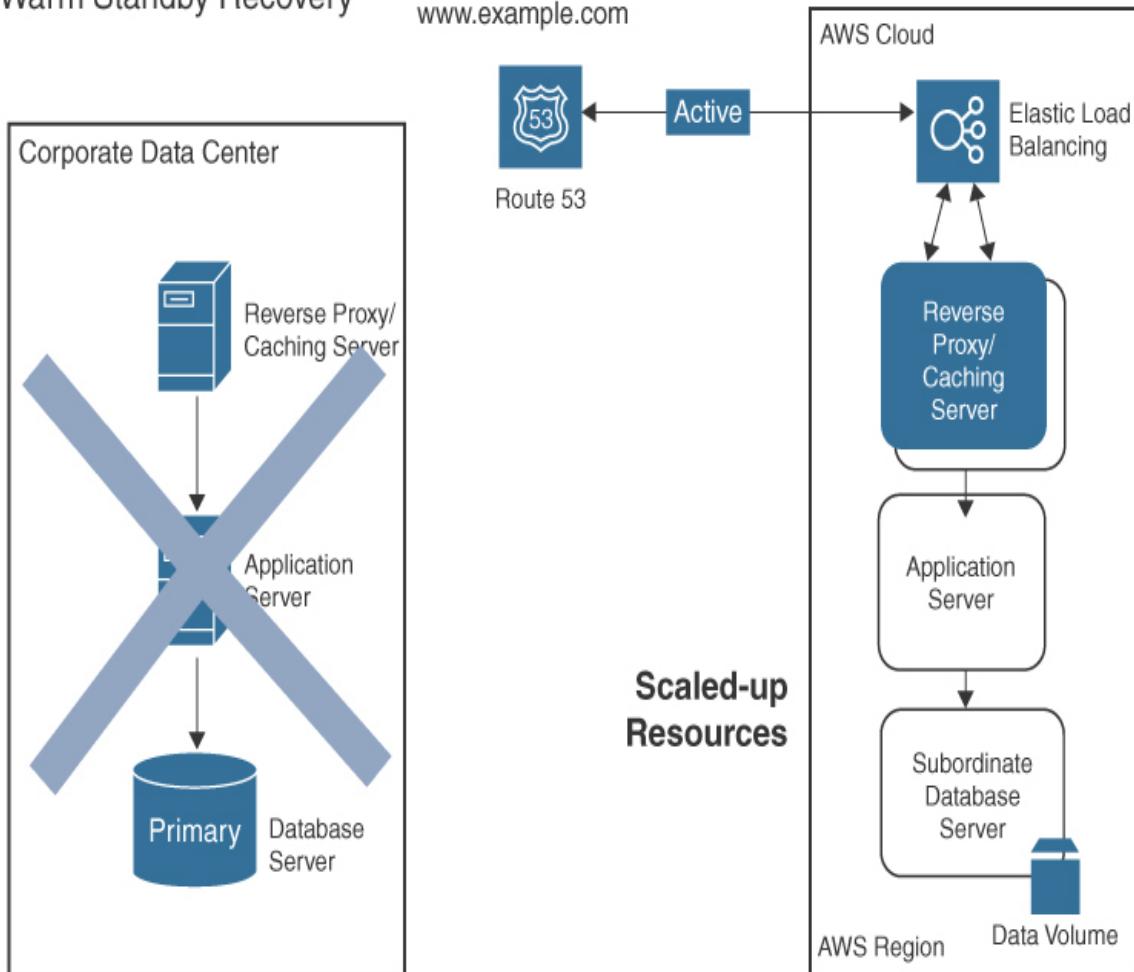


**Figure 7-28** Warm Standby Setup

The key variable, when compared to a pilot light deployment, is that the warm standby workload is active; when a disaster occurs, the recovery site can handle some amount of traffic at once. The additional EC2 instances will need to be scaled, and Route 53 reroutes DNS traffic requests to the standby location, as shown in [Figure 7-29](#). Because warm standby resources were already active, the recovery time is shorter than with a pilot light solution; however, a warm standby solution is more

expensive than a pilot light option as additional standby resources are running 24/7.

### Warm Standby Recovery



**Figure 7-29** Warm Standby Response

---

### Note

AWS Elastic Disaster Recovery can be used by organizations to set up automated disaster

recovery for on-premises physical or virtual server workloads and cloud-hosted applications to AWS.

---

## Multi-Region Scenarios

Workloads designed for minimal downtime with no data loss can be deployed across two or more availability zones, or across multiple AWS regions. Use cases requiring this level of availability include critical business workloads such as finance or banking workloads. Deployment of a workload across multiple AWS regions adds an additional level of redundancy and availability but has additional costs because each region must host a complete copy of the workload. Multi-region scenarios can be active-passive, or active-active with a minimum of two AWS regions hosting both the primary and alternate locations. An active-passive scenario uses one region as a primary site and the other region as a warm standby. In each region there is a minimum of two AZs in each AWS region hosting both the primary and alternate locations.

## Warm Standby with Amazon Aurora

In this example, the workload is deployed to each AWS region using the warm standby design, as shown in [Figure 7-30](#). The passive site is scaled down, but data records are kept consistent.

Data records are asynchronously replicated from the primary Aurora RDS cluster to the read replicas in the passive region with a global database solution providing strongly consistent data stores, keeping the workload data synchronized across multiple AWS regions.

Web and workload servers in both AWS regions are hosted using Workload Load Balancers integrated with EC2 Auto Scaling to automatically scale each workload to changes in workload demand. The ELB service is deployed to all AZs. Using an Auto Scale Group (ASG), the EC2 Auto-Scaling service operates across multiple AZs and mitigates availability zone failures by managing the required compute resources mandated by the ASG to the remaining AZs.

An Amazon Aurora global database has one primary region and can have up to five read-only secondary regions. Cross-region replication latency with Aurora is typically around 1 second. Amazon Aurora allows you to create up to 16 additional read-only database instances in each AWS region. If the Amazon Aurora primary region fails, one of the secondary regions can be promoted to take over the reading and writing responsibilities in less than 1 minute, with an effective RPO of 1 second and an RTO of less than 1 minute.

For a read-local/write-global strategy, an organization could define one region as the primary for all database writes. Data would be replicated for reads to the other AWS region. If the primary database region fails, failover to the secondary site occurs. Both workload tiers are online; when issues occur with one workload tier, traffic gets redirected automatically.

Amazon Route 53 can perform health checks to ensure each regional location remains available, routing requests to the other regional location when a resource becomes unhealthy.

---

#### Note

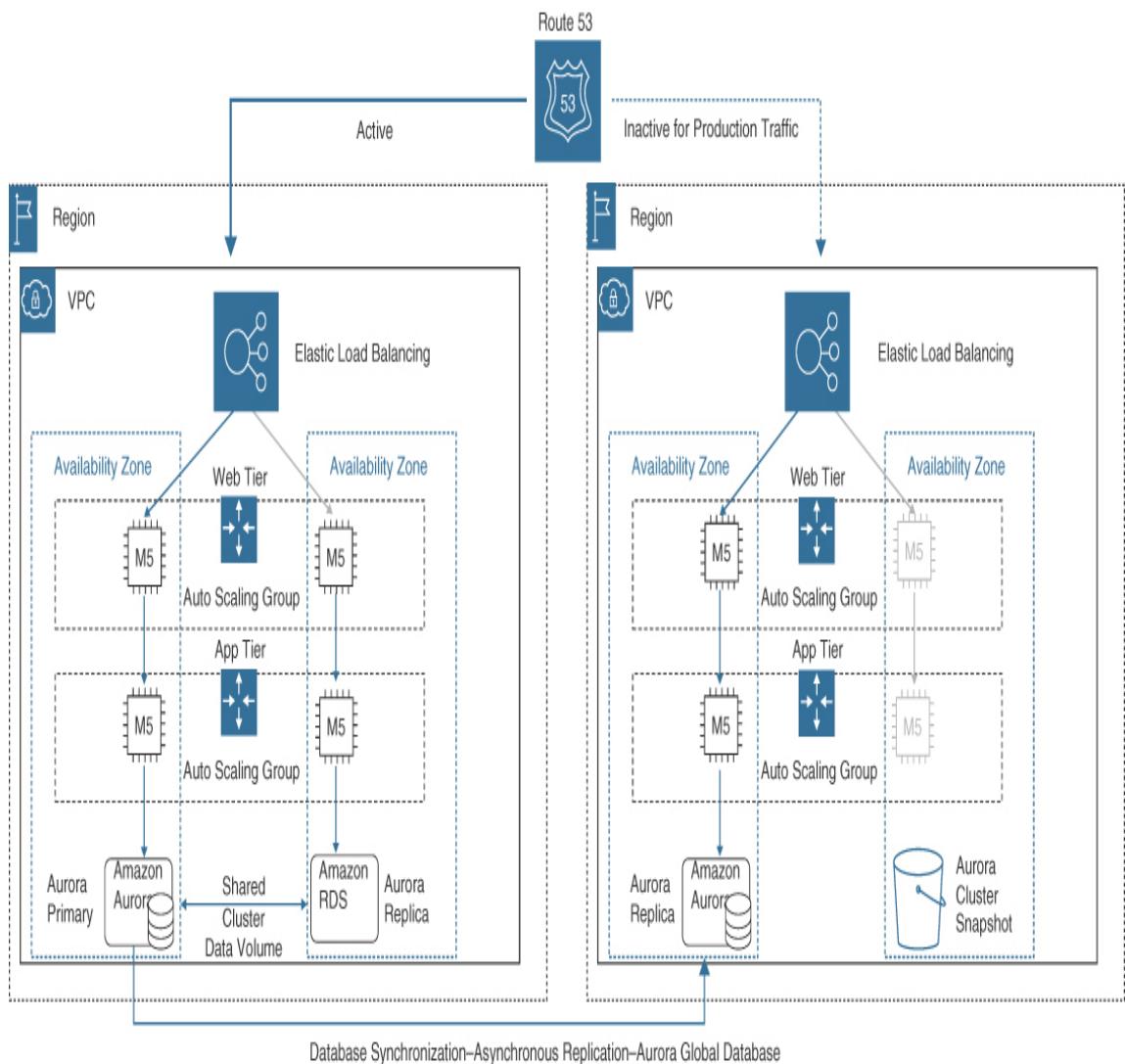
AWS services that operate across multiple regions with continuous asynchronous replication include Amazon S3 Multi-region access points, Amazon RDS read replicas, and Amazon DynamoDB multi-region tables. Up-to-date versions of data are synchronized and available in each of your active regions.

---

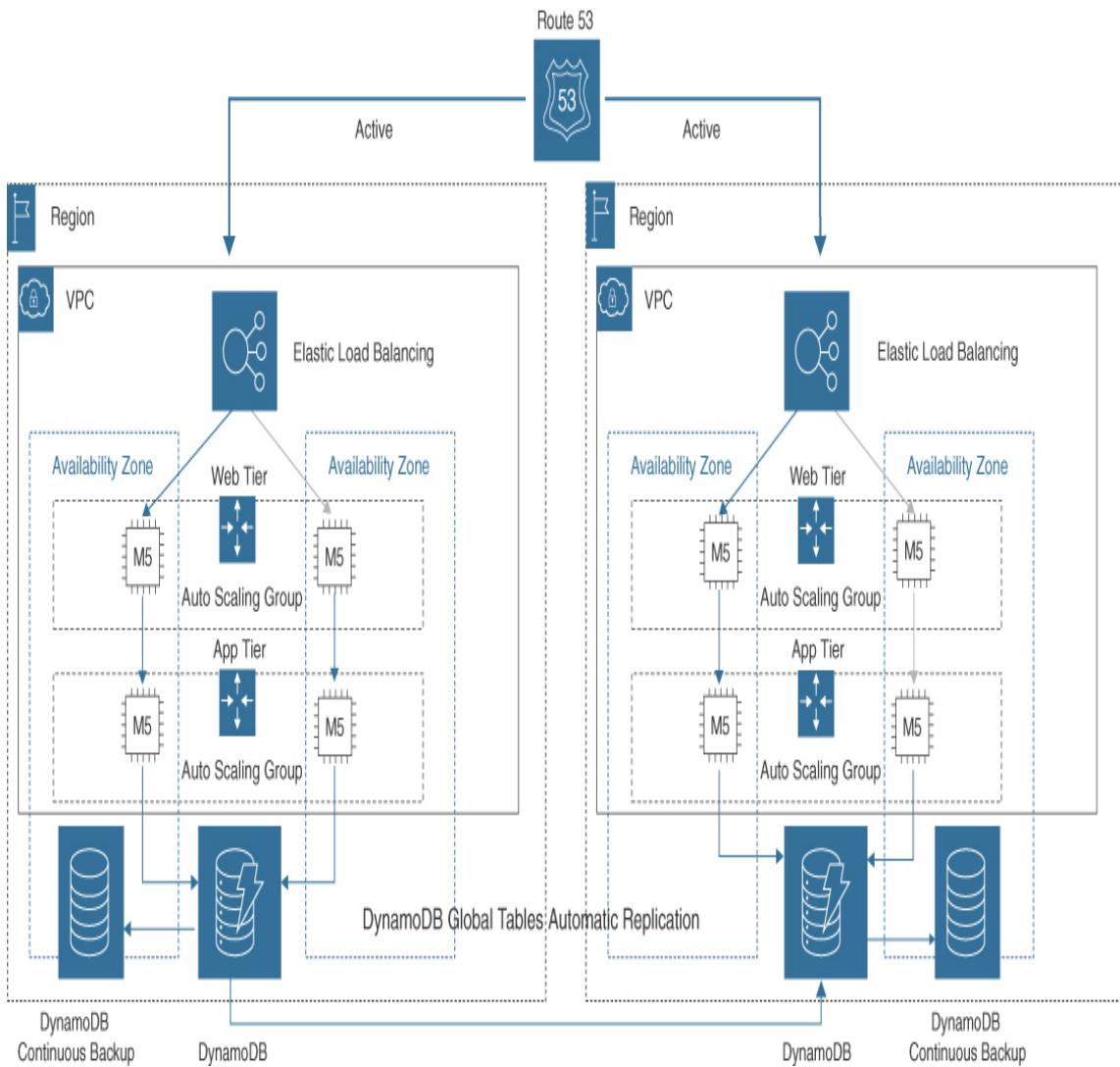
#### **Active-Active**

Active-active failover refers to a scenario in which two or more systems are operating simultaneously and either system can

take over if web, databases, or storage services fail or become unavailable. This type of failover allows for increased availability and reliability of the system because there is always a backup server ready to take over in the event of a failure. Therefore, at AWS with an ***active-active*** design, two or more AWS regions can accept requests. When a failure occurs, requests are rerouted from the region with issues to the other AWS region. [Figure 7-31](#) shows Amazon DynamoDB deployed as a global deployment with a local DynamoDB table deployed in each region and synchronized across the two AWS regions. Each local DynamoDB table has data changes propagated to the local table in the other region within seconds.



**Figure 7-30** Warm Standby with Aurora Global Database



**Figure 7-31** DynamoDB Deployed as a Global Table

---

### Note

Multi-region active-active deployments may have to consider possible data sovereignty requirements; if regions are hosted in the EU, data

storage will be bound by the General Data Protection Regulation (GDPR).

---

## Single and Multi-Region Recovery Cheat Sheet



Availability goals are typically defined as a percentage such as 99% or 99.9%. Recovery solutions could use availability zones within a single AWS region, or use multiple AWS regions, for failover and recovery. [Table 7-9](#) summarizes the relevant tasks for each chosen scenario.

**Table 7-9** Single or Multi-Region Scenario Tasks

| Single AWS Region |              |                |                 |
|-------------------|--------------|----------------|-----------------|
| Task              | 99% Scenario | 99.9% Scenario | 99.99% Scenario |
|                   |              |                |                 |

## Single AWS Region

| Task                     | 99% Scenario              | 99.9% Scenario   | 99.99% Scenario  |
|--------------------------|---------------------------|--|--|
| <b>Monitor resources</b> | Monitor website OK status | CloudWatch metrics and alarms for web and DB tier issues | CloudWatch metrics and alarms for web and DB tier issues |

|  |                                     |   |                                    |
|--|-------------------------------------|---|------------------------------------|
| <b>Adapt to changes in demand with automated responses</b> | Hardware failures, software updates | EC2 Auto-Scaling based on 70% CPU utilization | AWS Lambda, regional read replicas |
|--|-------------------------------------|---|------------------------------------|

## Single AWS Region

| Task                     | 99% Scenario          | 99.9% Scenario        | 99.99% Scenario   |
|--------------------------|-----------------------|-----------------------|---|
| <b>Implement changes</b> | CloudFormation stacks | CloudFormation stacks | CloudFormation stacks or CloudWatch Metrics CloudWatch Metrics CloudWatch Metrics |

|                     |  |  |   |
|---------------------|--|--|---|
| <b>Back up data</b> | Back up to Amazon S3; enable Versioning; WORM policy | Amazon RDS backups (snapshots transaction logs); backup to Amazon S3; enable Versioning; WORM policy | RDS (snapshots transaction logs) back up to Amazon S3; enable Versioning; WORM policy |
|---------------------|--|--|---|

## Single AWS Region

| Task                  | 99% Scenario       | 99.9% Scenario              | 99.99% Scenario   |
|-----------------------|--------------------|-----------------------------|---|
| <b>Architect</b>      | One region, one AZ | Two AZs; ELB; Auto Scaling; | Scenarios: Three AZs; RDS Multi-AZ; Aurora; ELB health checks |
| <b>for resiliency</b> |                    |                             | Average availability: 99.99% (with CDN)                       |
|                       |                    |                             | Cloud services: CloudFront (CDN)                              |

|                        |                                  |                                  |              |
|------------------------|----------------------------------|----------------------------------|--------------|
| <b>Test resiliency</b> | Limited testing of EC2 instances | Limited testing of EC2 instances | Game testing |
|------------------------|----------------------------------|----------------------------------|--------------|

## Single AWS Region

| Task                     | 99% Scenario                     | 99.9% Scenario         | 99.99% Scenario                       |
|--------------------------|----------------------------------|------------------------|---------------------------------------|
| <b>Disaster recovery</b> | Redeploy in same or alternate AZ | Recovery with runbooks | with runbooks using regional backends |

| Availability | Recover from failure in 2 hours | Recover from failure in 1 hour | Recovery time |
|--------------|---------------------------------|--------------------------------|---------------|
|              |                                 |                                | 15 minutes    |

## Disaster Recovery Cheat Sheet

Key Topic

Review the following cheat sheet statements to help solidify your knowledge of disaster recovery before taking the AWS Certified Solutions Architect – Associate (SAA-C03) exam:

- **Single region DR:** Multiple availability zones and Elastic Load Balancing provide high availability for Amazon EC2 instances and Amazon ECS.
- **Single region DR:** Multiple availability zones, Elastic Load Balancing, and EC2 Auto Scaling provide high availability, automatic scaling, and self-healing web and workload tiers.
- **Multi-region DR:** Route 53 health checks monitor the health and performance of web servers and are used to route traffic to healthy resources.
- **Multi-region DR:** Route 53 health checks monitor endpoints, other health checks, and CloudWatch alarms.
- **Multi-region DR:** Route 53 DNS active-active failover configurations are for scenarios where all resources are available. All records have the same record type and routing policy (for example, weighted, geoproximity, or latency policies).
- **Multi-region DR:** Route 53 DNS active-passive failover configurations use Failover as the routing policy. Scenarios are where the primary resource is available, with secondary resources on standby when primary resources become unavailable.

- **Multi-region DR:** Amazon Route 53 Workload Recovery Controller (ARC) is a special health check that determines when workloads and resources are ready for recovery and coordinates the failover process.
- **Single region DR:** Use AWS Backup to back up AWS data resources and hybrid VMware deployments to S3 storage.
- **Single/Multi-region DR:** AWS RDS can be deployed Multi-AZ or in a cluster (primary plus two standby DB replicas).
- **Single region DR:** Amazon Aurora can be deployed as a regional deployment across three AZs per region.
- **Single/Multi-region DR:** Amazon Aurora can also be deployed as a global database across multiple AWS regions.
- **Single/Multi-region DR:** Amazon DynamoDB can be deployed as a regional deployment across three AZs per region or as a global table across multiple AWS regions.
- **Single/Multi-region DR:** AWS CloudFormation can be used to define and deploy infrastructure across AWS Regions. AWS CloudFormation Stack Sets enables you to create and update CloudFormation stacks across multiple accounts and regions.

## AWS Service Quotas



When resources are ordered in each AWS account, the number of resources that can be ordered is controlled by default ***service quotas*** (formerly called *service limits*) that are defined for over 100 AWS cloud services. Quotas control the availability of AWS resources and attempt to prevent customers from provisioning more resources than needed without careful thought.

Customers can request quota increases using the AWS Service Quotas service, shown in [Figure 7-32](#). Customers need to be aware of the assigned default quotas and the increases that are required for the deployment of workload architecture.

**Request quota increase: Table-level write throughput limit**

**Description**  
The maximum number of write throughput allocated for a table or global secondary index. For more information, see <https://docs.aws.amazon.com/amazondynamodb/latest/developerguide/Limits.html#define-limits-throughput-capacity-modes>

**Utilization**  
Not available

**Applied quota value**  
40,000

**AWS default quota value**  
40,000

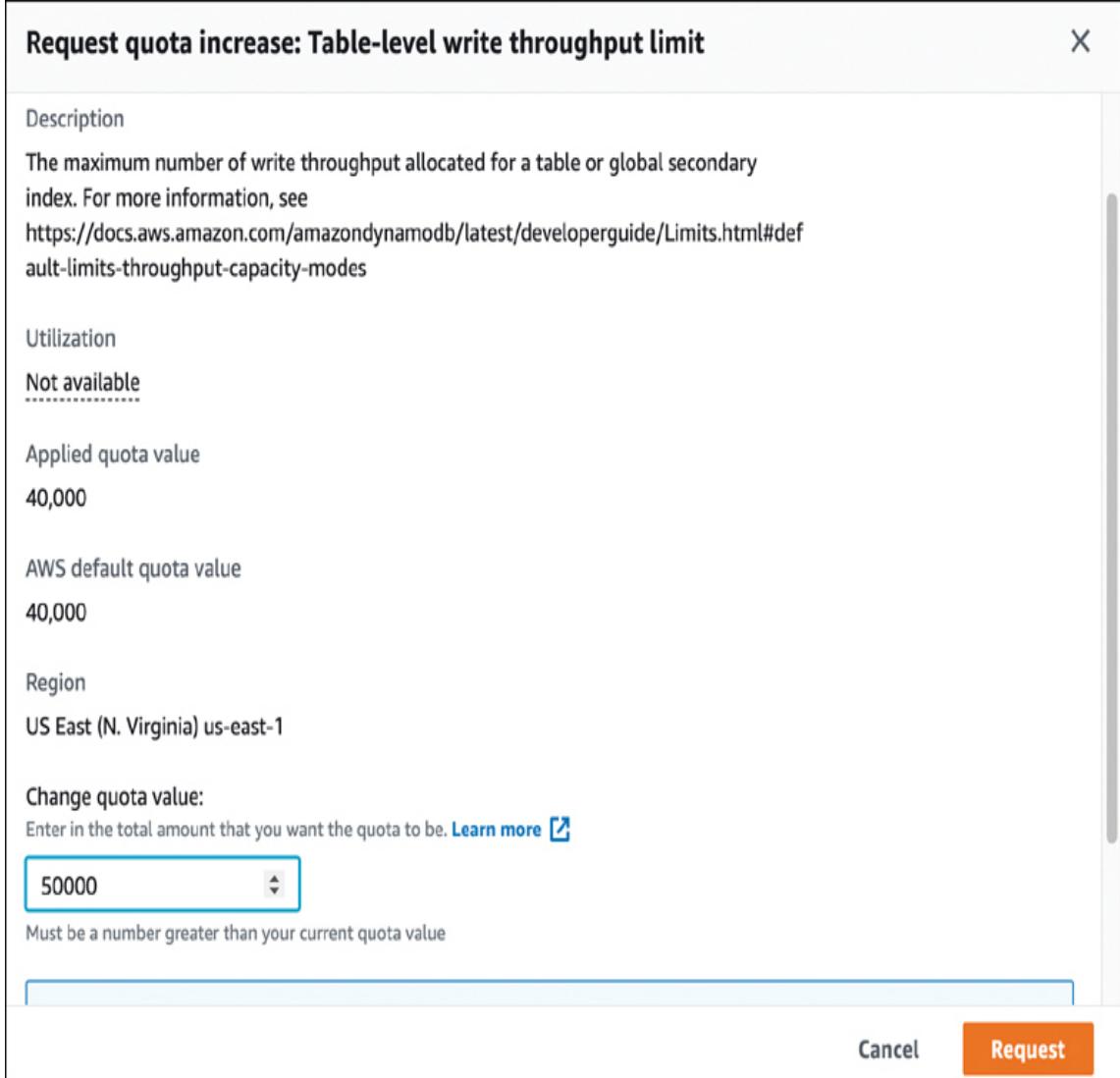
**Region**  
US East (N. Virginia) us-east-1

**Change quota value:**  
Enter in the total amount that you want the quota to be. [Learn more](#)

50000

Must be a number greater than your current quota value

**Cancel** **Request**



**Figure 7-32 Requesting a Quota Increase (Attached)**

Issues that could arise unexpectedly over time could include the number of VPCs that an AWS account can create per region, or the number of EC2 instances that can be launched per availability zone. If you want to scale the number of EC2 instances above your current service quota, for example, there

is a possibility that you might not have the ability to add additional EC2 instances when you need them. The AWS Service Quotas utility can also centrally control quota limits for an AWS Organization, which manages a collection of AWS accounts in a tree-like design.

Organizations can use the AWS Service Quotas console to review and request increases for most AWS quotas. AWS Trusted Advisor can also review your current use of AWS resources and display current quota values for some services, as shown in [Figure 7-33](#).

## Service limits

Choose a check name to see recommendations for services that use more than 80 percent of a service quota. Quota and usage data can take up to 24 hours to reflect any changes.

### Checks

► ✖ **VPC**

Checks for usage that is more than 80% of the VPC Limit.

2 of 16 items have usage that is more than 80% of the service limit.

► ⚠ **VPC Internet Gateways**

Checks for usage that is more than 80% of the VPC Internet Gateways Limit.

1 of 16 items have usage that is more than 80% of the service limit.

**Figure 7-33** Trusted Advisor Displays Current Quota Levels

Several AWS resources are also defined by specific constraints. For example, EBS volumes have maximum disk sizes, and EC2 instances have defined vCPU, RAM, storage, and network bandwidth constraints. Constraints can be changed for EBS volumes using the properties of each EBS volume to change the type, size, and network bandwidth; EC2 instances can be changed by type and family after the EC2 instance has been powered down.

## AWS Service Quotas Cheat Sheet



Review the following cheat sheet statements to help solidify your knowledge before taking the AWS Certified Solutions Architect – Associate (SAA-C03) exam.

- AWS Service Quota manages usage across AWS accounts and regions where workloads are running.
- Service quotas are tracked per AWS account.
- Service quotas are region-specific.

---

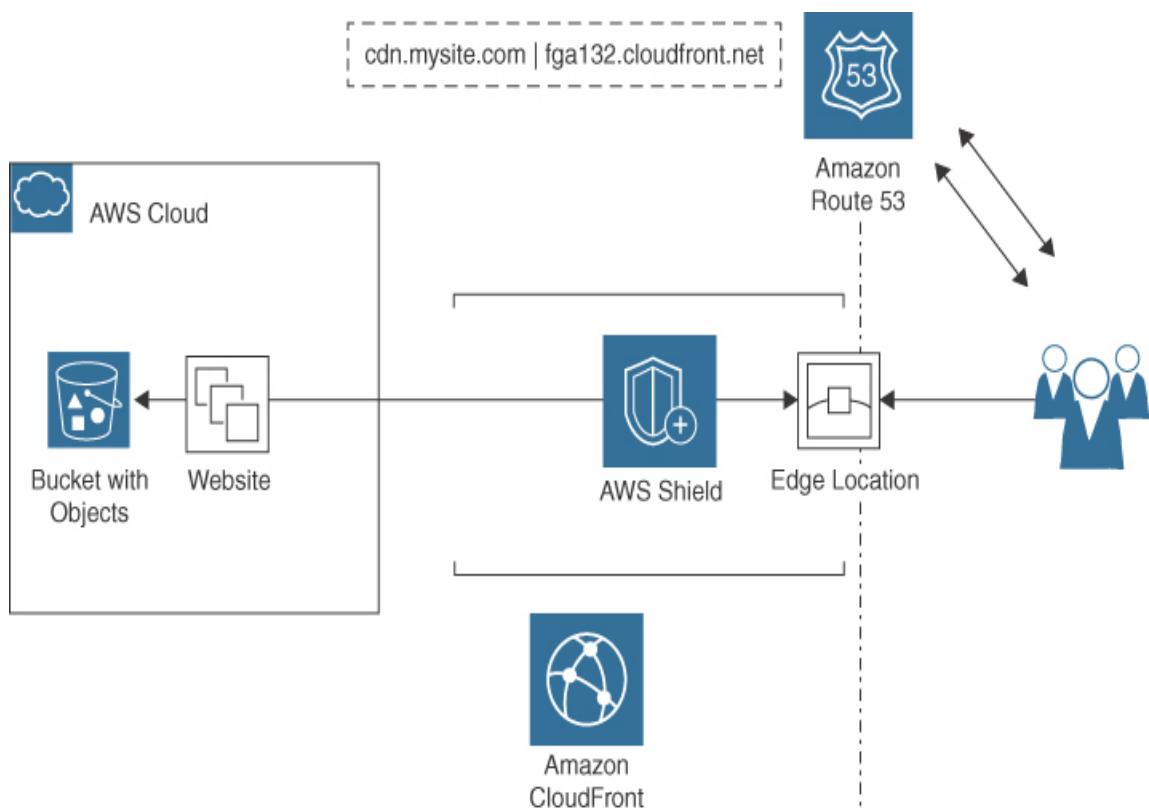
### Note

Most service quotas are specific to the AWS region where the service is located, as most AWS services are region-specific. You can also create CloudWatch alarms for a specific quota and be alerted when you are close to reaching your quota.

---

## Amazon Route 53

Amazon's hosted DNS service, Route 53, is named for the standard DNS port number. Route 53 has a public side tuned to the public edge locations that accepts incoming customer requests and resolves each query to the requested AWS resource located on Route 53's private side, as shown in [Figure 7-34](#). Route 53 knows about every resource created at AWS that is hosted on the private network. It supports both IPv4 and IPv6 address records, including the common A, AAAA, CNAME, MX, NS, PTR, SOA, SRV, and TXT records.



**Figure 7-34** Route 53 Operations at the Edge

If a customer accesses an application hosted at AWS from a device, a DNS query is carried out to find the application's location and connect. You need an Internet connection or a private network connection to AWS for a device to begin communicating with the AWS cloud. HTTP and HTTPS endpoints are available to connect to an AWS service. Endpoints are regional and, in some cases, global (for global services such as Route 53).

DNS services, namely Amazon Route 53, need to be operational for the initial query to be successful. Operating in the AWS public cloud, the scope of DNS services has changed from local DNS services per corporation to worldwide redundant global DNS services that are linked together with full knowledge of where the Amazon regions and availability zones are located and where each requested AWS service resides.

Amazon Route 53 is designed and operates using anycast DNS routing algorithms. Each destination AWS service location is known by the anycast DNS servers that are hosted across all the AWS regions and edge locations. The network location determines the edge location where Route 53 directs requests. When an application request reaches the selected edge location, the application request continues across Amazon's private network to the preferred AWS service location.

Route 53 offers the following services:

- **Domain name registration:** You use Route 53 to register the domain name for your website or application. After you choose a domain name and it is verified, the domain name is registered with Route 53. Route 53 DNS records can also be linked to an existing domain. When a domain has been registered with Route 53, it becomes the DNS service for the domain and creates a hosted zone for the domain. For redundancy, Route 53 assigns four name servers to the hosted zone; these name servers provide details about the location of your resources.
- **DNS resolution:** Route 53 routes Internet traffic to the resources for your domain both for public-facing resources and services and for AWS resources and services hosted at AWS. A public hosted zone is required for public-facing resources such as a company website. Private hosted zones are created to route traffic within a VPC. Each hosted zone has two records: a name server (NS) record and the start of authority (SOA) record. The name server record identifies the four name servers available to answer queries. The start of authority record indicates the authoritative location for answering queries for your domain.
- **Health checks:** Automated health checks can be delivered on a persistent schedule to cloud services and resources to

confirm they are still available and functioning. When resources become unavailable, Route 53 routes requests from unhealthy resources to healthy resources.

## Route 53 Health Checks

Health checks can monitor both the health and the performance of regional services including load balancers, web applications, and other external resources registered with Route 53. Each health check can monitor the following:

- **The health of resources:** Health checks can monitor selected endpoints based on their DNS name, IP address, or domain name to verify whether resources are reachable and functioning.
- **The status of other health checks:** Health checks can monitor multiple regional workloads to ensure that healthy resources are available across AWS regions or resources located on other public clouds or on premises.
- **The status of CloudWatch alarms:** Route 53 can monitor the same metrics being monitored by CloudWatch and make decisions without needing to communicate directly with the CloudWatch service.

## Route 53 Routing Policies

When Route 53 receives a query, a routing policy determines the response. [Table 7-10](#) lists routing policies supported by Route 53.

**Table 7-10** Route 53 Routing Policies

| Routing Policy | Function   | Use Case   |
|----------------|--|--|
| Simple         | A simple DNS response matching the requested resource with the current IP address. | A single resource, such as a web server, that is serving content for a single domain |

| Routing Policy | Function   | Use Case   |
|----------------|--|--|
| Failover       | If the primary resource is not responding, the request is sent to the secondary location.  | For active/passive failover design with multiple AWS regions       |
| Geolocation    | Choose resources based on the geographic locations of your users.<br>Content can be localized based on the language of your users or to restrict where content can be distributed. | For routing traffic based on the physical location of the end user |

| Routing Policy | Function  | Use Case  |
|----------------|---|---|
| Geoproximity   | <p>Route traffic to your resource based on the geographic locations of your users and requested resources. This routing requires you to create a defined Route 53 traffic policy.</p> | For routing traffic based on the location of your AWS resources |
| Latency        | <p>Use for applications hosted in multiple AWS regions.</p> <p>Requests are sent to the resources at the closest location with the lowest-latency route.</p>                          |   |

---

Latency

Use for applications hosted in multiple AWS regions.

Requests are sent to the resources at the closest location with the lowest-latency route.

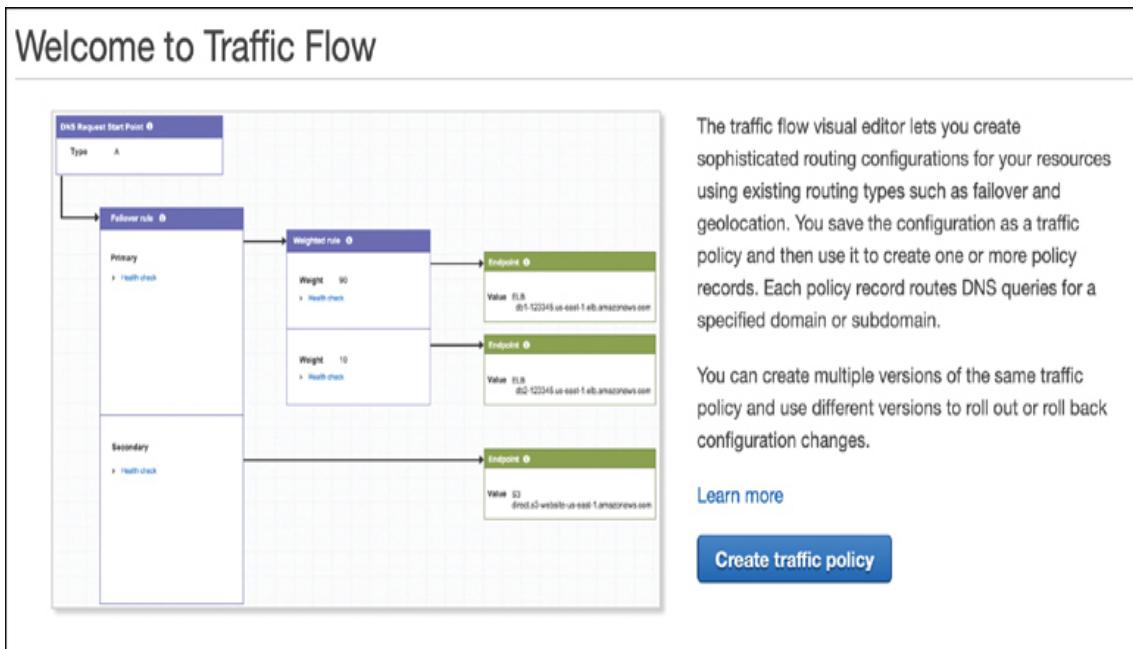
| Routing Policy    | Function   | Use Case  |
|-------------------|--|---|
| Multivalue answer | Using multiple IP addresses with health checks, Route 53 will return records for healthy resources that respond successfully to health checks. | For responding with up to eight healthy address records |

| Routing Policy | Function  | Use Case  |
|----------------|---|---|
| Weighted       | Associate multiple resources with a single domain name and choose how much traffic is routed to each resource stack. If a weight of 64 is assigned to one stack and a weight of 255 to the other, a weight of 64 would get 25% of the traffic requests. | For routing traffic to multiple resources at defined traffic levels |

## Route 53 Traffic Flow Policies

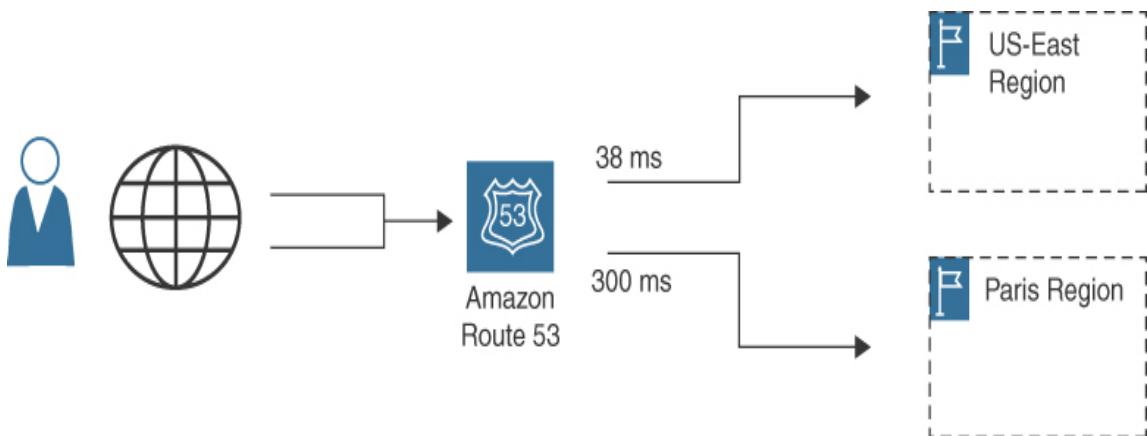
Route 53 traffic flow provides global traffic management services (GTMS) using traffic flow policies created using the

traffic flow visual editor (see [Figure 7-35](#)). This editor enables you to create specific routing configurations for your resources.



**Figure 7-35** Route 53 Traffic Flow Policies

You can create traffic flow policies for situations involving geoproximity, endpoint health, overall application load, and latency (see [Figure 7-36](#)). For example, you might want to route traffic based on the locations or the languages of your end users. In addition, you might want to control the amount of traffic sent to resources by using a *bias*, which enables you to change the size of the geographic region from which traffic is routed to a resource.



**Figure 7-36** Route 53 Latency Records Defining the Fastest Paths to Access Resources

### Note

With traffic policies, you can create a tree of alias records and routing policies.

### Alias Records

Alias records provide additional workload redundancy as multiple records can be provided for requests for AWS resources. When Route 53 receives a query for an AWS resource using an alias record, one or more IP addresses can return pointing to redundant AWS service locations. Route 53 responds with multiple IP addresses; if one IP address doesn't work due to maintenance or failures, one of the other IP addresses will resolve the request. The following AWS services support alias records:

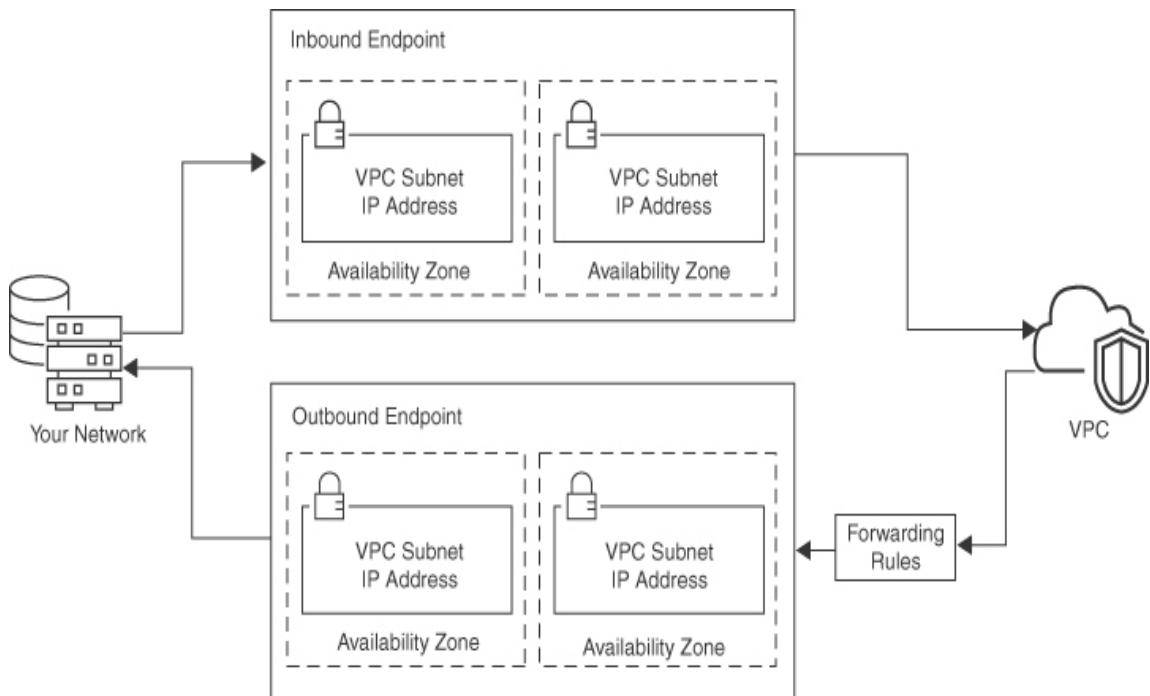
- **API gateway API request:** Route 53 provides one or more IP addresses for the requested API.
- **VPC interface endpoint:** Route 53 provides one or more IP addresses for the endpoint.
- **CloudFront distribution:** Route 53 provides one or more IP addresses for edge servers for servicing the request.
- **Elastic Beanstalk environment:** Route 53 provides one or more IP addresses for the environment.
- **ELB load balancer:** Route 53 provides one or more IP addresses for the load balancer.
- **Global accelerator:** Route 53 provides one or more IP addresses for each accelerator.
- **S3 bucket configured as a static website:** Route 53 provides one or more IP addresses for the S3 bucket.

Alias records also enable you to route traffic at the top node of a DNS namespace, known as the *zone apex*. For example, if the domain [mydomain.com](#) is registered with Route 53, the zone apex is [mydomain.com](#). At AWS, a CNAME record cannot be created for [mydomain.com](#); however, an alias record can be created to route traffic to [mydomain.com](#).

## Route 53 Resolver

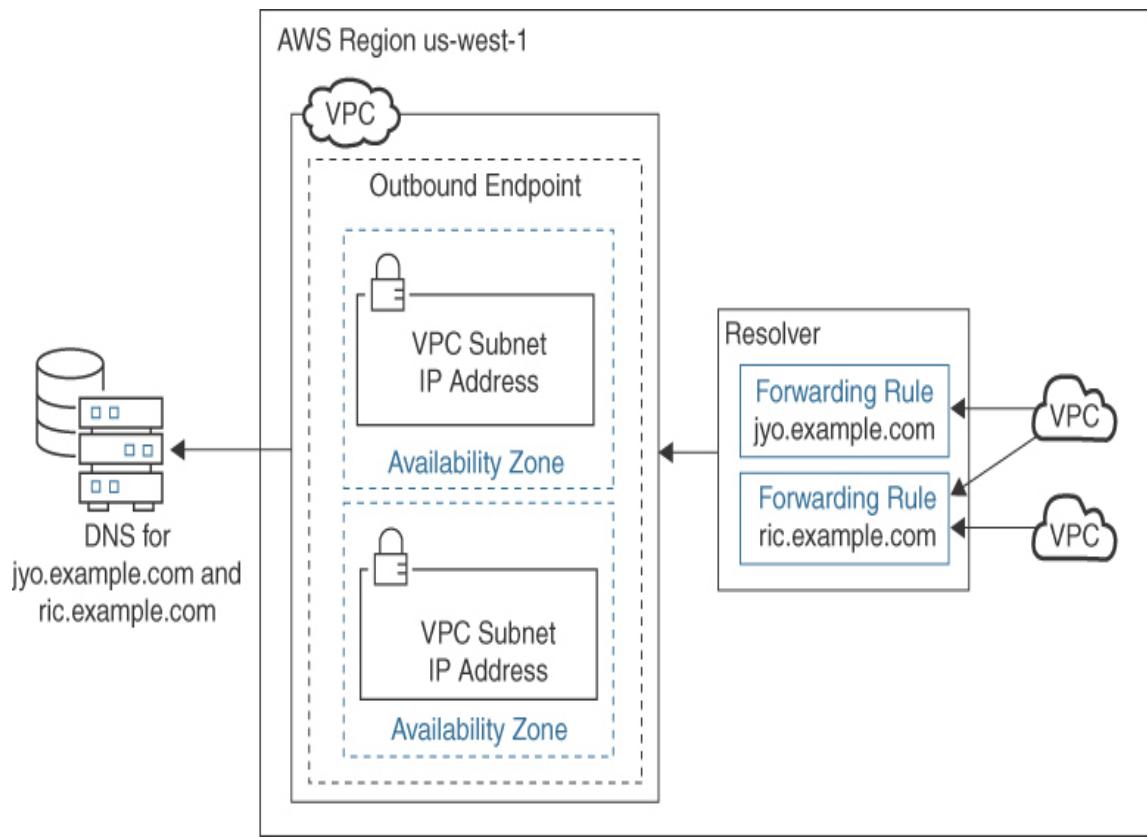
When an organization creates a VPC to host application servers running on EC2 instances, the Route 53 resolver automatically answers DNS queries from the EC2 instance's VPC-specific DNS names and Route 53 private hosted zones.

If you have a hybrid need for communication between on-premises resources and your VPC resources at AWS or between other peered VPCs, communication can be initiated with forwarding rules, as shown in [Figure 7-37](#). You first need to create resolver inbound and/or outbound endpoints. Inbound queries from the on-premises resource is provided by defining a Route 53 resolver endpoint that resolves to an AWS-hosted domain and the IP addresses of the DNS resolvers to which you want to forward the queries.



**Figure 7-37** Forwarding Rules

You enable outbound DNS queries by using conditional forwarding rules, as shown in [Figure 7-38](#).



**Figure 7-38** Conditional Forwarding Rules

## Exam Preparation Tasks

As mentioned in the section “[How to Use This Book](#)” in the Introduction, you have a couple of choices for exam preparation: the exercises here, [Chapter 16](#), “[Final Preparation](#),” and the exam simulation questions in the Pearson Test Prep Software Online.

## Review All Key Topics

Review the most important topics in the chapter, noted with the Key Topic icon in the outer margin of the page. [Table 7-11](#) lists these key topics and the page number on which each is found.

**Table 7-11** [Chapter 7](#) Key Topics

| Key Topic Element          | Description  | Page Number |
|----------------------------|--|-------------|
| Section                    | High Availability in the Cloud                       | 294         |
| Section                    | Reliability  | 295         |
| <a href="#">Figure 7-3</a> | Visualizing Availability Zones and Regional Services | 299         |
| Section                    | Availability Zones                                   | 300         |
| <a href="#">Figure 7-6</a> | AZ Balancing and Distribution of Resources           | 302         |
| Section                    | Planning Network Topology                            | 303         |

| Key Topic Element            | Description  | Page Number |
|------------------------------|--|-------------|
| Section                      | Local Zones  | 306         |
| Section                      | AWS Services Use Case  | 308         |
| <u>Figure</u><br><u>7-12</u> | Artifact Utility   | 311         |
| Section                      | Latency Concerns   | 319         |
| Section                      | Designing for High Availability and Fault Tolerance                      | 322         |
| <u>Figure</u><br><u>7-16</u> | Hosting Across Multiple AZs to Increase Availability and Fault Tolerance | 323         |
| <u>Figure</u><br><u>7-18</u> | Increasing Workload Availability by Adding Compute Workload              | 324         |

| Key Topic Element                  | Description   | Page Number |
|------------------------------------|---|-------------|
| <u><a href="#">Table 7-6</a></u>   | Avoiding Single Points of Failure                               | 325         |
| <u><a href="#">Table 7-7</a></u>   | Planning for High Availability, Fault Tolerance, and Redundancy | 326         |
| <u><a href="#">Figure 7-20</a></u> | Deploying Immutable Servers                                     | 328         |
| <u><a href="#">Table 7-8</a></u>   | Storage Service Characteristics                                 | 330         |
| Section                            | Failover Strategies   | 330         |
| Section                            | Single and Multi-Region Recovery Cheat Sheet                    | 343         |
| Section                            | Disaster Recovery Cheat Sheet                                   | 344         |

| Key Topic Element | Description                    | Page Number |
|-------------------|--------------------------------|-------------|
| Section           | AWS Service Quotas             | 345         |
| Section           | AWS Service Quotas Cheat Sheet | 347         |

## Define Key Terms

Define the following key terms from this chapter and check your answers in the glossary:

availability

reliability

region

availability zone (AZ)

Local Zones

AWS Artifact

FedRAMP

uptime

SLA

immutable

RTO

RPO

pilot light

warm standby

active-active

service quota

## **Q&A**

The answers to these questions appear in Appendix A. For more practice with exam format questions, use the Pearson Test Prep Software Online.

- 1.** First, you design with security, and then you make your workload as \_\_\_\_\_ and \_\_\_\_\_ as possible.

- 2.** Problems with one AWS region remain \_\_\_\_\_ to that region and should not affect other regions.
- 3.** Having multiple availability zones provides \_\_\_\_\_, \_\_\_\_\_, and \_\_\_\_\_.
- 4.** Costs are different for each \_\_\_\_\_ where an organization hosts workloads and cloud services.
- 5.** Artifact allows you to review \_\_\_\_\_ reports.
- 6.** Amazon Aurora global database deployments are deployed across multiple AWS \_\_\_\_\_.
- 7.** Amazon Route 53 routing provides global traffic management services using \_\_\_\_\_.
- 8.** The AWS resources that can be ordered are controlled by a default \_\_\_\_\_.

# Chapter 8

## High-Performing and Scalable Storage Solutions

This chapter covers the following topics:

- [AWS Storage Options](#)
- [Amazon Elastic Block Store](#)
- [Amazon Elastic File System](#)
- [Amazon FSx for Windows File Server](#)
- [Amazon Simple Storage Service](#)
- [Amazon S3 Glacier](#)
- [AWS Data Lake](#)

This chapter covers content that's important to the following exam domain and task statements:

### Domain 3: Design High-Performance Architectures

**Task Statement 1:** Determine high-performing and/or scalable storage solutions

**Task Statement 5:** Determine high-performing data ingestion and transformation solutions

The amount of public cloud storage utilized by public and private data records continues to grow, with no sign of stopping anytime soon. Workloads hosted at AWS consume a lot of data storage, and these out-of-sight data records become expensive to store. It's important to understand how each AWS storage service works and how cost savings and performance requirements can be achieved by picking the right storage solution.

According to <https://raconteur.net>, over 4 petabytes (PB) of data are created daily by Facebook users, 4 TIB of data are produced each day by connected cars, 65 billion messages are sent via WhatsApp daily, and wearable devices generate 28 PB of data. In addition, corporations and software as a service (SaaS) applications are using the cloud to store vast quantities of data, from Word and Excel files to machine learning, online gaming results, and, of course, cat pictures.

By 2025, over 463 exabytes (EB) of data will be created in the public cloud daily. That's 1,000,000,000,000,000,000 bytes! Any low advertised storage price becomes quite expensive over the long term as companies store data records for long periods of time.

---

Note

For additional details on these figures on data storage, see

<https://www.raconteur.net/infographics/a-day-in-data/>.

---

AWS has made great strides in providing “cloud” solutions for companies whose data records are still required to be stored on premises because of security concerns, rules, and regulations. (For example, some financial organizations are not allowed to move specific data resources to the cloud due to governance and security concerns.)

To combat compliance concerns AWS has launched AWS Outposts, which is designed to let customers run AWS services on premises, in their own data centers. AWS Outposts can be used for a variety of workloads, including data processing, machine learning, and low-latency applications, including compute and storage services (Amazon EC2, Amazon ECS/Amazon Kubernetes) and storage (Elastic Block Store [EBS] and Amazon Simple Storage Service [S3]). Organizations can now choose whether to operate in AWS public cloud regions or operate from an on-premises AWS Outposts deployment.

By the end of this chapter, you will understand the choices for storing your workload data using AWS managed storage

services. The AWS Certified Solutions Architect – Associate (SAA-C03) exam will expect you to be able to select the appropriate storage solution based on the use case presented in relevant exam questions. AWS storage services store many different types of data using the following storage services:

- Amazon EBS stores files in persistent scalable block storage volumes that are attached to EC2 instances. Boot and data volumes are supported.
- Amazon S3 offers unlimited object cloud storage in buckets for any data type, including logging services, CloudWatch logs, CloudTrail trails, Amazon Machine Images (AMIs), EBS snapshots, AWS managed services data, database backups, and more.
- Amazon S3 Glacier Flexible Retrieval provides archive and vault storage for records that need to be archived for long periods of time and retrieved at no cost.
- Amazon S3 Glacier Instant Retrieval provides archive and vault storage at a very low cost, with millisecond retrieval speeds when retrieval is necessary.
- Amazon S3 Glacier Deep Archive is offline archival and vault storage with data retrieval speeds within 12 hours.
- Amazon EFS or FSx storage is available for both AWS-hosted and on-premises Linux and Windows workloads that require access to a shared file system.

## “Do I Know This Already?”

The “Do I Know This Already?” quiz allows you to assess whether you should read this entire chapter thoroughly or jump to the “Exam Preparation Tasks” section. If you are in doubt about your answers to these questions or your own assessment of your knowledge of the topics, read the entire chapter. [Table 8-1](#) lists the major headings in this chapter and their corresponding “Do I Know This Already?” quiz questions. You can find the answers in [Appendix A](#), “[Answers to the ‘Do I Know This Already?’ Quizzes and Q&A Sections.](#)”

**Table 8-1** “Do I Know This Already?” Section-to-Question Mapping

| Foundation Topics Section          | Questions |
|------------------------------------|-----------|
| AWS Storage Options                | 1, 2      |
| Amazon Elastic Block Store         | 3, 4      |
| Amazon Elastic File System         | 5, 6      |
| Amazon FSx for Windows File Server | 7, 8      |

## Foundation Topics Section

## Questions

Amazon Simple Storage Service

9, 10

Amazon S3 Glacier

11, 12

AWS Data Lake

13, 14

---

### Caution

The goal of self-assessment is to gauge your mastery of the topics in this chapter. If you do not know the answer to a question or are only partially sure of the answer, you should mark that question as wrong for purposes of the self-assessment. Giving yourself credit for an answer you correctly guess skews your self-assessment results and might provide you with a false sense of security.

---

**1. What is the best AWS option for storing archival records?**

1. Amazon S3
2. Amazon S3 Deep Archive

3. Ephemeral storage

4. Snapshots

**2.** What type of AWS storage allows storage of an unlimited number of objects?

1. Amazon EFS

2. Amazon S3

3. Amazon FSx

4. Amazon EBS

**3.** How can Amazon EBS storage volumes be publicly accessed from the Internet?

1. Attach an Internet gateway.

2. EBS volumes are private.

3. Deploy Amazon Storage Gateway.

4. Attach a virtual private gateway.

**4.** What does the term *elastic* signify with Elastic Block Store volumes?

1. EBS volumes scale on demand.

2. EBS volume size and type can be changed after creation.

3. EBS volumes are replicated to multiple locations.

4. EBS volumes can be shared across availability zones.

**5.** How can Amazon EFS be accessed by on-premises systems and applications?

1. Internet Gateway connection
2. NAT Gateway service connection
3. Direct Connect connection
4. Internet connection

**6.** What is the recommended performance mode of operation to select when starting with Amazon EFS?

1. Max I/O
2. General Purpose
3. Bursting Throughput
4. Provisioned Throughput

**7.** What Microsoft Windows Server file system feature does FSx support?

1. NTFS
2. Distributed File System (DFS)
3. Services for NFS
4. Active Directory Federated Service

**8.** What Windows service does AWS FSx for Windows File Server use to replicate FSx data?

1. SMB 2.0
2. Volume Shadow Copy Service
3. SMB 3.0
4. NTFS

**9.** How can you protect Amazon S3 objects from being deleted?

1. Enable encryption
2. Enable versioning
3. Turn off public access
4. Create a lifecycle rule

**10.** What storage class provides automatic movement of objects?

1. Amazon S3 Glacier
2. Intelligent-Tiering
3. Reduced redundancy storage
4. Standard IA

**11.** Where are files uploaded into Amazon S3 Glacier?

1. Amazon S3 archives
2. Amazon S3 Glacier archives
3. Vaults
4. Buckets

**12.** How can objects be moved automatically from Amazon S3 into Amazon Glacier?

1. Lifecycle rule
2. S3 transfer acceleration
3. Versioning
4. AWS Snow Family device

**13.** Where is data stored when deploying AWS Lake Formation?

1. Amazon EMR
2. Amazon S3
3. Amazon S3 Glacier
4. Amazon EFS

**14.** What AWS service creates metadata from existing source data records when creating a data lake?

1. AWS Glue
2. AWS Athena
3. AWS Quicksight
4. Amazon Kinesis Data Firehose

## Foundation Topics

### AWS Storage Options

AWS offers a wide breadth of storage options—for block, object, and file storage—that are similar to but not the same as your on-premises storage options. [Figure 8-1](#) shows the available options, and the list that follows briefly describes the available options:

**Key Topic**



**Figure 8-1** AWS Storage Options

- **Sharable file system for Linux environment using Amazon EFS:** EFS scales on-demand from gigabytes to petabytes accessed through using NFSv4 mount points from selected VPC/subnets by Amazon Elastic Compute Cloud (EC2) instances, Amazon Elastic Container Service (ECS), Amazon Elastic Kubernetes Services (EKS), or from on-premises servers.
- **Object:** [\*\*Object storage\*\*](#) at AWS is the Amazon Simple Storage Service (S3). Amazon S3 storage is accessed and managed through simple API (application programming interface) calls, such as **GET** and **PUT**. Each file stored in an S3 bucket is

called an *object*, and each S3 object is identified and accessed by a unique key. The original design for S3 was for accessing content across the Internet. S3 is scalable, durable, and highly cost-effective object storage.

When an object is stored in S3, the entire object is uploaded. Changes to existing objects result in updating the entire object. S3 buckets can host a website's static content and media files and static websites that do not require server-side processing. S3 provides storage for big data analytics and machine learning and EBS **snapshots**, Amazon CloudWatch logs, and AWS Backup backups.

- **Block storage:** Block storage volumes are provided by Amazon EBS for Windows, Linux, or macOS instances. Formatted block storage is accessed at the block level by the operating system and applications. AWS **block storage** describes Amazon EBS volumes, and **ephemeral storage** SSD storage volumes that are attached directly to the bare-metal servers that host EC2 instances. Amazon EBS arrays use solid-state drives (SSDs) or hard disk drives (HDDs), providing persistent block storage only accessible across the private AWS network. Enabling direct public access to EBS storage volumes is not possible.

---

#### Note

Many EC2 instance families also include direct-attached storage as local storage volumes called ephemeral storage (or instance storage).

Ephemeral storage is located on the bare-metal server that hosts the EC2 instance; therefore, it's incredibly fast. However, ephemeral storage has no long-term durability; it is temporary storage. It can survive an EC2 instance reboot but is deleted when the EC2 instance is powered off or fails.

---

- **Sharable file system for Windows File Server:** Amazon FSx for Windows File Server provides fully managed native Microsoft Windows SSD or HDD storage accessible using Windows file shares. Amazon FSx for Windows File Server supports the ***Server Message Block (SMB)*** protocol.

---

#### Note

[Chapter 10, “Determining High-Performing Database Solutions,”](#) covers database storage offerings Amazon RDS, Amazon Aurora, Amazon DynamoDB, and Amazon ElastiCache.

---

## Workload Storage Requirements

Before choosing an AWS storage solution, organizations need to review their current storage requirements carefully. Each workload dictates what storage solution or solutions are required. Consult [Table 8-2](#) for reviewing storage requirements and the available storage solutions.

**Key Topic**

**Table 8-2** Storage Requirements

| Workload Requirements    | Storage Solution   |
|--------------------------|--|
| Operating system support | AWS supports the Linux, Windows, and macOS operating systems. Linux EC2, Windows, and macOS instances support EBS volumes and Amazon EFS shared storage and also can attach to FSx for Windows File Server shared storage. |

| Workload Requirements    | Storage Solution  |
|--------------------------|---|
| File-sharing protocols   | Amazon EFS supports the Network File System (NFSv4) versions 4.0 and 4.1, and Amazon FSx for Windows File Server supports the service message protocol SMB.   |
| Performance requirements | Shared storage solutions such as EFS and FSx provide baseline performance that can burst or scale up dramatically or be mapped to a defined throughput capacity. EBS General Purpose SSD 2 (gp2) volumes provide 3,000 IOPS and 125 MIB/s of consistent performance. EBS Io2 Block Express volumes offer sub-millisecond latency up to 256,000 IOPS and 4,000 MIB/s throughputs and up to 64 TiB in size for a single volume. |

| Workload Requirements   | Storage Solution  |
|-------------------------|---|
| Compliance requirements | <p>Cloud storage solutions at AWS support various levels of compliance. As discussed in <a href="#">Chapter 7</a>, you might need to follow your organization's compliance requirements, such as PCI DSS, FedRAMP requirements, or internal rules and regulations. Use AWS Artifact in the AWS Management Console to review the currently supported compliance standards.</p> |
| Capacity requirements   | <p>What are your daily, monthly, and yearly storage requirements? EBS volumes have capacity size limits ranging from 1 TiB up to 64 TiB.</p>  |

| Workload Requirements | Storage Solution  |
|-----------------------|---|
| Data encryption       | What needs to be encrypted for compliance—data records at rest or data in transit across the network? All data storage options can be encrypted using AES 256-bit encryption; Amazon S3 and S3 Glacier storage is encrypted by default. |
| Concurrent access     | Do virtual servers need to access shared storage? Amazon EFS and FSx for Windows File Server provide shared storage services.   |

| Workload Requirements           | Storage Solution   |
|---------------------------------|--|
| Input/output (I/O) requirements | What is a workload's percentage of reading and writing files? Is it balanced or unbalanced? Provisioned input/output requirements are supported by EBS gp3, io1, and io2 volumes, and Amazon EFS and FSx for Windows File Server throughput modes. |
| I/O performance requirements    | What input/output performance is required? EBS Block Express volumes offer up to 256,000 IOPS. Amazon EFS and FSx for Windows File Server offer adjustable performance modes.  |

| Workload Requirements   | Storage Solution  |
|-------------------------|---|
| Number of files         | How many files will you be storing in the cloud daily, weekly, or for the long term? Amazon S3 offers unlimited storage. Amazon EFS and FSx for Windows File Server support petabyte storage.                         |
| File size per directory | What's the largest required file size? For example, EFS storage has a maximum file size of 49 TiB. Amazon EBS volumes using the third extended file system (efs3) and 4 KB block size have a maximum 2 TiB file size. |

## Workload Requirements

### Storage Solution

Throughput for on-premises storage

What are your average data throughput requirements on premises? Are you expecting these or higher levels of performance at AWS? Do your workloads have specific IOPS requirements? Amazon EBS Block Express volumes offer up to 256,000 IOPS. Amazon EFS and FSx for Windows File Server offer burst and adjustable throughput performance levels. Amazon EBS Throughput Optimized HDDs provide a baseline for the performance of 40 MIB per TiB of storage.

## Workload Requirements

## Storage Solution

### Latency

What latency can your workload handle? Test workload latency using EC2 optimized instances with dedicated, high-bandwidth connections to EBS storage, which enables them to fully utilize the I/O capacity of the attached EBS volumes. This can significantly improve the performance of EBS-intensive workloads, such as databases, big data processing, and media processing.

---

### Note

The terms IOPS and PIOPS are often used in technical documentation related to storage in the cloud without any explanation. **IOPS** stands for **input/output operations per second**. The higher the number, the faster the HDD, SSD, or SAN. With provisioned IOPS (PIOPS), a specific amount of

IOPS can be ordered or provisioned. AWS supports up to 256,000 IOPS for EBS Block Express volumes. Every hard drive has a particular measurement of performance that defines how quickly data can be stored (written) or retrieved (read).

---

## **Amazon Elastic Block Store**

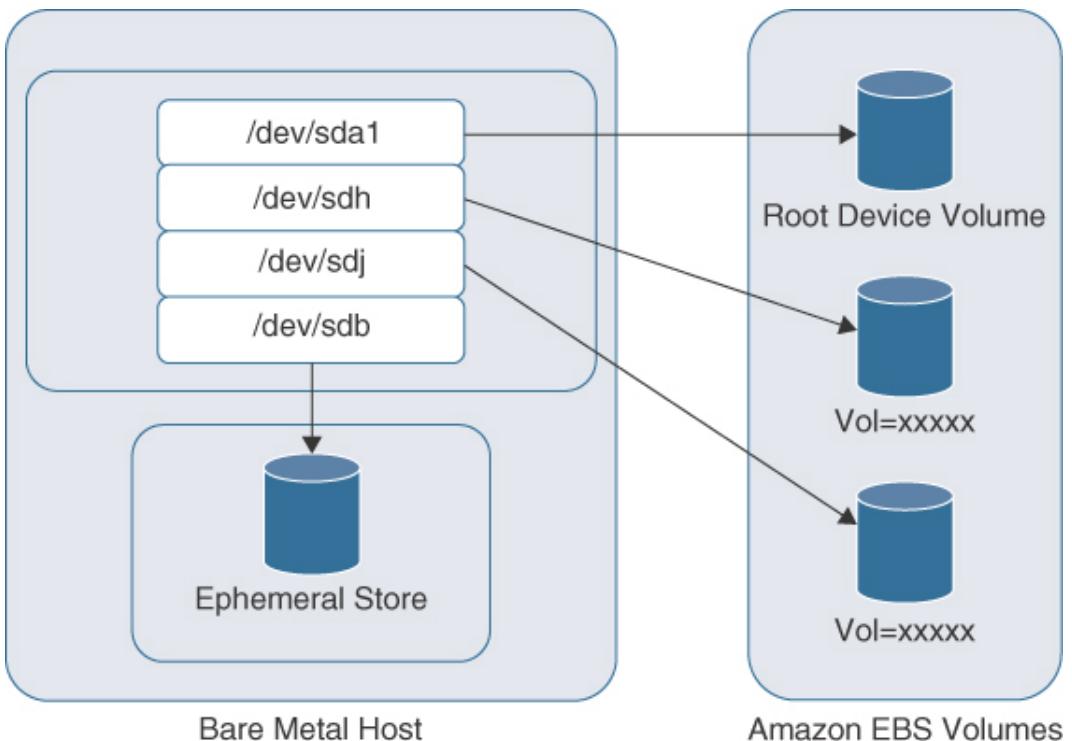
Amazon Elastic Block Store (EBS) is a block-level storage service for use with Amazon EC2 instances. It provides persistent storage, and it can be configured to deliver high levels of I/O performance. EBS volumes are attached to EC2 instances and are exposed as block devices. They can be used as raw block devices or can be formatted with a file system and used to store files. EBS volumes can also be backed up by creating snapshots of each EBS volume that are stored in Amazon S3 as a point-in-time backup.

The “E” in EBS defines the “elastic volumes” feature that enables you to increase your current volume capacity and performance or change your volume after an EBS volume has been created.

Amazon CloudWatch metrics and alarms can be used to define and monitor a desired baseline for each EBS volume. When

baseline values for capacity or performance have been breached, Amazon CloudWatch alarms can alert the Simple Notification Service (SNS). SNS, in turn, can notify Amazon Lambda, which can execute a custom function to increase capacity and performance or carry out any specific task as required.

When an EBS volume is created, the blocks for the volume are spread across multiple storage arrays, providing a high level of redundancy and durability. EBS volumes are stored within the same availability zone (AZ) where your instances reside, as shown in [Figure 8-2](#), providing 99.8% to 99.999% durability, depending on the type of EBS volume created.



**Figure 8-2** EBS Data Storage Architecture

Each EBS volume can be attached to any EC2 instance located in the AZ where the EBS volume resides. Multiple EBS volumes can also be attached to a single EC2 instance. A feature called Multi-attach allows up to 16 *Nitro*-backed EC2 instances to concurrently attach to a single EBS volume at the same time.

To use multi-attach, create a volume with the multi-attach attribute enabled, and then attach it to the desired EC2 instances. All instances will have read/write access to the volume, and any changes made to the volume by one instance will be visible to all other instances.

Multi-attach is only supported on Amazon EBS Provisioned IOPS SSD io1 and io2 volume types and is supported on a limited number of EC2 instance types.

Multi-attach can be useful in scenarios where you want to ensure high availability for your data using a cluster-aware file system such as Red Hat Global File System2 (GFS2).

---

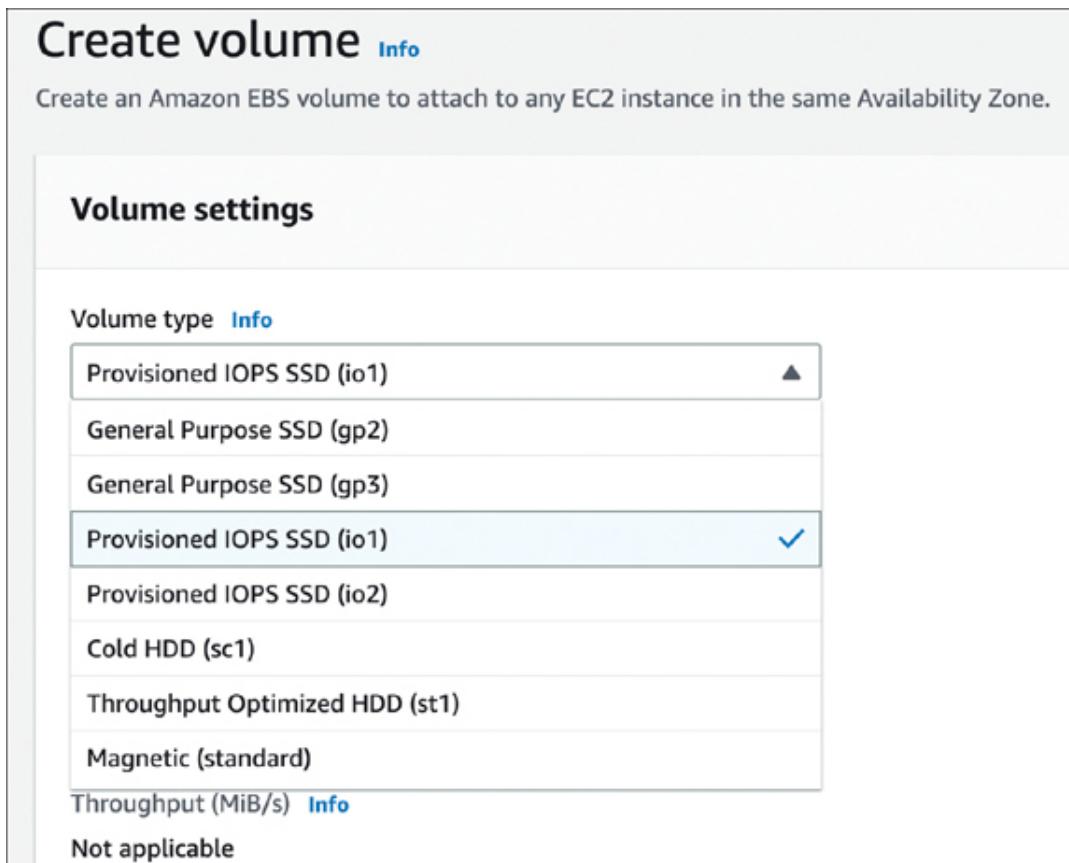
#### Note

Boot and data records should always be separated. Never store data records on the boot drive of an EC2 instance; boot volumes should only contain the operating system and the application. Storing data on separate EBS volumes allows you to back up (snapshot) the data volume as a separate process, making backup and restoration easier. Separating the boot and data volumes also makes it easier to plan the size, speed, and usage of your boot volumes. Separate EBS data volumes can have higher provisioned IOPS (for example, with database storage volumes).

---

## EBS Volume Types

Several EBS volume types are available depending on the required use case and workload requirements (see [Table 8-3](#)). Different workloads will require different performance needs; for example, various volume types can be selected for database storage volumes, or data that is not accessed as frequently, as shown in [Figure 8-3](#). For example, an EBS boot volume will not need to be as fast or as large as a high-performing EBS database volume.



**Figure 8-3** EBS Volume Options

The following EBS volumes can be chosen:

**Key Topic**

- General Purpose SSD gp2 volumes provide a baseline of 3 IOPS per GiB of volume storage with a minimum of 100 IOPS, burstable to 3000 IOPS with single-digit-millisecond latency and 99.8% to 99.9% durability. Volume sizes are Min: 1 GiB, Max: 16384 GiB.
- General Purpose SSD gp3 volumes also provide a minimum baseline of 3000 IOPS with a maximum of 16000 IOPS that can be selected with single-digit-millisecond latency, and 99.8% to 99.9% durability. Volume sizes are Min: 1 GiB, Max: 16384 GiB.
- Provisioned IOPS SSD io1 volumes can select IOPS from 100 to 4,950 IOPS with single-digit-millisecond latency, and 99.8% to 99.9% durability. Volume sizes are Min: 4 GiB, Max: 16384 GiB.
- Provisioned IOPS SSD io2 volumes can select from 100 IOPS up to a maximum 99,000 IOPS. Volume sizes are Min: 4 GiB, Max: 65536 GiB. Provisioned IOPS SSD (io2) volumes with a size greater than 16 TiB and IOPS greater than 64,000 are

supported only with instance types that support io2 Block Express volumes.

- EBS io2 IOPS SSD Block Express volumes are designed to provide 4,000 MIB/s throughputs per volume, 256,000 IOPS/volume, up to 64 TiB storage capacity, and 99.999% durability with sub-millisecond latency. These increased speeds are due to the Nitro hypervisor and Nitro I/O card dedicated for EBS I/O, which is embedded on the bare-metal server hosting the Nitro hypervisor. Selecting EC2 R5b instances in the EC2 Dashboard allows you to select io2 Block Express volumes, as shown in [Figure 8-4](#).

| Step 4: Add Storage |              |                        |            |   |  |
|---------------------|--------------|------------------------|------------|---|--|
| Volume Type         | Device       | Snapshot               | Size (GiB) | Volume Type   | IOPS                                   |
| Root                | /dev/xvda    | snap-08ccb15f1c8eb5387 | 8          | ✓ General Purpose SSD (gp2)<br>General Purpose SSD (gp3)<br>Provisioned IOPS SSD (io1)<br><b>Provisioned IOPS SSD (io2)</b><br>Magnetic (standard)<br>SSD (NVMe AMI required) | 100 / 3000<br>N/A<br>N/A<br>N/A<br>N/A |
| ephemeral0          | /dev/nvme0n1 | N/A                    | 900        |   |  |
| ephemeral1          | /dev/nvme1n1 | N/A                    | 900        |   |  |
| ephemeral2          | /dev/nvme2n1 | N/A                    | 900        |   |  |
| ephemeral3          | /dev/nvme3n1 | N/A                    | 900        | SSD (NVMe AMI required)   | N/A                                    |

**Figure 8-4** Provisioned IOPS SSD (io2) Selection

- **Throughput Optimized** HDD volumes (st1) provide good throughput performance for sequential I/O with a set

throughput/volume at 40 MiB/s per TiB. Volume sizes are Min: 4 GiB, Max: 16,384 GiB.

- Cold HHD (sc1) volumes have a max throughput/volume of 12 MiB/s per TiB of storage. Volume sizes are Min: 125 GiB, Max: 16,384 GiB.
- Magnetic HDD volumes average 100 IOPS. Volume sizes are Min: 1 GiB, Max: 1,024 GiB.

**Table 8-3** Performance Specifications for EBS Volume Types

|            | EBS         |               |               |
|------------|-------------|---------------|---------------|
|            | Provisioned |               |               |
| Parameter  | IOPS SSD    | io2           | io1           |
| Block      |             |               |               |
| Express    |             |               |               |
| Use case   | SAP HANA    | I/O-intensive | I/O-intensive |
|            |             | NoSQL         | NoSQL         |
| Durability | 99.999%     | 99.999%       | 99.8%–99.9%   |

|                              | EBS                    |             |             |
|------------------------------|------------------------|-------------|-------------|
| Parameter                    | IOPS SSD               | io2         | io1         |
| Size                         | Express<br>4 GB–64 TiB | 4 GB–16 TiB | 4 GB–64 TiB |
| <b>Max IOPS/volume</b>       | 256,000                | 64,000      | 64,000      |
| <b>Max IOPS/instance</b>     | 260,000                | 160,000     | 260,000     |
| <b>Max throughput/volume</b> | 7,500 MiB/s            | 4,750 MiB/s | 7,500 MiB/s |

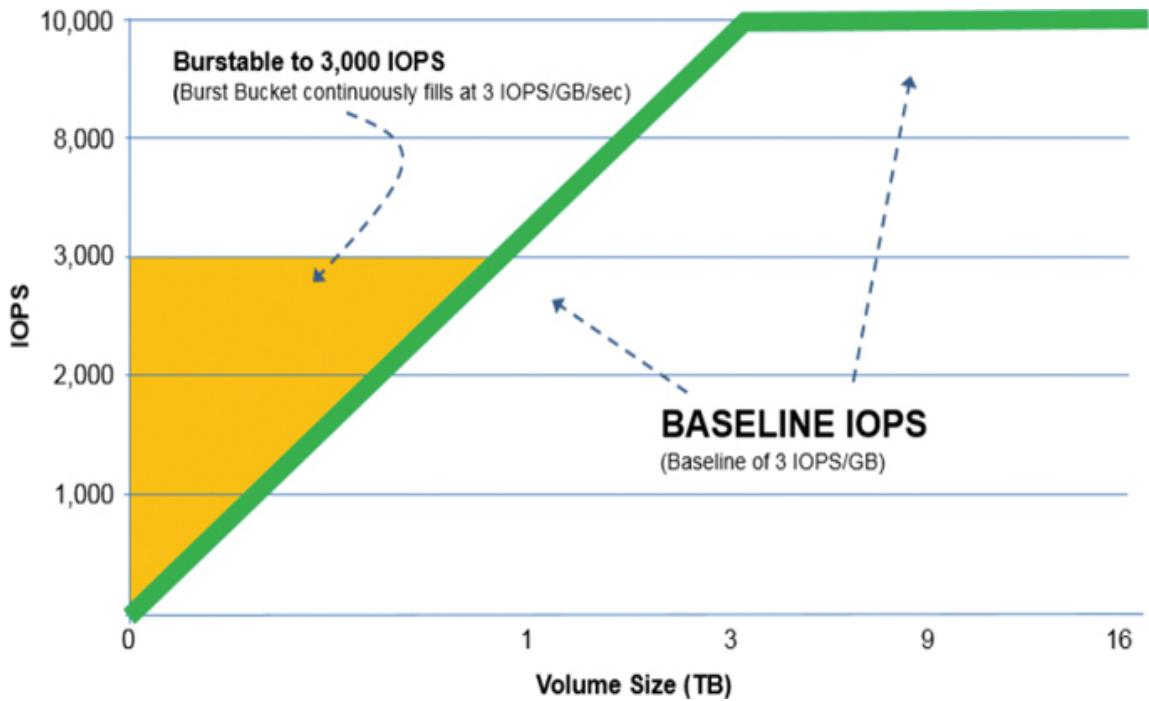
## General Purpose SSD (gp2/gp3)

General Purpose SSD gp2 is designed with a minimum baseline of 3 IOPS per GiB with a minimum of 100 IOPS, burstable to 3,000 IOPS. Gp2 volumes come with a baseline level of I/O performance and a maximum burst threshold, and the burst capability of a gp2 volume is determined by the number of I/O credits it has available. If a gp2 volume runs out of I/O credits,

its I/O performance will be limited to the baseline level until it earns more I/O credits. Bursting is designed for the use case where applications have periods of idle time followed by periods of high IOPS. The smallest gp2 drive can burst to 3,000 IOPS while maintaining a single-digit-millisecond latency with throughput up to 160 MiB/s.

All EBS General Purpose gp2 SSD storage volumes support bursting using a burst token bucket design, as shown in [Figure 8-5](#). During quiet idle times, each volume accumulates burst tokens at 3 IOPS per gigabyte per second. When the driver needs additional performance using the acquired burst tokens, it can burst up to higher levels of IOPS. A larger storage volume is assigned additional ***burst credits***, allowing the drive to burst for a longer time frame. For example, a 300-GB volume can burst up to 40 minutes, 500-GB volumes can burst for almost an hour, and even larger 900-GB volumes can burst for almost 10 hours.

**Key Topic**



**Figure 8-5 EBS Burst Credit Architecture**

General Purpose SSD gp3 volumes do not support bursting but include 3000 IOPS per second and 125 MiB/s of consistent performance at no additional cost. EBS volumes created with the desired IOPS value meet their desired performance requirements 99.9% of the time.

For ***cold storage*** and Throughput-Optimized volumes, burst credits accumulate based on the size of the drive. A storage-optimized volume accumulates burst credits at 40 MiB/s per terabyte; a cold storage volume accumulates burst credits at 12 MiB per terabyte.

If you have a large sequential workload running for 3 hours on a Throughput-Optimized volume, approximately 5.4 TIB of data will be transferred. On the other hand, if your workload storage is random, using the same drive in the same working time frame of 3 hours, 87 GB of data will be transferred.

---

#### Note

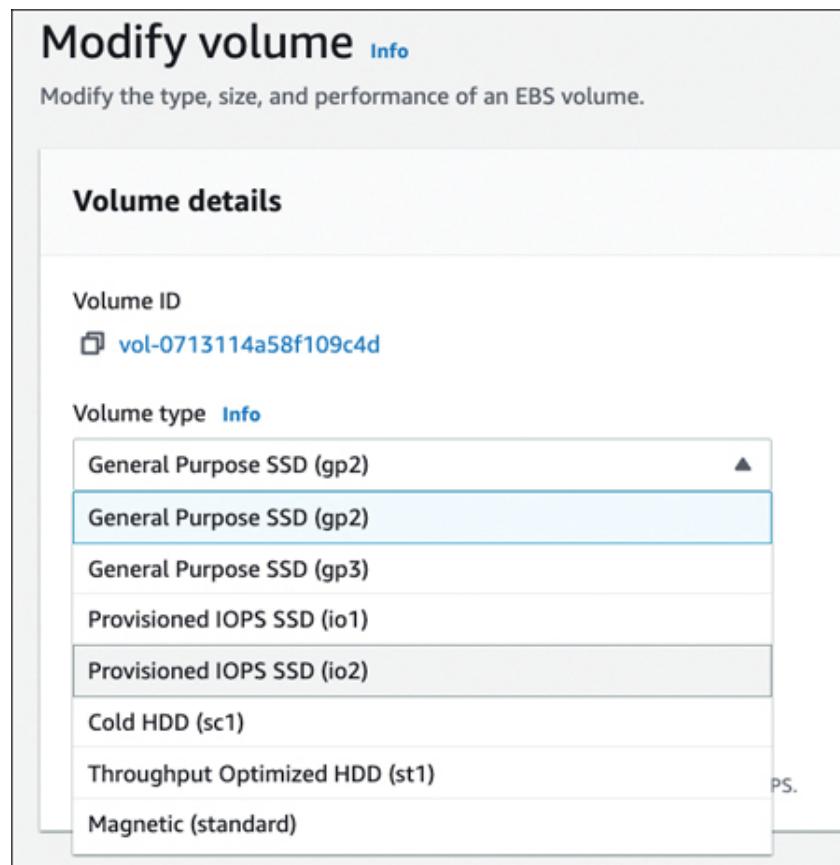
To carry out workload testing on a Linux system, you could use the utility **iostat**; for Windows systems, use the built-in **Perfmon**. Testing results are displayed in sectors. You then need to know the number of bytes you're using per sector and multiply the sectors by the number of bytes to calculate the current workload transfer rate.

---

## Elastic EBS Volumes

Amazon EBS enables you to increase the current volume size and change the volume type, size, and, if applicable, change provisioned IOPS in near real time, as shown in [Figure 8-6](#). There is no maintenance window required when manually scaling an EBS volume. Elastic volumes provide cost savings; organizations do not have to overprovision storage volumes by planning for future growth at the start. Amazon RDS MySQL

and Microsoft SQL Server deployments support Storage autoscaling; EBS volumes can be set to a maximum storage threshold providing dynamic scaling support when additional storage is required. AWS CloudWatch can also monitor EBS volumes' read and write throughput and latency using Amazon CloudWatch metrics and send alarms to the AWS Simple Notification Service when issues arise. To monitor the free space on your EBS volumes using the CloudWatch Agent, you can configure the CloudWatch Agent to collect the VolumeFreeSpace metric for your EBS volumes. Next, set up a CloudWatch alarm to trigger an alert when the VolumeFreeSpace metric falls below a certain threshold.



**Figure 8-6** Modifying EBS Volumes

## Attaching an EBS Volume

When an EBS volume is attached to an EC2 instance, several background processes begin running, depending on the hypervisor being used to host the EC2 instance:

- **Instances hosted on the Xen hypervisor:** Instances are associated with a system component called an I/O domain. When an EBS volume is attached, it is attached to the I/O domain. Another system component, called a *shared memory*

*segment*, is initiated at the hypervisor architecture level, acting as a queue to handle data access from the instance to the EBS volume.

When an instance wants to perform an I/O call, it first submits a system call to the kernel. Assuming that it's a read request, a data block is placed in the hypervisor queue. Next, the I/O domain takes the data from the hypervisor queue and sends it to the defined queue, which delivers the data blocks to the EBS volume.

- **Instances hosted on the Nitro hypervisor:** A PCI interface is used to directly interface with the Nitro card, which presents a direct storage interface to the EC2 instance. Storage requests in this design are submitted directly to the PCI interface; the Nitro hypervisor has no part to play in communicating with EBS storage volumes. The Nitro interface provides a huge performance boost with minimal latency to local instance storage hosted on the Xen hypervisor.



## Amazon EBS Cheat Sheet

For the AWS Certified Solutions Architect – Associate (SAA-C03) exam, review the details for creating and managing EBS volumes:

- EBS volumes are network-attached storage volumes attached to an EC2 instance.
- There are eight volume types: Provisioned IOPS SSD (io2 Block Express, io2, io1), General Purpose SSD (gp2, gp3), Throughput Optimized (st1), and Cold HDD (sc1).
- EBS volumes support three storage categories: SSD storage for transactional workloads, HDD storage for cold storage and throughput optimized workloads, and magnetic hard drives.
- Multiple EBS volumes can be attached to a single EC2 instance.
- EBS volumes can be attached and detached from a running EC2 instance.
- EBS volumes must be in the same AZ where the EC2 instance is located to be attached.
- Use AWS Backup to back up your attached EBS volumes with EBS snapshots on a schedule.
- By default, root EBS volumes are set for deletion on termination.
- By default, non-boot volumes are not deleted on termination.

- EBS data volumes should have a snapshot schedule in place for backing up EBS data volumes as required.
- Don't store application data on your root EBS boot volumes.
- EC2 instances can be created with encrypted EBS boot and data volumes.
- EBS volumes allow volume size, volume type, and IOPS changes while online.
- Create separate EBS volumes for boot volumes.
- Ephemeral storage volumes are faster than EBS volumes for storing temporary files.

## EBS Snapshots

An EBS snapshot is a point-in-time copy of an EBS volume. Snapshots are stored in controlled Amazon S3 object storage linked to your AWS account. *Controlled storage* means that AWS creates and manages the S3 storage location, but each customer has access to their snapshots through the EBS console or CLI.

The first time a snapshot of an EBS volume is taken, every block of the EBS volume is part of the primary snapshot captured and stored in controlled S3 storage. From this point forward, every additional snapshot is created as an incremental snapshot that records all the changes since the last snapshot. Only newly written and changed volume blocks are pushed to the

incremental snapshot. When you delete a snapshot, only the data exclusive to the snapshot copy is retained. Each snapshot has a unique identifier, and new EBS volumes can be created from any snapshot.

Snapshots can be shared by modifying the permissions of the snapshot. Snapshots can be shared publicly or privately with other AWS accounts using the EC2 Dashboard, selecting Private or Public Snapshots.

---

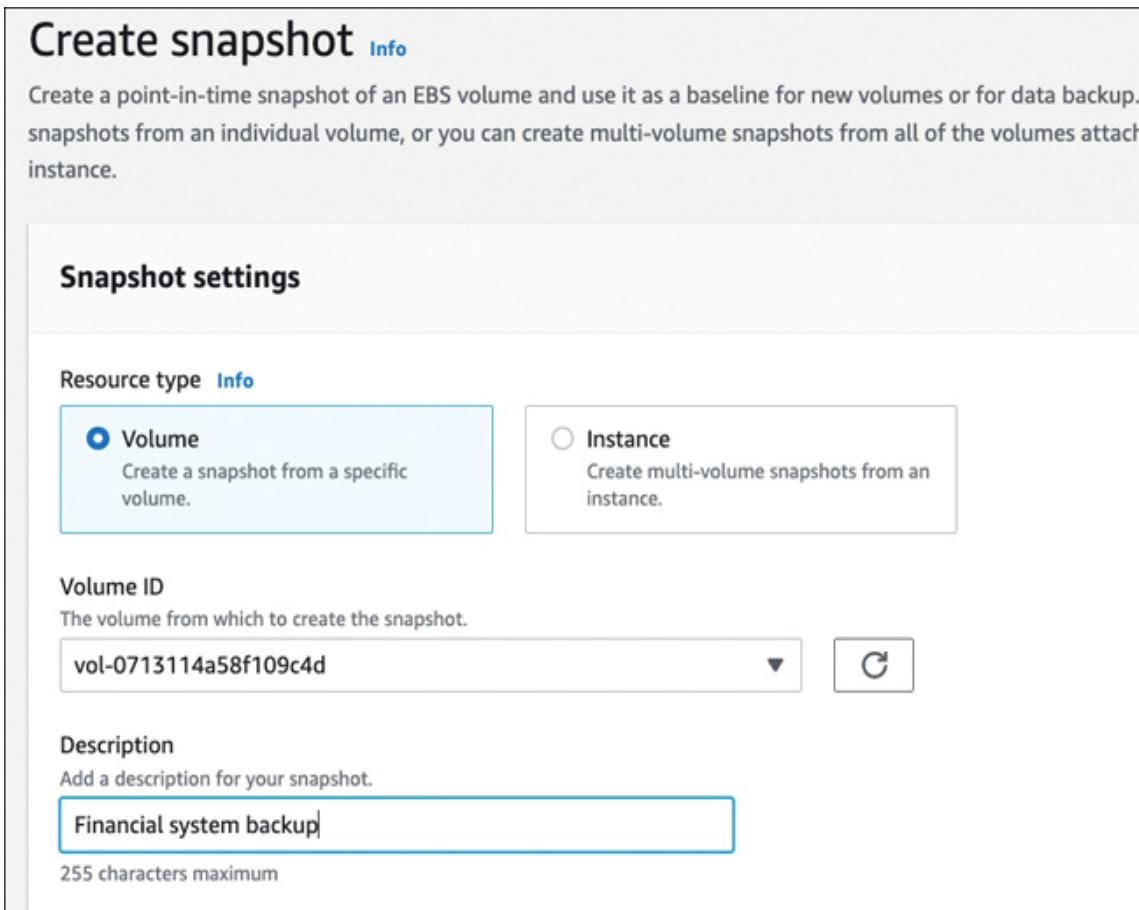
#### Note

Deploying a daily snapshot schedule greatly reduces the possibility of data loss. AWS has three options for managing snapshots. First, the Data Lifecycle Manager allows you to schedule the creation and deletion of EBS snapshots. Second, AWS Backup allows you even more control of your storage by centrally managing EBS snapshots, including any Amazon RDS snapshots, Amazon Storage Gateway snapshots, Amazon EFS backups, and Amazon DynamoDB backups. The third option is to create a CloudWatch Event to trigger a custom Lambda function on a set schedule to carry out the snapshot process.

---

## Taking a Snapshot from a Linux Instance

After you begin the snapshot process, the snapshot process continues in the background. If there are a small number of volume blocks to change, the snapshot will be created quickly; with many changes, the snapshot process will take longer to complete. Before beginning the snapshot process for an EBS boot volume, make sure to detach the volume to stop any ongoing I/O processes. After starting the snapshot process, as shown in [Figure 8-7](#), the snapshot ID confirms that the snapshot is being created.



**Figure 8-7** Creating a Snapshot

## Taking a Snapshot from a Windows Instance

For Windows instances, the windows operating system uses the **sync** utility to *quiesce* the file system and create the snapshot. Windows Server has a built-in operating system service utility called the Volume Shadow Copy Service (VSS) for creating application-consistent snapshots. The AWS Systems Manager (SSM) can automate Windows VSS commands such as **ec2-create-snapshot** to schedule snapshots of Windows EC2

instances. Windows Server images AMI 2017.11.21 and higher include the SSM and VSS snapshot utilities.



## Fast Snapshot Restore

Fast snapshot restore is a feature of Amazon EBS that enables you to quickly restore an EBS volume from a snapshot, creating a new volume from the snapshot and attaching it to an Amazon Elastic Compute Cloud (EC2) instance.

Fast snapshot restore enables you to skip the process of creating a new volume and then copying the data from the snapshot to the new volume. Instead, the new volume is created directly from the snapshot, which can save a significant amount of time.

Amazon EBS fast snapshot restore (FSR) enables you to speed up restoring data to multiple EBS volumes from a snapshot. FSR has been designed to help speed up the snapshot restore process for virtual desktop environments and custom AMIs.



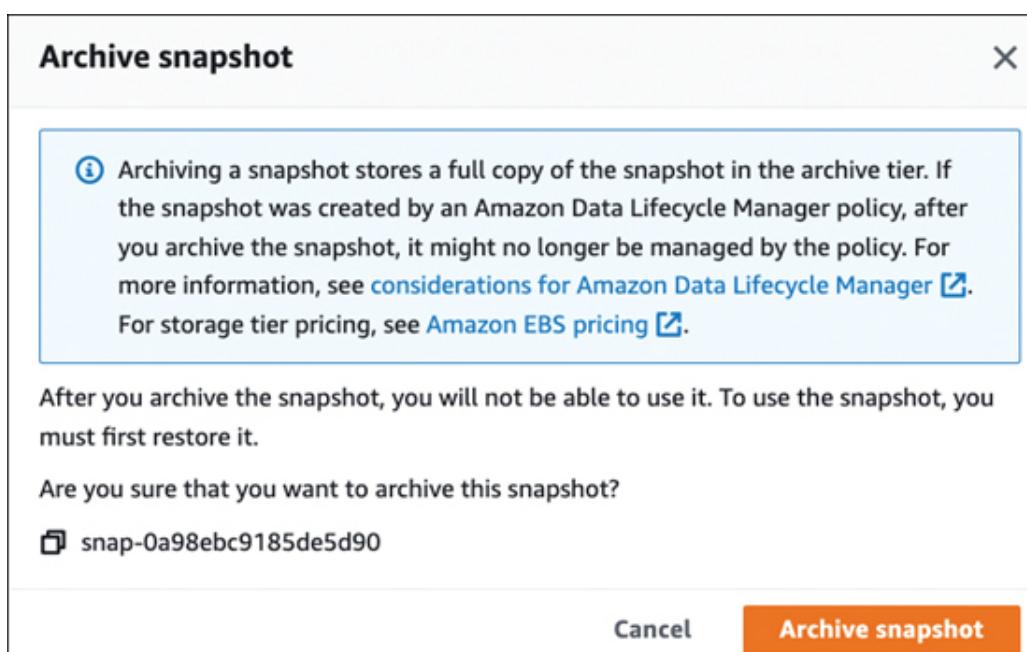
## **Snapshot Administration**

Various administrative tasks need to be considered when planning for disaster recovery, or creating EBS volumes from existing snapshots:

- **Within an AWS region:** Snapshots can be used to create an EBS volume within any AZ within the same region where the snapshot is stored.
- **To another AWS region:** By using the EBS copy utility via the AWS CLI or Amazon Data Lifecycle Manager, snapshots can be copied to multiple AWS regions.
- **Launching EC2 instances:** EBS boot volumes can be used to create a new EC2 instance.
- **Rebuilding a database:** AWS RDS database instance snapshots can be used to create a new database instance.
- **Creating volumes:** Existing snapshots can be used to create new EBS volumes.

Typically, snapshots are stored in the EBS Snapshot Standard tier, which supports incremental EBS snapshots. A new storage tier called Amazon EBS Snapshots Archive is available for low-cost, long-term storage of rarely accessed snapshots that do not require fast retrieval, as shown in [Figure 8-8](#). Organizations can archive a snapshot for long-term storage using the Amazon EBS

Snapshots Archive. The incremental snapshot is converted to a full snapshot and moved to the EBS Snapshots Archive tier. Amazon EBS Snapshots Archive storage has 75% lower snapshot storage costs for archived snapshots stored over 90 days. When an archived snapshot needs to be restored, it is moved from the archive tier to the standard tier and then restored.



**Figure 8-8** EBS Snapshots Archive

## EBS Recycle Bin

The Amazon EBS Recycle Bin feature enables you to recover EBS snapshots and volumes that were deleted within the past

90 days. When an EBS snapshot or volume is deleted, it is moved to the Recycle Bin, where it can be recovered if needed.

To use the EBS Recycle Bin, you must first enable the feature in the AWS Management Console. View and recover deleted snapshots and volumes by navigating to the EBS Dashboard and select the Recycle Bin, selecting the snapshots or volumes that you want to recover. Both tag-level and region-level retention rules are supported.



## Snapshot Cheat Sheet

For the AWS Certified Solutions Architect – Associate (SAA-C03) exam, you need to understand these snapshots details:

- Snapshots are saved incrementally in Amazon S3 controlled storage.
- Snapshots are region-specific, whereas EBS volumes are AZ-specific.
- A snapshot can be copied to another AWS region.
- Snapshots can be archived for low-cost, long-term retention.
- When a snapshot is encrypted, encryption keys are created and managed by the AWS Key Management Service (KMS).

- Snapshots use AES-256 encryption.
- The Amazon Data Lifecycle Manager can be used to create a snapshot management schedule that controls the creation and deletion of snapshots.
- The Amazon Data Lifecycle Manager can be used to create a snapshot management schedule that stores snapshots across multiple AWS Regions.
- Snapshots that have accidentally been deleted can be restored from the EBS Recycle Bin.
- EBS snapshots can be stored locally in a local AWS Outposts deployment.

## **Local EC2 Instance Storage Volumes**

There are two main types of storage that EC2 instances can directly use: persistent EBS block storage volumes and temporary block storage (also called ephemeral storage).

An ephemeral storage volume is a hard drive (SSD/HDD) physically attached to the bare-metal server where the EC2 instance is hosted.

The local temporary block storage volume is shared between the hosted EC2 instances that support ephemeral storage. Depending on the EC2 instance type selected, there could be one

or more SSD volumes exposed as ephemeral storage devices. Ephemeral storage volumes are numbered from 0 to 23 and are labeled as ephemeral 0 to ephemeral 23, as shown in [Figure 8-9](#).

| Volume Type | Device       | Snapshot               | Size (GiB) | Volume Type               | IOPS       |
|-------------|--------------|------------------------|------------|---------------------------|------------|
| Root        | /dev/xvda    | snap-08ccb15f1c8eb5387 | 8          | General Purpose SSD (gp2) | 100 / 3000 |
| ephemeral0  | /dev/nvme0n1 | N/A                    | 7500       | SSD (NVMe AMI required)   | N/A        |
| ephemeral1  | /dev/nvme1n1 | N/A                    | 7500       | SSD (NVMe AMI required)   | N/A        |
| ephemeral2  | /dev/nvme2n1 | N/A                    | 7500       | SSD (NVMe AMI required)   | N/A        |
| ephemeral3  | /dev/nvme3n1 | N/A                    | 7500       | SSD (NVMe AMI required)   | N/A        |
| ephemeral4  | /dev/nvme4n1 | N/A                    | 7500       | SSD (NVMe AMI required)   | N/A        |
| ephemeral5  | /dev/nvme5n1 | N/A                    | 7500       | SSD (NVMe AMI required)   | N/A        |
| ephemeral6  | /dev/nvme6n1 | N/A                    | 7500       | SSD (NVMe AMI required)   | N/A        |
| ephemeral7  | /dev/nvme7n1 | N/A                    | 7500       | SSD (NVMe AMI required)   | N/A        |

**Figure 8-9** Instance Storage

Ephemeral storage volumes can be useful for buffers and caches and for storing temporary data records. However, ephemeral storage volumes are *non-persistent data stores*; when an EC2 instance is turned off or terminated, the storage is discarded (however, it *does* survive a reboot). Ephemeral storage data records do not have any long-term durability because they are not automatically replicated to another location, and there is no integrated support with the existing EBS Snapshots service. A use case for choosing ephemeral

storage for high-performance database storage could involve the following components:

- Choosing ephemeral storage—for example, an i2 EC2 instance for a large Microsoft SQL Server database—would provide incredible speed for the database records at the local storage level.
- For redundancy, additional EBS volumes designed for a further level of throughput optimization or IOPS could be added as volumes for full and partial backups of the ephemeral storage volumes and transaction logs.
- A third-party VSS-aware snapshot utility could be installed locally on the Microsoft SQL server EC2 instance performing backup from the ephemeral storage volumes to the additional EBS volumes.
- EBS snapshots could then be taken from the additional EBS volumes for safekeeping.

Testing has shown that EC2 NVMe instance storage is more than five times faster than EBS SSD general drives for the cached reads, and it is more than ten times faster for writes. (For specifications, see

<https://docs.aws.amazon.com/AWSEC2/latest/UserGuide/storage-optimized-instances.html>.) EC2 instances with both EBS *and* ephemeral volumes deployed could provide some advantages:

- Boot drives could be EBS SSD volumes, providing fast boot times, and the ephemeral storage could be used to store cached data or logs of the hosted web application.
  - Ephemeral storage could be considered for workloads where you do not need the data volumes and storage records to persist.
- 

#### Note

With proper planning, you may be able to take advantage of ephemeral storage volumes. And ephemeral storage, if available, is included in the computed price of each EC2 instance.

---

## Amazon Elastic File System

Amazon Elastic File System (Amazon EFS) is a fully managed, elastic file storage service (see [Figure 8-10](#)) that scales on demand up to petabytes for Linux, Windows, and macOS instances. EFS removes the need to provision and attach EBS volumes for data storage, and operates as a shared storage service that allows you to concurrently serve EC2 instances hosted on subnets within select VPCs by using NFS **mount points**. The key features of EFS include the following:

- **Availability and durability:** Choose Regional to store data redundantly across multiple availability zones. Choose One Zone to store data redundantly within a single availability zone. Amazon EFS can store and share data access for Amazon EC2 instances, containerized applications, and on-premises servers.
- **No networking or file layer to manage:** There are no EBS drives to provision, manage, and pay for. Pay for the storage used and the level of performance.
- **Elastic:** The EFS file system automatically scales as you add and remove files; you do not need to select an initial storage size.
- **Scale:** EFS can scale to petabytes of capacity and performance can scale along with the increase in size.
- **Performance:** Two performance modes are available: General Purpose and Max I/O. Max I/O is designed for thousands of instances that need access to the same files at the same time.
- **Compatible:** Amazon EFS supports the Network File System (NFS) protocol supported by a wide range of applications and operating systems.
- **Lifecycle management:** Amazon EFS Intelligent-Tiering automatically transitions files in and out of Standard-

infrequent Access storage based on a defined number of days since last access.

### General

**Name - optional**  
Name your file system.

Graphics\_Depot

Name must not be longer than 256 characters, and must only contain letters, numbers, and these characters: + - = . \_ : /

**Availability and durability**  
Choose Regional (recommended) to create a file system using regional storage classes. Choose One Zone to create a file system using One Zone storage class.

**Regional**  
Stores data redundantly across multiple AZs

**One Zone**  
Stores data redundantly within a single AZ

**Automatic backups**  
Automatically backup your file system data with AWS Backup using recommended settings. Additional pricing applies. [Learn more](#)

**Enable automatic backups**

**Lifecycle management**  
EFS Intelligent-Tiering uses Lifecycle Management to automatically achieve the right price and performance blend for your application by moving your files between Standard and Infrequent Access storage classes. [Learn more](#)

**Transition into IA**  
Transition files from Standard to Standard-Infrequent Access.

▾

**Transition out of IA**  
Transition files from Standard-Infrequent Access to Standard.

▾

**Performance mode**  
Set your file system's performance mode based on IOPS required. [Learn more](#)

**General Purpose**  
Ideal for latency-sensitive use cases, like web serving environments and content management systems

**Max I/O**  
Scale to higher levels of aggregate throughput and operations per second

**Figure 8-10** EFS Configuration Options

## EFS Performance Modes

**Key  
Topic**

There are two EFS performance modes (shown in [Figure 8-10](#)):

- **General Purpose:** The general purpose mode is assigned a throughput performance profile that supports up to 7,000 operations per second per file system deployment; general purpose is recommended as the starting mode of operation. Amazon recommends that you continue to monitor each EFS file system using the CloudWatch metric PercentIOLimit; if the file system is operating close to 100%, choose Max I/O.
- **Max I/O:** Max I/O scales to a much higher level of throughput and operations per second and was designed for situations where thousands of EC2 instances are attached to a single EFS file system deployment, or for big data and data warehousing workloads.

To select a performance mode for an Amazon EFS file system, use the AWS Management Console or the AWS CLI. The performance mode of an existing EFS file system can be changed at any time.

**Key  
Topic**

## EFS Throughput Modes

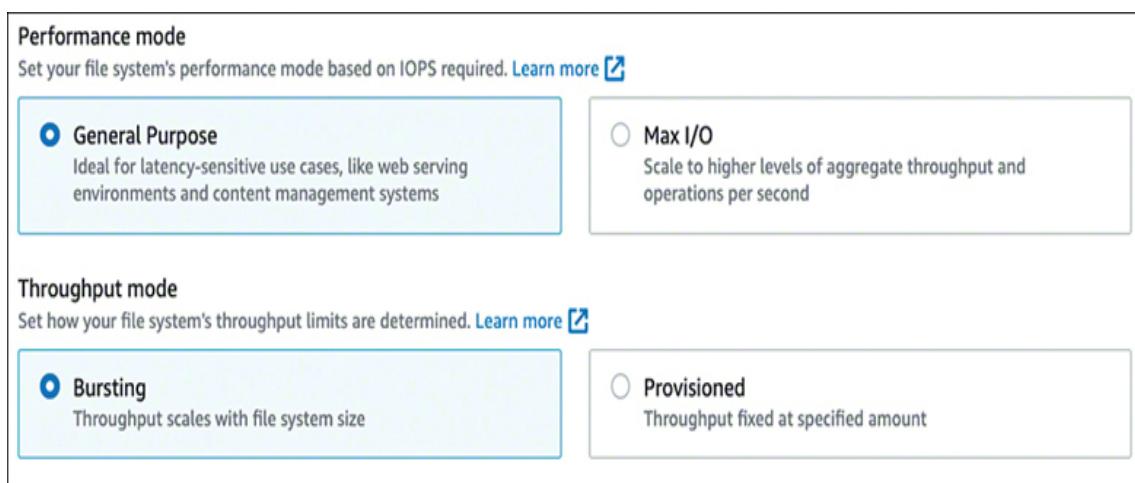
There are two EFS throughput modes:

- **Bursting:** In bursting mode, Amazon EFS automatically adjusts the throughput of the file system up or down in response to changes in demand, enabling you to burst to higher levels of throughput when required. With the default throughput mode selected, when the EFS file system throughput remains below the assigned baseline rate, it earns burst credits that are saved for future throughput requirements. When the file system requires additional throughput, the saved burst credits are utilized for read and write performance throughput above the current baseline. New file systems have an initial credit burst balance of 2.1 TiB.

The overall throughput is designed to increase as the number of stored files increases. As files are added to the file system, the amount of throughput allowed is increased based on the allotted file size. For example, a 5-TiB EFS file system can burst to 500 MiB/s of throughput ( $5 \text{ TiB} \times 100 \text{ MiB/s per terabyte}$ ); a 10 TiB file system can burst to 1000 MiB/s of throughput. Using the CloudWatch metric `BurstCreditBalance`, you can monitor the current burst credit

balance. Move up to the Provisioned throughput mode with a few mouse clicks, as shown in [Figure 8-11](#).

- **Provisioned:** In provisioned mode, specify the desired level of throughput for the file system, and Amazon EFS will automatically provision the necessary resources to meet the requested throughput. Provisioned Throughput can scale up to 3 GiB/s for read operations and 1 GiB/s for write operations per file system deployment. Customers are billed for the EFS storage used and any throughput that has been provisioned above the baseline.

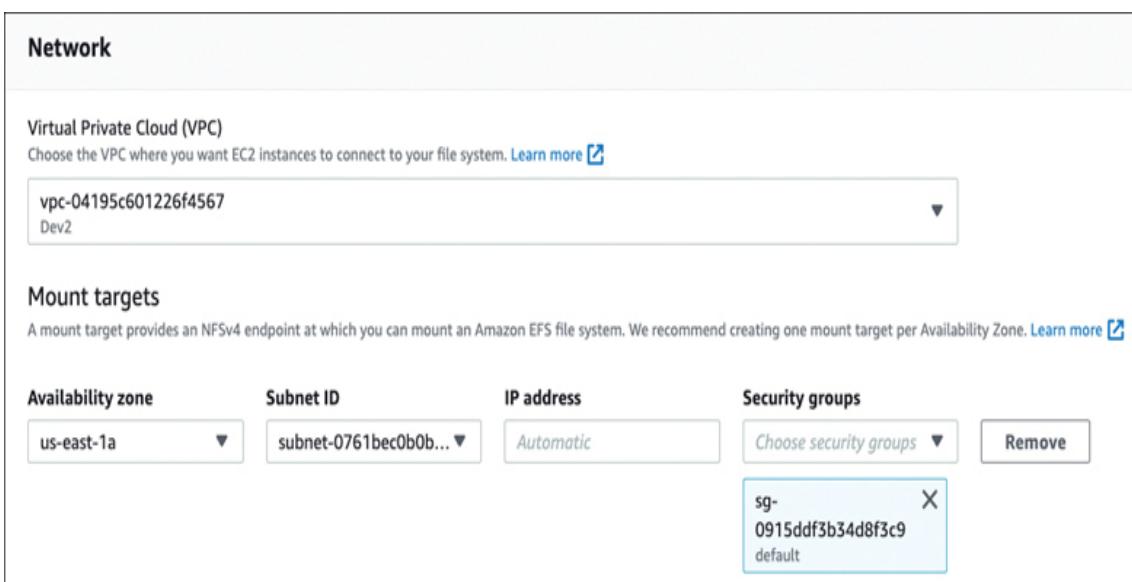


**Figure 8-11** Selecting EFS Performance and Throughput

## EFS Security

After creating an EFS file system and mount points from the selected VPC, you can use security groups to control the EC2

instance access to the EFS mount points (see [Figure 8-12](#)). Access to the EFS files and directories is controlled by application user and group permissions. All data stored within EFS can be encrypted at rest, and encryption keys are managed by AWS KMS.



**Figure 8-12** EFS Security

## EFS Storage Classes

Amazon EFS supports three storage classes:

- **Standard:** The default storage class is selected by default.
- **Infrequent Access:** The infrequent storage (IA) storage class is for files that are accessed infrequently but require rapid access when needed. Transition into the IA tier can be set

from 1 day to 90 days since the file was last accessed. The Infrequent Access tier provides a lower storage cost than the Standard storage tier, but has higher access fees.

- **One-Zone Infrequent Access:** This storage class is similar to the Infrequent Access storage class with the added benefit of being stored in a single availability zone. This can be useful for data that is not defined as critical; it provides a lower storage cost than Standard storage and Infrequent Access but has higher access fees.

## EFS Lifecycle Management

EFS Lifecycle Management provides cost-effective file management. Files that have not been accessed for a defined period of time are moved to the Standard-Infrequent Access (IA) storage class from the Standard storage class. A *lifecycle policy* defines the period of time before lifecycle management transitions files into or out of Standard-Infrequent Access storage (see [Figure 8-13](#)). Lifecycle transitions into IA storage can be from 7 to 90 days since the file was last accessed. Files remain in IA storage and are transitioned out of IA storage when accessed once again.

**Lifecycle management**

EFS Intelligent-Tiering uses Lifecycle Management to automatically achieve the right price and performance blend for your application by moving your files between Infrequent Access storage classes. [Learn more](#)

**Transition into IA**

Transition files from Standard to Standard-Infrequent Access.

|                           |        |
|---------------------------|--------|
| 30 days since last access | ▲      |
| None                      |        |
| 7 days since last access  |        |
| 14 days since last access |        |
| 30 days since last access | more ▾ |
| 60 days since last access |        |
| 90 days since last access |        |

**Transition out of IA**

Transition files from Standard-Infrequent Access to Standard.

|                               |  |
|-------------------------------|--|
| On first access               | ▼  |
| <input type="radio"/> Max I/O | Scale to higher levels of aggregate throughput and operations per second |

**Figure 8-13** EFS Lifecycle Policy

## Amazon EFS Cheat Sheet



For the AWS Certified Solutions Architect – Associate (SAA-C03) exam, you need to understand the following critical aspects of EFS:

- EFS storage is accessed using NFS mount points using the NFS protocol.
- Storage capacity is elastic; you pay for what you use.
- EFS storage can be attached from on-premises systems using a VPN connection or Direct Connect.

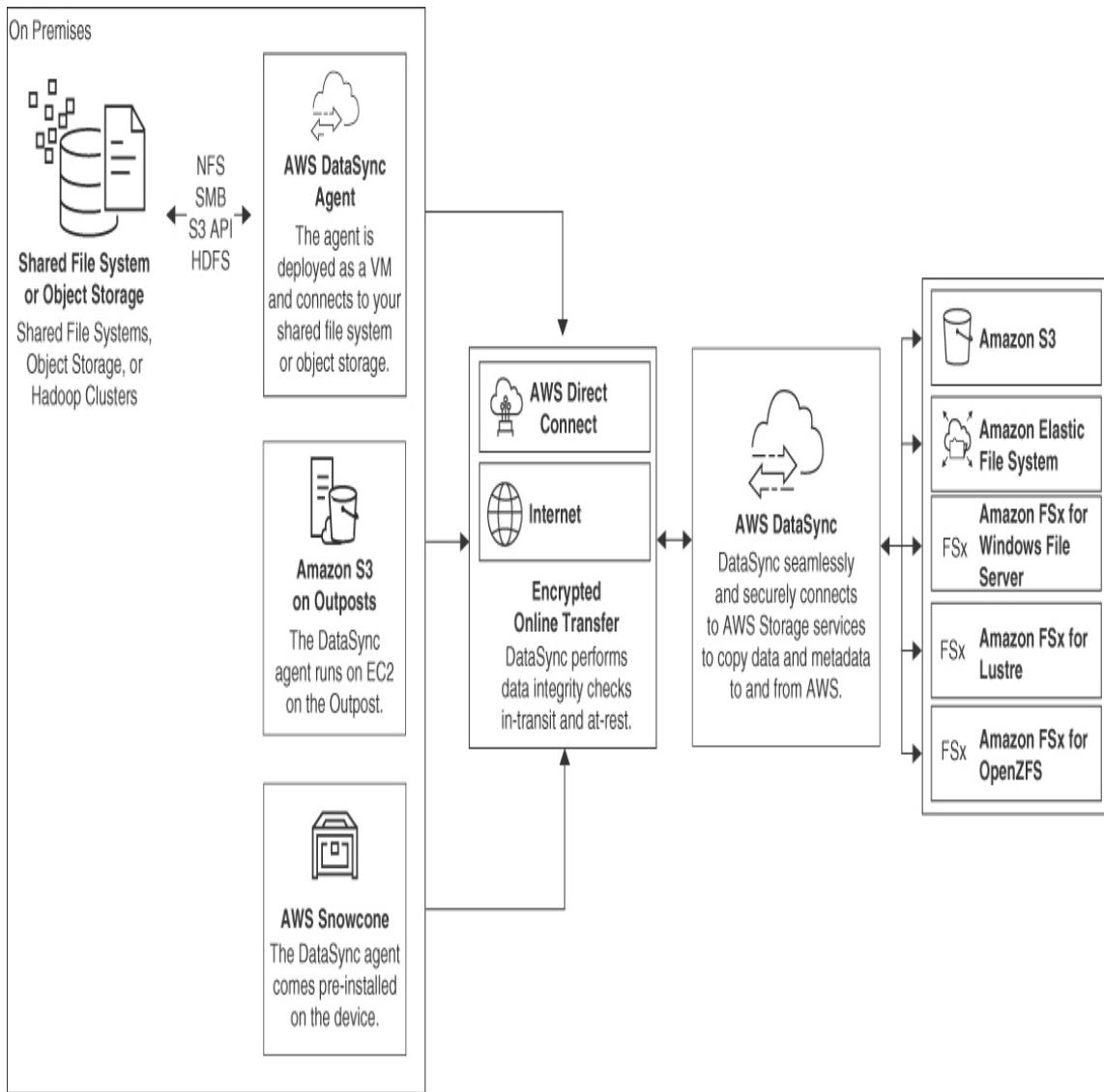
- Concurrent connections to an EFS file system can be made from multiple subnets.
- There are two EFS performance modes: General Purpose and Max I/O.
- Access to files and directories can be controlled with POSIX-compliant user and group-level permissions.
- AWS Key Management Service (KMS) manages encryption keys for encrypting and decrypting the EFS file system.
- Data encryption in transit uses TLS 1.3.
- EFS supports VMware ESXi.
- EFS supports AWS Outposts.

## AWS DataSync



AWS DataSync enables you to sync data from an on-premises NFS storage array into EFS storage, as shown in [Figure 8-14](#). AWS DataSync provides end-to-end security with encryption and integrity validation for in-transit and at-rest security. AWS DataSync is available as an installable agent that is installed to your AWS account. It also can be used to copy files from EFS file systems hosted in different AWS regions and to transfer data between AWS storage services with the exception of EBS

volumes. AWS DataSync is much faster than standard Linux copy commands: uploads utilize encrypted parallel data transfer to the selected storage destination. AWS DataSync securely accesses AWS storage controlled by a service-linked IAM role. AWS DataSync supports connections to VPC endpoints, allowing the transfer of data using the private AWS network.



**Figure 8-14** AWS DataSync

## Amazon FSx for Windows File Server

Windows servers can use Amazon FSx for Windows File Server to set up a compatible Windows file system.

Amazon FSx for Windows File Server is supported by EC2 instances and containers, as well as Amazon WorkSpaces instances. FSx for Windows File Server also supports the use of the Distributed File System (DFS) namespace. Amazon FSx for Windows File Server integrates with on-premises Microsoft Active Directory and with AWS Managed Microsoft AD. Specify the level of redundancy for your FSx for the Windows File Server file system by selecting either a Single-AZ or Multi-AZ deployment.

Amazon FSx for Windows File Server file systems can be accessed from servers located in select VPCs or from an on-premises server using either a VPN or a Direct Connect connection. Amazon FSx for Windows File Server features include:

- **Single-AZ deployment:** Amazon FSx for Windows File Server automatically replicates data within the selected availability zone. Amazon FSx for Windows File Server uses the Windows Volume Shadow Copy Service to create daily backups stored in Amazon S3.
- **Multi-AZ deployment:** A Multi-AZ deployment, shown in [Figure 8-15](#), provides continuous availability to your data when one AZ becomes unavailable; each AZ has a dedicated file server. All file changes are synchronously replicated to

both file servers in each AZ. If an AZ becomes available, control is passed to the active file server, which services all read and write requests. When resources in the preferred AZ are once again available, Amazon FSx for Windows File Server falls back to the preferred file server in the preferred AZ; failover and recovery typically take less than 30 seconds. Automatic failover occurs when any of these situations occur:

- Availability zone outage
- The preferred file server is unavailable
- The preferred file server is down due to maintenance

**File system details**

File system name - optional [Info](#)

Shared accounting records

Maximum of 256 Unicode letters, whitespace, and numbers, plus + - = . \_ : /

Deployment type [Info](#)

Multi-AZ  
 Single-AZ

Storage type [Info](#)

SSD  
 HDD

Storage capacity [Info](#)

3000  GiB

Minimum 32 GiB; Maximum 65536 GiB

**Figure 8-15** FSx Deployment Options

- **Throughput:** Amazon FSx for Windows File Server file servers use an in-memory cache for accessing active files to maintain performance. Storage choices include SSD storage, which provides sub-millisecond file operations, or HDD storage, which provides single-digit-millisecond performance. FSx throughput capacity can be adjusted from 8 MiB/s to 2,048 MiB/s. Network baseline capacity ranges from 16 MiB/s to over 3,000 MiB/s.

An Amazon FSx for Windows File Server file system configured with 2 TiB of hard disk drive storage capacity and 32 MiB/s throughput capacity has the following throughput parameters:

- Network throughput of 32 MiB/s baselines and bursting up to 600 MiB/s when required
- Disk throughput of 24 MiB/s baselines and 160 MiB/s when required
- **Windows shares:** Amazon FSx for Windows File Server is built using Windows file servers and accessed using SMB Version 2.0 to 3.11, allowing you to support older Windows 7 clients and Windows Server 2008 up to present-day versions.
- **File system:** Amazon FSx for Windows File Server file systems, built on SSDs, can be up to 64 TIB in size with more than 2 MIB/s of throughput. Multi-AZ support for Amazon FSx for Windows File Server allows you to use the Microsoft

DFS namespace to replicate between multiple locations with up to 300 PB of storage.

- **Redundancy:** Amazon FSx for Windows File Server data is stored within a single AZ or multiple AZs. Incremental snapshots are automatically taken every day. Manual snapshots are supported for additional redundancy concerns.
- **Data deduplication:** Data deduplication can be enabled with compression to automatically reduce costs for redundant data records storing duplicated files.
- **Active Directory:** Existing Microsoft Windows environments can be integrated with Active Directory deployments and Amazon FSx for Windows File Server file systems; end users can use their existing identities for access to FSx resources. Your Active Directory deployment can be self-managed or hosted and deployed using Managed AD Directory Services for Microsoft Active Directory.

## Amazon FSx for Windows File Server Cheat Sheet



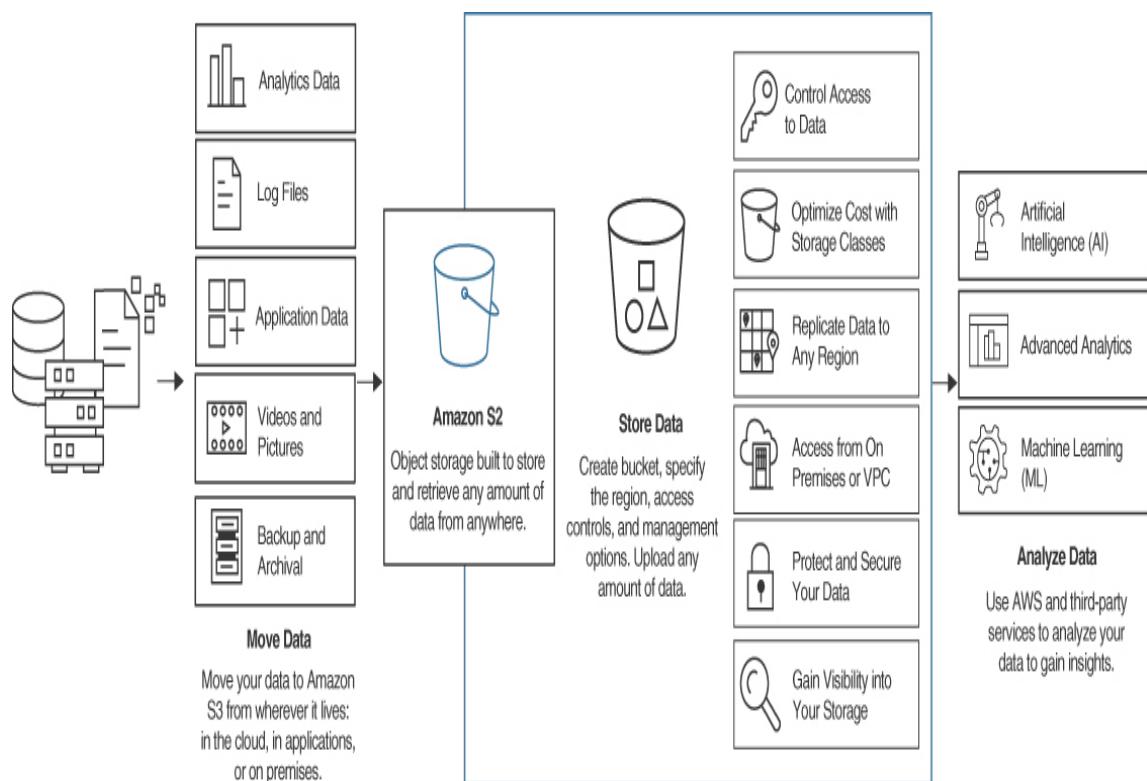
For the AWS Certified Solutions Architect – Associate (SAA-C03) exam, understand the following aspects of FSx:

- Amazon FSx for Windows File Server supports the SMB protocol, allowing connections to EC2 instances and ECs containers, VMware Cloud on AWS, and Linux and Windows applications.
- Amazon FSx for Windows File Server supports all Windows versions from Windows Server 2012 and Windows 7 forward.
- Amazon FSx for Windows File Server supports on-premises access via AWS Direct Connect or AWS VPN connections.
- Amazon FSx for Windows File Server supports access across multiple VPCs using VPC peering or AWS Transit Gateway connections.
- Amazon FSx for Windows File Server file systems are encrypted automatically at rest and in transit with keys managed by AWS Key Management Service.
- Daily backups are automatically stored in S3 storage.
- Amazon FSx for Windows File Server is integrated with the AWS Backup service.

## Amazon Simple Storage Service

Amazon Simple Storage Service (S3) was one of the first AWS services offered, launched on March 14, 2006. Amazon S3 provides unlimited storage in a logical container called a *bucket* that can be made available publicly across the Internet or

privately across Amazon's private network, as shown in [Figure 8-16](#). AWS customers that use S3 storage can store any amount of data for many use cases, including data lakes, websites, mobile applications, backups, archived data, and big data analysis. Internally at AWS, almost every data component makes its way to S3 storage, including AMIs, EBS snapshots, and continuous backups from DynamoDB, Redshift, CloudWatch logs, and CloudTrail trails.



**Figure 8-16** Amazon Simple Storage Service

Pricing for Amazon S3 storage is low at the start, but storage at AWS is assessed a monthly charge, plus a data transfer out fee that is applied when you retrieve, replicate, or download files from AWS storage. It's a great advantage to have unlimited storage, but organizations need to make sure they are getting the best bang for their money. When you have content stored at S3, charges are charged every month; fortunately, there are several storage classes designed to minimize S3 storage costs.

Amazon S3 storage can't be used as a boot drive, or as a hard drive location to install software applications. Amazon S3 is primarily designed for strong read after write consistency for both **PUT** and **DELETE** requests of stored objects. S3 provides read-after-write consistency for PUTS of new objects and eventual consistency for PUTS and Deletes. This ensures that when a new object is added to an Amazon S3 bucket, it is immediately available for reading. However, when an S3 object is overwritten or deleted, it can take some time for the updated version to be reflected when reading the object. S3 objects are replicated to multiple locations; eventually, all locations will be updated with the new or changed object. This replication process is defined as eventual consistency. Objects stored in Amazon S3 are also subjected to continuous integrity checks through checksums, from the source upload to the final bucket destination.

---

## Note

In a single AWS region, the Amazon S3 service manages daily access peaks of over 60 TiB/s.

---

## Amazon S3 Bucket Concepts

Following are the key terms and concepts to understand when working with Amazon S3 storage:

- **Buckets:** Buckets are the containers that store S3 objects. You can upload as many objects into an S3 bucket as you want; buckets do not have item or size limits. Use the multipart upload API to upload single objects larger than 5 MIB; once the upload is complete, the file components are combined into a single object.
- **Objects:** Each S3 object consists of the object data (the file itself) and associated metadata that describes the stored object, such as date and content type. Each object is identified within each S3 bucket by a unique key name.
- **Keys:** Each object key name uniquely identifies each object stored in an S3 bucket. For example, the object *cat.jpg*, stored in the Amazon S3 bucket *my-bucket*, stored in the us-west-2 region, can be addressed through the associated URL:

<https://my-bucket.s3.us-west-1.amazonaws.com/photos/mycat.jpg>.

- **Access levels:** You can define public or private access levels to S3 buckets and objects with AWS Identity and Access Management (IAM) users and security policies, bucket policies, and access Control lists (ACLs).
- **Service quotas:** 100 Amazon S3 buckets can be created per AWS account; the default quota value can be increased using the Service Quota utility.
- **Object size limit:** There is no limit to the number of objects stored in a single S3 bucket; however, each individual object size is limited to 5 TiB.
- **AWS region:** Before selecting a particular AWS region for an S3 bucket, consider factors such as compliance, latency, and cost of the S3 service based on bucket/region location. As part of the S3 service-level agreement (SLA) with AWS, objects stored in a specified AWS region never leave the region unless you decide to transfer them to another location.
- **S3 bucket names:** All names for S3 buckets are Domain Name System (DNS) names stored in Amazon Route 53, AWS's global DNS service. S3 bucket names are therefore DNS names and must be globally unique across all AWS customers.

- **S3 object metadata:** Each object in an S3 bucket contains some mandatory components: the object itself and the associated metadata. Optionally, you can add additional custom metadata to each object when stored. Keep in mind that custom metadata is not encrypted. Review the mapping for each object with the bucket/key/version of objects, as shown in [Figure 8-17](#).

| Object overview                        |   |
|--|---|
| Owner                                  | S3 URI  |
| wilkinsolutions                        | <a href="s3://3733538949474494033/Labs for Storage.docx">s3://3733538949474494033/Labs for Storage.docx</a>   |
| AWS Region                             | Amazon Resource Name (ARN)  |
| US East (N. Virginia) us-east-1        | <a href="#">arn:aws:s3:::3733538949474494033/Labs for Storage.docx</a>  |
| Last modified                          | Entity tag (Etag)   |
| October 20, 2020, 10:43:07 (UTC-04:00) | <a href="#">708b60101be27253fa3cfa621378e507</a>  |
| Size                                   | Object URL  |
| 36.5 KB                                | <a href="https://3733538949474494033.s3.amazonaws.com/Labs+for+Storage.docx">https://3733538949474494033.s3.amazonaws.com/Labs+for+Storage.docx</a> |
| Type                                   |   |
| docx                                   |   |
| Key                                    |   |
| <a href="#">Labs for Storage.docx</a>  |   |

**Figure 8-17** S3 Object Details

Many options are available when you create an S3 bucket, as shown in [Table 8-4](#). Understand these options for the AWS Certified Solutions Architect – Associate (SAA-C03) exam.

**Key Topic**

**Table 8-4** S3 Bucket Configuration Options

| S3 Feature       | Details   |
|------------------|---|
| AWS Region       | The region where the Amazon S3 bucket is created.   |
| S3 bucket policy | Bucket policies allow or deny requests to IAM users and groups and can grant cross-account access to other AWS accounts. Only the owner of the bucket can associate a bucket policy with an Amazon S3 bucket. |
| IAM policies     | Create IAM users within your AWS account to grant access to Amazon S3 buckets and objects.  |

---

| S3 Feature             | Details   |
|------------------------|---|
| S3 Block Public Access | Block Public Access settings are enabled at the bucket level. There are three options that can be individually selected: Grant access with IAM identities, Bucket policies, or Buckets in a VPC setting.  |
| S3 Access Points       | Simplify data access for any AWS service or customer application that requires access to shared datasets stored in an S3 bucket. Separate S3 access points can be created and tailored for application accces or groups of IAM users. Access points can be configured to restrict access to a specific virtual private cloud (VPC). |

| S3 Feature           | Details  |
|----------------------|--|
| Access Control Lists | ACLs can be used to grant read and write permissions to authorized users, groups, or AWS accounts for individual buckets and objects. AWS recommends that ACLs be disabled and to use the Bucket Owner Enforced setting for S3 object ownership.   |
| Website hosting      | An S3 bucket can be configured to host a static website that does not require service-side processing.   |
| Logging              | Track access requests to Amazon S3 buckets using Amazon CloudTrail, Amazon S3 Access Logs, AWS CloudWatch S3 metrics, or AWS GuardDuty analysis of S3 data events. Server access logging provides detailed records of requests made to an Amazon S3 bucket, assisting in security and access audits. |

| S3 Feature         | Details  |
|--------------------|--|
| SNS notifications  | Receive notifications when S3 bucket events (such as <b>GET</b> , <b>PUT</b> , and <b>DELETE</b> ) occur.  |
| S3 Versioning      | Store multiple versions of the same object within a single Amazon S3 bucket. Versioning must be enabled for many other Amazon S3 features including Lifecycle policies and Bucket replication options. |
| Lifecycle policies | Create lifecycle rules for Amazon S3 controlling object retention and movement to other Amazon S3 storage classes to help manage storage costs.  |

| S3 Feature                   | Details  |
|------------------------------|--|
| Bucket replication           | Automatically replicate objects to other Amazon S3 buckets hosted in the same region (Same Region Replication) or other Amazon S3 regions (Cross Region Replication) to help with disaster recovery. Replication can also be to another AWS account. |
| Server-Side Encryption (SSE) | As of January 5, 2003, all new object uploads to Amazon S3 will be automatically encrypted with Amazon S3 Managed keys (SSE-S3). Other choices are AWS-Key Management Service Keys (SSE-KMS) or SSE with customer provided keys (SSE-C).             |

| S3 Feature     | Details   |
|----------------|---|
| Object tagging | Add up to ten tags to each Amazon S3 object to assist in controlling access with bucket or IAM policies. Tags can be used in lifecycle and replication policies. To ensure that the latest version of an object is always available for reading, S3 provides a mechanism called object tagging, which allows users to specify a version of an object to be the current version. This can be useful in scenarios where read-after-write consistency is required. |
| Requester pays | Charge the data transfer costs to the end user that requests the object.  |
| Object lock    | Enable write-once/read-many (WORM) policies on objects or buckets to manage compliance requirements. Define a “retain until date” or “legal hold” protection.   |

| S3 Feature                    | Details   |
|-------------------------------|---|
| S3 Transfer acceleration      | Speed up the transfer of larger Amazon S3 objects over long distances from the end user to the S3 bucket using CloudFront global edge locations and the AWS private network for uploads and downloads.  |
| Multi-Region access point     | Define a global endpoint to access data sets stored in Amazon S3 buckets located in multiple AWS regions. Multi-Region access uses the AWS Global Accelerator to select the S3 bucket with the lowest network latency.                        |
| AWS PrivateLink for Amazon S3 | AWS PrivateLink for S3 provides private connectivity between Amazon S3 and on-premises locations using interface VPC endpoints for S3 in your VPC to connect your on-premises applications directly to S3 over AWS Direct Connect or AWS VPN. |

---

### Note

S3 Object Ownership is an S3 bucket-level setting used to disable ACLs and take ownership of all objects in your S3 bucket. S3 Object Ownership allows the owner of an S3 bucket to take control of all bucket objects.

---

## **Amazon S3 Data Consistency**

Objects stored in an Amazon S3 bucket are replicated many times to at least three other separate physical storage locations (availability zones) within the AWS region where your Amazon S3 bucket is located, providing a high level of durability for each stored object. After a new object has been written to an Amazon S3 bucket, read requests retrieve the latest version of the object.

Working with multiple copies of data, replicating updates and deletions takes some time to complete. All S3 objects eventually in all linked storage locations will be the same.

## **Amazon S3 Storage Classes**

When storing objects in Amazon S3 buckets, the available Amazon S3 storage classes have been designed for different use cases, as shown in [Table 8-5](#).

**Key Topic**

**Table 8-5** S3 Storage Classes

| Storage Class    | S3 Standard     | S3 Intelligent-Tiering   | S3                         |
|------------------|-----------------|--|----------------------------|
| Access frequency | No restrictions | Automatically moves objects to the Standard-Infrequent Access tier after 30 days | Infrequent minimum 30 days |
| Cost             | Lowest cost     | Medium cost  | Highest cost               |

|                     |              |                        |                                |
|---------------------|--------------|------------------------|--------------------------------|
|                     |              |                        | S3                             |
| Storage Class       | S3 Standard  | S3 Intelligent-Tiering | Standard-I (Infrequent Access) |
| <b>Access speed</b> | Milliseconds | Milliseconds           | Milliseconds                   |

**Minimum Number of AZs**

|                       |     |     |        |
|-----------------------|-----|-----|--------|
| <b>Retrieval cost</b> | N/A | N/A | Per GB |
|-----------------------|-----|-----|--------|

**Minimum duration (days)**

---

| Storage Class       | S3 Standard | S3 Intelligent-Tiering | S3 Standard-I (Infrequent Access) |
|---------------------|-------------|------------------------|-----------------------------------|
| Availability        | 99.99%      | 99.9%                  | 99.9%                             |
| Minimum object size | N/A         | N/A                    | 128 KB                            |

There are six Amazon S3 storage classes to consider:

- **Amazon S3 Standard:** Designed for data regularly accessed by online cloud-hosted applications. It is designed for high performance and durability, offering eleven 9s durability and four 9s availability. Amazon states, “If you store 10,000 objects in S3, you may lose one object every 10 million years.” Exact pricing depends on the amount you store and the Amazon S3 bucket’s region. The first 50 TiB is charged at \$0.023 per gigabyte per month; at 500 TiB of object storage, the price drops to \$0.021 per gigabyte per month. Note that there are also charges for querying and retrieving objects. Amazon S3 Standard supports SSL for data in transit and data encryption can be enabled for objects at rest.

- **Amazon S3 Intelligent-Tiering:** Rules for intelligent-tiering analyze and move less frequently accessed objects to lower-cost storage classes from Amazon S3 Standard to Amazon S3, Amazon S3-IA, One Zone-IA, or Amazon S3 Glacier (Instant Retrieval, Flexible Retrieval, or Deep Archive). S3 Intelligent-Tiering optimizes your storage costs by monitoring your access patterns at the object level, automating cost savings for your stored objects.

After 30 days of inactivity, your objects are automatically moved from the Frequent Access tier to the Infrequent Access tier on an object-by-object basis. If the infrequently accessed data objects begin to be accessed more frequently, they are moved back to the Frequent Access tier. Intelligent-Tiering might save organizations a great deal of money after deploying one of these use cases:

- Moving all objects older than 90 days to Amazon S3-IA
- Moving all objects after 180 days to Amazon S3 Glacier
- Moving all objects over 365 days to Amazon S3 Glacier Deep Archive
- **Amazon S3 Standard-Infrequent-Access (IA):** Designed for less frequently accessed data while maintaining eleven 9s durability. If you don't need to access your data frequently and object access is greater than 30 days, Amazon S3 Standard-IA is a cheaper option than Amazon S3 Standard.

- **Amazon S3 One Zone-IA:** Designed for less frequently accessed data that could be re-created if necessary, Amazon S3 One Zone-IA provides less durability than Amazon S3 Standard-IA because the data is stored in a single AZ instead of across three or more AZs. The price point of Amazon S3 Standard-IA is 20% less than the price of Amazon S3-IA.
- **Amazon S3 Glacier:** Long-term data archival storage stored in archives and vaults. If you need to access your archived data quickly, you can retrieve it within minutes—but you pay additional fees for the expedited access.
  - Glacier Instant Retrieval storage has a minimum storage requirement of 90 days. Retrievals are carried out in milliseconds and are allowed once a month for objects that might need to be accessed quickly, such as medical images or news media assets.
  - Glacier Flexible Retrieval storage has a minimum storage requirement of 90 days. Retrievals are carried out in minutes of large archived datasets such as backup or disaster recovery records, or using free bulk retrievals, which take 5 to 12 hours.
- **S3 Glacier:** The concept of this archive storage option is to remove the need for on-premises long-term archiving vaults. It is the cheapest archive option at AWS: just \$0.00099 per gigabyte per month with a minimum storage requirement of

180 days. Retrieval times are within 12 hours or less. Organizations can request faster retrieval times, but additional fees apply.

## Amazon S3 Management

S3 buckets have a variety of powerful management controls to help manage your stored data:

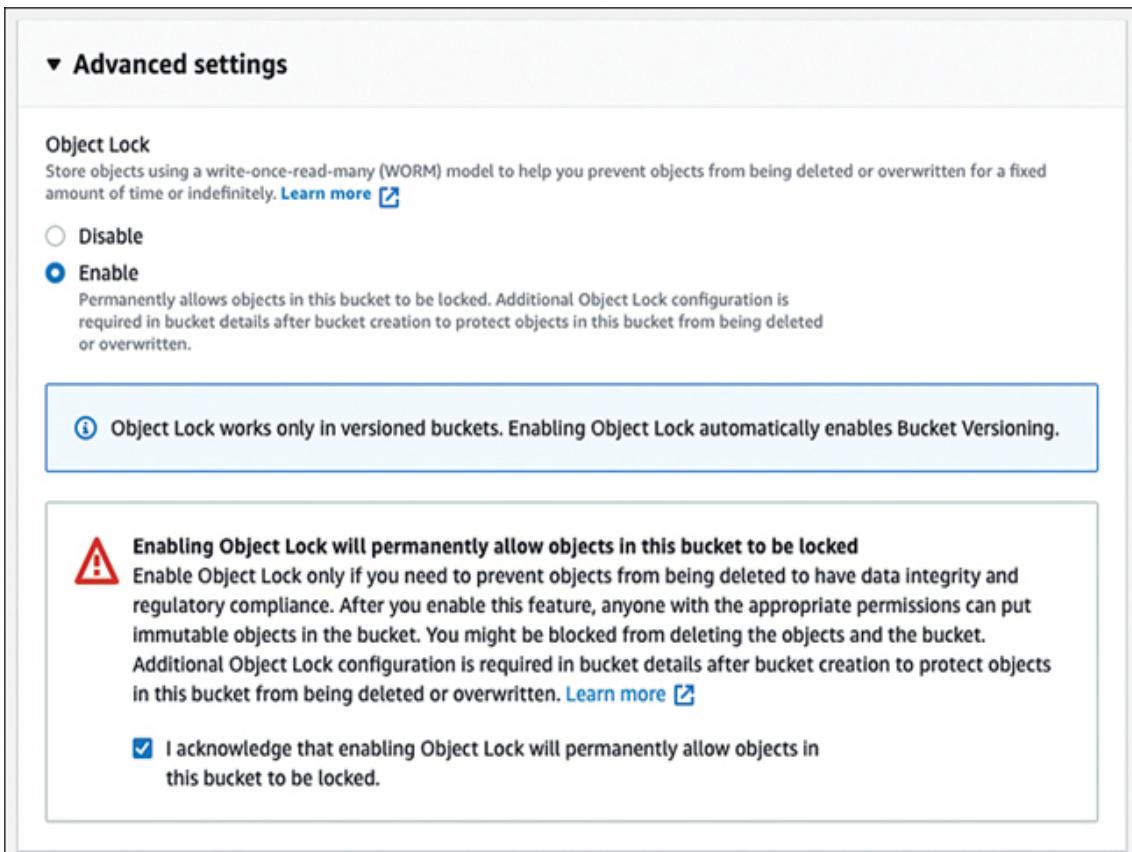


- **S3 Batch Operations:** S3 Batch Operations can be used to perform operations on a large number of objects, making it an efficient way to manage data in S3. It can be particularly useful for scenarios where administrators need to perform the same operation on a large number of objects, such as migrating data between S3 buckets or applying data retention policies. To use S3 Batch Operations, create a job that specifies the operation to be performed, the objects to be affected, and any additional parameters. S3 Batch Operations can be initiated using the AWS Management Console, the AWS CLI, or using an AWS SDK.
- **S3 Object Lock:** Organizations can enforce retention policies based on defined “retain until” dates or legal hold dates. All

storage classes can enable S3 Object Lock settings; however, versioning must also be enabled at the Amazon S3 bucket level before the Object Lock feature can be enforced.

Once Object Lock has been enabled, you can lock objects in that bucket. Object Lock can be deployed at the object level or Amazon S3 bucket level. For example, you could add a 3-year or a 5-year retention policy at the Amazon S3 bucket level; from that point forward, all objects placed in the specific Amazon S3 bucket would inherit the retention policy. S3 bucket level protection is especially for customers that need to adhere to SEC Rule 17-a 4(f), CFTC Regulation 1.3.1, and FINRA Rule 4511. There are two modes of protection available for locking Amazon S3 objects (see [Figure 8-18](#)):

- **Retention period:** Specifies a fixed period of time in which an S3 object remains locked. During this period, the object is WORM-protected and can't be overwritten or deleted.
- **Legal Hold:** Adds S3 Object Lock protection to an Amazon S3 object that remains until it is explicitly removed by an authorized administrator.



**Figure 8-18** Enabling Object Lock on an S3 Bucket

**Key Topic**

- **S3 Replication:** This feature provides replication of objects between buckets within the same AWS region or across different AWS regions. Two-way replication between two or more buckets within the same or different AWS region is also supported. Predictable replication times can be accomplished

by using Replication Time Control (RTC), which replicates objects in less than 15 minutes.

- **Cross-Region Replication (CRR):** This feature can be used from any AWS region to any other region to any Amazon S3 storage class and across AWS accounts. The replication process is carried out by enabling CCR at the source bucket for the entire bucket contents or by a prefix or tag, as shown in [Figure 8-19](#). In addition, lifecycle rules control the replication of objects stored in Amazon S3 Standard to other Amazon S3 storage classes on the destination bucket or, optionally, to Amazon S3 Glacier. CCR replication is encrypted to maintain data security. The cost for using CRR is calculated based on the copy request and the inter-region data transfer charge for each replicated object.

## Create replication rule

### Replication rule configuration

#### Replication rule name

Replication from US to Paris

Up to 255 characters. In order to be able to use CloudWatch metrics to monitor the progress of your replication rule, the replication rule name must only contain English characters.

#### Status

Choose whether the rule will be enabled or disabled when created.

- Enabled
- Disabled

#### Priority

The priority value resolves conflicts that occur when an object is eligible for replication under multiple rules to the same destination. The rule is added to the configuration at the highest priority and the priority can be changed on the replication rules table.

0

### Source bucket

#### Source bucket name

3733538949474494033

#### Source Region

US East (N. Virginia) us-east-1

#### Choose a rule scope

- Limit the scope of this rule using one or more filters
- Apply to all objects in the bucket

**Figure 8-19** Enabling Cross-Region Replication

- **Same-Region Replication (SRR):** This feature allows you to achieve data compliance for storing your data in a separate AWS account within the same region as the original bucket contents. SRR can be used within any AWS region to any Amazon S3 storage class and across AWS accounts. The

replication process enables SRR at the source bucket for the entire bucket contents or based on a prefix or tag. Configure lifecycle rules to replicate objects stored in Amazon S3 Standard to other Amazon S3 storage classes on the destination bucket or to Amazon S3 Glacier. To ensure data security, SRR replication is encrypted.

- **S3 Storage Lens:** This feature allows you to gain visibility into your Amazon S3 storage usage and activity across your entire AWS organization or AWS account. Amazon S3 Storage Lens provides recommendations to improve your cost efficiency and the data security of your stored objects.
- **S3 Inventory:** If you want to find details about your current object inventory in Amazon S3, you can run an inventory process using S3 Inventory, shown in [Figure 8-20](#). After you select the source bucket for analysis, S3 Inventory creates a flat CSV file based on your query criteria and stores the inventory listing in a specified destination Amazon S3 bucket. The inventory listing encompasses current objects and their associated metadata, including the object's key name, version ID, encryption and replication status, retention date, storage class, object hold status, and object size. The inventory list file can be encrypted using S3-managed or KMS-managed keys.

**Additional fields - optional**

Choose the metadata that should be included for each listed object in the report. [Learn more](#) 

**Object**

Size  
 Last modified  
 Multipart upload  
 Replication status  
 Encryption  
 Bucket key status

**Storage class**

Storage class  
 Intelligent-Tiering: Access tier

**Data integrity**

ETag  
 Additional checksums function

**Object Lock**

All Object Lock configurations

- Object Lock: Retention mode
- Object Lock: Retain until date
- Object Lock: Legal hold status

[Cancel](#) [Create](#)

**Figure 8-20** Creating an S3 Inventory Report

- **Storage Class Analytics:** Through machine learning processes, your Amazon S3 storage is categorized into groups of less frequently accessed data based on an analysis of retrievals against the stored objects. Analysis can also be performed based on Amazon S3 buckets or object tags.

- **Object tags:** Up to ten tags can be added to each Amazon S3 object. The following are some of the actions that can be performed based on the assigned S3 object tag:
  - **Searching and filtering:** S3 enables users to search for and filter objects based on their tags.
  - **Lifecycle management:** S3 enables users to define lifecycle policies that automatically transition objects to different storage classes or delete them based on the object's tags.
  - **Access control:** S3 enables users to specify access controls on objects based on the object's tags.
  - **Cost optimization:** S3 enables users to optimize storage costs by transitioning objects to different storage classes using lifecycle policies that are based on the object's tags.
  - **Auditing and compliance:** S3 enables users to use object tags to track and audit data access and usage.

## S3 Bucket Versioning



Versioning can be enabled on each Amazon S3 bucket to further protect your objects from accidental deletion. As a best practice, versioning could be enabled before any objects are stored in an

Amazon S3 bucket to ensure that all objects will be protected from deletion. Enabling versioning guarantees the following:

- Every new **PUT** of an existing object is created as a new object with a new version ID.
- The newest version is defined as the current version, and the previous versions are retained and not overwritten.
- When you request just the S3 key name of an object, you are presented with the current version of the versioned object.

After versioning has been enabled, additional lifecycle management rules for the versioned content can be created, as shown in [Figure 8-21](#). These rules can define a lifecycle expiration policy that dictates the number of versions that you want to maintain. Lifecycle rules help you manage previous versions of objects by transitioning or expiring specific objects after a defined number of days.

**Lifecycle rule configuration**

Lifecycle rule name  
**Rule 1**  
Up to 255 characters

Choose a rule scope  
 Limit the scope of this rule using one or more filters  
 Apply to all objects in the bucket

**⚠️ Apply to all objects in the bucket**  
If you want the rule to apply to specific objects, you must use a filter to identify those objects. Choose "Limit the scope of this rule using one or more filters". [Learn more](#)

I acknowledge that this rule will apply to all objects in the bucket.

**Lifecycle rule actions**  
Choose the actions you want this rule to perform. Per-request fees apply. [Learn more](#) or see [Amazon S3 pricing](#)

Move current versions of objects between storage classes  
 Move noncurrent versions of objects between storage classes  
 Expire current versions of objects  
 Permanently delete noncurrent versions of objects  
 Delete expired object delete markers or incomplete multipart uploads  
These actions are not supported when filtering by object tags or object size.

**Transition current versions of objects between storage classes**  
Choose transitions to move current versions of objects between storage classes based on your use case scenario and performance access requirements. These transitions start from when the objects are created and are consecutively applied. [Learn more](#)

Choose storage class transitions  
Glacier Deep Archive ▾

Days after object creation  
90 ▾ Remove

**Figure 8-21 Lifecycle Rules**

---

**Note**

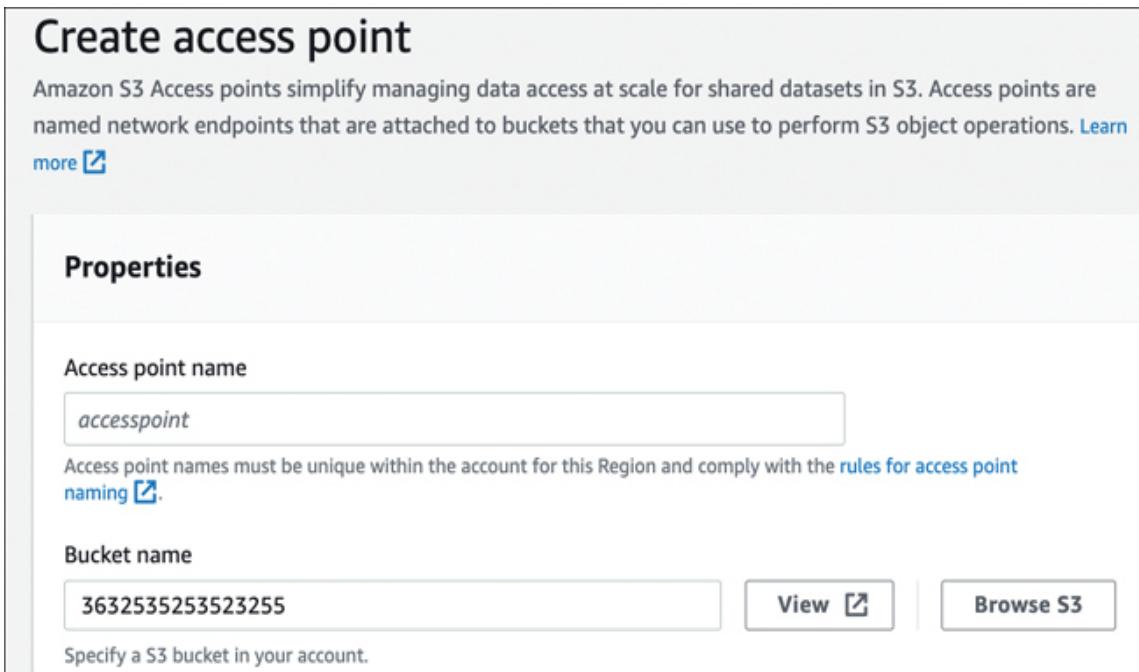
Amazon S3 supports SNS notifications that can alert you at no additional cost when S3 object deletions occur. You can also enable multi-factor authentication (MFA) on S3 buckets that is activated when objects are selected to be deleted.

---

## Amazon S3 Access Points

You can simplify access to objects stored in Amazon S3 buckets from multiple AWS locations by creating Amazon S3 access endpoints that are directly attached to specific Amazon S3 buckets, as shown in [Figure 8-22](#). Each Amazon S3 access point can have different permissions and network controls for requests made through the access point. The following are key features of S3 access points:

- S3 access points can deploy an optional AWS IAM policy.
- Access points block public access by default.
- Access points can have permissions defined for IAM users and groups, and specific applications.
- Access points can be configured to accept requests only from a VPC.



**Figure 8-22** Amazon S3 Access Points Setup

S3 access can also be controlled using a VPC endpoint, which provides secure access using either a gateway or interface connection to an S3 bucket from applications running in a VPC without requiring an Internet Gateway or NAT gateway. Access to an S3 bucket using a VPC gateway endpoint connection is not charged.

## Multi-Region Access Points

Multi-region S3 access points can also be created that provide a global endpoint, allowing applications to request content from S3 buckets located in multiple AWS regions.

When creating a S3 multi-region access point, multiple AWS regions are selected where the replicated S3 buckets will be served through a global endpoint. Application requests use the AWS Global Accelerator service to route requests across the private AWS global network to the S3 bucket with the lowest network latency.

## Preselected URLs for S3 Objects

Amazon S3 allows the generation of preselected URLs for objects stored in the service. These URLs can be shared with others, enabling them to access the S3 object without the need for an AWS account or AWS IAM credentials.

There are several methods to generate preselected URLs for Amazon S3 objects:

- **Query String Authentication:** S3 Query String Authentication enables users to generate a time-limited URL for an object that includes a signature calculated using the user's AWS secret access key. This URL can be shared with others and used to access the object without the need for AWS credentials.
- **Presigned URLs:** S3 Presigned URLs enable users to generate a time-limited URL that can be used to access an object or

perform an operation on the object, such as uploading or downloading it. Presigned URLs are signed using the user's AWS access key and secret access key.

- **Object URLs:** S3 Object URLs can be used to access objects stored in S3 using the S3 endpoint and the object's key. Object URLs can be accessed by anyone with the URL, but they do not include any security or access controls.

## S3 Cheat Sheet



For the AWS Certified Solutions Architect – Associate (SAA-C03) exam, you need to understand the following aspects of Amazon S3 storage:

- An Amazon S3 bucket is a flat container that contains file objects.
- Amazon S3 buckets cannot be nested.
- Amazon S3 buckets are region-specific.
- Amazon S3 bucket names must be unique globally across the AWS cloud, as the S3 namespace is actually a DNS namespace hosted by Amazon Route 53.

- Objects stored in Amazon S3 storage can be addressed through a service endpoint, by bucket name, by object name, and by object version.
- There are five methods for controlling access to Amazon S3 storage: bucket policies, IAM policies, access control lists, presigned URLs, and query-string authentication.
- An Amazon S3 bucket owner can define cross-account permissions for controlling access from another AWS account.
- Membership in the Authenticated Users group allows S3 bucket access from any AWS account. By default, this group includes all AWS accounts.
- Membership in the All Users group allows anyone outside AWS to access the S3 Bucket. Requests can be signed or unsigned.
- **Multipart uploads** should be used for objects larger than 5 GB. Uploads occur synchronously and in parallel.
- Amazon S3 Transfer Acceleration uses edge locations to speed up transfers of files over long distances from the end user's location to the selected S3 bucket.
- An Amazon S3 bucket can be used to host a static website.
- A custom domain name can be used with a Route 53 alias record for an Amazon S3 bucket that is hosting a static website.

- **Versioning** stores all versions of an object and protects against accidental deletion or overwrites.

## Amazon S3 Glacier

Amazon S3 Glacier is an extension of Amazon S3 storage that offers the lowest cost of any AWS storage services. S3 Glacier costs on average \$0.004 per gigabyte per month, and S3 Glacier Deep Archive costs just \$0.00099 per gigabyte per month.

The premise with Amazon S3 Glacier archival storage data is that customers don't require regular access to the content stored in S3 Glacier vaults and archives. Therefore, the minimum storage time is 90 days; accessing the Amazon S3 Glacier content sooner results in a financial penalty. Amazon S3 Glacier storage pricing is based on monthly storage capacity and the total number of lifecycle transition requests for moving data into Amazon S3 Glacier. Objects archived into Amazon S3 Glacier must remain a minimum of 90 days, or additional charges apply.

Amazon S3 Glacier has the same eleven 9s durability as S3 Standard storage. Amazon S3 Glacier data is stored in vaults and **archives**, set up through the Amazon S3 management console or using the AWS CLI. Amazon S3 Glacier automatically

encrypts all stored objects. Content can be delivered to Amazon S3 Glacier by using the following methods:

- Amazon S3 Lifecycle or Intelligent-Tiering rules
- CLI commands
- Direct use of the REST API or AWS SDK
- AWS Storage Gateway: Tape Gateway, which integrates with S3 Glacier
- AWS Snowball, AWS Snowcone, and AWS Snowmobile devices, which can directly migrate data records into S3 storage
- A direct **PUT** stores objects directly in S3 Glacier using the S3 API
- Cross-Region Replication or Single-Region Replication can replicate objects into S3 Glacier storage

## Vaults and Archives

You can store an unlimited number of archives within a single Amazon S3 Glacier vault: the term *archive* refers to the objects (such as documents, photos, or videos) that are stored in each archive. Amazon S3 Glacier archives can be from 1 byte up to 40 TiB; archives up to 4 GiB can be uploaded in a single operation. Archives from 100 MiB up to 40 TiB use the multipart upload API and are synchronously uploaded and stored in an

Amazon S3 Glacier vault (see [Figure 8-23](#)) stored in your chosen AWS region. Each AWS customer can have up to 1,000 vaults per AWS region.

|   |  |
|---|--|
| <p><b>Create Vault</b></p> <p>Step 1: Vault Name</p> <p>Step 2: Event Notifications</p> <p>Step 3: Event Notification Details</p> <p>Step 4: Review</p> | <p><b>Welcome to Amazon S3 Glacier</b></p> <p>Data is stored in S3 Glacier in "archives." An archive can be any data such as a photo, video, or document. You can upload a single file as an archive or aggregate multiple files into a TAR or ZIP file and upload as one archive.</p> <p>A single archive can be as large as 40 terabytes. You can store an unlimited number of archives and an unlimited amount of data in S3 Glacier. Each archive is assigned a unique archive ID at the time of creation, and the content of the archive is immutable, meaning that after an archive is created it cannot be updated.</p> <p>Vaults allow you to organize your archives and set access policies and notification policies. Get started by giving your vault a name. You can then create your vault now or click <b>Next Step</b> to set up your vault's properties.</p> <p>Region: US East (N. Virginia) </p> <p>Vault Name*: King Street Productions </p> |
|---|--|

**Figure 8-23** Creating S3 Glacier Vaults

## S3 Glacier Retrieval Policies



To retrieve objects from S3 Glacier, create an archive retrieval job using the Management Console or the S3 API. The archive retrieval job is a separate temporary copy of your data placed in either Amazon S3 Reduced Redundancy Storage or Amazon Standard-IA storage, leaving the actual archived data in its original location in Amazon S3 Glacier. Temporary archived

data is accessed using an Amazon S3 **GET** request. To retrieve an archive from S3 Glacier, use the AWS CLI to get the ID of the archive to retrieve. Next, initiate a job requesting Amazon S3 to prepare the archive request. Three retrieval options are available:

- **Expedited:** Retrieval is within 1 to 5 minutes for archives less than 250 MiB. Provisioned capacity can be purchased to ensure that expedited retrieval requests under all circumstances will be carried out.
- **Standard:** Retrieve any archive within several hours, typically 3 to 5 hours. This is the default option when retrieval options are not specified.
- **Bulk:** Retrieve any archive within 5 to 12 hours. Bulk retrievals are the lowest-cost S3 Glacier retrieval option.

## S3 Glacier Deep Archive

If archive records are rarely viewed, organizations should consider using Amazon S3 Glacier Deep Archive, for which you are charged about \$1 per terabyte per month for storage. Larger customers can move on-premises magnetic tape libraries into Amazon S3 Glacier Deep Archive and save money. Objects archived to Amazon S3 Glacier Deep Archive have a minimum storage requirement of 180 days. Amazon S3 Glacier

Deep Archive data can be accessed by two retrieval processes. Free standard retrieval delivers your data within 12 hours. Bulk retrieval returns your data within 48 hours. Expedited data retrieval is not supported.

## Amazon S3 Glacier Cheat Sheet



For the AWS Certified Solutions Architect – Associate (SAA-C03) exam, you need to understand the following critical aspects of S3 Glacier:

- Data in Amazon S3 Glacier is resilient even if one availability zone is unavailable.
- Archived objects are visible upon retrieval request through an Amazon S3 temporary storage location and not directly through Amazon S3 Glacier.
- Amazon S3 Glacier automatically encrypts all stored data at rest using AES 256-bit encryption.
- The Amazon S3 **PUT** API allows direct uploads to Amazon S3 Glacier.
- Amazon Glacier Deep Archive is lower cost than Amazon S3 Glacier but has longer retrieval times.

- Once an archive has been uploaded, it cannot be modified.
- Amazon Glacier Instant Retrievals can be expedited within milliseconds.
- Amazon Glacier Flexible Retrieval bulk retrievals are free of charge once a year.
- Amazon Glacier Flexible Retrieval bulk retrieval times are typically between 5 and 12 hours.

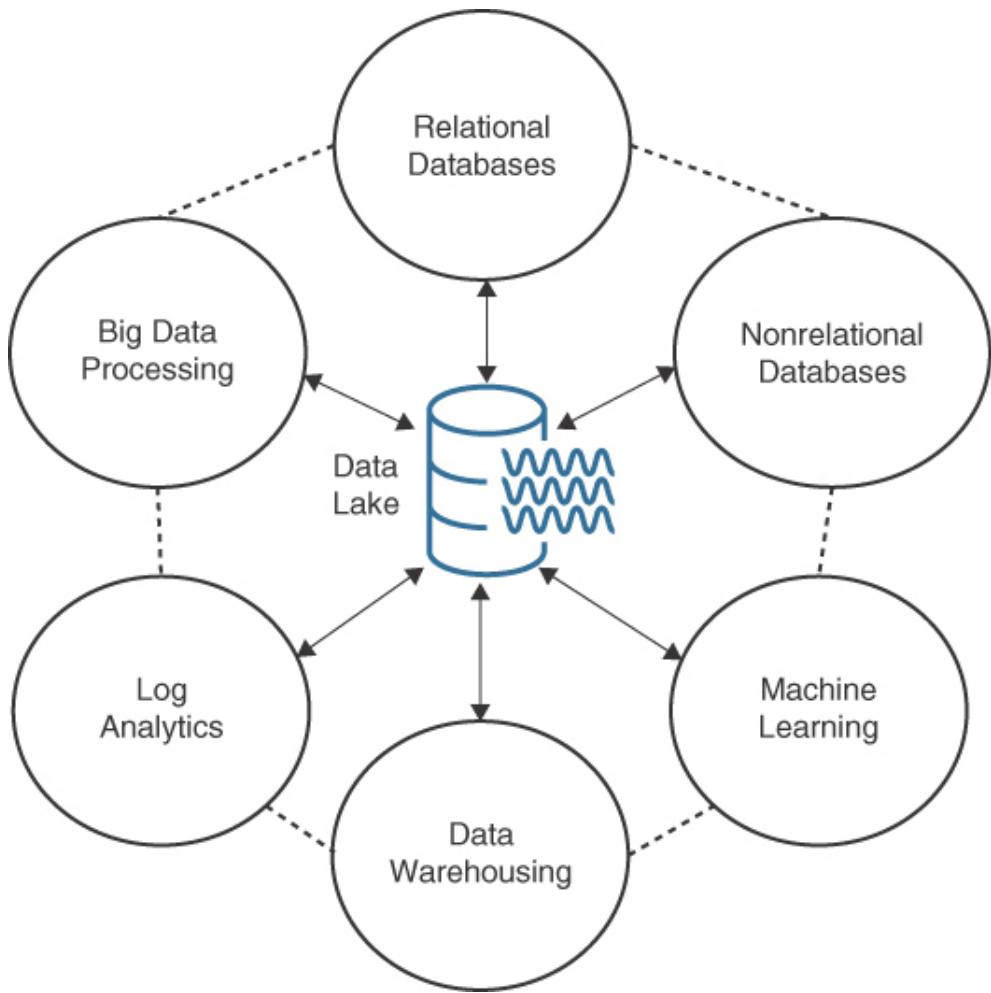
## AWS Data Lake

A *data lake* is a central repository that enables you to store all your structured and unstructured data at any scale. Data records stored in a data lake come from a variety of sources, such as transactional databases, log files, sensors, and social media feeds. One of the key benefits of a data lake is that it enables organizations to store data in its native format, without the need to transform or structure it upfront. A data lake stores both structured and unstructured data records. Many different types of analytics can be performed on stored data, including data visualizations and dashboard views, big data processing, and machine learning. The following are key features of a data lake:

- **Multiple data sources:** Data is collected in real time from multiple sources, including relational databases, business

applications, and nonrelational data types such as IoT devices, social media, and mobile applications (see [Figure 8-24](#)).

- **Catalog:** A centralized catalog presents data lake content built from analyzing, cataloging, and indexing data records in multiple locations.
- **Analysis:** Data scientists, developers, and analysts can access the data lake using a variety of analytic tools and frameworks such as Apache Hadoop, Apache Spark, and Presto.
- **Machine Learning:** Generate insights on historical data using models that can forecast outcomes.
- **Combine customer data:** Retail sales can be compared against incident tickets for additional customer insights.
- **Improve operational efficiencies:** Collect and analyze real-time data from connected IoT devices.



**Figure 8-24** Corporate Data Sources

---

### Note

Data lakes enable customers to run analytics without first having to move data records from their source locations.

---

Before data lakes, many larger customers used data warehouses, which stored data from a business's line of business applications, and the daily transactions between customers (CRM) and human resources (HRM). A data warehouse's data structure and schema were designed for efficient SQL queries that created reports and detailed analysis and provided a single source of truth for the company. A data warehouse can still be a component of a modern data lake, as detailed in [Table 8-6](#).

**Table 8-6** Amazon Redshift Compared to AWS Data Lake

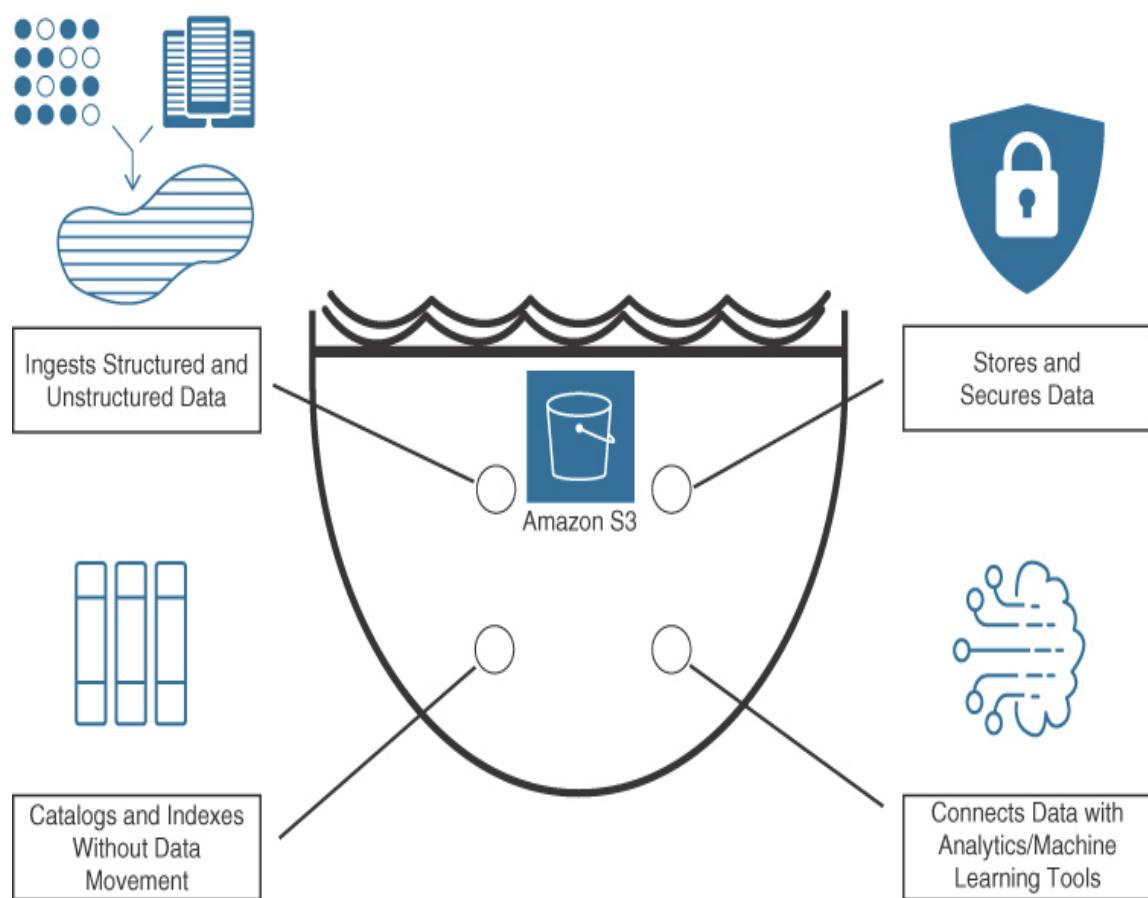
| Characteristics | Data Warehouse<br>(Amazon<br>Redshift) | AWS Data Lake |
|-----------------|--|---------------|
|-----------------|--|---------------|

| Characteristics      | Data Warehouse<br>(Amazon Redshift)  | AWS Data Lake  |
|----------------------|--|--|
| Data types           | Relational data for transactional systems, databases, and LOB applications | Both relational and nonrelational (social media, mobile apps, websites, IoT devices) |
| Schema               | Designed before deployment   | Defined when analyzed  |
| Cost and performance | Fast queries require high IOPS storage                                     | Fast queries use low-cost storage  |

|                         |  |   |
|-------------------------|--|---|
| Characteristics         | Data Warehouse<br>(Amazon Redshift)                              | AWS Data Lake   |
| Quality of data records | Central source of truth  | Curated and noncurated (raw) data                       |
| End users               | Business analysts  | Data scientists, data developers, and business analysts |
| Analytics               | Business intelligence (BI), batch reporting, data visualizations | Machine learning, predictive analysis                   |

## AWS Lake Formation

Today's modern data lake architecture enables customers to integrate an on-premises data lake, a data warehouse, and custom data stores into a unified data lake stored in Amazon S3. A data lake provides secure data access utilizing compliance and governance across an organization. Deploying AWS Lake Formation, data can be connected from external data silos into a data lake for analytics and machine learning analysis (see [Figure 8-25](#)).

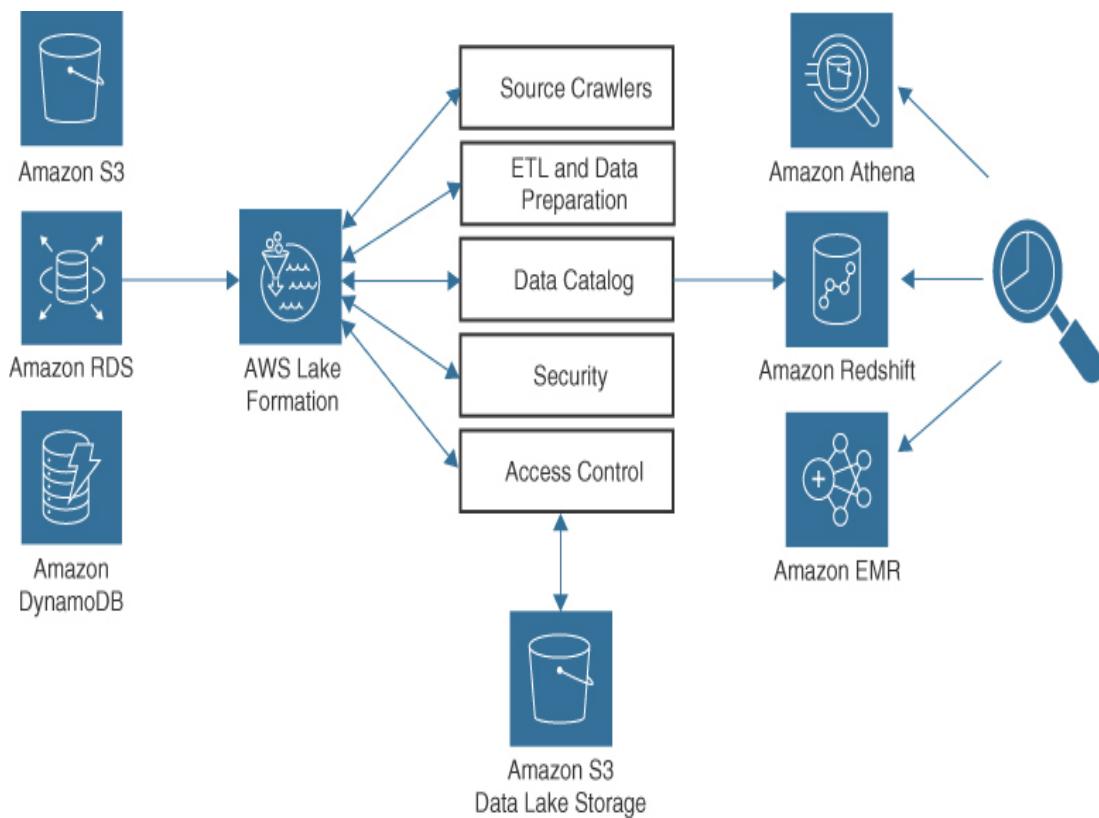


**Figure 8-25** AWS Lake Formation

- AWS Lake Formation automates the creation of a secure centralized data lake that stores all customer data in its original format and is prepared for analysis.
- AWS Lake Formation uses Amazon S3 storage to create the data lake, ensuring unlimited storage and durable storage across multiple availability zones.
- AWS Lake Formation does the work of loading selected data from source locations, turning on encryption and key management, and reorganizing data into a columnar format for analysis by various analytical tools.

An AWS data lake is a central repository for storing a vast amount of raw data in its native format, which can then be used for various analytics and machine learning purposes. Data records that are part of a data lake are stored in both structured and unstructured data formats. AWS offers a range of services and tools for creating and managing data lakes, including AWS Glue, Amazon S3, Amazon Redshift, Amazon Kinesis, and Amazon EMR (see [Figure 8-26](#)). These services can be used to collect, process, and store data from a variety of organizational sources, including transactional systems, IoT devices, and social media platforms. Building a data lake involves the following steps:

1. Identify the organizational sources of data that will be included in the data lake, such as transactional systems, IoT devices, and social media platforms. Data is stored in two formats: structured and semi-structured.



**Figure 8-26** Data Lake Services

2. Extract the data from these sources and transform it into a format that is suitable for storage in the data lake. This may involve cleaning and normalizing the data and applying any necessary transformations to make it consistent and usable.

3. Load the data into the data lake, typically using Amazon S3.
4. Organize the data in the data lake by applying metadata tags and creating a logical structure that makes it easy to find and access the data.
5. Secure the data lake by implementing proper access controls and permissions using AWS IAM policies or service control policies. Policies include setting up authentication and authorization mechanisms as well as encrypting sensitive data. Authentication can be managed using Amazon Cognito. It is also important to monitor the data lake for any security threats or anomalies and to implement appropriate safeguards for protection.
6. Monitor the data lake for any security threats or anomalies, and implement appropriate safeguards to protect against them.
7. Use the data in the data lake for various analytics and machine learning purposes, using built-in AWS services or by integrating with third-party tools and services.

## **Structured and Unstructured Data**

Structured data is organized in a specific format and is easy to search, process, and analyze. It is typically organized in a way that enables both humans and machines to easily interpret it. Examples of structured data storage AWS include databases,

tables, and spreadsheets. Unstructured or semi-structured data is data that doesn't fit neatly into a predefined data model or doesn't follow a fixed format, such as data stored in text files or images. AWS provides a number of cloud services that can be used to store, process, and analyze unstructured data:

- **Amazon S3:** An object storage service that can be used to store unstructured data of any size, at any scale.
- **Amazon Textract:** A machine learning service used to extract text and data from scanned documents and images.
- **Amazon Comprehend:** A natural language processing (NLP) service used to extract insights from unstructured text data.
- **Amazon Rekognition:** A machine learning service used to analyze images and videos and extract insights.

Structured data storage services can be used individually or together to store, process, and analyze unstructured data. For example, a data scientist could use Amazon S3 to store images, Amazon Textract to extract text and data from the images, and Amazon Comprehend to extract insights from the extracted text data. AWS provides a number of services that can be used to store, process, and analyze structured data:

- **Amazon RDS:** Supports a variety of database engines, including MySQL, Microsoft SQL, PostgreSQL, and Oracle.

- **Amazon Redshift:** A data warehousing service that analyzes structured data using SQL and business intelligence tools.
- **Amazon DynamoDB:** A NoSQL database service that provides fast and predictable performance with seamless scalability.
- **Amazon EMR:** A big data processing service that supports a range of big data frameworks, such as Apache Spark and Hadoop.

Unstructured data storage services can be used individually or together to store, process, and analyze structured data. For example, you could use Amazon RDS to store structured data in a relational database, Amazon Redshift to analyze the data using SQL and business intelligence tools, and Amazon EMR to process the data using a big data framework.

## Analytical Tools and Datasets

After the data lake has been created, administrators can set up access to existing datasets and analytical tools. End users access the centralized data catalog that lists the available datasets. Each dataset can then be used with a choice of analytics and machine learning services, listed in [Table 8-7](#).

**Table 8-7** Tools for Analytics, Data Movement, and AWS Data Lake

| Category  | Use Case              | AWS Service                      |
|-----------|-----------------------|----------------------------------|
| Analytics | Interactive analytics | Amazon Athena,<br>Amazon Kinesis |
|           | Big data              | Amazon EMR                       |
|           | Data warehouse        | Amazon Redshift                  |
|           | Operational analysis  | Amazon OpenSearch Services       |
|           | Data visualizations   | Amazon Quicksight                |
|           | Data preparation      | AWS Glue<br>DataBrew             |

| Category      | Use Case                | AWS Service  |
|---------------|-------------------------|--|
| Data movement | Real-time data movement | AWS Glue<br>Amazon Managed Streaming for Apache Kafka (MSK)<br>Amazon Kinesis Data Streams<br>Amazon Kinesis Data Firehose<br>Amazon Kinesis Video Streams<br>Amazon Kinesis |
| AWS data lake | Object storage          | Amazon S3<br>Amazon S3   |
|               | Backup and archive      | Glacier, AWS Backup  |
|               | Data catalog            | AWS Glue, AWS  |
|               | Third-party data        | Lake Formation<br>AWS Data Exchange  |

| Category                                 | Use Case              | AWS Service                             |
|--|-----------------------|---|
| Predictive analysis and machine learning | Frameworks ML Service | AWS Deep Learning AMIs Amazon SageMaker |

## AWS Glue

Amazon Glue is a fully managed extract, transform, and load (ETL) service that automatically discovers and profiles data, creates and updates metadata, and enables users to create and orchestrate ETL jobs using the AWS Glue console. AWS Glue also provides several useful features, such as data classification, data discovery, and data lineage, to help customers understand, cleanse, and transform their data. AWS Glue can also be used to move data between data stores, transform data, and process data for analysis. It is often used with Amazon Redshift, Amazon S3, and Amazon EMR, as well as other AWS data storage and analytics services. AWS Glue consists of several key components:

- **ETL jobs:** Python or Scala code that defines the ETL logic for extracting data from sources, transforming the data, and

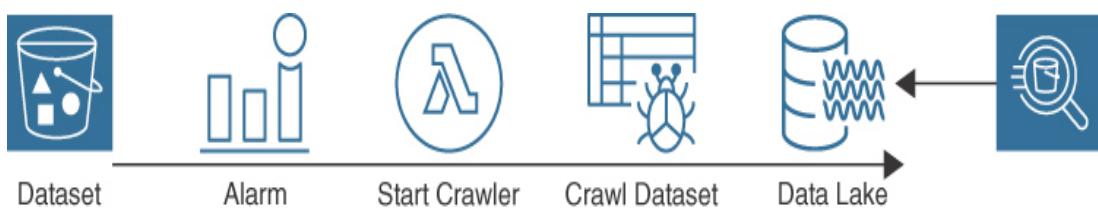
loading it into targets.

- **ETL libraries:** Customizable Apache Spark libraries for common ETL tasks such as reading and writing data, data type conversion, and data processing.
- **Data catalog:** A central repository for storing metadata about data sources and targets, as well as ETL jobs and development endpoints. The data catalog is used to access data sources and targets using AWS Glue ETL jobs and development endpoints.
- **Development endpoints:** Apache Spark environments used to develop, test, and run ETL jobs.
- **Triggers:** Start ETL jobs on a schedule, in response to an event, or on-demand.
- **Crawlers:** Discover data stored in data stores and populate the AWS Glue data catalog with metadata about the discovered data.
- **AWS Glue Studio:** A visual interface for creating, debugging, and managing ETL jobs and development endpoints.
- **Glue schema registry:** Validate and control streaming data using registered schemas for Apache Avro and JSON.
- **Glue DataBrew:** Clean and normalize data without having to write code.
- **Glue elastic views:** Use SQL queries to combine and replicate data to S3 buckets or Amazon Redshift.

The process of using AWS Glue to create and execute ETL jobs involves the following steps:

- 1. Define source and target data stores:** Identify the data stores that contain the source data and the data stores that will receive the transformed data. Data sources include data stored in databases, data lakes, or file systems at AWS or on premises.
- 2. Connect to the data stores:** Create connections in the AWS Glue data catalog to the source and target data stores, enabling AWS Glue to access the data and metadata stored in the data stores.
- 3. Discover data and schema:** AWS Glue discovers the data and its schema by crawling through the data stored in the data stores and extracting metadata, for example data types and field names.
- 4. Create an ETL job:** Use AWS Glue to create an ETL job. Each job is a defined set of ETL tasks that are executed on a schedule or on demand.
- 5. Define the ETL process:** Define the ETL process using Python or Scala code using a development endpoint to extract data from the source data stores, transform the data as required, and load the transformed data into the target data stores.

**6. Execute the ETL job:** The ETL job can now be executed either on demand or on a schedule (see [Figure 8-27](#)). The ETL job will extract, transform, and load the transformed data into the target data stores.



**Figure 8-27** AWS Glue ETL Job Flow

**7. Monitor and maintain the ETL process:** While the ETL job is being executed use the AWS Glue console to monitor the job's progress, view the job's logs, and view the metadata for the transformed data. You can also edit, update, and maintain each ETL job as needed.

## Analytic Services

Several analytic services analyze structured and unstructured data stored in a variety of storage locations at AWS and when working with AWS Lake Formation:

- **Amazon Athena:** A serverless query service that analyzes Amazon S3 data. Amazon Athena enables users to query data

stored in Amazon S3 using SQL, without the need to set up any infrastructure or manage any databases. Queries can be performed in parallel in a variety of standards, including CSV, JSON, ORC, Avro, and Parquet, resulting in extremely high performance. Amazon Athena integrates with other AWS services such as Amazon Glue, providing a seamless way to analyze data from a variety of sources. Amazon Athena is easy to use, with a simple web-based interface and the capability to run ad-hoc queries or save and reuse commonly used queries.

- **Amazon OpenSearch Service:** Performs log analysis and real-time application monitoring, providing visibility into your workload performance. Find relevant data within applications, websites, and data lakes using SQL query syntax. Data can be read using CSV tables or JSON documents.
- **Amazon QuickSight:** Enables users to create dashboards and interactive reports using data from a variety of sources, including relational databases, data warehouses, and data lakes. Amazon QuickSight offers a range of visualization tools and features, including the ability to create custom charts and graphs and share dashboards and reports with others. Amazon QuickSight integrates with other AWS services, such as Amazon Redshift and Amazon Athena, making it easy to

analyze and visualize data stored in those services and providing data visualizations and insights from an organization's data sources. Data records can be stored at AWS or stored in external locations, including on-premises SQL Server, MySQL, and PostgreSQL databases, or in Amazon Redshift, Amazon RDS, Amazon Aurora, and Amazon S3 storage.

- **Amazon Elastic Map Reduce (EMR):** A big data platform for data processing and analysis and machine learning using Apache Spark, Apache Hive, and Presto. Runs petabyte-scale analysis much cheaper than traditional on-premises solutions. EMR integrates with other AWS services, such as Amazon S3 and Amazon Kinesis, enabling users to easily move and analyze data stored in those services.
- **Amazon Managed Streaming for Apache Kafka (Amazon MSK):** Streaming data can be consumed using a full-managed Apache Kafka and Kafka Connect Clusters hosted at AWS, allowing Kafka applications and Kafka connectors to run at AWS without requiring expert knowledge in operating Apache Kafka. Amazon MSK can be used to build a real-time data pipeline that streams data between applications and data stores, or for building a real-time streaming application that can process and analyze data in real time. Amazon MSK used with other AWS services, such as Amazon S3, Amazon

Redshift, and Amazon EMR, can build complete data architectures on AWS. Amazon MSK is built using Apache Kafka; an open-source distributed event streaming platform used for building real-time data pipelines and streaming applications.

- **AWS Data Exchange:** The secure exchange of third-party data files and data tables into AWS. Customers can use the AWS Data Exchange API to copy selected third-party data from AWS Data Exchange into S3 storage. Data Exchange third-party products include weather, healthcare, data sciences, geospatial, and mapping services.
- **AWS Data Pipeline:** Process and move data between different AWS compute and storage services and from on-premises siloed data sources, and transfer the results into S3 buckets, Amazon RDS, DynamoDB, and Amazon Elastic Map Reduce.
- **AWS DataSync:** A data transfer service for moving large amounts of data between on-premises storage and Amazon S3, Amazon EFS, or Amazon FSx for Windows File Server. AWS DataSync can transfer data between any NFS, SMB, or Amazon S3-compatible storage, so you can use it to move data between a variety of different storage solutions.
- **Amazon Storage Gateway:** Connects an on-premises software appliance with cloud-based storage providing

secure integration between an organization's on-premises IT environment and the AWS S3 storage. The Storage Gateway appliance is available in three different deployments:

- File Gateway enables users to store files in Amazon S3, with local caching for low-latency access.
- Volume Gateway provides block-level storage volumes that can be used as iSCSI targets, with local caching for low-latency access.
- Tape Gateway emulates a tape library providing virtual tape infrastructure (VTI) that can be used with existing backup applications.

The Storage Gateway appliance is a virtual machine installed on premises that connects to the cloud via secure, encrypted channels using the Amazon S3 API for communication and supports data transfer rates of up to 100 MiBps.

## **Amazon Kinesis Data Streams**

Amazon Kinesis is a fully managed, cloud-based service offered by AWS for real-time data processing and analysis. Kinesis makes it easy to collect, process, and analyze streaming data, providing the ability to build custom applications that can process and analyze large amounts of data in real time. Kinesis is highly scalable and can handle data streams of any size. It

offers several different types of data streams, including the Amazon Kinesis Data Streams, Amazon Kinesis Data Firehose, Amazon Kinesis Video Streams, and Amazon Kinesis Data Analytics:

- Amazon Kinesis Data Streams enables users to collect, process, and analyze streaming data, providing the ability to build custom applications that can process and analyze large amounts of data in real time. Amazon Data Streams offers a flexible, scalable, and durable platform for streaming data, with the ability to process hundreds of thousands of data streams per second. Amazon Data Streams also integrates with other AWS services, such as Amazon S3, Amazon Redshift, and Amazon EMR, allowing users to easily move and analyze data stored in those services.
- Amazon Kinesis Data Firehose captures, transforms, and loads streaming data into services such as Amazon S3, Amazon Redshift, and Amazon Elasticsearch Service, without the need to write any custom code. It enables users to set up a data stream and specify a destination for the data, such as an S3 bucket or a Redshift cluster. Amazon Data Firehose will then automatically deliver the data to the specified destination, applying any necessary transformations or enrichments along the way.

- The Amazon Kinesis Video Streams SDK enables connected camera devices, such as phones, drones, and dashcams, to securely stream video to custom real-time or batch-oriented applications running on AWS EC2 instances. The video streams can also be stored and encrypted for further monitoring and analytics.
- Amazon Kinesis Data Analytics is a service offered by AWS for real-time analysis of streaming data. Amazon Data Analytics enables users to run SQL queries on streaming data, providing the ability to build applications that can analyze and process data in real time. Amazon Data Analytics also integrates with other AWS services, such as Amazon S3, Amazon Redshift, and Amazon EMR, allowing users to easily move and analyze data stored in those services.

## Exam Preparation Tasks

As mentioned in the section “[How to Use This Book](#)” in the Introduction, you have a couple of choices for exam preparation: the exercises here, [Chapter 16](#), “[Final Preparation](#),” and the exam simulation questions in the Pearson Test Prep Software Online.

## Review All Key Topics

Review the most important topics in the chapter, noted with the Key Topic icon in the margin of the page. [Table 8-8](#) lists these key topics and the page number on which each is found.



**Table 8-8** [Chapter 8](#) Key Topics

| Key Topic Element          | Description                                     | Page Number |
|----------------------------|---|-------------|
| <a href="#">Figure 8-1</a> | AWS Storage Options                             | 362         |
| <a href="#">Table 8-2</a>  | Storage Requirements                            | 363         |
| List                       | EBS volume choices                              | 367         |
| <a href="#">Table 8-3</a>  | Performance Specifications for EBS Volume Types | 369         |
| <a href="#">Figure 8-5</a> | EBS Burst Credit Architecture                   | 370         |
| Section                    | Amazon EBS Cheat Sheet                          | 372         |

| Key Topic Element | Description                     | Page Number |
|-------------------|---------------------------------|-------------|
| Section           | Fast Snapshot Restore           | 374         |
| Section           | Snapshot Administration         | 375         |
| Section           | Snapshot Cheat Sheet            | 376         |
| Section           | EFS Performance Modes           | 380         |
| Section           | EFS Throughput Modes            | 381         |
| Section           | Amazon EFS Cheat Sheet          | 383         |
| Section           | AWS DataSync                    | 384         |
| Section           | Amazon FSx Cheat Sheet          | 388         |
| <u>Table 8-4</u>  | S3 Bucket Configuration Options | 392         |
| <u>Table 8-5</u>  | S3 Storage Classes              | 394         |

| Key Topic Element | Description                   | Page Number |
|-------------------|-------------------------------|-------------|
| Paragraph         | S3 object lock                | 396         |
| Paragraph         | S3 replication                | 397         |
| Section           | S3 Bucket Versioning          | 400         |
| Section           | S3 Cheat Sheet                | 403         |
| Section           | S3 Glacier Retrieval Policies | 405         |
| Section           | Amazon S3 Glacier Cheat Sheet | 406         |

## Define Key Terms

Define the following key terms from this chapter and check your answers in the glossary:

object storage

snapshot

block storage

ephemeral storage

Server Message Block (SMB)

input/output operations per second (IOPS)

Nitro

Throughput Optimized

burst credits

cold storage

mount point

lifecycle policy

bucket

multipart upload

versioning

archive

## **Q&A**

The answers to these questions appear in [Appendix A](#). For more practice with exam format questions, use the Pearson Test Prep Software Online.

- 1.** What is the advantage of EFS shared storage over EBS volumes?
- 2.** What is the difference between files stored in object storage and files stored using block storage?
- 3.** What is the difference between EBS io2 and io1 volumes regarding attaching the volumes to instances?
- 4.** What happens when an EBS snapshot is deleted in the background?
- 5.** What is the fastest storage that can be ordered at AWS?
- 6.** To use most management features of Amazon S3 storage, what must first be enabled?
- 7.** How can S3 objects be shared with clients that do not have AWS credentials?

**8.** After a WORM policy has been applied to an Amazon S3 bucket in Compliance mode, when can the policy be removed?

# Chapter 9

## Designing High-Performing and Elastic Compute Solutions

This chapter covers the following topics:

- [AWS Compute Services](#)
- [AWS Lambda](#)
- [Amazon Container Services](#)
- [Monitoring with AWS CloudWatch](#)
- [Auto Scaling Options at AWS](#)

This chapter covers content that's important to the following exam domain and task statement:

### **Domain 3: Design High-Performing Architectures**

Task Statement 2: Design high-performing and elastic compute solutions

Designing and deploying high-performing computing resources for workloads involves Amazon Elastic Compute Cloud (EC2) instances and understanding which instance family and type to select. Web and application servers, databases, and containerized applications are hosted on EC2 instances. The

compute service AWS Lambda is also covered in this chapter. AWS EC2 instances also host container deployments at AWS, including Amazon Elastic Container Service (ECS) and the Amazon Elastic Container Service for Kubernetes (EKS).

For the AWS Certified Solutions Architect – Associate (SAA-C03) exam, you also need to understand monitoring using AWS CloudWatch, the built-in monitoring service. AWS CloudWatch is a key component of AWS autoscaling services. Autoscaling compute resources at AWS is carried out using the EC2 Auto Scaling service, managing a workload's autoscaling resources using AWS Auto Scaling, which are both covered in this chapter.

## “Do I Know This Already?”

The “Do I Know This Already?” quiz allows you to assess whether you should read this entire chapter thoroughly or jump to the “Exam Preparation Tasks” section. If you are in doubt about your answers to these questions or your own assessment of your knowledge of the topics, read the entire chapter. [Table 9-1](#) lists the major headings in this chapter and their corresponding “Do I Know This Already?” quiz questions. You can find the answers in [Appendix A, “Answers to the ‘Do I Know This Already?’ Quizzes and Q&A Sections.”](#)

**Table 9-1** “Do I Know This Already?” Section-to-Question Mapping

## Foundation Topics Section

## Questions

AWS Compute Services 1, 2

AWS Lambda 3, 4

Amazon Container Services 5, 6

Monitoring with AWS CloudWatch 7, 8

Auto Scaling Options at AWS 9, 10

---

### Caution

The goal of self-assessment is to gauge your mastery of the topics in this chapter. If you do not know the answer to a question or are only partially sure of the answer, you should mark that question as wrong for purposes of the self-assessment.

Giving yourself credit for an answer you correctly guess skews your self-assessment results and might provide you with a false sense of security.

---

**1.** What compute service is designed for processing high volumes of batch jobs?

1. Amazon Elastic Container Service
2. Amazon EC2
3. AWS Batch
4. AWS Lambda

**2.** When an Amazon EC2 instance is ordered, what additional software component must be chosen?

1. Security group
2. AMI
3. Enhanced Network Adapter
4. Subnet

**3.** What type of service is AWS Lambda?

1. Database service
2. Storage service
3. Management service
4. Event-driven service

**4.** What is the main component of AWS Lambda?

1. Automation

2. Functions

3. Monitoring

4. Compliance

**5.** What AWW service supports Docker deployments?

1. AWS EC2

2. AWS ECS

3. AWS S3

4. AWS EKS

**6.** What is required to launch a container at AWS?

1. Container registry

2. Task definition

3. Launch template

4. Fargate

**7.** Which of the following does AWS CloudWatch use to monitor AWS resources for EC2 Auto Scaling?

1. Alarms

2. Alerts

3. Metrics

4. Rules

**8.** Where does AWS CloudWatch store monitoring records?

1. Amazon S3 buckets
2. Log group
3. Trails
4. Amazon DynamoDB table

**9.** How does EC2 Auto Scaling determine what type of EC2 instance to add to an Auto Scaling group?

1. AWS CloudWatch
2. Launch template
3. Registration
4. Deregistration

**10.** How does EC2 Auto Scaling maintain a defined level of compute capacity automatically?

1. Scheduled scaling
2. Target tracking
3. Simple scaling
4. Step scaling

## Foundation Topics

### AWS Compute Services

Planning and running well-architected applications at AWS requires designing, deploying, and managing workload compute resources. Tasks include

- **Provisioning compute infrastructure:** Compute choices include EC2 instances hosting virtual servers or containerized applications running in a virtual private cloud (VPC); database services such as the Amazon Relational Database Service or Amazon DynamoDB.
- **Configuring infrastructure:** Security, high availability, and performance efficiency (compute, database, network).
- **Automated management:** AWS Fargate for Docker or EKS container management, AWS Lambda custom functions for executing event-driven responses, or AWS Elastic Beanstalk for creating and maintaining application infrastructure for virtual server or containerized applications.
- **Scaling:** Auto-scaling services for EC2 instances (EC2 Auto Scaling) or workload auto-scaling (AWS Auto Scaling).
- **Monitoring:** Visibility into all cloud services that make up the application architecture tracking resource usage, deployment success and failure, workload health and automated responses using AWS CloudWatch ***metrics*** and alarms, Amazon EventBridge rules and alerts, Amazon Simple Notification Service (SNS) notifications, and AWS

Lambda functions that carry out tasks in response to metric alarms or alerts.

[Table 9-2](#) details the available AWS compute services and use cases.



**Table 9-2** Compute Services and Use Cases

| Compute Type        | Compute Service   | Use Case   |
|---------------------|---|--|
| EC2 instances (VMs) | Amazon EC2— resizable compute capacity with Intel, AMD, and Graviton processors | Web and application hosting, data processing, high-performance computing |

| Compute Type     | Compute Service  | Use Case   |
|------------------|--|--|
| Batch processing | AWS Batch—<br>Dynamically run massive batch computing workloads based on the number and resource requirements of the batch job | Data processing, machine learning training, and financial modeling     |
| Scaling          | Amazon EC2 Auto Scaling—<br>add and remove compute capacity to match application demand  | Scale EC2 instances and containers with dynamic and predictive scaling |

| Compute Type | Compute Service                                   | Use Case   |
|--------------|---|--|
| Containers   | Amazon Elastic Container Service<br>(Amazon ECS)  | Host Docker containers with AWS ECS                                      |
|              | Amazon Elastic Container Registry<br>(Amazon ECR) | Store, manage, and deploy container images                               |
|              | Amazon Elastic Kubernetes Service<br>(Amazon EKS) | Fully managed Kubernetes service   |
|              | Amazon EKS Anywhere                               | Run Kubernetes containers on customer-managed on-premises infrastructure |

Compute  
Type

Compute  
Service

Use Case

AWS Fargate

Serverless  
compute  
management for  
Docker and  
Kubernetes  
container  
deployments at  
AWS

Amazon ECS  
Anywhere

Run Dockers  
containers on  
customer-managed  
on-premises  
infrastructure

| Compute Type   | Compute Service   | Use Case |
|----------------|---|----------|
| AWS App Runner | Configure applications and deploy to AWS using a Git repository or a Docker image, without having to set up the underlying infrastructure |          |

|            |            |   |
|------------|------------|---|
| Serverless | AWS Lambda | Run custom functions for event-driven solutions without having to provision or manage servers |
|------------|------------|---|

| Compute Type    | Compute Service | Use Case  |
|-----------------|-----------------|---|
| Edge and hybrid | AWS Outposts    | Bring native AWS services and infrastructure to your on-premises environments |
| AWS Local Zones |                 | Run latency-sensitive workloads closer to on-premises data centers            |

| Compute Type | Compute Service      | Use Case   |
|--------------|----------------------|--|
|              | AWS Wavelength Zones | AWS cloud services and infrastructure hosted in third-party telecommunication data centers for developing and hosting 5G workloads |

|                     |                       |  |
|---------------------|-----------------------|--|
| Capacity management | AWS Compute Optimizer | Recommend compute resources to improve workload performance and reduce costs |
|---------------------|-----------------------|--|

| Compute Type | Compute Service              | Use Case  |
|--------------|------------------------------|---|
|              | AWS Elastic Beanstalk        | Deploy web applications running on instances and containers on AWS infrastructure services      |
|              | Elastic Load Balancing (ELB) | Distribute requests to targeted EC2 instances and containers across multiple availability zones |
|              | Amazon EC2 Auto Scaling      | Scale workloads (web servers and databases) using dynamic and predictive scaling                |

## AWS EC2 Instances

Amazon Elastic Compute Cloud EC2 instances are virtual servers that let you run applications in the AWS cloud. Each EC2 instance is a virtual server running Windows, Linux, or the macOS as a guest operating system hosted by the Xen or Nitro hypervisor. All new EC2 instances offered by AWS since 2017 are hosted on the Nitro hypervisor, called the Nitro System.

---

### Note

Additional details on EC2 instance types can be found in [Chapter 13, “Designing Cost-Effective Compute Solutions.”](#)

---

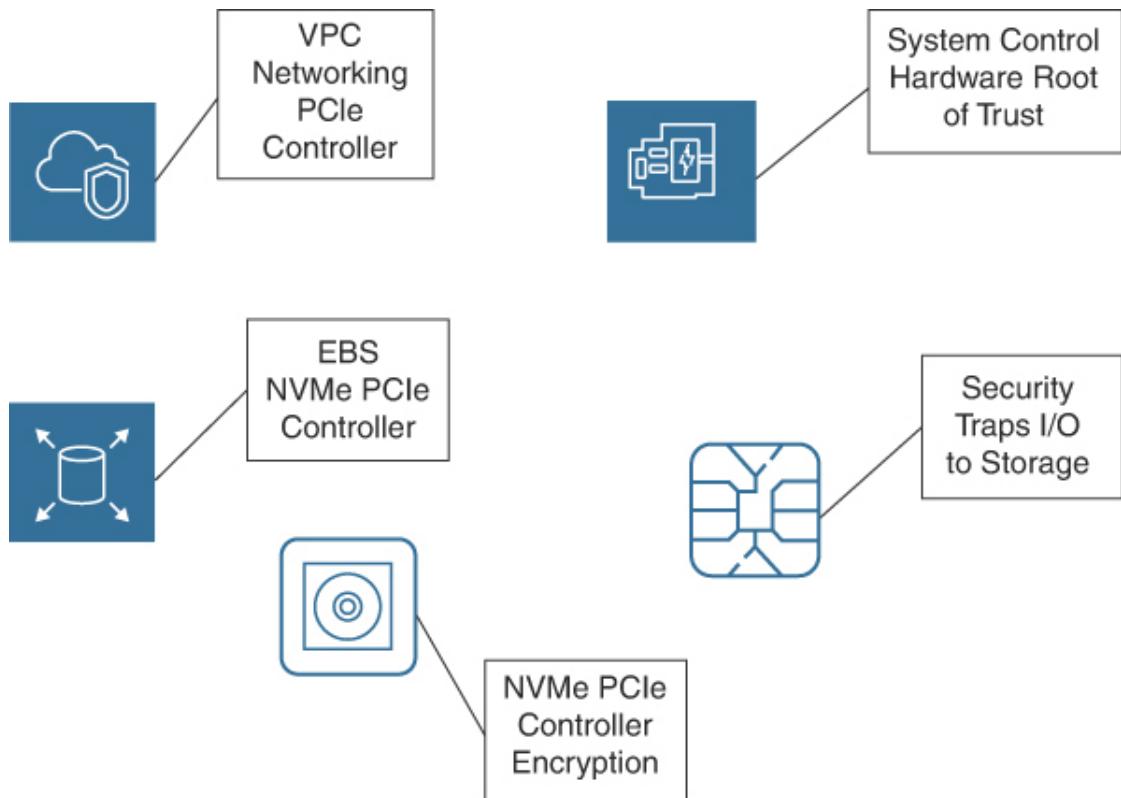
Each EC2 instance is configured at launch with an allotted amount of RAM and virtual CPU (vCPU) cores, storage, and networking bandwidth. The Nitro System is designed to provide high performance, security, and reliability for EC2 instances using internal Nitro services.

The Nitro System (see [Figure 9-1](#)) consists of several key components:

- **Nitro Hypervisor:** A lightweight hypervisor designed to provide efficient resource isolation and high performance for

EC2 and ECS instances at bare-metal speeds.

- **Nitro Security Chip:** Minimizes the attack surface for EC2 and ECS instances, protecting against physical attacks, and prohibits all administrative access, including AWS employees.
- **Nitro Controller:** Several what are called nitro cards offload I/O operations for increased performance, including the Nitro Card for VPC, Nitro Card for EBS, Nitro Card for Instance Storage, and the Nitro Security Chip.
- **Nitro TPM:** Provides cryptographic proof of EC2 instance integrity. The Trusted Platform Module (TPM) 2.0 helps migrate existing on-premises TPM workloads to EC2.
- **AWS Nitro Enclaves:** Enables organizations to create isolated and secure compute environments deploying EC2 instances deployed with hardware-based isolation and memory protection to securely process highly sensitive workloads such as financial transactions, healthcare data, and government applications.



**Figure 9-1** The Nitro System Architecture and Components

The following components are part of each EC2 configuration:

- **Amazon Machine Images (AMIs)**
- Authentication using a unique public/private key pair
- Amazon Elastic Block Store (EBS) and/or temporary storage volumes
- A mandatory firewall called a *security group* that protects each basic or elastic network interface
- Basic or elastic network interfaces (ENI)

- Multi-tenant, single-tenant, dedicated, or bare-metal instance deployment

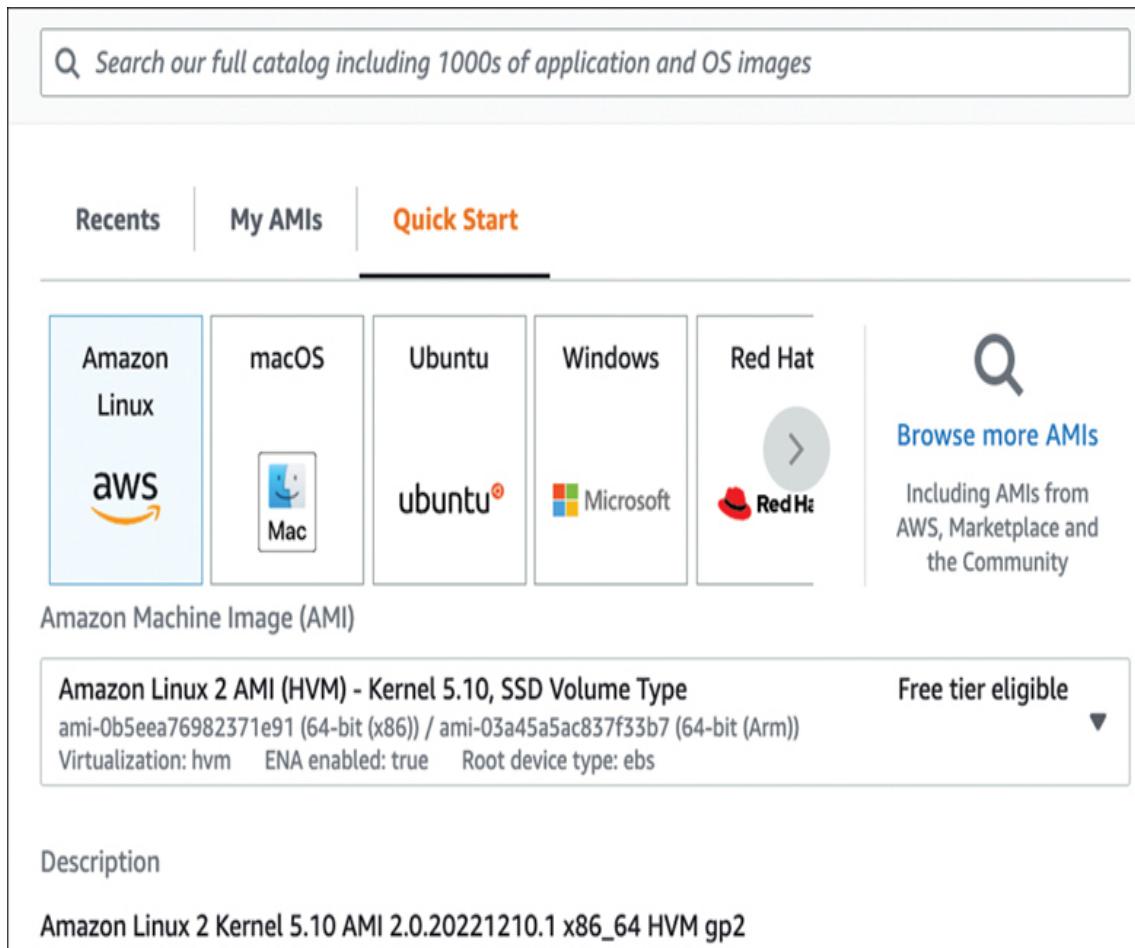
## Amazon Machine Images



An Amazon Machine Image (AMI) is a virtual machine image used to launch EC2 instances.

Organizations can create their own AMIs or use existing AMIs provided by AWS or third parties. An instance launched with a particular AMI creates a virtual machine that is identical to the selected AMI.

AWS provides a wide range of AMIs for various operating systems and applications, including Windows, Linux, macOS, and various open-source operating systems and application stacks (see [Figure 9-2](#)). Create custom AMIs using the EC2 Dashboard or the AWS command-line interface (CLI).



**Figure 9-2** AMI Options at AWS

Each AMI is a template containing a software configuration including the operating system, application software, and any additional software or configuration required.

Each organization's needs and requirements will dictate which type of AMI to create and use; if you want super-fast block storage and your design works with temporary storage volumes, perhaps a local instance with ephemeral storage

makes sense. If you require virtual hard disks, EBS-backed AMIs are widely available. You can also mix and match by deploying an EBS boot volume and additional EBS and ephemeral volumes. Each AMI includes the following components, described by an associated XML manifest:

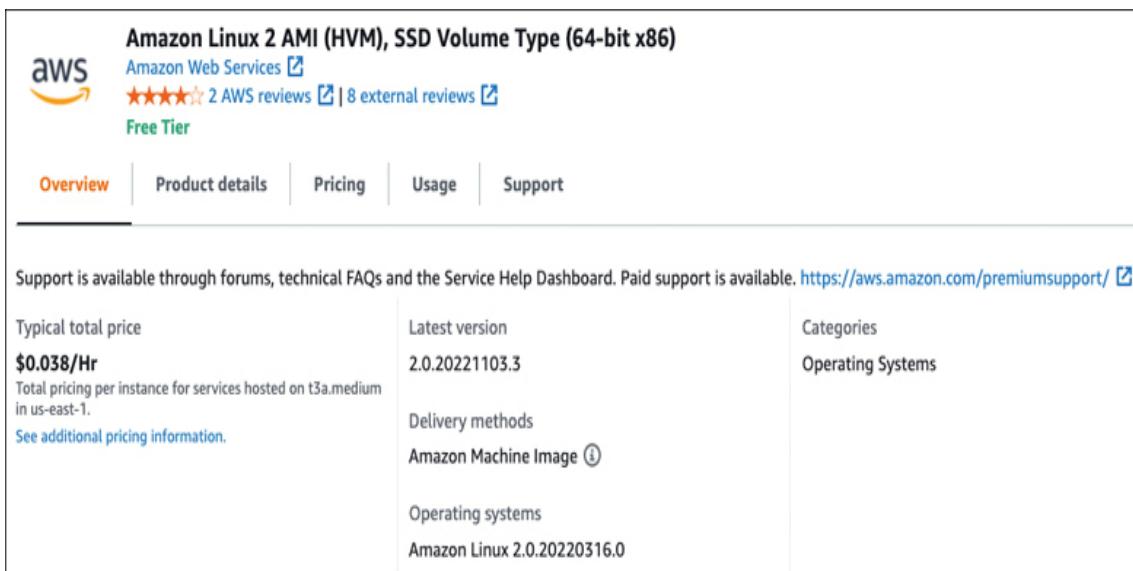
- **Boot volume:** The root boot volume for an EC2 instance can be either an EBS boot volume created from a snapshot or a local instance storage volume copied from an Amazon S3 bucket.
- **Launch permissions:** Launch permissions define the AWS account permitted to use the AMI to launch instances. Default launch permissions are set to private, which means that only the AWS account where the AMI was created can use that AMI. Launch permissions can also define a select list of AWS accounts. Switching AMI launch permissions from private to public means any organization in the overall AWS cloud has access.
- **Volumes to attach:** Volumes attached to the EC2 instance at launch are contained in a block device mapping document. Local instance temporary volumes are listed as ephemeral0 to ephemeral23, depending on the number of instance store volumes created. Instance ephemeral volumes are SSD or NVMe drives.

- **Default region:** AMIs are stored in the local AWS region where they are created. After creation, AMIs can be manually copied or backed up to other AWS regions as necessary.
- **Operating system:** Choices are Linux, Windows, or macOS.
- **Root device storage:** Amazon EBS or an EC2 instance storage volume.

EBS boot volumes are created from EBS snapshots stored in AWS-controlled storage. An EC2 instance storage volume has the root volume created from a template is stored in S3 storage.

## AWS Linux AMIs

Prebuilt AMIs supplied by Amazon include Amazon Linux 2 AMI, as shown in [Figure 9-3](#), and Amazon Linux AMI. Amazon Linux 2 is the latest version of Amazon Linux; Amazon's Linux distribution is based on Red Hat Enterprise Linux (RHEL). The Amazon Linux 2 AMI supports EC2 instances, including EC2 bare-metal instances, and also supports Docker container deployments.



**Figure 9-3** Amazon Linux 2 AMI in the EC2 Console

The Amazon Linux 2 AMI can be used on all EC2 instance types that support hardware virtual machine (HVM) AMIs. The Amazon Linux 2 AMI does not support paravirtualization functionality. Paravirtualization is an older virtualization technique that Linux supported to allow direct execution of user requests from the guest VM to the host computer; increasing the speed of the guest VM operating system calls for networking and storage. With advances in HVM architecture and the AWS Nitro hypervisor, paravirtualization AMIs have extremely limited support at AWS.

The Linux 2 AMI includes a variety of software packages and configurations that seamlessly integrate with many AWS services, such as Amazon CloudFront monitoring and AWS

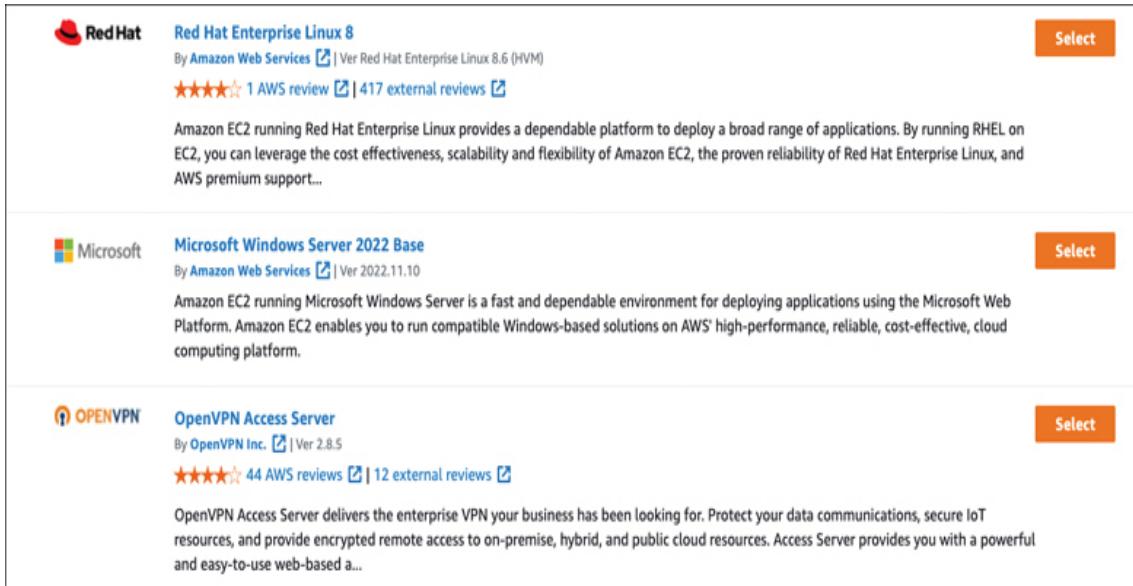
Systems Manager Patch Manager, the AWS CLI, and **cloud-init**, which is used for automating user data scripts at boot. AWS provides long-term support (5 years), including updates from a private repository stored in the AWS region where the instance is located.

## Windows AMIs

Amazon has worked with Microsoft to make available a library of AMIs, including Windows Server versions from 2016 to 2022. Windows instances bundle the licensing fees in their cost (Windows Server and SQL Server licenses). AWS's Windows AMIs are, by default, patched within 5 days of Microsoft's "Patch Tuesday" release.

## AWS AMIs

AWS Marketplace has thousands of AMI options (see [Figure 9-4](#)). Many third-party software appliances are available; for example, Cisco, Juniper, F5, and OpenVPN Access Server images. After selecting an AMI from the AWS Marketplace, decisions to be made include software license fees, the AWS region, and the EC2 instance type.



**Figure 9-4** AWS Marketplace AMI Choices

## Creating a Custom AMI

Creating a custom AMI enables you to create the desired software build for a specific workload.

Follow these steps to create a custom AMI:

1. Launch an EC2 instance and select the latest version of the Amazon Linux AMI. This initial deployment will be used as the base for a custom AMI.
2. Connect to the EC2 instance using SSH, and install and configure any software or applications to include in your custom AMI. Make any necessary changes to the configuration of the operating system or applications.

Additional tasks to complete include defragmenting the attached hard drives for faster operation, deleting temporary files, and creating the desired user accounts and passwords.

3. Stop the EC2 instance. From the EC2 dashboard, select the stopped instance, then choose Create Image from the Actions menu (see [Figure 9-5](#)).

The screenshot shows the 'Create image' dialog box. At the top, it says 'Create image [Info](#)'. Below that, a note states: 'An image (also referred to as an AMI) defines the programs and settings that are applied when you launch an EC2 instance. You can create an image from the configuration of an existing instance.' The 'Instance ID' field contains 'i-0b96d33e33dfd82a1 (web)'. The 'Image name' field is set to 'web\_server\_financial\_01'. The 'Image description - optional' field contains 'Financial web app for budget forecasting'. Under 'No reboot', there is an unchecked checkbox labeled 'Enable'. In the 'Instance volumes' section, there is a table with one row. The columns are: Volume type (EBS), Device (/dev/...), Snapshot (Create new snapshot fr...), Size (8), Volume type (EBS General Purpose S...), IOPS (100), Throughput (100), and Delete on termination (checked). A note at the bottom right of the table says 'Enable'.

**Figure 9-5** AMI Creation

4. Enter a name and description for your custom AMI, and choose Create Image.
5. Wait for the AMI to be created. This process may take several minutes.

6. Once the AMI has been created, launch a new EC2 instance selecting the custom AMI.

During the creation of the AMI, snapshots of the EC2 instance's root volume and any other attached EBS volumes are created.

After the AMI build process has been completed, the EC2 instance is rebooted to check the file system integrity of the snapshots and AMI that was just created. Once a custom AMI is created, tested, and finalized for production, it should be considered a *golden AMI*; the image should be as perfect as possible; customizations or changes should not be allowed to a finalized production AMI.

You can deploy Windows or Linux AMIs using any of the following options:

- **EC2 dashboard:** Create an EBS-backed AMI from an EBS-backed instance.
- **AWS CLI:** Use the `create-image` command to create an EBS-backed AMI from an EBS-backed instance.
- **Amazon Marketplace:** Many commercial Windows and Linux operating system images and third-party virtual software appliance images are available for deployment.
- **My AMIs:** This EC2 dashboard location stores custom AMIs created in your AWS account.

- **AWS Application Migration Service (AMS):** This service enables you to automate the migration of physical, virtual, and cloud-based servers to Amazon EC2. The AWS Application Migration Service creates AMIs of your on-premises servers.
- **AWS Database Migration Service (DMS):** This service enables you to migrate databases to and from Amazon RDS and Amazon Redshift, and migrate on-premises database engines into AWS.

### Custom Instance Store AMIs

[Table 9-3](#) outlines the differences between instances backed by EBS volumes and instances using attached local instance stores.

**Table 9-3** Differences in Instance Storage

| Parameter | EBS Root Device | Instance Store Root Device |
|-----------|-----------------|----------------------------|
|-----------|-----------------|----------------------------|

| Parameter   | EBS Root Device               | Instance Store Root Device  |
|-------------|-------------------------------|---|
| Boot time   | Fast (under 1 minute)         | Not so fast (approximately 5 minutes) because the root drive image must be copied from S3 storage at boot |
| Root drive  | gp2, gp3, io1, io2 SSD drives | 10 GB maximum   |
| Volume type | EBS block storage             | Local instance with block storage located on the bare-metal server hosting the instance                   |

| Parameter        | EBS Root Device   | Instance Store Root Device   |
|------------------|---|--|
| Data persistence | By default, EBS root volumes are deleted when the instance terminates                     | No persistence when the instance store root device is turned off or terminated |
| Backup           | AMI, snapshot, Amazon Data Lifecycle Manager, AWS Backup                                  | AMI storage in Amazon S3 storage   |
| State            | When the EBS instance is turned off, the root volume is persistently stored in Amazon EBS | Running or in a terminated state   |

---

## Note

AMIs that support the Nitro hypervisor must be HVM in design and support enhanced networking, with the ability to boot from EBS storage using an NVMe interface. The latest AWS AMIs for Linux and Windows are HVM by design, as are the latest AMIs for Ubuntu, Debian, Red Hat Enterprise Linux, SUSE Enterprise Linux, CentOS, and FreeBSD.

---

## AMI Build Considerations

Over time, you will develop a standardized procedure for building and maintaining AMIs. Consider the following questions when designing workload compute requirements:

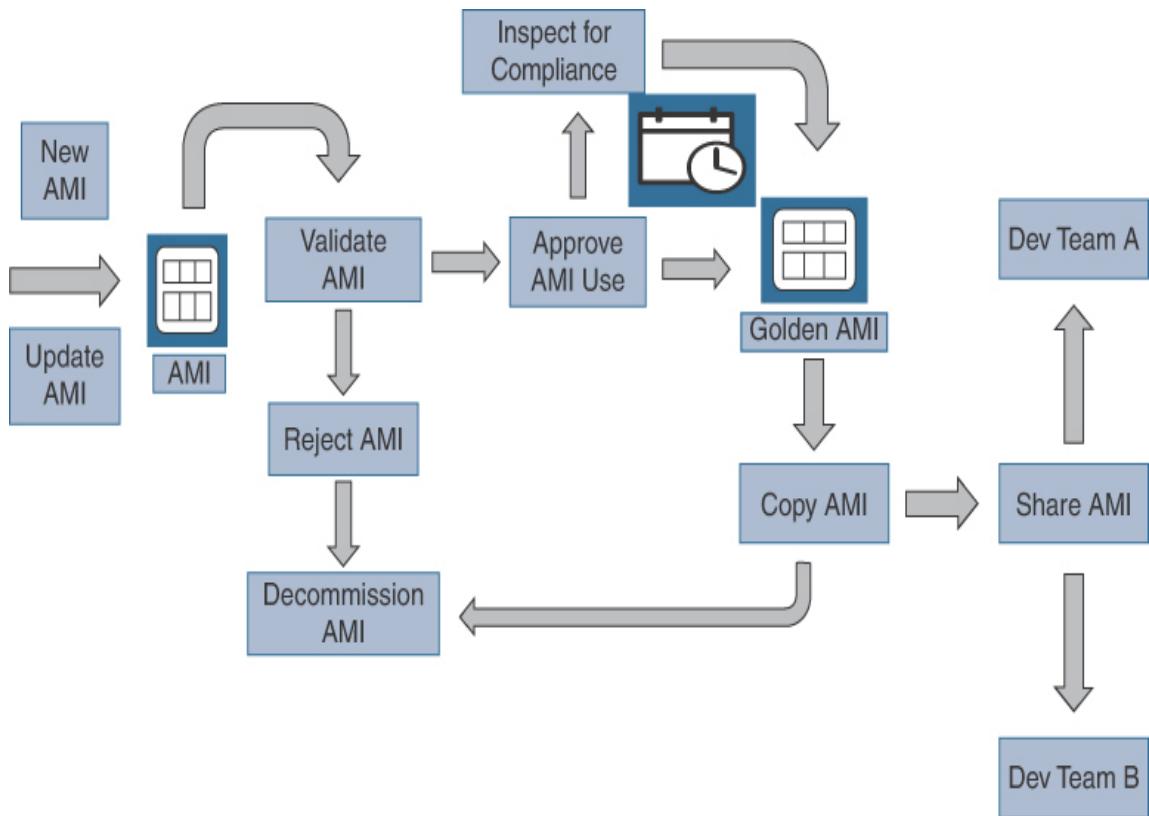
- Are your EC2 instances in an Auto Scaling Group? The EC2 instance and AMI are documented in the Launch Template or Launch Configuration used by the Auto Scaling Group.
- Are there any licensing requirements? AWS License Manager can help manage software licenses from third-party software vendors that are part of your AMI software build.
- How often do you plan to update your AMIs? By default, AMIs supplied by AWS have security and maintenance

updates applied to the repository for each supported AMI, such as Amazon Linux 2 and currently supported Microsoft Windows Server builds.

## Amazon EC2 Image Builder

Amazon EC2 Image Builder helps organizations create, manage, and maintain customized AMIs. With EC2 Image Builder, you can automate the process of building, testing, and distributing AMIs. Here are some key features of EC2 Image Builder:

- **AMI creation:** EC2 Image Builder provides prebuilt image pipelines that enable you to quickly create custom AMIs without having to write any code.
- **Automated testing:** EC2 Image Builder includes built-in testing capabilities that allow a variety of supplied tests and custom tests to validate the functionality and security of your AMIs before they are distributed (see [Figure 9-6](#)).
- **Image distribution:** EC2 Image Builder integrates with AWS IAM and Amazon S3 for secure distribution of AMIs to designated administrators and AWS accounts.



**Figure 9-6** Building and Maintaining AMIs

Here are several considerations when creating an AMI for production:

- Don't embed passwords or security information in an AMI. Instead, use AWS Identity and Access Management (IAM) roles and AWS Secrets Manager.
- At a minimum, use a standard bootstrapping process with user data scripts to customize the initial boot process. User data scripts allow you to automate the initial boot of your instances with required updates and configurations.

- Properly tag your AMIs for identification purposes. Tags can be used for monitoring, automation, security, and creating custom cost and billing reports.
- 

### Note

Amazon has released a golden AMI pipeline sample configuration that is available at

<https://github.com/aws-samples/aws-golden-ami-pipeline-sample>. Here, you will find a readme file with step-by-step instructions, including cloud formation templates to help you set up a golden AMI pipeline for the creation and distribution of AMIs.

---

## AWS Lambda

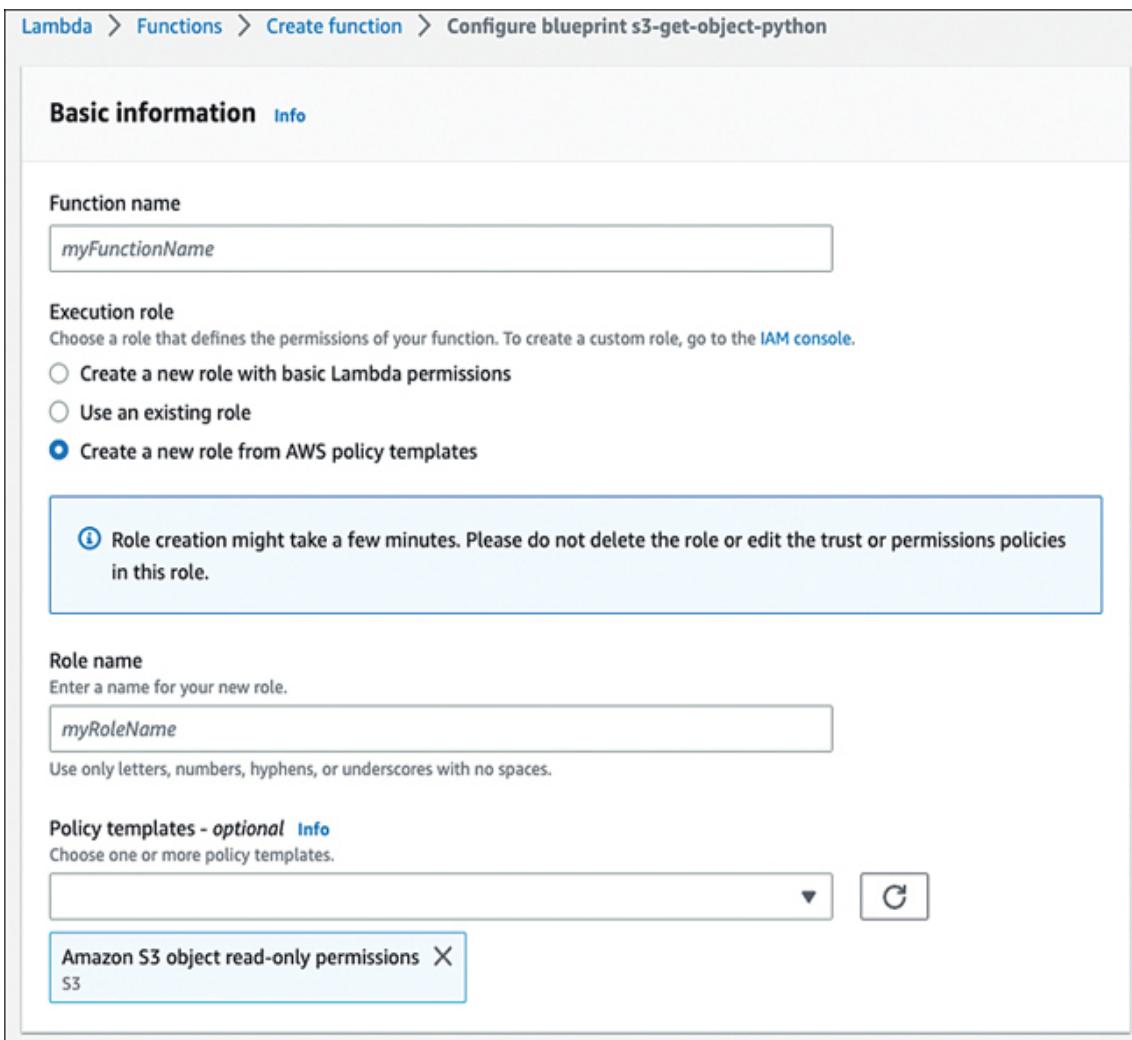


AWS Lambda is a serverless computing service that enables you to run custom functions in response to events or specific triggers. AWS Lambda provides compute processing power on demand, executing functions that you've either written or

selected from a library of functions created by AWS. With AWS Lambda, you can create functions without the need to provision or maintain any infrastructure. Here are some key features of AWS Lambda:

- **Pay-per-use:** AWS Lambda charges you only for the time your code executes.
- **Event-driven:** AWS Lambda functions can be triggered by a wide range of events, such as object changes in an Amazon S3 bucket or an Amazon DynamoDB table, or from an HTTP request to an Amazon API Gateway endpoint.
- **AWS service integration:** AWS Lambda can communicate with other AWS services.
- **Execution environment:** AWS Lambda is compatible with code written in a variety of programming languages, including Node.js, Java, Python, and C#.

To use AWS Lambda, create a function and specify the trigger or event that will trigger your function (see [Figure 9-7](#)). AWS Lambda also provides monitoring and logging features to help you troubleshoot and debug each function's operation. Lambda supports two types of deployment packages: container images and .zip file archives. Container images are stored in the Amazon Elastic Container Registry. Zip archives are stored in an S3 bucket.



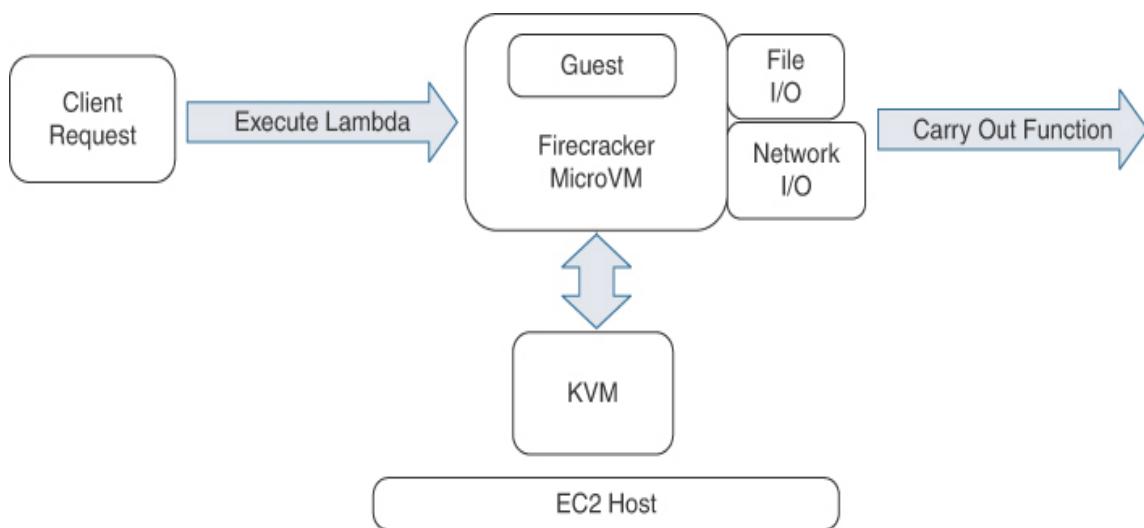
**Figure 9-7** Creating a Custom Lambda Function for a Specific Task

Every time a Lambda function executes, you are charged based on the RAM/CPU and processing time the function uses.

Lambda functions run in a specialized virtualization format called Firecracker hosted on EC2 instances. It was developed by Amazon and is used as the virtualization technology for AWS Lambda and AWS Fargate.

Firecracker uses a microVM architecture, creating a lightweight VM for each function that is executed. Firecracker microVMs are based on the Kernel-based Virtual Machine (KVM) hypervisor and are designed to be small, fast, and secure.

Firecracker utilizes a small footprint of approximately 5 MB of RAM (see [Figure 9-8](#)). Each Firecracker microVM is launched in less than 100 ms. To secure each AWS Lambda function during execution, each Firecracker VM runs in an isolated guest mode, using a network device for communication, a block I/O device to store the function code, and a programmable interval timer. The libraries required for execution are included in the local executable code; no outside libraries are required or allowed.



**Figure 9-8** Firecracker Micro VM Architecture

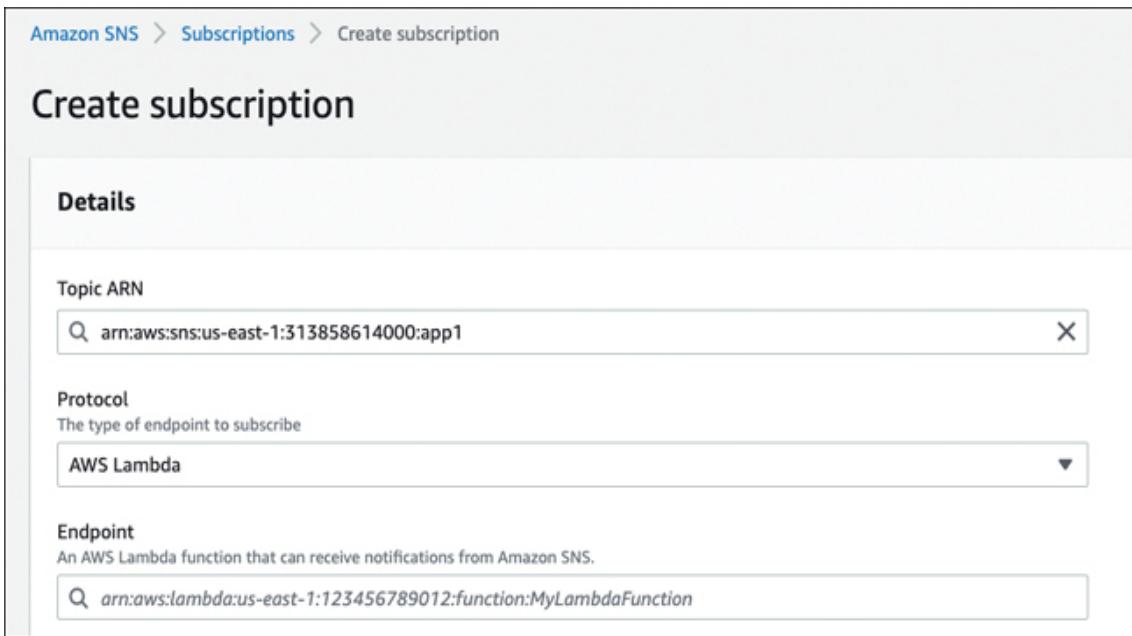
## AWS Lambda Integration

**Key  
Topic**

Many AWS management services have been integrated with Lambda functions:

- **S3 bucket:** AWS Lambda can be triggered when an object is uploaded to an S3 bucket. The uploaded object triggers a Lambda function that converts the object into three different resolutions storing the converted object in three different S3 buckets.
- **DynamoDB table:** AWS Lambda can be triggered when an entry is made in a DynamoDB table, triggering a Lambda function that performs a custom calculation.
- **Amazon Kinesis:** AWS Lambda can be triggered when data is added to an Amazon Kinesis stream. AWS Lambda functions can be executed to process, transform, or analyze real-time data streams.
- **Amazon SNS:** AWS Lambda can be triggered when a message is published to an SNS topic (see [Figure 9-9](#)).
- **Amazon Cognito:** AWS Lambda can be triggered when a user signs up or signs in to an Amazon Cognito user pool, customizing the user experience or sending confirmation emails.

- **Amazon API Gateway:** AWS Lambda can be integrated with Amazon API Gateway to build serverless backends for web, mobile, and IoT applications.
- **Amazon CloudWatch logs:** Content delivered to an AWS CloudWatch log triggers an SNS notification that calls an AWS Lambda function to carry out a specific task.
- **AWS Config:** AWS Config rules analyze whether resources created in an AWS account follow a company's deployment guidelines. When AWS resource deployments don't meet defined compliance requirements, an AWS Lambda function can be executed to delete the out-of-bounds resource.
- **Application Load Balancer (ALB):** Incoming mobile application requests can be directed to AWS Lambda functions hosted by a target group.
- **AWS Step Functions:** Step Functions can utilize Lambda functions as part of the state machine workflow.
- **CloudFront:** Both ingress and egress traffic flow to and from an edge location can be intercepted and queried using Lambda@Edge functions hosted at edge locations for a CloudFront CDN distribution.



**Figure 9-9** SNS Subscription and Lambda Function Executing Actions

## Lambda Settings

When you create an AWS Lambda function, there are several basic settings that you must configure to specify how each function will operate:

- **Function name:** The function name must be unique within the region and account in which you are creating the function.
- **Runtime:** AWS Lambda supports a variety of runtimes, including Node.js, Python, Java, C#, and Go.
- **Role:** The AWS IAM role that defines the permissions that your function has to access other AWS resources or perform

actions.

- **Memory:** Specify a value between 128 MB and 3,008 MB in 64 MB increments.
- **Timeout:** The maximum amount of time that your function is allowed to run before it is terminated. You can specify a value between 1 second and 15 minutes.
- **VPC:** Specify a VPC to give your function access to resources in a private network, such as an Amazon RDS database.

Once a Lambda function is created, the required amount of memory, ephemeral storage, and performance is allocated to your function (see [Figure 9-10](#)). The maximum execution CPU time is 15 minutes, and the minimal execution time is 1 second. Memory can be requested at 64 MB increments from 128 to 10.24 GB.

Lambda > Functions > remove\_instances > Edit basic settings

## Edit basic settings

### Basic settings [Info](#)

#### Description - optional

#### Memory [Info](#)

Your function is allocated CPU proportional to the memory configured.

128 MB

Set memory to between 128 MB and 10240 MB

#### Ephemeral storage [Info](#)

You can configure up to 10 GB of ephemeral storage (/tmp) for your function. [View pricing](#)

512 MB

Set ephemeral storage (/tmp) to between 512 MB and 10240 MB.

#### SnapStart [Info](#)

Reduce startup time by having Lambda cache a snapshot of your function after the function has initialized. To evaluate whether your function code is resilient to snapshot operations, review the [SnapStart compatibility considerations](#).

None



Supported runtimes: Java 11 (Corretto).

#### Timeout

0 min 3 sec

**Figure 9-10** Lambda Function Basic Settings

## AWS Lambda Cheat Sheet

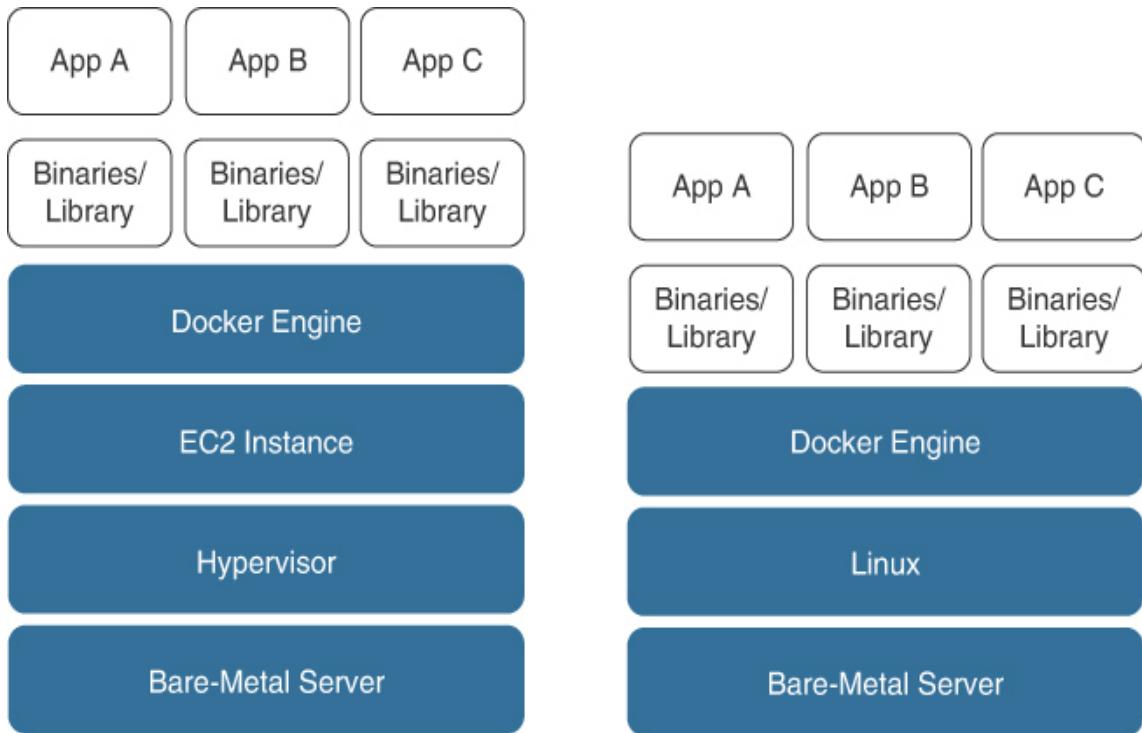
**Key Topic**

For the AWS Certified Solutions Architect – Associate (SAA-C03) exam, you need to understand the following critical aspects of AWS Lambda:

- AWS Lambda allows you to run code as custom functions without having to provision servers to host and execute the function.
- AWS Lambda manages the required vCPU/RAM, storage, and execution of Lambda functions.
- AWS Lambda functions consist of programming code and any associated dependencies.
- Firecracker runs AWS Lambda functions in a lightweight virtual machine (microVM).
- Uploaded AWS Lambda code is stored in an S3 bucket or the Elastic Container Registry.
- Each AWS Lambda function receives 500 MB of temporary disk space for use during execution.
- AWS Lambda monitors executing functions using real-time Amazon CloudWatch metrics.
- A Savings Plan can reduce the cost of running AWS Lambda functions.

## Amazon Container Services

Another way of running applications at AWS is by using containers. Using containers involves changing the underlying virtualization from a complete emulation of a computer system to just the virtualization of the operating system components required for communication: the necessary runtime, libraries, and system tools. A container shares the host operating system's kernel and its system libraries using a Kubernetes control plane or the Docker engine. Containerized applications have read-only access to the host operating system's file system and network stack, as shown in [Figure 9-11](#). When compared to virtual machine deployments, there is less duplication of operating system components when running applications or micro-systems in containers.



**Figure 9-11** Container Architecture Options

The operating system that hosts the Docker or Kubernetes container deployments could run on a bare-metal server or on EC2 instances, providing flexibility in how containers can be hosted at AWS. Organizations could choose a bare-metal instance, a large on-demand or reserved instance hosted within a VPC, to install a custom Docker or Kubernetes environment. When you compare the operating concepts of containers and EC2 instances, containers offer interesting advantages, as detailed in [Table 9-4](#).

**Table 9-4** VMs Versus Containers

| Parameter        | VMs/Instances           | Containers                          |
|------------------|-------------------------|-------------------------------------|
| Stack components | Full application stack  | Lightweight—just the application    |
| Isolation        | Each app has its own VM | Containers share the host OS kernel |
| Startup          | Startup in minutes      | Startup in seconds                  |
| Size             | Gigabytes in size       | Megabytes in size                   |
| Security         | FIPS 140-2 Approved     | FIPS 140-2 validation               |

Containerized applications are moving to the forefront for most public cloud providers due to the interest in multi-cloud—the ability to run an application across different cloud providers. Whether or not you accept or embrace the multi-cloud model, AWS supports Dockers and Kubernetes container

environments, which can run in multi-cloud environments hosted by AWS, Microsoft Azure, or Google Cloud.

AWS, Google, and Microsoft Azure also all have multi-cloud container solutions. AWS has AWS Outposts, which focuses on on-premises deployments using hardware and infrastructure services provided by AWS. Google offers Anthos, a managed application platform that extends the management of containerized applications that are running at Google, AWS, and Azure and on-premises locations. Microsoft Azure has Azure Arc, which can leverage containerized applications on premises, in the Azure cloud, or on a competitor's cloud.

## **Amazon Elastic Container Service**



Amazon Elastic Container Service (Amazon ECS) is a fully managed container orchestration service that makes it easy to deploy, run, and scale containerized applications on AWS running Docker containers running on a managed cluster across multiple availability zones (AZs) within an AWS region. Here are the key features of AWS ECS:

- **Container orchestration:** Amazon ECS enables you to easily deploy and manage your containers across a fleet of Amazon EC2 instances.
- **Auto Scaling:** Amazon ECS automatically scales your containers using Amazon EC2 Auto Scaling to automatically scale your EC2 instances and Docker container workloads.
- **Task Definition:** Specify the desired state of your containerized applications and AWS ECS or Fargate automatically schedules tasks to meet requirements.
- **Monitoring and Logging:** AWS ECS is integrated with the following AWS services: AWS IAM, Amazon EC2 Auto Scaling, ELB Application Load Balancer, Amazon RDS, and the Amazon Elastic Container Registry (ECR).

## AWS ECS Task Definition Choices

An Amazon Elastic Container Service (ECS) task definition is a blueprint that describes the containers that make up an ECS task. It specifies the container image, CPU and memory requirements, networking settings, and other information needed to run a containerized application. The management of the container infrastructure that your tasks and services are hosted on is determined by the launch type selected (see [Figure 9-12](#)).

## Select launch type compatibility

Select which launch type you want your task definition to be compatible with based on where you want to launch your task.

### FARGATE



Price based on task size

Requires network mode awsvpc

AWS-managed infrastructure, no Amazon EC2 instances to manage

### EC2



Price based on resource usage

Multiple network modes available

Self-managed infrastructure using Amazon EC2 instances

### EXTERNAL



Price based on instance-hours and additional charges for other AWS services used

Self-managed on-premise infrastructure with ECS  
Anywhere

**Figure 9-12** Task Definition Options

- **AWS Fargate launch type:** Fargate is a container management service you can select when creating a task definition when running ECS. Fargate relieves you of having to manage the underlying instances and clusters. When you select Fargate, you have the option to specify the container

image and memory and CPU requirements by using an ECS task definition. Fargate then takes over scheduling the orchestration, management, and placement of the containers throughout the cluster, providing a highly available operation environment. Optionally, you can still launch your container environment using manual EC2 launch patterns using the ECS service.

AWS Fargate with ECS can also be used to run containers without having to manage (provision, configure, or scale) your clusters of EC2 instances. Applications launched by Fargate are packaged in containers with the defined CPU, memory at the task level, networking, and security policies. AWS Fargate supports Amazon Linux 2 and Microsoft Windows Server 2019 and 2022 Full and Core editions.

EKS running Kubernetes pods can be also managed by Fargate. Fargate profiles control which pods use Fargate management when launched. Each Fargate profile defines the pod's execution role, subnets where pods will be launched, and namespace.

- **EC2 launch type:** Run containerized applications that are hosted on a cluster of EC2 instances that you manage. Task definitions are created that define the container images that

will run across your clusters. Direct fine-grained control can be run using the EC2 launch type that deploys EC2 instances in your cluster. ECS follows what is called a *task placement strategy* and uses a defined task definition when launching your containers. For example, your strategy might be to spread the containers across multiple EC2 instances and across multiple AZs. Task placement strategies might include running a task per EC2 instance or running your own custom-designed tasks. ECS monitors all running tasks and restarts tasks as a service restart when failure occurs.

- **External launch type:** Self-managed ECS on-premises deployments on an organization's own hardware resources.
- 

#### Note

Amazon ECR is Amazon's own private container registry store for storing and deploying containers both publicly and privately.

---

Each ECS ***task definition*** must define the following criteria:

- The container image to be pulled from the private registry to create the containerized application.
- Container definitions, each of which includes the container image, the command to run when the container starts, the

CPU and memory requirements, and other settings.

- The launch type to use for your task: AWS Fargate, EC2, or ECS Anywhere deployments on premises.
- The task execution role, which is the IAM role the ECS agent uses to perform tasks on your behalf.
- Links that need to be established between containers.
- Volumes, which can be used to store data that is persisted across container restarts or to share data between containers.
- Network and port settings. ECS supports several networking modes, including bridge mode, host mode, and awsvpc mode.

---

#### Note

ECS supports both rolling updates and blue/green deployments using AWS CodeDeploy.

---

## Amazon Elastic Kubernetes Service



Another container management service supported by AWS is Amazon Elastic Kubernetes Service (EKS). EKS can be used to run Kubernetes at AWS without having to install and maintain

your own Kubernetes control plane. EKS clusters have two main components: the Kubernetes control plane and clusters of EKS EC2 instances. EKS clusters contain pods that contain one or more containers. Pods can be deployed as self-managed nodes, EKS-managed node groups, or using AWS Fargate.

The EKS control plane has a minimum of two highly available API server instances. The API server is the primary point of interaction for users and other components of the Kubernetes system, and it is responsible for handling requests to create, update, and delete resources within the cluster. There are also three etcd instances hosted across three availability zones within each AWS region. The etcd instances communicate with each other to form a distributed system that can be used to store and retrieve configuration data. EKS automatically scales the control plane instances based on load, detects and replaces unhealthy instances, and automatically patches and updates control plane instances. Amazon EKS is integrated with AWS services such as Amazon CloudWatch, EC2 Auto Scaling groups, IAM, and ELB Application Load Balancers:

- Amazon CloudWatch logs are directly updated from the EKS control plane audit and diagnostic logs.
- EC2 Auto Scaling communicates with the Kubernetes Cluster Autoscaler with Auto Scaling groups and Launch templates.

- The AWS Load Balancer Controller manages AWS ELB Load Balancers for each Kubernetes cluster.
- AWS IAM security creates IAM roles for role-based access control (RBAC). Access to an EKS cluster using IAM entities is enabled by the AWS Authenticator for Kubernetes, which allows authentication to a Kubernetes cluster.

Amazon EKS can be deployed via the following methods:

- **Amazon EKS:** Deploy managed EKS control plane and nodes managed by AWS.
- **EKS on AWS Outposts:** AWS Outposts allows the running of EKS and other AWS infrastructure services in on-premises locations.
- **Amazon EKS Anywhere:** Deploy and manage Kubernetes clusters on premises supported by AWS using customer hardware and VMware vSphere.
- **EKS Distro:** Deploy open-source deployment of Kubernetes clusters on premises using customer hardware *and* customer support.

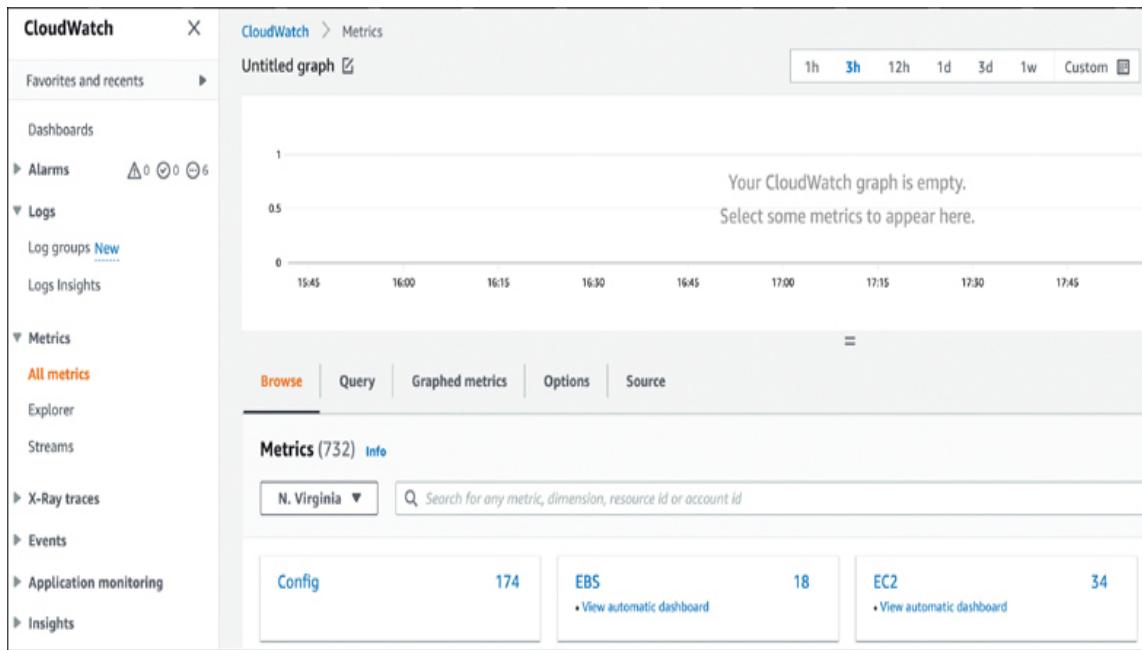
EKS workload clusters can be deployed on any type of EC2 instance. Currently, EKS supports Kubernetes clusters, with support for versions up to 1.23.14.

EKS has been certified Kubernetes conformant, so any third-party plug-ins or tools that you are using or have developed can be migrated to AWS for use in Kubernetes deployments.

## Monitoring with AWS CloudWatch

Amazon CloudWatch is a monitoring service provided by AWS that enables you to monitor and manage various resources and more than 70 cloud services in the AWS cloud. With CloudWatch, you can monitor metrics, set alarms, and automatically react to changes in your resources.

Amazon CloudWatch provides data and operational insights for various AWS resources, such as Amazon Elastic Compute Cloud (Amazon EC2) instances and Amazon Relational Database Service (Amazon RDS) instances. You can use Amazon CloudWatch to collect and track metrics, shown in [Figure 9-13](#), which are variables you can measure for your resources and applications. You can also use Amazon CloudWatch to set metric alarms, which enable you to receive notifications or automatically make changes to the resources you are monitoring when a threshold is breached.



**Figure 9-13** CloudWatch Metrics

You don't *have* to use CloudWatch; you might have specific needs and requirements that CloudWatch can't match. However, if you're using hosted AWS resources that employ EC2 instances or containers hosting applications—or database servers hosted by Amazon Relational Database Service (RDS)—and you don't have another monitoring solution, then you should consider using CloudWatch monitoring because AWS service metrics are already integrated into the CloudWatch monitoring service. You may already have a monitoring solution, such as Loggly, Service Now, or Datadog, that can integrate with CloudWatch data points using the CloudWatch

application programming interface (API) with your third-party monitoring solution.

The following are useful features of CloudWatch to know:



- ***Auto Scaling* and CloudWatch metrics and alarms:** You can automatically adjust your application's compute power as needed with EC2 Auto Scaling and EC2 metrics linked to Amazon CloudWatch alarms.
- **Log filtering with metric filters and alerts:** You can arrange to be notified when specific data patterns occur in your logs and act accordingly using Amazon CloudWatch alarms and Amazon SNS notifications that call AWS Lambda to run custom functions providing automated solutions.
- **Billing Alerts:** You can monitor and control AWS cloud costs by using Billing Alerts and SNS notifications.
- **Logging of CloudTrail API calls to CloudWatch logs:** You can use CloudTrail to log all the API calls made to your AWS resources and send the logs to CloudWatch Logs for storage and monitoring. This can be useful for a variety of purposes, such as auditing and compliance, debugging, and security analysis.

---

## Note

Any third-party monitoring service you use today, such as Splunk, Datadog, or New Relic, supports integration with AWS CloudWatch data records and S3 buckets using RESTful APIs.

---

## CloudWatch Basic Monitoring

Basic monitoring provided by CloudWatch is free of charge and, depending on the AWS service, a select number of basic metrics are available for monitoring the operation of the service.

Metrics, once enabled, report to Amazon CloudWatch on a set schedule. For example, for EC2 instances and containers (ECS), metric data is sent to CloudWatch every 5 minutes. For RDS and Elastic Load Balancing (ELB), a selection of metric data is sent to CloudWatch every 60 seconds. EC2 instances can also enable detailed monitoring, which increases the reporting period to every 60 seconds; however, detailed monitoring is not free.

Make sure to check what basic metrics are available for each service in the AWS documentation; new metrics are being added to CloudWatch all the time to further enhance its monitoring ability. With every AWS service, there are additional metrics you can choose to enable to assist in

monitoring. For example, [Figure 9-14](#) shows the current CloudWatch metrics that are available.

The screenshot shows the CloudWatch Overview page. At the top, there is a dropdown menu set to "All resources". Below it, a table titled "Services" displays the status of various AWS services. The columns are "Status", "Alarm", "Insufficient", and "OK". The "Status" column includes icons: a yellow triangle for EC2, a question mark for others, and a green checkmark for Usage. The "Alarm" column shows the number of alarms for each service. The "Insufficient" and "OK" columns both show a dash (-). The services listed are EC2, AWS Auto Scaling, Application ELB, CloudWatch Events, DynamoDB, EFS, Elastic Block Store, RDS, S3, Simple Queue Service, and Usage.

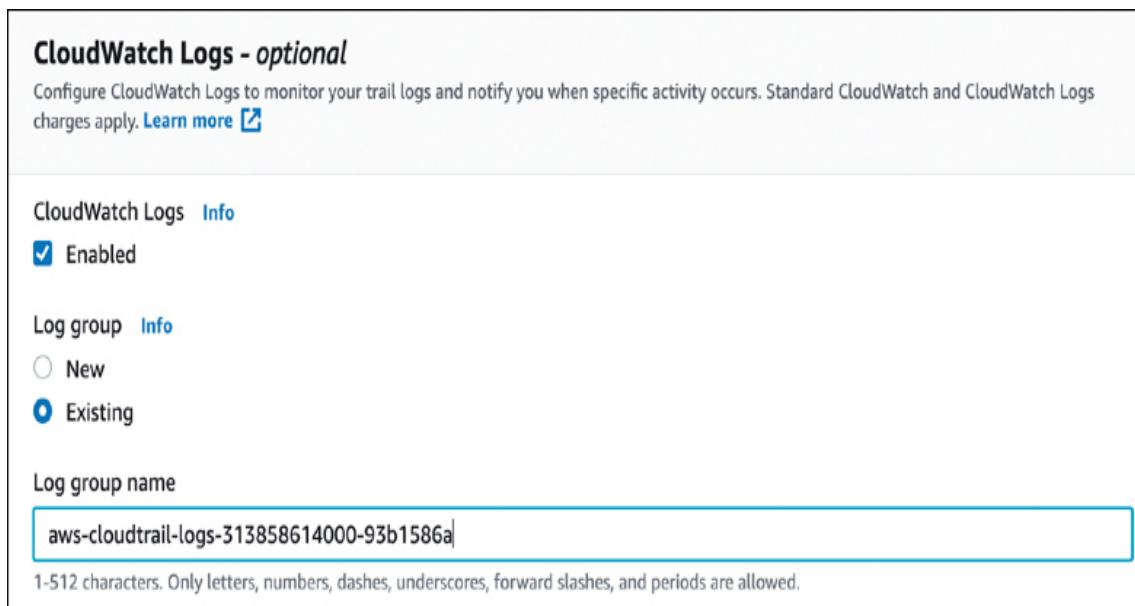
| Status   | Alarm | Insufficient | OK |
|--|-------|--------------|----|
| <span style="color: orange;">⚠</span> EC2                | -     | 3            | -  |
| <span style="color: gray;">?</span> AWS Auto Scaling     | -     | -            | -  |
| <span style="color: gray;">?</span> Application ELB      | -     | -            | -  |
| <span style="color: gray;">?</span> CloudWatch Events    | -     | -            | -  |
| <span style="color: gray;">?</span> DynamoDB             | -     | -            | -  |
| <span style="color: gray;">?</span> EFS                  | -     | -            | -  |
| <span style="color: gray;">?</span> Elastic Block Store  | -     | -            | -  |
| <span style="color: gray;">?</span> RDS                  | -     | -            | -  |
| <span style="color: gray;">?</span> S3                   | -     | -            | -  |
| <span style="color: gray;">?</span> Simple Queue Service | -     | -            | -  |
| <span style="color: gray;">?</span> Usage                | -     | -            | -  |

**Figure 9-14** CloudWatch Metric Choices

## CloudWatch Logs



Amazon CloudWatch allows you to send your event data from AWS CloudTrail custom trails to CloudWatch log groups. This enables Amazon CloudWatch to analyze and search for any specific patterns, such as errors or system issues to analyze further (see [Figure 9-15](#)).



**Figure 9-15** CloudTrail Event Logging to CloudWatch Logs

The CloudTrail service tracks all API calls and authentications made to each AWS account. Stored CloudWatch log data can be reviewed by creating a metric filter that looks for specific events or patterns.

VPC flow logs also store network traffic log data in [\*\*CloudWatch log groups\*\*](#). Network traffic across a VPC, subnet, or elastic

network adapter can be captured and stored in a CloudWatch log group for analysis. Collected data stored in a CloudWatch log can also be analyzed by one of the following options (see [Figure 9-16](#)):

- **Export Data to Amazon S3:** Log information for a defined date range can be exported to an S3 bucket for analysis by any third-party monitoring application.
- **Create an AWS Lambda subscription filter:** When a log event matches a specific filter, AWS Lambda functions can execute a custom task based on the event type generated.

## Flow log settings

Name - *optional*

**Filter**  
The type of traffic to capture (accepted traffic only, rejected traffic only, or all traffic).

Accept  
 Reject  
 All

**Maximum aggregation interval** [Info](#)  
The maximum interval of time during which a flow of packets is captured and aggregated into a flow log record.

10 minutes  
 1 minute

**Destination**  
The destination to which to publish the flow log data.

Send to CloudWatch Logs  
 Send to an Amazon S3 bucket  
 Send to Kinesis Firehose in the same account  
 Send to Kinesis Firehose in a different account

**Figure 9-16** CloudWatch Log Data Export Options

Developers can use CloudWatch logs to analyze network traffic captured in a VPC flow log; company auditors can analyze AWS CloudTrail API calls made to any of the company's AWS accounts.

---

**Note**

Retention time for CloudWatch logs is forever; however, you can choose a retention time frame of up to 10 years. Retention of records stored in S3 buckets can be managed with lifecycle rules.

---

## Collecting Data with the CloudWatch Agent



Amazon CloudWatch receives and stores metrics or log data from the CloudWatch agent installed on each EC2 instance or, optionally, on servers located on premises. AMIs have the CloudWatch agent installed by default.

The CloudWatch agent installation steps for on-premises servers are

**Step 1.** At the AWS CLI command prompt download the CloudWatch Agent installer package using this command syntax:

[Click here to view code image](#)

```
curl https://s3.amazonaws.com/amazoncloudwatch-agent/
linux/amd64/latest/amazon-cloudwatch-agent.rpm -O
```



## Step 2. Install the CloudWatch Agent using the AWS CLI:

[Click here to view code image](#)

```
sudo yum install -y ./amazon-cloudwatch-agent.rpm
```

## Step 3. Configure the CloudWatch agent using the AWS CLI:

[Click here to view code image](#)

```
sudo /opt/aws/amazon-cloudwatch-agent/bin/  
amazon-cloudwatch-agent-config-wizard
```

This command will start the CloudWatch Agent configuration wizard, which will guide you through the process of configuring the CloudWatch Agent to collect and send metrics and log data to CloudWatch. You can also use the CloudWatch Agent configuration file to specify the metrics and log files that you want to collect and send to CloudWatch.

## Planning for Monitoring

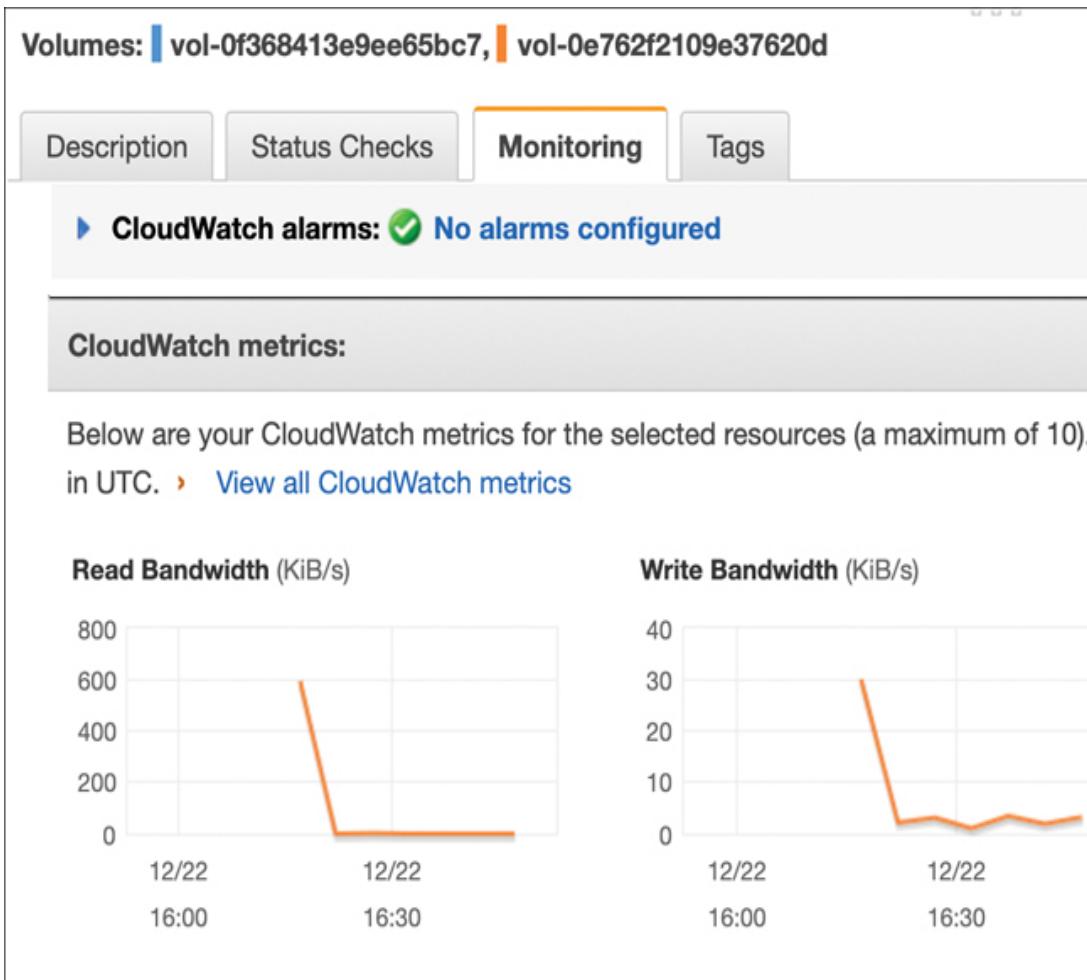
When you are operating in the cloud, monitoring helps you know when your AWS applications and associated services are

operating properly, or there are issues that need to be addressed. You are ultimately monitoring to be kept abreast of potential problems before they occur, and to learn when and why problems occur. Monitoring allows you to be proactive in solving problems. Using CloudWatch, customers can monitor application services and compute resources, including the following variables:

- **Performance:** Monitoring your application's compute speed (database, application, or web server) over time allows you to develop an initial baseline of operation and what you deem to be an acceptable level of performance. For example, by monitoring an application server over a relatively long time period, such as multiple weeks or months, you will get valuable data insights about when EC2 instances, ECS containers, and RDS instances get busy, which times are quieter, and when systems get overloaded, such as at the end of the month or at certain times of each day.
- **Instance resources:** The initial components to monitor with regard to compute performance are CPU, memory, storage, and networking.
- **CPU and RAM utilization:** On EC2 instances running Windows Server, the CloudWatch agent can collect all counters supported by Performance Monitor. Linux instances

collect system information using CloudWatch metrics for CPU, disk, memory, and networking.

- **Available disk space:** EBS volumes have disk performance and disk read and write operation metrics, and the CloudWatch agent can report on total disk space, used space, percentage of total disk space, and many other metrics.
- **IOPS:** CloudWatch has metrics for EBS volumes, as shown in [Figure 9-17](#). In addition, you can monitor the overall read and write performance of EBS volumes and input/output operations per second (IOPS) performance.



**Figure 9-17 EBS CloudWatch Metrics**

- **Network traffic:** Traffic can include subnet traffic and load-balancing traffic and VPN and Direct Connect connections. CloudWatch metrics are available for the ELB service, the network address translation (NAT) gateway, the transit gateway, and VPN connections. VPC flow logs also capture pertinent network information that is stored in CloudWatch. Additional metrics are available for EC2 instance networking.

---

### Note

At AWS, you can set up notifications that are emailed, texted, or sent to an SQS queue, an SNS topic, or AWS Lambda, which could carry out an automated solution using a custom function.

---

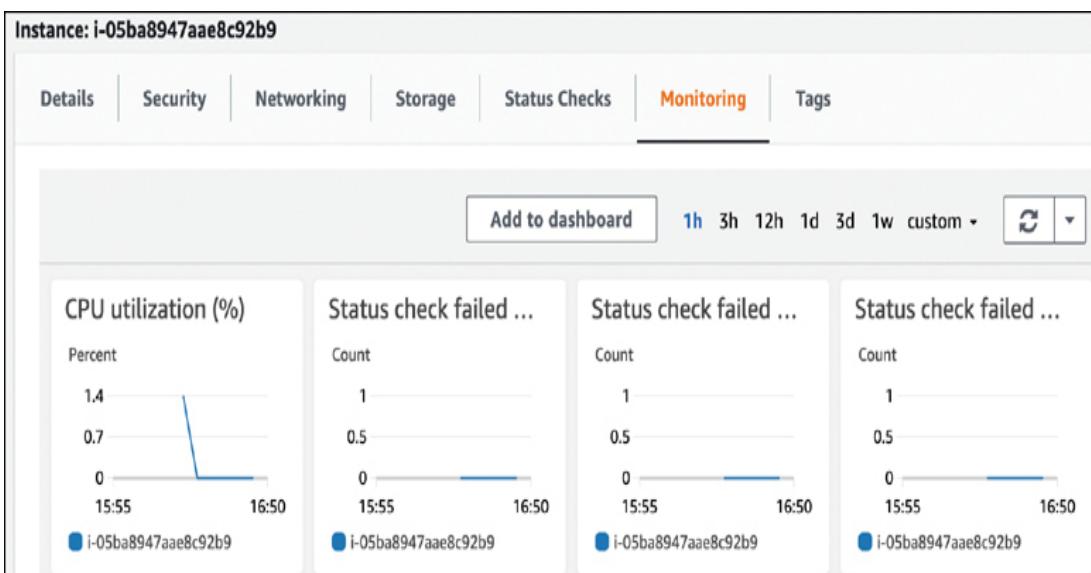
## Amazon CloudWatch Integration



The following are some of the infrastructure cloud AWS services that are embedded with CloudWatch metrics:

- **Amazon SNS:** SNS is used to send alerts when CloudWatch metric alarms are triggered.
- **Elastic Load Balancing (ELB):** Load-balancing metrics available include active connection count, request count, healthy host count, Transport Layer Security (TLS) connection errors, HTTP responses, and errors.
- **Amazon S3:** Storage metrics detail the number of objects and bucket size; request metrics include all requests, GET requests, bytes uploaded and downloaded, and 4xx and 5xx errors.

- **Amazon EC2:** Once an instance has been launched, from the Monitoring tab, 14 metrics are displayed, including options for CPU utilization, disk read and write operations, network traffic and packet flow, and status checks (see [Figure 9-18](#)).



**Figure 9-18** EC2 Instance Metrics

- **EC2 Auto Scaling:** Launch or terminate instances using CloudWatch metrics and alarms that trigger EC2 Auto Scale Groups.
- **AWS CloudTrail:** After a custom trail has been created, CloudWatch can be configured to write all API calls and authentications in your AWS account to a CloudWatch log file.

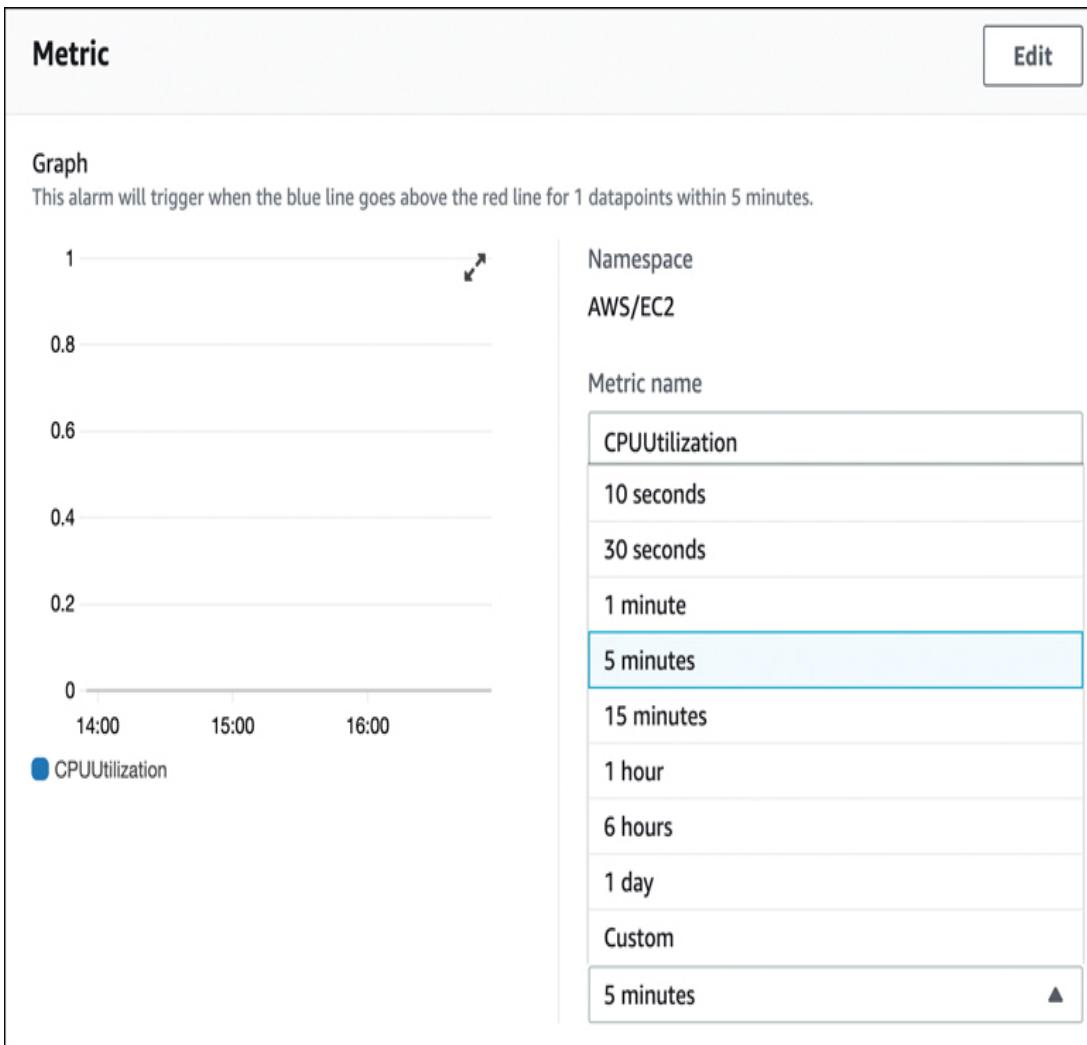
- **AWS Config:** Rules that discover resources that are out of compliance can invoke a custom AWS Lambda function to perform remediation.
- **Amazon RDS:** Metrics include database connections, disk queue length, free storage space, read and write throughput, SSD burst balance, and CPU credit usage.
- **AWS IAM:** All authentication attempts, both successful and unsuccessful, can be monitored with Amazon CloudWatch. SNS notifications can notify humans and automated subscribers.

## Amazon CloudWatch Terminology

Amazon CloudWatch has its own specific terms and definitions. For each alarm, you set a *threshold*. You can choose whether the alarm will trigger when the value is greater than (>), greater than or equal to (>=), less than (<), or less than or equal to (<=) the defined statistic. You need to understand the following common terms when using CloudWatch and for the AWS Certified Solutions Architect – Associate (SAA-C03) exam:

- **Namespace:** Each AWS service stores its CloudWatch metrics and associated data in its own container. At this writing, there are more than 74 AWS services that use CloudWatch metrics.

- **Metrics:** Each metric is a variable within an AWS service. Each monitored variable produces a data set that is collected over a time period, resulting in a graph defined by data points. The data points represent the metric data received from the variable being monitored at an exact point in time, based on the range of times selected. For example, with EC2 instances, you can monitor the metric CPU usage. You can see in [Figure 9-19](#) on the x-axis (which represents time) that the data points represent the data collected over the past hour, in 5-minute increments. The y-axis shows the percentage of CPU utilization.



**Figure 9-19** Data Points Summarized Every 5 Minutes

- **Statistics:** Each metric that you select for analysis collects data based on a defined time period. Graphed data is categorized statistically using some of the following terms:
  - **Minimum:** The lowest value seen during the specified time period

- **Maximum:** The highest value seen during the specified time period
- **Sum:** All values added together, based on a specific time period
- **SampleCount:** The number of data points over a specific time period
- **Average:** Calculated from Sum divided by SampleCount, based on the time period
- **Dimensions:** A dimension describes the metric and what data it stores. Multiple dimensions can be multiple instances assigned to the metric CPU utilization.
- **Units of measurement:** Statistics are defined by bytes, seconds, counts, or percentages.
- **Timestamp:** Each metric is stamped with a timestamp that references the exact time when data was received. Each timestamp includes the date, hours, minutes, and seconds based on the current time in UTC format.
- **Time range (period):** This is the length of time data is collected based on a metric calculated on the defined statistical value. Periods of time can be set from 1 minute up to 15 months. The number of periods determines the number of data points presented on the graph.
- **Alarms:** An alarm starts an action based on the state of the metric's data over the defined time period. Alarms can

trigger notifications using SNS topics, an EC2 action, or an Auto Scaling action. You can also analyze the data output for each of the CloudWatch metrics against a custom baseline of defined measurement; if the data is below a defined threshold, all is well. However, once the metric's results cross or exceed the baseline for a defined time period, the CloudWatch alarm fires, notifying you of potential issues. CloudWatch can also alert an AWS Lambda function and the problem can be fixed—automatically. Once an alarm has been enabled, there are three possible states:

- **OK:** This means that the data collected and evaluated by CloudWatch still fits within the defined alarm *threshold*. For example, you may have defined the CPU utilization at 60%. CloudWatch's analysis of the metric's data points over a defined evaluation period indicates that CPU utilization is currently at 52%; therefore, everything is still okay.
- **ALARM:** The metric's data indicates that the established baseline of acceptable CPU utilization has been breached.
- **INSUFFICIENT DATA:** There is not enough evaluated data to make a definitive analysis.
- **Events:** CloudWatch provides a near-real-time stream of system events for most AWS services based on a defined pattern, such as API calls indicating root account usage within the AWS account or any IAM API calls. Events can be

stored in a CloudTrail log group and tracked using a metric filter, as shown in [Figure 9-20](#). CloudTrail typically delivers events to the configured log group within 15 minutes of the API call. The target that is notified when the event rule fires can be one of several AWS services, including an SNS topic, a Lambda function, or an SQS queue.

# Define pattern

## Create filter pattern

You can use metric filters to monitor events in a log group as they are sent to CloudWatch Logs. You can monitor and count specific terms or extract values from log events and associate the results with a metric. [Learn more about pattern syntax.](#)

### Filter pattern

Specify the terms or pattern to match in your log events to create metrics.

### Test pattern

Select log data to test

### Log event messages

Type log data to test with your Filter Pattern. Please use line breaks to separate log events.

```
[83078518-fcc1-4d30-9573-8b9737671438] BENCHMARK : Running Start Crawl f
[83078518-fcc1-4d30-9573-8b9737671438] BENCHMARK : Classification comple
[83078518-fcc1-4d30-9573-8b9737671438] INFO : Crawler configured with Sche
[83078518-fcc1-4d30-9573-8b9737671438] INFO : Created table gluetest in data
[83078518-fcc1-4d30-9573-8b9737671438] BENCHMARK : Finished writing to Ca
[83078518-fcc1-4d30-9573-8b9737671438] BENCHMARK : Crawler has finished ..
```

**Figure 9-20** Defining a CloudWatch Metric Filter

All metric data is stored with timestamps referencing specific points in time; at AWS, these timestamps are defined as data points. By default, metrics are defined as standard resolution—that is, a resolution of 1 minute. Changing the resolution of a data point from 1 minute to 1 second will provide a much more granular distribution of data points. Data stored in a high-resolution format can be retrieved in periods ranging from 1 second up to multiple minutes.

---

#### Note

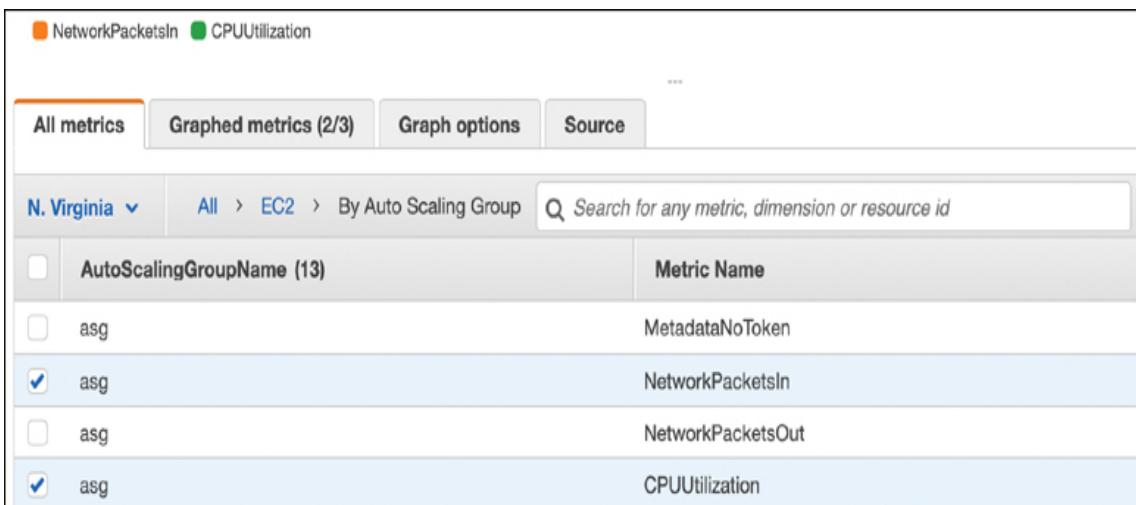
Pricing for CloudWatch is based on the type of metric, the number of dashboards, and the number of alarms that fire. The first 5 GiB of CloudWatch log data storage is free; additional storage is charged. The AWS services that are integrated with CloudWatch send their associated basic metric information to the default CloudWatch dashboard at no additional charge.

---

One of the best examples of AWS service integration and automated problem solving is the relationship between the three essential AWS services, Amazon CloudWatch, EC2 Auto

Scaling, and Elastic Load Balancers (ELB), which work together as a team. Here's how these cloud services work together:

**Step 1.** EC2 instances hosted on subnets in different AZs could be hosted on private subnets and targeted by a public-facing load balancer. EC2 instances can be monitored by Amazon CloudWatch on a select metric such as NetworkPacketsIn or CPUUtilization, as shown in [Figure 9-21](#).



**Figure 9-21** CloudWatch and EC2 Auto Scaling

**Step 2.** Once the selected metric on the instances has exceeded the defined performance threshold for a defined time period—such as 75% for 5 minutes—CloudWatch can fire an alarm that calls the EC2 Auto Scaling service.

**Step 3.** EC2 Auto Scaling starts the build of an additional EC2 instance; when it is ready, the instance is added to the pool of instances targeted by the load balancer. The performance problem is automatically solved without human intervention.

### Creating a CloudWatch Alarm

When creating a CloudWatch alarm, the first task is to choose the metric that you want to link with an alarm function. You then need to define the threshold that, when breached, fires the alarm. To create a CloudWatch alarm, follow these steps:

**Step 1.** From the CloudWatch console, choose Alarms. Choose Create Alarm and then Select Metric.

**Step 2.** From the service namespace (for example, EC2), choose your metric or metrics.

**Step 3.** Select the Graphed metrics tab, as shown in [Figure 9-22](#), and set the following options:

- **Statistics:** Choose Minimum, Maximum, Sum, or Average.
- **Period:** Choose the time frame that data is sampled, such as 1 minute.



**Figure 9-22** Defining Metric Behaviors

**Step 4.** Define the alarm with a name and the threshold at which the alarm should fire (see [Figure 9-23](#)):

- **Whenever:** Define the metric, which in this example is CPUUtilization.
- **Is:** This can be defined as greater than ( $>$ ), less than ( $<$ ), or greater than or equal to ( $\geq$ ).
- **For:** This is the number of data points and the number of sampling periods (in this case, three data points).

## Conditions

Threshold type

Static  
Use a value as a threshold

Anomaly detection  
Use a band as a threshold

Whenever CPUUtilization is...

Define the alarm condition.

Greater  
> threshold

Greater/Equal  
>= threshold

Lower/Equal  
<= threshold

Lower  
< threshold

than...

Define the threshold value.

65

Must be a number

▼ Additional configuration

Datapoints to alarm

Define the number of datapoints within the evaluation period that must be breaching to cause the alarm to go to ALARM state.

3 out of 3

Missing data treatment

How to treat missing data when evaluating the alarm.

Treat missing data as ignore (maintain the alarm state) ▾

**Figure 9-23** Setting CloudWatch Alarm Details

## Additional Alarm and Action Settings

**Key Topic**

There are some complicated settings that can define how the stream of metric data is handled when it is stored in CloudWatch. Say that an instance runs into problems and doesn't send data; in this case, the default setting "Missing" means that the alarm doesn't worry about any missing data points in its evaluation of whether to change the state from OK to ALARM. In other words, the missing data isn't considered critical.

You could also choose to treat the missing data points as being within the defined threshold; in this case, you would choose Not Breaching. Or you could choose to treat the missing data points as reaching the threshold, in which case you would choose Breaching. Our example uses Breaching under the assumption that if data points are not being delivered to CloudWatch, there's a problem with the EC2 instance; as a result, the missing data points are critical.

Amazon SNS actions define the type of notification used when an alarm fires. For EC2 instance metrics, you have several choices:

- Send an SNS notification via email or text.
- Choose an Auto Scaling action that adds additional instances to be added, reducing the CPU utilization.

## **Amazon CloudWatch Cheat Sheet**

For the AWS Certified Solutions Architect – Associate (SAA-C03) exam, you need to understand the following critical aspects of CloudWatch:

- Amazon CloudWatch monitors resources such as EC2 instances, EBS volumes, and RDS instances.
- Amazon CloudWatch monitors custom metrics and log files generated by hosted applications.
- Amazon CloudWatch monitors application performance and operational health.
- Amazon CloudWatch logs allow you to monitor and troubleshoot systems and applications.
- Amazon CloudWatch logs allow you to store log files from EC2, AWS CloudTrail, ELB, and Amazon Route 53 and other third-party logs.

## **Auto Scaling Options at AWS**

Imagine that you have a web application designed and operating on EC2 instances; you have followed best practices and have the application load spread across multiple AZs on multiple instances. The application works well with adequate performance for about 4,000 users. But last week the

application was made available to 1,000 additional users, and the application performance started to slow down. Developers added some additional instances, and everyone was happy once again.

This type of manual solution does work, and if the application in question is going to have additional users in the future, why not just add a few more instances into the mix when required? Users won't complain. Or will they? Performing a manual analysis of the existing EC2 instances and waiting for complaints from the end users is not a proactive approach. Another issue is cost; if more compute instances are running 24/7, the cost of hosting the application increases.

Amazon faced exactly this problem in its early days of running the [Amazon.com](#) website. Running excessive compute capacity was expensive. Amazon solved this problem with EC2 Auto Scaling.

Scaling is the secret sauce for all public cloud providers; all public clouds offer a version of auto scaling. Every application that you use on your smartphone or tablet is controlled by some elements of compute scale. If your application is slow, perhaps it's because there are too many users currently accessing the application. If your application has decent performance, the

odds are that the application is being carefully monitored, and resources are being added or removed based on demand.

Perhaps you're watching a movie on Netflix. If it is a popular movie, the odds are that other people are also watching it.

When Netflix can scale or increase the resources that host the movie based on user demand, it has happy customers. When you shop online, you have probably experienced slow page loads and slow response time. The odds are that this rarely happens at [Amazon.com](#) because of AWS's ability to scale its cloud resources based on demand.

In [Chapter 1, “Understanding the Foundations of AWS Architecture,”](#) we looked at the definition of the public cloud based on the National Institute of Standards and Technology (NIST) definitions and characteristics of the public cloud. One of the NIST characteristics of the public cloud is *rapid elasticity*, defined as follows in NIST SP 800-145: “capabilities can be elastically provisioned and released, in some cases automatically, to scale rapidly outward and inward commensurate with demand.”

[Amazon.com](#), which is powered by the AWS cloud, relies on auto-scaling. On Black Friday events or during the holiday season, one could reasonably expect additional web and

application servers to be required to handle the unknown number of customers who will be buying goods on [Amazon.com](#). However, in the initial years of operation of the [Amazon.com](#) store, the number of servers running was based on expectations; Amazon tried to have enough resources available when there were lots of customers. It was a proactive but very expensive design.

Running too many underutilized resources costs Amazon a lot of money, and this practice will cost your organization a lot of money running hosted resources online and available but not fully utilized. The Amazon online retail store uses monitoring and automation to horizontally scale out its web and application compute resources (adding more compute power) and scaling in (removing compute power) based on the demand. Running just the right number of servers, based on current demand, and dynamically scaling out and in as required, greatly improves workload cost utilization.

---

### Note

The automatic scaling of compute resources at AWS is dependent on monitoring the compute resources that need to scale using CloudWatch metrics and alarms.

---

AWS uses CloudWatch monitoring, ELB load balancing, and Auto Scaling services to run its managed cloud services at the required scale.

## EC2 Auto Scaling

The Amazon EC2 Auto Scaling service enables you to automatically scale your Amazon EC2 resources in response to changes in demand for your application. It helps you ensure that you have the right number of Amazon EC2 instances available to meet the needs of your application, without having to manually create or terminate instances. Here are some key technical details about Amazon EC2 Auto Scaling:

- **Policies:** Amazon EC2 Auto Scaling uses scaling policies to determine when to scale your Amazon EC2 resources up or down. You can create simple scaling policies based on a single metric, such as CPU utilization, or you can create more complex policies that use multiple metrics based on the size of your Amazon EC2 fleet of instances.
- **Scaling actions:** When a scaling policy is triggered, Amazon EC2 Auto Scaling takes a scaling action to either launch or terminate Amazon EC2 instances. The number of instances to launch or terminate can be defined, or you can use a

percentage to scale the number of instances relative to the size of your Amazon EC2 fleet.

- **Scaling groups:** Amazon EC2 Auto Scaling groups are logical collections of Amazon EC2 instances that are managed as a single entity. Multiple scaling groups can be created with their own scaling policies and configurations.
- **Health checks:** Amazon EC2 Auto Scaling uses *health checks* to ensure that only healthy Amazon EC2 instances are used to serve traffic. If an instance fails a health check, it is terminated and replaced with a new instance.
- **CloudWatch alarms:** Amazon EC2 Auto Scaling uses CloudWatch alarms and SNS notifications to trigger scaling actions based on metrics with defined thresholds linked to alarms. For example, an alarm can send an SNS notification when the CPU utilization of your Amazon EC2 instances exceeds a certain threshold, triggering a scaling action to launch additional instances.

## EC2 Auto Scaling Operation

**Key Topic**

EC2 Auto Scaling works with three main components: a launch template or launch configuration, an Auto Scaling group, and a

defined ***scaling policy***.

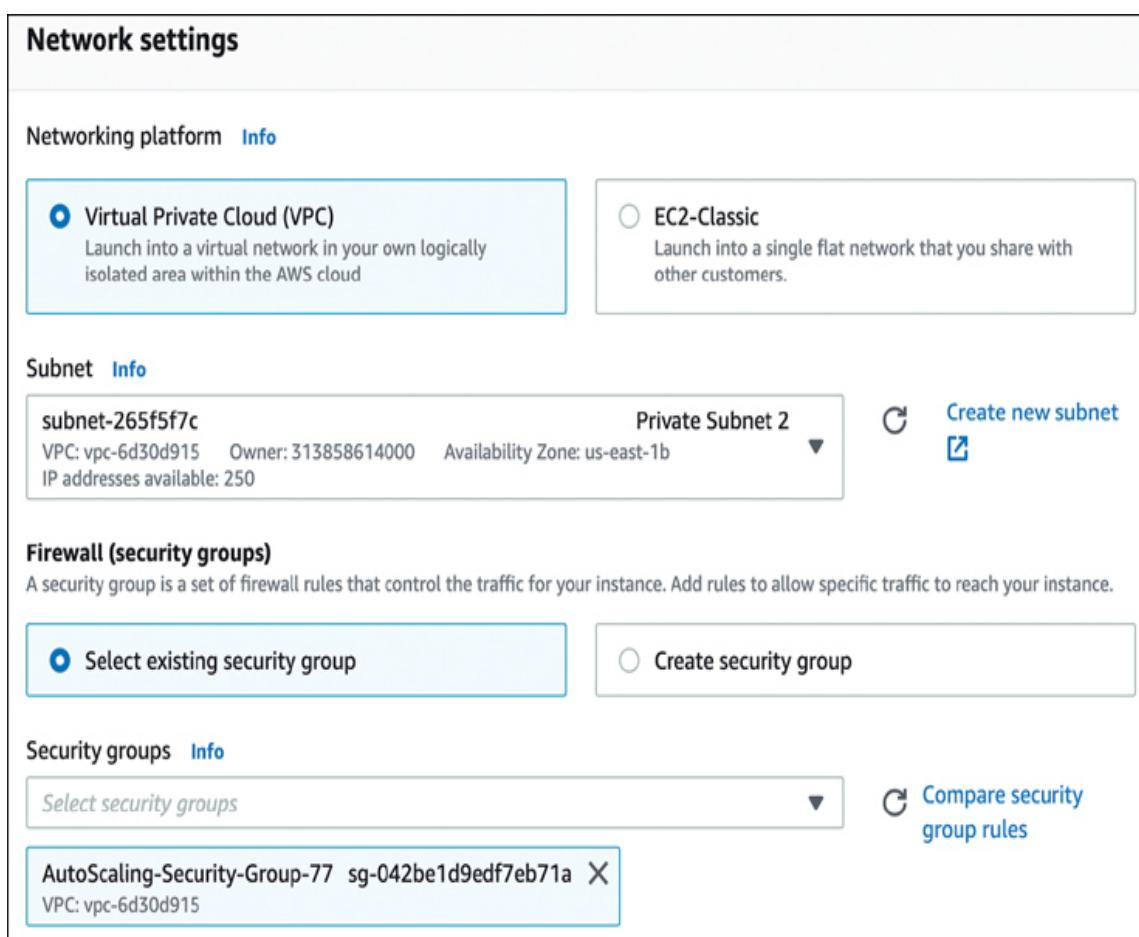
## Launch Configuration

A launch configuration is a simple template used by an ASG to launch EC2 instances. The process of creating a launch configuration is much like the process you would follow to manually launch an EC2 instance from the AWS Management Console. The launch configuration contains the installation details for the EC2 instances that will be built by the ASG. Each launch configuration is associated with one ASG. A launch configuration includes numerous system components, including the instance ID of the AMI, the instance type to build, the key pair for authentication, the desired security groups, and a block storage device. Launch configurations are slowly being superseded by the launch template, which has many additional settings that can be used when deploying instances.

## Launch Templates

A ***launch template*** is similar to a launch configuration, with added features related to versioning an existing template, so you can make changes. In addition, launch templates support all new AWS features related to EC2 instances and Auto Scaling, whereas launch configurations do not. A default launch template can be created as a source template; then, other

versions of the template can be created and saved. AWS recommends that you use launch templates rather than launch configurations because a launch template has many deployment options when compared to a launch template (see [Figure 9-24](#)).



**Figure 9-24** Launch Template Network Settings

## Auto Scaling Groups

An Auto Scaling group (ASG) is built from a collection of EC2 instances that have been generated from the associated launch configuration or launch template. Each ASG launches instances, following the parameters of the launch template, to meet the defined scaling policy. An ASG can function independently or be associated with a load balancer, as shown in [Figure 9-25](#).

The screenshot shows the 'Load balancing - optional' section of the AWS Auto Scaling Group creation wizard. It includes three options: 'No load balancer' (radio button unselected), 'Attach to an existing load balancer' (radio button selected, highlighted in blue), and 'Attach to a new load balancer' (radio button unselected). Below this, the 'Attach to an existing load balancer' section is expanded, showing two choices: 'Choose from your load balancer target groups' (radio button selected, highlighted in blue) and 'Choose from Classic Load Balancers' (radio button unselected). A dropdown menu labeled 'Select target groups' contains one item: 'TG1 | HTTP Application Load Balancer: alb'. There is also a small 'X' icon next to the target group name.

**Load balancing - optional** Info

Use the options below to attach your Auto Scaling group to an existing load balancer, or to a new load balancer that you define.

No load balancer  
Traffic to your Auto Scaling group will not be fronted by a load balancer.

Attach to an existing load balancer  
Choose from your existing load balancers.

Attach to a new load balancer  
Quickly create a basic load balancer to attach to your Auto Scaling group.

**Attach to an existing load balancer**

Select the load balancers that you want to attach to your Auto Scaling group.

Choose from your load balancer target groups  
This option allows you to attach Application, Network, or Gateway Load Balancers.

Choose from Classic Load Balancers

Existing load balancer target groups  
Only instance target groups that belong to the same VPC as your Auto Scaling group are available for selection.

Select target groups ▾ C

TG1 | HTTP X  
Application Load Balancer: alb

**Figure 9-25** Auto Scaling Group Settings and Options

Scaling policies are attached to ASGs, which automatically increase or decrease the number of EC2 instances based on CloudWatch alarms and notifications.

The EC2 instance types that can be added to an ASG include on-demand, spot, or reserved instances across multiple AZs. You can also define the percentage of on-demand and spot instances to deploy for additional cost savings.

An ASG performs health checks on instances added to the ASG; ELB health checks can also be chosen. EC2 Auto Scaling performs health checks for recently launched EC2 instances using the EC2 status check (see [Figure 9-26](#)). If instances added to an ASG fail their status checks after boot, they are considered unhealthy, and EC2 Auto Scaling terminates, relaunches, and re-adds them to the ASG.

**Health checks - optional**

**Health check type** [Info](#)  
EC2 Auto Scaling automatically replaces instances that fail health checks. If you enabled load balancing, you can enable ELB health checks in addition to the EC2 health checks that are always enabled.

EC2       ELB

**Health check grace period**  
The amount of time until EC2 Auto Scaling performs the first health check on new instances after they are put into service.

300 seconds

**Figure 9-26** Health Check Options

ELB health checks are a little more rigorous than Auto Scaling health checks, and ASGs can and should be configured to also use ELB health checks. If an ASG or ELB health check fails, the unhealthy instances are terminated and relaunched.

---

#### Note

If an ASG is not associated with a load balancer, the ASG status health checks of each EC2 instance are used to determine the health of the new instance.

---

## Scaling Options for Auto Scaling Groups



For the AWS Certified Solutions Architect – Associate (SAA-C03) exam, you need to understand several scaling options for deploying Auto Scaling groups:

- **Target tracking scaling policy:** You can select the metric type and target value to maintain the desired capacity (see [Figure 9-27](#)) and automatically have any instances that are determined to be unhealthy (by the Auto Scaling health check or the load-balancing health check) replaced.

### Scaling policies - optional

Choose whether to use a scaling policy to dynamically resize your Auto Scaling group to meet changes in demand.

Target tracking scaling policy

Choose a desired outcome and leave it to the scaling policy to add and remove capacity as needed to achieve that outcome.

None

Scaling policy name

Target Tracking Policy

Metric type

Average CPU utilization

Target value

60

Instances need

300

seconds warm up before including in metric

**Figure 9-27** Target Tracking Variables for Increasing Group Size

- **Simple scaling:** You can increase and decrease the size of an ASG based on a single metric and automatically manage the size of the EC2 instances in the Auto Scaling group. An Auto Scaling group has three key parameters that determine the

size and capacity of the group: minimum size, maximum size, and desired capacity.

- **Minimum size:** The minimum size is the minimum number of Amazon EC2 instances that you want to have running in your Auto Scaling group at any given time. The minimum size cannot be set to a value lower than the current size of the group.
- **Maximum size:** The maximum size is the maximum number of Amazon EC2 instances that you want to have running in your Auto Scaling group at any given time. The maximum size cannot be set to a value higher than the current size of the group.
- **Desired capacity:** The desired capacity is the number of Amazon EC2 instances that you want to have running in your Auto Scaling group at a given time. The desired capacity can be any integer value between the minimum size and the maximum size, inclusive. The minimum size, maximum size, and desired capacity of an Auto Scaling group determine the range of values that Amazon EC2 Auto Scaling can use to scale your Amazon EC2 fleet.

You can select from EC2 instance metrics, such as CPU utilization, to support target tracking. Setting CPUUtilization to 60% results in instances being added to or removed from the Auto Scaling group as required to maintain the desired CPU

utilization (see [Figure 9-28](#)). Slightly more instances may be added than necessary when maintaining the desired CPU utilization because the math behind the scaling calculations rounds up the number of instances to add or remove.

The screenshot shows the configuration for a simple scaling policy. The 'Policy type' is set to 'Simple scaling'. The 'Scaling policy name' is 'Financial'. Under 'CloudWatch alarm', it shows a selected alarm 'awsec2-i-0b70d1f1b2baf29a9-CPU-Utilization' and a threshold of 'CPUUtilization >= 0.55 for 1 consecutive periods of 300 seconds'. Below this, it specifies 'InstanceId = i-0b70d1f1b2baf29a9'. In the 'Take the action' section, there is an 'Add' button, a value '2', and a dropdown for 'capacity units'. At the bottom, there is a 'And then wait' section with a value '300' and the text 'seconds before allowing another scaling activity'.

**Figure 9-28** Simple Scaling Parameters

**Simple scaling** allows the use of a single custom metric that determines the scaling out and in of your defined fleet of EC2 instances. Multiple policies can be attached to an ASG that can control the scaling out and in. Multiple scaling policies can provide scaling control. For example, a scaling policy might react to the CloudWatch metric Network Utilization and scale out when network traffic is greater than (>) a certain

percentage. Or a scaling policy could measure the depth of messages in an SQS queue and scale out when the number of messages is over a certain value. Both simple scaling and step scaling, discussed next, support the following parameters for scaling instances:

- **ChangeInCapacity:** This parameter increases or decreases the capacity of the ASG by the defined number of instances.
- **ExactCapacity:** This parameter defines the capacity of the ASG based on a defined number. For example, if the current capacity is four instances and an adjustment to the ASG is three instances, the capacity is set to seven instances.
- **PercentChangeInCapacity:** This parameter changes the capacity of the ASG by either a positive or a negative percentage value.

---

#### Note

Amazon recommends that metrics used for target tracking should be set for a 1-minute frequency to ensure faster response time. Detailed monitoring must be enabled to use 1-minute intervals.

---

- **Step scaling:** A step scaling policy enables you to define a series of steps, or thresholds, for a metric, and specify a

different number of Amazon EC2 instances or capacity units to launch or terminate for each step. For example, you could specify that if the average CPU utilization of your Amazon EC2 instances exceeds 70%, Amazon EC2 Auto Scaling should launch two additional instances, and if the average CPU utilization falls below 50%, it should terminate two instances. Step scaling enables you to define lower and upper boundaries for the metric being used and to define the amount by which to scale in or scale out the instances, as shown in [Figure 9-29](#), with incremental steps in or out:

- A first instance is added when CPU utilization is between 40% and 50%.
- The next step adds two instances when CPU utilization is between 50% and 70%.
- In the third step, three instances are added when CPU utilization is between 70% and 90%.
- When CPU utilization is greater than 90%, a further four instances are added.

Policy type  
Step scaling

Scaling policy name  
Thanksgiving sale

CloudWatch alarm  
Choose an alarm that can scale capacity whenever:  
awsec2-i-0b70d1f1b2baf29a9-CPU-Utilization

Create a CloudWatch alarm [Create a CloudWatch alarm](#)

breaches the alarm threshold: CPUUtilization >= 0.55 for 1 consecutive periods of 300 seconds for the metric dimensions:  
InstanceId = i-0b70d1f1b2baf29a9

Take the action  
Add

|   |                |         |                               |
|---|----------------|---------|-------------------------------|
| 1 | capacity units | when 45 | <= CPUUtilization < 55        |
| 2 | capacity units | when 55 | <= CPUUtilization < 65        |
| 2 | capacity units | when 65 | <= CPUUtilization < +infinity |

Add step

The screenshot shows the configuration of a Step Scaling policy. The policy type is set to 'Step scaling' and the scaling policy name is 'Thanksgiving sale'. A CloudWatch alarm named 'awsec2-i-0b70d1f1b2baf29a9-CPU-Utilization' is chosen as the trigger. The 'Take the action' section defines three steps: 1. Add capacity units when CPUUtilization is less than or equal to 45 and greater than 55. 2. Remove capacity units when CPUUtilization is less than or equal to 55 and greater than 65. 3. Remove capacity units when CPUUtilization is less than or equal to 65 and greater than infinity. The 'Add step' button is visible at the bottom.

**Figure 9-29** Step Scaling Parameters

The step scaling policy also defines a warmup period. This is a period of time that you can specify during which Amazon EC2 Auto Scaling will not take any scaling actions in response to a trigger to add additional instances. The warm-up period gives EC2 instances time to start up and become fully operational before they start serving traffic. Keep in mind that newly launched EC2 instances might need to apply updates, finish configuration, and pass health checks before they are operational. Defined scale-down steps are carried out when scaling in.

When you create or update an Auto Scaling group, you can specify the desired capacity in terms of Amazon EC2 instances or in terms of capacity units. A capacity unit is a unit of capacity that represents the number of Amazon EC2 instances that you want to have running in your Auto Scaling group.

By using capacity units, you can specify the desired capacity of your Auto Scaling group in a more flexible and granular way. For example, you can specify that you want to have two capacity units of a particular instance type, which would equate to two Amazon EC2 instances of that type. This enables you to specify the desired capacity of your Auto Scaling group in a way that is independent of the specific instance type that you are using.

- **Scale based on a schedule:** Scaling can be defined based on time and date values that instruct Auto Scaling to scale up or down at a specific time. The start time and the minimum, maximum, and desired sizes can be set for recurring actions.

## Management Options for Auto Scaling Groups

**Key Topic**

There are several options available for managing ASGs, including Predictive scaling, Warm pools, and Instance replace.

- **Predictive scaling:** A feature called predictive scaling uses machine learning models to analyze the deployed application and traffic pattern history to forecast recommended scaling. In the case of a workload that has high usage during normal business hours and lower usage overnight, predictive scaling can add capacity before an increase in daily traffic occurs.
- **Warm pools:** Having the ability to scale quickly in response to real-time application demand is sometimes hard to manage due to the latency that occurs when the EC2 instance or container is initialized at startup. Latency can sometimes be several minutes or longer until the instance is ready to be added to an Auto Scaling group. EC2 Auto Scale warm pools allow you to reduce scale-out latency by maintaining a pool of pre-warmed EC2 instances ready to be immediately placed into service when a scale-out event is issued.
- **Instance refresh:** Instance refresh can be used to update the instances currently in an ASG instead of replacing the instances one at a time. Instance refresh can be useful to deploy a new AMI or new user data script. Instances can also be replaced by specifying the maximum amount of time an instance can be in service before it is terminated and

replaced. A minimum instance refresh value of at least 1 day must be initially set.

---

### Note

You can make manual changes to your ASGs at any time by changing the maximum, minimum, or desired state values to start capacity changes.

---

## Cooldown Period

Simple scaling policies are bound to a ***cooldown period***. Let's look at an example of where a cooldown period is useful. Suppose that your company has a three-tier application running in test mode on AWS, consisting of web servers, application servers, and the AWS RDS database tier. Separate ASGs are created for the web tier and for the application tier. Each tier uses CloudWatch alarms to scale out whenever the CPU utilization for each tier exceeds 70%. The triggering of the CloudWatch alarm instructs the ASG to launch and configure additional instances.

For this example, the EC2 instances are using a user data script for installing updates at first boot. These additional tasks can take time—perhaps 4 or 5 minutes. When an instance has

finished updating and is ready for use and marked as “healthy,” it enters the *InService* state.

While the new EC2 instance is being readied, if an unexpected increase in CPU utilization occurs to the web tier, the CloudWatch alarm may trigger once again, ordering the ASG to launch another EC2 instance. However, because a cooldown period is in force, even after the ASG is directed to launch an EC2 instance on a scale-out request, all scaling requests are ignored until the cooldown period finishes. The default cooldown period is 300 seconds; you can change this value when creating an ASG or at a later point if you wish to make modifications.

## Termination Policy



When a scale-in event occurs, defined default termination policies control which EC2 instances are first terminated. The default termination policy is designed to ensure that your instances are evenly spaced across the availability zones to maintain high availability. Termination of unhealthy instances

occurs first; then Auto Scaling attempts to launch new instances to replace the terminated instances.

Other termination options that can be chosen for a custom termination policy include the following:

- **Oldest launch template:** Remove instances that are using an older launch template.
- **Oldest launch configuration:** Remove instances that are using an older launch configuration.
- **Closest to next instance hour:** Terminate instances that are closest to the next billing hour.
- **Newest Instance:** Terminate the newest instance in the ASG.
- **Oldest Instance:** Terminate the oldest instance in the ASG.
- **Allocation strategy:** Terminate instances based on on-demand or spot instance strategies.

---

#### Note

The scale-out setting defines which instances are launched by the defined scaling policy. The scale-in setting defines which instances are terminated by the scaling policy.

---

## Lifecycle Hooks

## Key Topic

You can create custom actions that are carried out on instances that an ASG launches or terminates. With such an action, an instance is placed into a wait state and held from becoming registered in the ASG or from being terminated for a period of time. While it is being held in the wait state, custom actions can be performed on the instance. Think of a wait state as an opportunity to perform any task you want on an EC2 instance before it is added to or removed from the ASG. An instance can be held in a wait state for a maximum of 48 hours. During the wait time, the following custom actions are allowed:

- Call an AWS Lambda function to perform a specific task.
- Send a message to a defined SNS notification.
- Execute a script as the instance starts and remains in the defined wait state.
- Add a ***lifecycle hook*** to an ASG by using AWS CLI commands that populate the user data location in a launch template, as shown here:

[Click here to view code image](#)

```
AWS autoscaling put-lifecycle-hook --lifecycle-hook
code> --auto-scaling-group-name <ASG here > --lifec
autoscaling:EC2 INSTANCE LAUNCHING
```



## EC2 Auto Scaling Cheat Sheet



For the AWS Certified Solutions Architect – Associate (SAA-C03) exam, you need to understand the following critical aspects of EC2 Auto Scaling:

- EC2 Auto Scaling ensures that you have the correct amount of compute power required by an application at all times.
- An Auto Scaling group is a collection of EC2 instances that can provide horizontal scaling across multiple AZs.
- An Auto Scaling group requires an attached Auto Scaling policy.
- EC2 Auto Scaling can be integrated with ELB target groups and Amazon CloudWatch metrics and alarms.
- A launch template is a template used by Auto Scaling to create additional EC2 instances as required.
- Application Load Balancers, Network Load Balancers, and Gateway Load Balancers can be attached to an Auto Scaling group.

- Load balancers must be deployed in the same region as the Auto Scaling deployment.

## AWS Auto Scaling



You now know about EC2 Auto Scaling, but there's another scaling option called AWS Auto Scaling. What's the difference? AWS Auto Scaling can manage auto scaling for the following AWS workload after they have been deployed using a scaling plan. Organizations can use AWS Auto Scaling to manage scaling for these AWS resources.

- **Amazon Aurora:** Increase or decrease the number of read replicas that have been provisioned for an Aurora DB cluster.
- **EC2 Auto Scaling:** Increase or decrease the number of EC2 instances in an Auto Scaling group.
- **Elastic Container Service:** Increase or decrease the desired task count in ECS.
- **DynamoDB:** Increase or decrease the provisioned read and write capacity of a DynamoDB table or global secondary index.

- **Spot fleet:** Increase or decrease the target capacity of a spot fleet. A spot fleet enables you to launch a desired number of instances, called a *fleet* of instances, based on the desired price and number of spot instance types.

A scaling plan tells AWS Auto Scaling how to optimize the utilization of supported AWS resources defined in your scaling plan. You can optimize for availability, for cost, or a balance of both options. AWS Auto Scaling can also be used to create predictive scaling for EC2 instances that are members of a target tracking EC2 auto scaling group. Predictive scaling looks at historic traffic patterns and forecasts and schedules changes in the number of EC2 instances running at the appropriate time.

## Exam Preparation Tasks

As mentioned in the section “[How to Use This Book](#)” in the Introduction, you have a couple of choices for exam preparation: the exercises here, [Chapter 16](#), “[Final Preparation](#),” and the exam simulation questions in the Pearson Test Prep Software Online.

## Review All Key Topics

Review the most important topics in the chapter, noted with the Key Topic icon in the margin of the page. [Table 9-5](#) lists these key topics and the page number on which each is found.



**Table 9-5** [Chapter 9](#) Key Topics

| Key Topic Element         | Description                    | Page Numbers |
|---------------------------|--------------------------------|--------------|
| <a href="#">Table 9-2</a> | Compute Services and Use Cases | 425          |
| Section                   | Amazon Machine Images          | 429          |
| Section                   | Creating a Custom AMI          | 432          |
| Section                   | AWS Lambda                     | 436          |
| Section                   | AWS Lambda Integration         | 438          |
| Section                   | AWS Lambda Cheat Sheet         | 441          |

| Key Topic Element | Description                               | Page Numbers |
|-------------------|---|--------------|
| Section           | Amazon Elastic Container Service          | 443          |
| Section           | Amazon Elastic Kubernetes Service         | 446          |
| List              | Useful features of CloudWatch             | 448          |
| Section           | CloudWatch Logs                           | 449          |
| Section           | Collecting Data with the CloudWatch Agent | 451          |
| Section           | CloudWatch Integration                    | 453          |
| Section           | Additional Alarm and Action Settings      | 460          |
| Section           | CloudWatch Cheat Sheet                    | 461          |

| Key Topic Element | Description                                | Page Numbers |
|-------------------|--|--------------|
| Section           | EC2 Auto Scaling Operation                 | 463          |
| Section           | Scaling Options for Auto Scaling Groups    | 466          |
| Section           | Management Options for Auto Scaling Groups | 470          |
| Section           | Termination Policy                         | 471          |
| Section           | Lifecycle Hooks                            | 472          |
| Section           | EC2 Auto Scaling Cheat Sheet               | 473          |
| Section           | AWS Auto Scaling                           | 473          |

## Define Key Terms

Define the following key terms from this chapter and check your answers in the glossary:

metric

Amazon Machine Image (AMI)

task definition

Auto Scaling

CloudWatch log group

alarm

health check

scaling policy

launch template

simple scaling

step scaling

cooldown period

lifecycle hook

**Q&A**

The answers to these questions appear in [Appendix A](#). For more practice with exam format questions, use the Pearson Test Prep Software Online.

- 1.** Each AMI is a \_\_\_\_\_ containing the desired software configuration.
- 2.** What is the best reason for creating golden AMIs for your web and application servers?
- 3.** Every time a Lambda function executes, you are charged based on the \_\_\_\_\_ and \_\_\_\_\_ the function uses.
- 4.** What would be a good reason to consider customizing the CloudWatch agent?
- 5.** When planning an Auto Scaling deployment, should you use a launch configuration or a launch template?
- 6.** What is the recommended EC2 Auto Scaling policy setting to start with?
- 7.** When should simple scaling policies be deployed to help improve application performance?
- 8.** How do step scaling policies help you save additional money on compute?

# Chapter 10

## Determining High-Performing Database Solutions

This chapter covers the following topics:

- [AWS Cloud Databases](#)
- [Amazon Relational Database Service](#)
- [Amazon Aurora](#)
- [Amazon DynamoDB](#)
- [Amazon ElastiCache](#)
- [Amazon Redshift](#)

This chapter covers content that's important to the following exam domain and task statement:

### **Domain 3: Design High-Performing Architectures**

Task Statement 3: Determine high-performing database solutions

Just as it's likely that you're using networking services at AWS, you are almost certainly using one or more databases for your workload data records. I can't think of a single application that's hosted in the cloud or on premises that doesn't have a