Practice Test #2 - AWS Certified Solutions Architect Associate - Results

Attempt 1

Question 1 Incorrect

An engineering team wants to examine the feasibility of the `user data` feature of Amazon EC2 for an upcoming project.

Which of the following are true about the Amazon EC2 user data configuration? (Select two)

Correct selection

By default, user data runs only during the boot cycle when you first launch an instance

By default, scripts entered as user data do not have root user privileges for executing

Your selection is correct

By default, scripts entered as user data are executed with root user privileges

When an instance is running, you can update user data by using root user credentials

Your selection is incorrect

By default, user data is executed every time an Amazon EC2 instance is re-started

Overall explanation

Correct options:

User Data is generally used to perform common automated configuration tasks and even run scripts after the instance starts. When you launch an instance in Amazon EC2, you can pass two types of user data - shell scripts and cloud-init directives. You can also pass this data into the launch wizard as plain text or as a file.

By default, scripts entered as user data are executed with root user privileges

Scripts entered as user data are executed as the root user, hence do not need the sudo command in the script. Any files you create will be owned by root; if you need non-root users to have file access, you should modify the permissions accordingly in the script.

By default, user data runs only during the boot cycle when you first launch an instance

By default, user data scripts and cloud-init directives run only during the boot cycle when you first launch an instance. You can update your configuration to ensure that your user data scripts and cloud-init directives run every time you restart your instance.

Incorrect options:

By default, user data is executed every time an Amazon EC2 instance is re-started - As discussed above, this is not a default configuration of the system. But, can be achieved by explicitly configuring the instance.

When an instance is running, you can update user data by using root user credentials - You can't change the user data if the instance is running (even by using root user credentials), but you can view it.

By default, scripts entered as user data do not have root user privileges for executing - Scripts entered as user data are executed as the root user, hence do not need the sudo command in the script.

Reference:

https://docs.aws.amazon.com/AWSEC2/latest/UserGuide/user-data.html

Domain

Design High-Performing Architectures

Question 2 **Correct**

A global media agency is developing a cultural analysis project to explore how major sports stories have evolved over the last five years. The team has collected thousands of archived news bulletins and magazine spreads stored in PDF format. These documents are rich in unstructured text and come from various sources with differing layouts and font styles. The agency wants to better understand how public tone and narrative have shifted over time. The team has chosen to use Amazon Textract for its ability to accurately extract printed and scanned text from complex PDF layouts. They need a solution that can then analyze the emotional tone and subject matter of the extracted text with the least possible operational burden, using fully managed AWS services where possible.

Which solution will best meet these requirements?

Use Amazon SageMaker to train a custom sentiment analysis model. Store the model outputs in Amazon DynamoDB for structured querying by analysts

Process the extracted output with AWS Lambda to convert the text into CSV format. Query the data using Amazon Athena and visualize it using Amazon QuickSight.

Ingest the extracted data into Amazon Redshift using AWS Glue, and use Amazon Rekognition to analyze the tone of the document layouts for sentiment classification.

Your answer is correct

Send the extracted text to Amazon Comprehend for entity detection and sentiment analysis. Store the results in Amazon S3 for further access or visualization.

Overall explanation

Correct option:

Send the extracted text to Amazon Comprehend for entity detection and sentiment analysis. Store the results in Amazon S3 for further access or visualization.

This solution offers the least operational overhead and leverages fully managed services. Amazon Textract efficiently extracts structured and unstructured text from complex PDF documents. The extracted content is then passed to Amazon Comprehend, which provides pre-trained NLP models capable of performing sentiment analysis, entity recognition, and topic classification. There is no need to build or train custom models. The results can be stored in Amazon S3, making them easily accessible for downstream processing or reporting. This approach is serverless, scalable, and ideal for unstructured data.

Amazon Textract with Amazon Comprehend:

Extracting custom entities from documents with Amazon Textract and Amazon Comprehend

by Yuan Jiang, Kashif Imran, Nishant Dhiman, Rohit Raj, and Sonali Sahu | on 22 JUL 2020 | in Amazon Comprehend, Amazon Textract, Artificial Intelligence | Permalink | Comments | Share

July 2024: This post was reviewed and updated for accuracy.

Amazon Textract is a machine learning (ML) service that makes it easy to extract text and data from scanned documents. Textract goes beyond simple optical character recognition (OCR) to identify the contents of fields in forms and information stored in tables. This allows you to use Amazon Textract to instantly "read" virtually any type of document and accurately extract text and data without needing any manual effort or custom code.

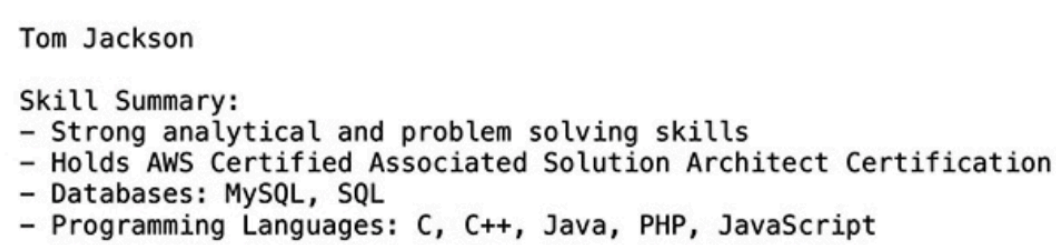
Amazon Textract has multiple applications in a variety of fields. For example, talent management companies can use Amazon Textract to automate the process of extracting a candidate's skill set. Healthcare organizations can extract patient information from documents to fulfill medical claims.

When your organization processes a variety of documents, you sometimes need to extract entities from unstructured text in the documents. A contract document, for example, can have paragraphs of text where names and other contract terms are listed in the paragraph of text instead of as a key/value or form structure. Amazon Comprehend is a natural language processing (NLP) service that can extract key phrases, places, names, organizations, events, sentiment from unstructured text, and more. With custom entity recognition, you can to identify new entity types not supported as one of the preset generic entity types. This allows you to extract business-specific entities to address your needs.

In this post, we show how to extract custom entities from scanned documents using Amazon Textract and Amazon Comprehend.

Use case overview

For this post, we process resume documents from the Resume Entities for NER dataset to get insights such as candidates' skills by automating this workflow. We use Amazon Textract to extract text from these resumes and Amazon Comprehend custom entity recognition to detect skills such as AWS, C, and C++ as custom entities. The following screenshot shows a sample input document.



```
Tom Jackson

Skill Summary:
- Strong analytical and problem solving skills
- Holds AWS Certified Associated Solution Architect Certification
- Databases: MySQL, SQL
- Programming Languages: C, C++, Java, PHP, JavaScript
```

The following screenshot shows the corresponding output generated using Amazon Textract and Amazon Comprehend.

Start Offset	End Offset	Confidence	Text	Type
9	39	0.9918249249458313	analytical and problem solving	SKILLS
8	11	0.932351291179657	AWS	SKILLS
33	41	0.9178061485290527	Solution	SKILLS
20	23	0.9526545405387878	SQL	SKILLS
2	13	0.9611709713935852	Programming	SKILLS
25	27	0.9993765354156494	C,	SKILLS
28	32	0.9994931817054749	C++,	SKILLS
33	37	0.9995038509368896	Java	SKILLS
39	42	0.9900853633880615	PHP	SKILLS

via - https://aws.amazon.com/blogs/machine-learning/extracting-custom-entities-from-documents-with-amazon-textract-and-amazon-comprehend/

Incorrect options:

Use Amazon SageMaker to train a custom sentiment analysis model. Store the model outputs in Amazon DynamoDB for structured querying by analysts - Although SageMaker provides a powerful platform for building custom ML models, using it here adds unnecessary complexity and overhead. You would need to collect training data, develop a sentiment model, and manage training infrastructure, which contradicts the requirement for low operational overhead. Also, DynamoDB is not optimized for full-text analysis, making it an awkward choice for storing sentiment-rich textual data.

Process the extracted output with AWS Lambda to convert the text into CSV format. Query the data using Amazon Athena and visualize it using Amazon QuickSight. - This solution involves multiple processing steps and intermediate transformations. Converting text to CSV via Lambda and querying it using Athena adds operational complexity. Additionally, Athena is not built for sentiment analysis—it's a query engine for structured data. QuickSight provides visualization, but without prior NLP processing (e.g., through Comprehend), it cannot extract sentiment insights from plain text.

Ingest the extracted data into Amazon Redshift using AWS Glue, and use Amazon Rekognition to analyze the tone of the document layouts for sentiment classification. - This architecture includes components (e.g., Amazon Rekognition) that are not suitable for text or sentiment analysis. Rekognition is a computer vision service, designed to analyze images and videos—not the emotional tone of text. Loading textual data into Redshift via AWS Glue also involves multiple ETL steps and schema management, which increases overhead without improving the sentiment analysis capabilities.

References:

https://docs.aws.amazon.com/textract/latest/dg/what-is.html

https://docs.aws.amazon.com/comprehend/latest/dg/what-is.html

https://aws.amazon.com/blogs/machine-learning/extracting-custom-entities-from-documents-with-amazon-textract-and-amazon-comprehend/

https://docs.aws.amazon.com/sagemaker/latest/dg/whatis.html

https://docs.aws.amazon.com/athena/latest/ug/what-is.html

https://docs.aws.amazon.com/rekognition/latest/dg/what-is.html

Domain

Design High-Performing Architectures

Question 3 Correct

An e-commerce application uses an Amazon Aurora Multi-AZ deployment for its database. While analyzing the performance metrics, the engineering team has found that the database reads are causing high input/output (I/O) and adding latency to the write requests against the database.

As an AWS Certified Solutions Architect Associate, what would you recommend to separate the read requests from the write requests?

Provision another Amazon Aurora database and link it to the primary database as a read replica

Configure the application to read from the Multi-AZ standby instance

Your answer is correct

Set up a read replica and modify the application to use the appropriate endpoint

Activate read-through caching on the Amazon Aurora database

Overall explanation

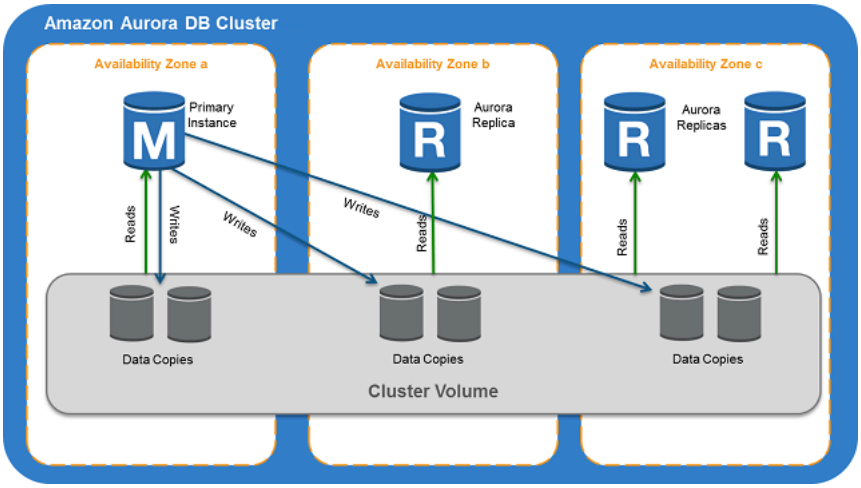
Correct option:

Set up a read replica and modify the application to use the appropriate endpoint

An Amazon Aurora DB cluster consists of one or more DB instances and a cluster volume that manages the data for those DB instances. An Aurora cluster volume is a virtual database storage volume that spans multiple Availability Zones (AZs), with each Availability Zone (AZ) having a copy of the DB cluster data. Two types of DB instances make up an Aurora DB cluster:

Primary DB instance – Supports read and write operations, and performs all of the data modifications to the cluster volume. Each Aurora DB cluster has one primary DB instance.

Aurora Replica – Connects to the same storage volume as the primary DB instance and supports only read operations. Each Aurora DB cluster can have up to 15 Aurora Replicas in addition to the primary DB instance. Aurora automatically fails over to an Aurora Replica in case the primary DB instance becomes unavailable. You can specify the failover priority for Aurora Replicas. Aurora Replicas can also offload read workloads from the primary DB instance.

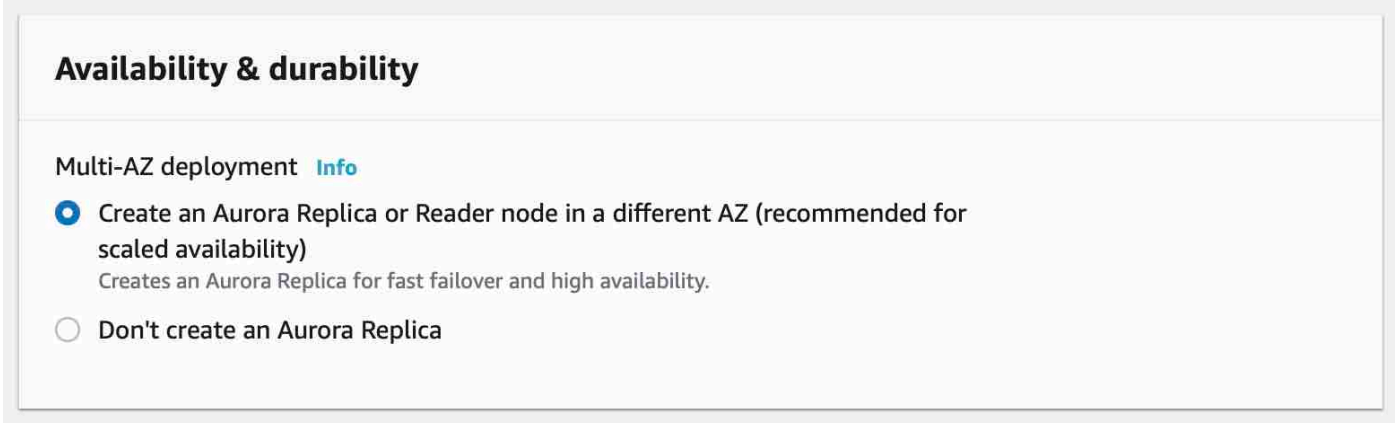


via - https://docs.aws.amazon.com/AmazonRDS/latest/AuroraUserGuide/Aurora.Overview.html

Aurora Replicas have two main purposes. You can issue queries to them to scale the read operations for your application. You typically do so by connecting to the reader endpoint of the cluster. That way, Aurora can spread the load for read-only connections across as many Aurora Replicas as you have in the cluster. Aurora Replicas also help to increase availability. If the writer instance in a cluster becomes unavailable, Aurora automatically promotes one of the reader instances to take its place as the new writer.

While setting up a Multi-AZ deployment for Aurora, you create an Aurora replica or reader node in a different Availability Zone (AZ).

Multi-AZ for Aurora:



You use the reader endpoint for read-only connections for your Aurora cluster. This endpoint uses a load-balancing mechanism to help your cluster handle a query-intensive workload. The reader endpoint is the endpoint that you supply to applications that do reporting or other read-only operations on the cluster. The reader endpoint load-balances connections to available Aurora Replicas in an Aurora DB cluster.

Types of Aurora endpoints

An endpoint is represented as an Aurora-specific URL that contains a host address and a port. The following types of endpoints are available from an Aurora DB cluster.

Cluster endpoint

A *cluster endpoint* (or *writer endpoint*) for an Aurora DB cluster connects to the current primary DB instance for that DB cluster. This endpoint is the only one that can perform write operations such as DDL statements. Because of this, the cluster endpoint is the one that you connect to when you first set up a cluster or when your cluster only contains a single DB instance.

Each Aurora DB cluster has one cluster endpoint and one primary DB instance.

You use the cluster endpoint for all write operations on the DB cluster, including inserts, updates, deletes, and DDL changes. You can also use the cluster endpoint for read operations, such as queries.

The cluster endpoint provides failover support for read/write connections to the DB cluster. If the current primary DB instance of a DB cluster fails, Aurora automatically fails over to a new primary DB instance. During a failover, the DB cluster continues to serve connection requests to the cluster endpoint from the new primary DB instance, with minimal interruption of service.

The following example illustrates a cluster endpoint for an Aurora MySQL DB cluster.

```
mydbcluster.cluster-123456789012.us-east-1.rds.amazonaws.com:3306
```

Reader endpoint

A *reader endpoint* for an Aurora DB cluster provides load-balancing support for read-only connections to the DB cluster. Use the reader endpoint for read operations, such as queries. By processing those statements on the read-only Aurora Replicas, this endpoint reduces the overhead on the primary instance. It also helps the cluster to scale the capacity to handle simultaneous `SELECT` queries, proportional to the number of Aurora Replicas in the cluster. Each Aurora DB cluster has one reader endpoint.

If the cluster contains one or more Aurora Replicas, the reader endpoint load-balances each connection request among the Aurora Replicas. In that case, you can only perform read-only statements such as `SELECT` in that session. If the cluster only contains a primary instance and no Aurora Replicas, the reader endpoint connects to the primary instance. In that case, you can perform write operations through the endpoint.

The following example illustrates a reader endpoint for an Aurora MySQL DB cluster.

```
mydbcluster.cluster-ro-123456789012.us-east-1.rds.amazonaws.com:3306
```

Custom endpoint

A *custom endpoint* for an Aurora cluster represents a set of DB instances that you choose. When you connect to the endpoint, Aurora performs load balancing and chooses one of the instances in the group to handle the connection. You define which instances this endpoint refers to, and you decide what purpose the endpoint serves.

An Aurora DB cluster has no custom endpoints until you create one. You can create up to five custom endpoints for each provisioned Aurora cluster. You can't use custom endpoints for Aurora Serverless clusters.

The custom endpoint provides load-balanced database connections based on criteria other than the read-only or read/write capability of the DB instances. For example, you might define a custom endpoint to connect to instances that use a particular AWS instance class or a particular DB parameter group. Then you might tell particular groups of users about this custom endpoint. For example, you might direct internal users to low-capacity instances for report generation or ad hoc (one-time) querying, and direct production traffic to high-capacity instances.

Because the connection can go to any DB instance that is associated with the custom endpoint, we recommend that you make sure that all the DB instances within that group share some similar characteristic. Doing so ensures that the performance, memory capacity, and so on, are consistent for everyone who connects to that endpoint.

This feature is intended for advanced users with specialized kinds of workloads where it isn't practical to keep all the Aurora Replicas in the cluster identical. With custom endpoints, you can predict the capacity of the DB instance used for each connection. When you use custom endpoints, you typically don't use the reader endpoint for that cluster.

The following example illustrates a custom endpoint for a DB instance in an Aurora MySQL DB cluster.

```
myendpoint.cluster-custom-123456789012.us-east-1.rds.amazonaws.com:3306
```

Instance endpoint

An *instance endpoint* connects to a specific DB instance within an Aurora cluster. Each DB instance in a DB cluster has its own unique instance endpoint. So there is one instance endpoint for the current primary DB instance of the DB cluster, and there is one instance endpoint for each of the Aurora Replicas in the DB cluster.

The instance endpoint provides direct control over connections to the DB cluster, for scenarios where using the cluster endpoint or reader endpoint might not be appropriate. For example, your client application might require more fine-grained load balancing based on workload type. In this case, you can configure multiple clients to connect to different Aurora Replicas in a DB cluster to distribute read workloads. For an example that uses instance endpoints to improve connection speed after a failover for Aurora PostgreSQL, see Fast failover with Amazon Aurora PostgreSQL. For an example that uses instance endpoints to improve connection speed after a failover for Aurora MySQL, see MariaDB Connector/J failover support – case Amazon Aurora.

The following example illustrates an instance endpoint for a DB instance in an Aurora MySQL DB cluster.

```
mydbinstance.123456789012.us-east-1.rds.amazonaws.com:3306
```

via - https://docs.aws.amazon.com/AmazonRDS/latest/AuroraUserGuide/Aurora.Overview.Endpoints.html

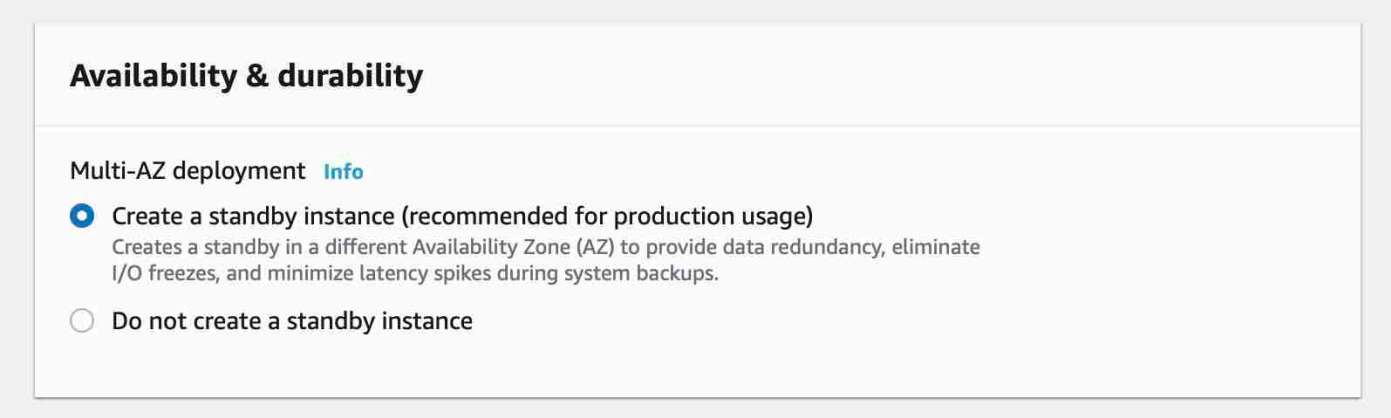
Incorrect options:

Provision another Amazon Aurora database and link it to the primary database as a read replica - You cannot provision another Aurora database and then link it as a read-replica for the

primary database. This option is ruled out.

Configure the application to read from the Multi-AZ standby instance - This option has been added as a distractor as Aurora does not have any entity called standby instance. You create a standby instance while setting up a Multi-AZ deployment for Amazon RDS and NOT for Aurora.

Multi-AZ for Amazon RDS:



Activate read-through caching on the Amazon Aurora database - Amazon Aurora does not have built-in support for read-through caching, so this option just serves as a distractor. To implement caching, you will need to integrate something like Amazon ElastiCache and that would need code changes for the application.

References:

https://docs.aws.amazon.com/AmazonRDS/latest/AuroraUserGuide/Aurora.Overview.html

https://docs.aws.amazon.com/AmazonRDS/latest/AuroraUserGuide/Concepts.AuroraHighAvailability.htm

https://docs.aws.amazon.com/AmazonRDS/latest/AuroraUserGuide/Aurora.Overview.Endpoints.html

Domain

Design Resilient Architectures

Question 4 **Incorrect**

A manufacturing company receives unreliable service from its data center provider because the company is located in an area prone to natural disasters. The company is not ready to fully migrate to the AWS Cloud, but it wants a failover environment on AWS in case the on-premises data center fails. The company runs web servers that connect to external vendors. The data available on AWS and on-premises must be uniform.

Which of the following solutions would have the LEAST amount of downtime?

Correct answer

Set up a Amazon Route 53 failover record. Run application servers on Amazon EC2 instances behind an Application Load Balancer in an Auto Scaling group. Set up AWS Storage Gateway with stored volumes to back up data to Amazon S3

Set up a Amazon Route 53 failover record. Set up an AWS Direct Connect connection between a VPC and the data center. Run application servers on Amazon EC2 in an Auto Scaling group. Run an AWS Lambda function to execute an AWS CloudFormation template to create an Application Load Balancer

Set up a Amazon Route 53 failover record. Run an AWS Lambda function to execute an AWS CloudFormation template to launch two Amazon EC2 instances. Set up AWS Storage Gateway with stored volumes to back up data to Amazon S3. Set up an AWS Direct Connect connection between a VPC and the data center

Your answer is incorrect

Set up a Amazon Route 53 failover record. Execute an AWS CloudFormation template from a script to provision Amazon EC2 instances behind an Application Load Balancer. Set up AWS Storage Gateway with stored volumes to back up data to Amazon S3

Overall explanation

Correct option:

Set up a Amazon Route 53 failover record. Run application servers on Amazon EC2 instances behind an Application Load Balancer in an Auto Scaling group. Set up AWS Storage Gateway with stored volumes to back up data to Amazon S3

If you have multiple resources that perform the same function, you can configure DNS failover so that Route 53 will route your traffic from an unhealthy resource to a healthy resource.

Elastic Load Balancing is used to automatically distribute your incoming application traffic across all the Amazon EC2 instances that you are running. You can use Elastic Load Balancing to manage incoming requests by optimally routing traffic so that no one instance is overwhelmed.

Your load balancer acts as a single point of contact for all incoming web traffic to your Auto Scaling group.

AWS Storage Gateway is a hybrid cloud storage service that gives you on-premises access to virtually unlimited cloud storage. It provides low-latency performance by caching frequently accessed data on-premises while storing data securely and durably in Amazon cloud storage services. Storage Gateway optimizes data transfer to AWS by sending only changed data and compressing data. Storage Gateway also integrates natively with Amazon S3 cloud storage which makes your data available for in-cloud processing.

Incorrect options:

Set up a Amazon Route 53 failover record. Execute an AWS CloudFormation template from a script to provision Amazon EC2 instances behind an Application Load Balancer. Set up AWS Storage Gateway with stored volumes to back up data to Amazon S3

Set up a Amazon Route 53 failover record. Run an AWS Lambda function to execute an AWS CloudFormation template to launch two Amazon EC2 instances. Set up AWS Storage Gateway with stored volumes to back up data to Amazon S3. Set up an AWS Direct Connect connection between a VPC and the data center

Set up a Amazon Route 53 failover record. Set up an AWS Direct Connect connection between a VPC and the data center. Run application servers on Amazon EC2 in an Auto Scaling group. Run an AWS Lambda function to execute an AWS CloudFormation template to create an Application Load Balancer

AWS CloudFormation is a convenient provisioning mechanism for a broad range of AWS and third-party resources. It supports the infrastructure needs of many different types of applications such as existing enterprise applications, legacy applications, applications built using a variety of AWS resources, and container-based solutions.

These three options involve AWS CloudFormation as part of the solution. Now, AWS CloudFormation takes time to provision the resources and hence is not the right solution when LEAST amount of downtime is mandated for the given use case. Therefore, these options are not the right fit for the given requirement.

References:

https://aws.amazon.com/route53/

https://aws.amazon.com/storagegateway/

Domain

Design Resilient Architectures

Question 5 Correct

A social photo-sharing web application is hosted on Amazon Elastic Compute Cloud (Amazon EC2) instances behind an Elastic Load Balancer. The app gives the users the ability to upload their photos and also shows a leaderboard on the homepage of the app. The uploaded photos are stored in Amazon Simple Storage Service (Amazon S3) and the leaderboard data is maintained in Amazon DynamoDB. The Amazon EC2 instances need to access both Amazon S3 and Amazon DynamoDB for these features.

As a solutions architect, which of the following solutions would you recommend as the MOST secure option?

Configure AWS CLI on the Amazon EC2 instances using a valid IAM user's credentials. The application code can then invoke shell scripts to access Amazon S3 and Amazon DynamoDB via AWS CLI

Your answer is correct

Attach the appropriate IAM role to the Amazon EC2 instance profile so that the instance can access Amazon S3 and Amazon DynamoDB

Encrypt the AWS credentials via a custom encryption library and save it in a secret directory on the Amazon EC2 instances. The application code can then safely decrypt the AWS credentials to make the API calls to Amazon S3 and Amazon DynamoDB

Save the AWS credentials (access key Id and secret access token) in a configuration file within the application code on the Amazon EC2 instances. Amazon EC2 instances can use these credentials to access Amazon S3 and Amazon DynamoDB

Overall explanation

Correct option:

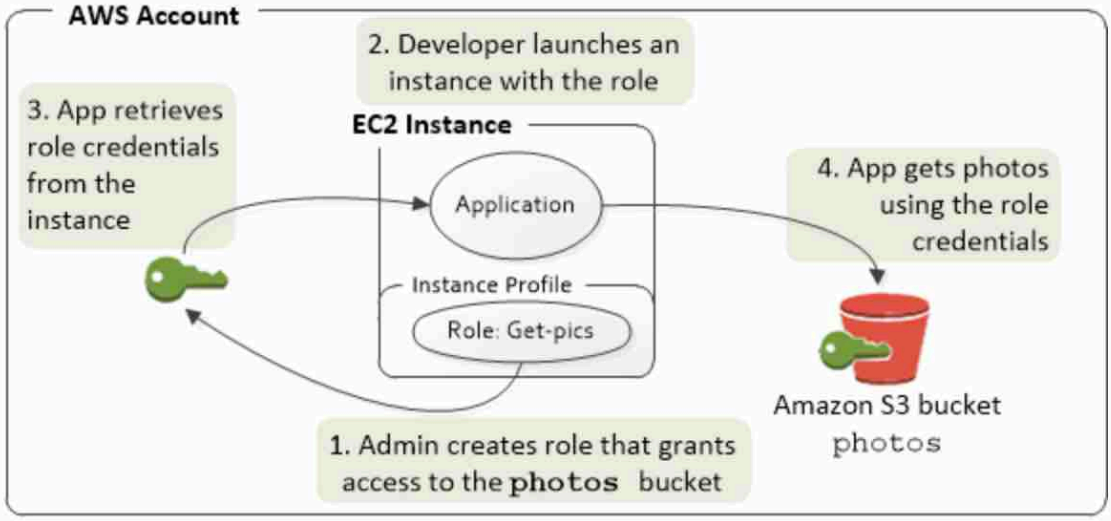
Attach the appropriate IAM role to the Amazon EC2 instance profile so that the instance can access Amazon S3 and Amazon DynamoDB

Applications that run on an Amazon EC2 instance must include AWS credentials in their AWS API requests. You could have your developers store AWS credentials directly within the Amazon EC2 instance and allow applications in that instance to use those credentials. But developers would then have to manage the credentials and ensure that they securely pass the credentials to each instance and update each Amazon EC2 instance when it's time to rotate the credentials.

Instead, you should use an IAM role to manage temporary credentials for applications that run on an Amazon EC2 instance. When you use a role, you don't have to distribute long-term credentials (such as a username and password or access keys) to an Amazon EC2 instance. The role supplies temporary permissions that applications can use when they make calls to other AWS resources. When you launch an Amazon EC2 instance, you specify an IAM role to associate with the instance. Applications that run on the instance can then use the role-supplied temporary credentials to sign API requests. Therefore, this option is correct.

How Do Roles for EC2 Instances Work?

In the following figure, a developer runs an application on an EC2 instance that requires access to the S3 bucket named `photos`. An administrator creates the `Get-pics` service role and attaches the role to the EC2 instance. The role includes a permissions policy that grants read-only access to the specified S3 bucket. It also includes a trust policy that allows the EC2 instance to assume the role and retrieve the temporary credentials. When the application runs on the instance, it can use the role's temporary credentials to access the photos bucket. The administrator doesn't have to grant the developer permission to access the photos bucket, and the developer never has to share or manage credentials.



via - https://docs.aws.amazon.com/IAM/latest/UserGuide/id_roles_use_switch-role-ec2.html

Incorrect options:

Save the AWS credentials (access key Id and secret access token) in a configuration file within the application code on the Amazon EC2 instances. Amazon EC2 instances can use these credentials to access Amazon S3 and Amazon DynamoDB

Configure AWS CLI on the Amazon EC2 instances using a valid IAM user's credentials. The application code can then invoke shell scripts to access Amazon S3 and Amazon DynamoDB via AWS CLI

Encrypt the AWS credentials via a custom encryption library and save it in a secret directory on the Amazon EC2 instances. The application code can then safely decrypt the AWS credentials to make the API calls to Amazon S3 and Amazon DynamoDB

Keeping the AWS credentials (encrypted or plain text) on the Amazon EC2 instance is a bad security practice, therefore these three options using the AWS credentials are incorrect.

Reference:

https://docs.aws.amazon.com/IAM/latest/UserGuide/id_roles_use_switch-role-ec2.html

Domain

Design Secure Architectures

Question 6 Correct

Your company has a monthly big data workload, running for about 2 hours, which can be efficiently distributed across multiple servers of various sizes, with a variable number of CPUs. The solution for the workload should be able to withstand server failures.

Which is the MOST cost-optimal solution for this workload?

Your answer is correct

Run the workload on a Spot Fleet

Run the workload on Reserved Instances (RI)

Run the workload on Dedicated Hosts

Run the workload on Spot Instances

Overall explanation

Correct option:

Run the workload on a Spot Fleet

The Spot Fleet selects the Spot Instance pools that meet your needs and launches Spot Instances to meet the target capacity for the fleet. By default, Spot Fleets are set to maintain target capacity by launching replacement instances after Spot Instances in the fleet are terminated.

A Spot Instance is an unused Amazon EC2 instance that is available for less than the On-Demand price. Spot Instances provide great cost efficiency, but we need to select an instance type in advance. In this case, we want to use the most cost-optimal option and leave the selection of the cheapest spot instance to a Spot Fleet request, which can be optimized with the `lowestPrice` strategy. So this is the correct option.

Key Spot Instance Concepts:

Concepts

Before you get started with Spot Instances, you should be familiar with the following concepts:

- *Spot Instance pool* – A set of unused EC2 instances with the same instance type (for example, `m5.large`), operating system, Availability Zone, and network platform.
- *Spot price* – The current price of a Spot Instance per hour.
- *Spot Instance request* – Provides the maximum price per hour that you are willing to pay for a Spot Instance. If you don't specify a maximum price, the default maximum price is the On-Demand price. When the maximum price per hour for your request exceeds the Spot price, Amazon EC2 fulfills your request if capacity is available. A Spot Instance request is either *one-time* or *persistent*. Amazon EC2 automatically resubmits a persistent Spot request after the Spot Instance associated with the request is terminated. Your Spot Instance request can optionally specify a duration for the Spot Instances.
- *Spot Fleet* – A set of Spot Instances that is launched based on criteria that you specify. The Spot Fleet selects the Spot Instance pools that meet your needs and launches Spot Instances to meet the target capacity for the fleet. By default, Spot Fleets are set to *maintain* target capacity by launching replacement instances after Spot Instances in the fleet are terminated. You can submit a Spot Fleet as a one-time *request*, which does not persist after the instances have been terminated. You can include On-Demand Instance requests in a Spot Fleet request.
- *Spot Instance interruption* – Amazon EC2 terminates, stops, or hibernates your Spot Instance when the Spot price exceeds the maximum price for your request or capacity is no longer available. Amazon EC2 provides a Spot Instance interruption notice, which gives the instance a two-minute warning before it is interrupted.

via - https://docs.aws.amazon.com/AWSEC2/latest/UserGuide/using-spot-instances.html

Incorrect options:

Run the workload on Spot Instances - A Spot Instance is an unused Amazon EC2 instance that is available for less than the On-Demand price. Because Spot Instances enable you to request unused Amazon EC2 instances at steep discounts, you can lower your Amazon EC2 costs significantly. The hourly price for a Spot Instance is called a Spot price. Only spot fleets can maintain target capacity by launching replacement instances after Spot Instances in the fleet are terminated, so spot instances, by themselves, are not the right fit for this use-case.

Run the workload on Reserved Instances (RI) - Reserved Instances are less cost-optimized than Spot Instances, and most efficient when used continuously. Here the workload is once a month, so this is not efficient.

Run the workload on Dedicated Hosts - Amazon EC2 Dedicated Hosts allow you to use your eligible software licenses from vendors such as Microsoft and Oracle on Amazon EC2 so that you get the flexibility and cost-effectiveness of using your licenses, but with the resiliency, simplicity, and elasticity of AWS. An Amazon EC2 Dedicated Host is a physical server fully dedicated for your use, so you can help address corporate compliance requirement. They're not particularly cost-efficient. So this option is not correct.

References:

https://docs.aws.amazon.com/AWSEC2/latest/UserGuide/using-spot-instances.html

https://docs.aws.amazon.com/AWSEC2/latest/UserGuide/spot-fleet.html#spot-fleet-allocation-strategy

Domain

Design Cost-Optimized Architectures

Question 7 Correct

Your company has an on-premises Distributed File System Replication (DFSR) service to keep files synchronized on multiple Windows servers, and would like to migrate to AWS cloud.

What do you recommend as a replacement for the DFSR?

Your answer is correct

Amazon FSx for Windows File Server

Amazon Elastic File System (Amazon EFS)

Amazon Simple Storage Service (Amazon S3)

Amazon FSx for Lustre

Overall explanation

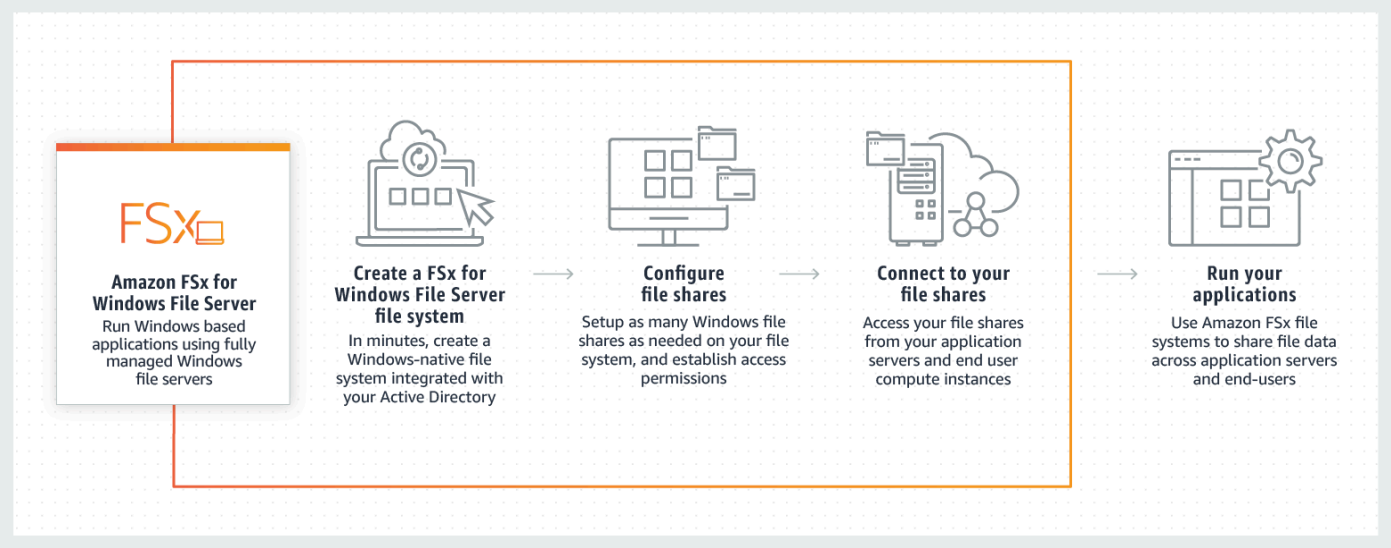
Correct option:

Amazon FSx for Windows File Server

Amazon FSx for Windows File Server provides fully managed, highly reliable file storage that is accessible over the industry-standard Service Message Block (SMB) protocol. It is built on Windows Server, delivering a wide range of administrative features such as user quotas, end-user file restore, and Microsoft Active Directory (AD) integration. The Distributed File System Replication (DFSR) service is a new multi-master replication engine that is used to keep folders synchronized on multiple servers. Amazon FSx supports the use of Microsoft's Distributed File System (DFS) to organize shares into a single folder structure up to hundreds of PB in size.

Amazon FSx for Windows is a perfect distributed file system, with replication capability, and can be mounted on Windows.

How Amazon FSx for Windows Works:



via - https://aws.amazon.com/fsx/windows/

Incorrect options:

Amazon FSx for Lustre - Amazon FSx for Lustre makes it easy and cost-effective to launch and run the world's most popular high-performance file system. It is used for workloads such as machine learning, high-performance computing (HPC), video processing, and financial modeling. The open-source Lustre file system is designed for applications that require fast storage – where you want your storage to keep up with your compute. Amazon FSx enables you to use Lustre file systems for any workload where storage speed matters. FSx for Lustre integrates with Amazon S3, making it easy to process data sets with the Lustre file system. Amazon FSx for Lustre is for Linux only, so this option is incorrect.

Amazon Elastic File System (Amazon EFS) - Amazon Elastic File System (Amazon EFS) provides a simple, scalable, fully managed elastic NFS file system for use with AWS Cloud services and on-premises resources. It is built to scale on-demand to petabytes without disrupting applications, growing and shrinking automatically as you add and remove files, eliminating the need to provision and manage capacity to accommodate growth. Amazon EFS is a network file system but for Linux only, so this option is incorrect.

Amazon Simple Storage Service (Amazon S3) - Amazon Simple Storage Service (Amazon S3) is an object storage service that offers industry-leading scalability, data availability, security, and performance. Amazon S3 cannot be mounted as a file system on Windows, so this option is incorrect.

References:

https://docs.microsoft.com/en-us/previous-versions/windows/desktop/dfsr/dfsr-overview

https://aws.amazon.com/fsx/windows/

https://aws.amazon.com/fsx/lustre/

Domain

Design High-Performing Architectures

Question 8 **Correct**

An application runs big data workloads on Amazon Elastic Compute Cloud (Amazon EC2) instances. The application runs 24x7 all round the year and needs at least 20 instances to maintain a minimum acceptable performance threshold and the application needs 300 instances to handle spikes in the workload. Based on historical workloads processed by the application, it needs 80 instances 80% of the time.

As a solutions architect, which of the following would you recommend as the MOST cost-optimal solution so that it can meet the workload demand in a steady state?

Purchase 20 on-demand instances. Use Auto Scaling Group to provision the remaining instances as spot instances per the workload demand

Purchase 80 spot instances. Use Auto Scaling Group to provision the remaining instances as on-demand instances per the workload demand

Purchase 80 on-demand instances. Provision additional on-demand and spot instances per the workload demand (Use Auto Scaling Group with launch template to provision the mix of on-demand and spot instances)

Your answer is correct

Purchase 80 reserved instances (RIs). Provision additional on-demand and spot instances per the workload demand (Use Auto Scaling Group with launch template to provision the mix of on-demand and spot instances)

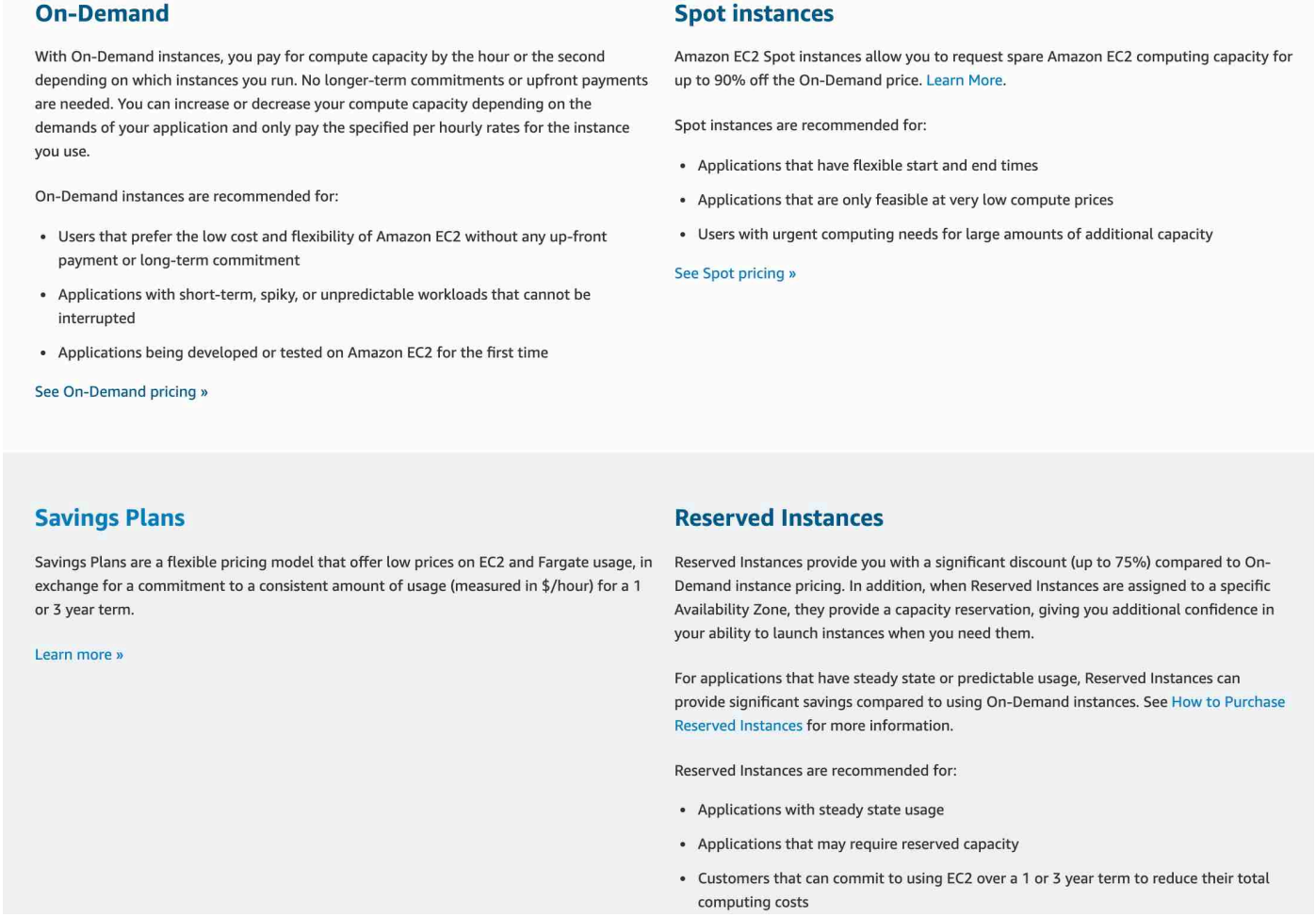
Overall explanation

Correct option:

Purchase 80 reserved instances (RIs). Provision additional on-demand and spot instances per the workload demand (Use Auto Scaling Group with launch template to provision the mix of on-demand and spot instances)

As the steady-state workload demand is 80 instances, we can save on costs by purchasing 80 reserved instances. Based on additional workload demand, we can specify a mix of on-demand and spot instances using Application Load Balancer with a launch template to provision the mix of on-demand and spot instances.

Please see this detailed overview of various types of Amazon EC2 instances from a pricing perspective:



via - https://aws.amazon.com/ec2/pricing/

Incorrect options:

Purchase 20 on-demand instances. Use Auto Scaling Group to provision the remaining instances as spot instances per the workload demand - Provisioning 20 on-demand instances implies that there would be a shortfall of 60 instances 80% of the time. Provisioning all of these 60 instances as spot instances is highly risky as there is no guarantee regarding the availability of the spot instances, which means we may not even meet the steady-state requirement for the workload, so this option is incorrect.

Purchase 80 on-demand instances. Provision additional on-demand and spot instances per the workload demand (Use Auto Scaling Group with launch template to provision the mix of on-demand and spot instances) - Provisioning 80 on-demand instances would end up costlier than the option where we provision 80 reserved instances. So this option is ruled out.

Purchase 80 spot instances. Use Auto Scaling Group to provision the remaining instances as on-demand instances per the workload demand - The option to purchase 80 spot instances is incorrect, as there is no guarantee regarding the availability of the spot instances, which means we may not even meet the steady-state workload.

Reference:

https://aws.amazon.com/ec2/pricing/

Domain

Design Cost-Optimized Architectures

Question 9 Correct

A big data consulting firm needs to set up a data lake on Amazon S3 for a Health-Care client. The data lake is split in raw and refined zones. For compliance reasons, the source data needs to be kept for a minimum of 5 years. The source data arrives in the raw zone and is then processed via an AWS Glue based extract, transform, and load (ETL) job into the refined zone. The business analysts run ad-hoc queries only on the data in the refined zone using Amazon Athena. The team is concerned about the cost of data storage in both the raw and refined zones as the data is increasing at a rate of 1 terabyte daily in each zone.

As a solutions architect, which of the following would you recommend as the MOST cost-optimal solution? (Select two)

Setup a lifecycle policy to transition the refined zone data into Amazon S3 Glacier Deep Archive after 1 day of object creation

Use AWS Glue ETL job to write the transformed data in the refined zone using CSV format

Your selection is correct

Use AWS Glue ETL job to write the transformed data in the refined zone using a compressed file format

Your selection is correct

Setup a lifecycle policy to transition the raw zone data into Amazon S3 Glacier Deep Archive after 1 day of object creation

Create an AWS Lambda function based job to delete the raw zone data after 1 day

Overall explanation

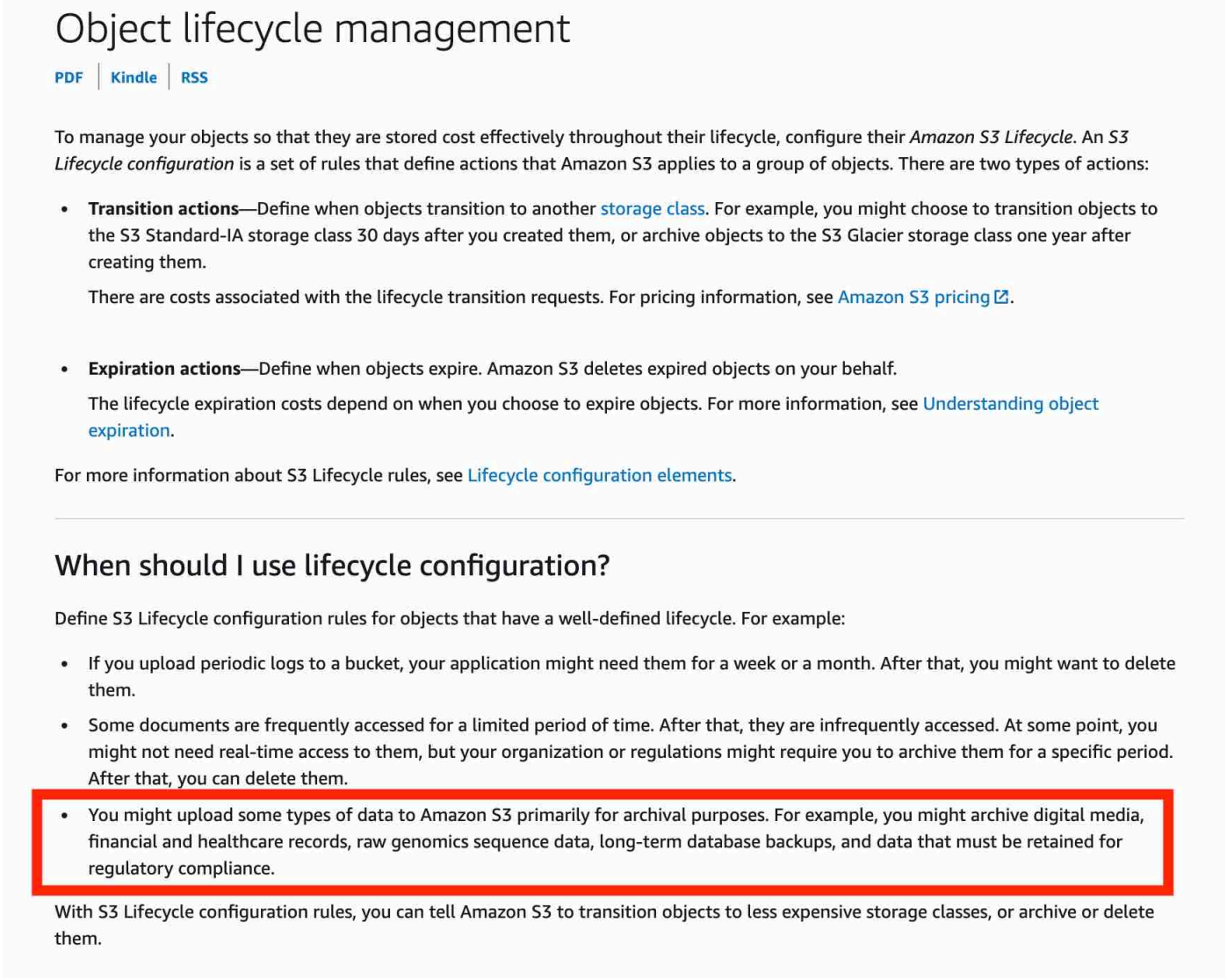
Correct options:

Setup a lifecycle policy to transition the raw zone data into Amazon S3 Glacier Deep Archive after 1 day of object creation

You can manage your objects so that they are stored cost-effectively throughout their lifecycle by configuring their Amazon S3 Lifecycle. An S3 Lifecycle configuration is a set of rules that define actions that Amazon S3 applies to a group of objects. For example, you might choose to transition objects to the Amazon S3 Standard-IA storage class 30 days after you created them, or archive objects to the Amazon S3 Glacier storage class one year after creating them.

For the given use-case, the raw zone consists of the source data, so it cannot be deleted due to compliance reasons. Therefore, you should use a lifecycle policy to transition the raw zone data into Amazon S3 Glacier Deep Archive after 1 day of object creation.

Please read more about Amazon S3 Object Lifecycle Management:



Object lifecycle management

PDF | Kindle | RSS

To manage your objects so that they are stored cost effectively throughout their lifecycle, configure their *Amazon S3 Lifecycle*. An *S3 Lifecycle configuration* is a set of rules that define actions that Amazon S3 applies to a group of objects. There are two types of actions:

- **Transition actions**—Define when objects transition to another storage class. For example, you might choose to transition objects to the S3 Standard-IA storage class 30 days after you created them, or archive objects to the S3 Glacier storage class one year after creating them.

  There are costs associated with the lifecycle transition requests. For pricing information, see Amazon S3 pricing.

- **Expiration actions**—Define when objects expire. Amazon S3 deletes expired objects on your behalf.

  The lifecycle expiration costs depend on when you choose to expire objects. For more information, see Understanding object expiration.

For more information about S3 Lifecycle rules, see Lifecycle configuration elements.

When should I use lifecycle configuration?

Define S3 Lifecycle configuration rules for objects that have a well-defined lifecycle. For example:

- If you upload periodic logs to a bucket, your application might need them for a week or a month. After that, you might want to delete them.
- Some documents are frequently accessed for a limited period of time. After that, they are infrequently accessed. At some point, you might not need real-time access to them, but your organization or regulations might require you to archive them for a specific period. After that, you can delete them.
- You might upload some types of data to Amazon S3 primarily for archival purposes. For example, you might archive digital media, financial and healthcare records, raw genomics sequence data, long-term database backups, and data that must be retained for regulatory compliance.

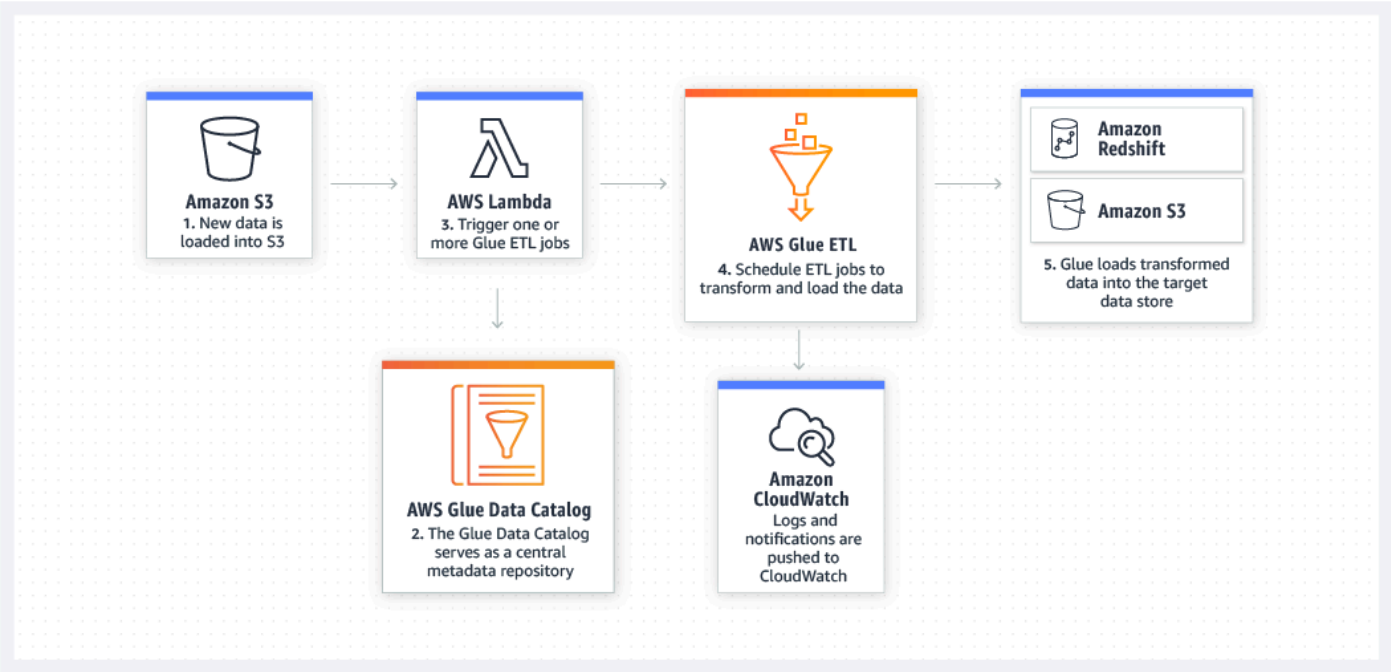
With S3 Lifecycle configuration rules, you can tell Amazon S3 to transition objects to less expensive storage classes, or archive or delete them.

via - https://docs.aws.amazon.com/AmazonS3/latest/dev/object-lifecycle-mgmt.html

Use AWS Glue ETL job to write the transformed data in the refined zone using a compressed file format

AWS Glue is a fully managed extract, transform, and load (ETL) service that makes it easy for customers to prepare and load their data for analytics. You cannot transition the refined zone data into Amazon S3 Glacier Deep Archive because it is used by the business analysts for ad-hoc querying. Therefore, the best optimization is to have the refined zone data stored in a compressed format via the Glue job. The compressed data would reduce the storage cost incurred on the data in the refined zone.

Please see this example for a AWS Glue ETL Pipeline:



via - https://aws.amazon.com/glue/

Incorrect options:

Create an AWS Lambda function based job to delete the raw zone data after 1 day - As mentioned in the use-case, the source data needs to be kept for a minimum of 5 years for compliance reasons. Therefore the data in the raw zone cannot be deleted after 1 day.

Setup a lifecycle policy to transition the refined zone data into Amazon S3 Glacier Deep Archive after 1 day of object creation - You cannot transition the refined zone data into Amazon S3 Glacier Deep Archive because it is used by the business analysts for ad-hoc querying. Hence this option is incorrect.

Use AWS Glue ETL job to write the transformed data in the refined zone using CSV format - It is cost-optimal to write the data in the refined zone using a compressed format instead of CSV format. The compressed data would reduce the storage cost incurred on the data in the refined zone. So, this option is incorrect.

References:

https://docs.aws.amazon.com/AmazonS3/latest/dev/object-lifecycle-mgmt.html

https://aws.amazon.com/glue/

Domain

Design High-Performing Architectures

Question 10 **Incorrect**

A company is developing a global healthcare application that requires the least possible latency for database read/write operations from users in several geographies across the world. The company has

hired you as an AWS Certified Solutions Architect Associate to build a solution using Amazon Aurora that offers an effective recovery point objective (RPO) of seconds and a recovery time objective (RTO) of a minute.

Which of the following options would you recommend?

Set up an Amazon Aurora provisioned Database cluster

Correct answer

Set up an Amazon Aurora Global Database cluster

Your answer is incorrect

Set up an Amazon Aurora serverless Database cluster

Set up an Amazon Aurora multi-master Database cluster

Overall explanation

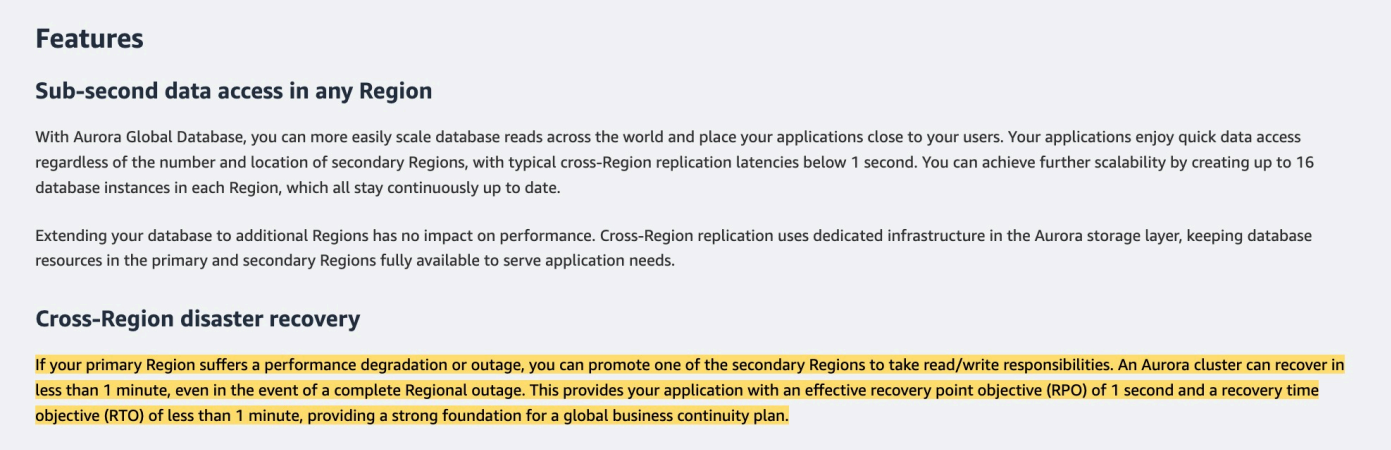
Correct option:

Set up an Amazon Aurora Global Database cluster

Amazon Aurora Global Database is designed for globally distributed applications, allowing a single Amazon Aurora database to span multiple AWS Regions. It replicates your data with no

impact on database performance, enables fast local reads with low latency in each Region, and provides disaster recovery from Region-wide outages.

If your primary Region suffers a performance degradation or outage, you can promote one of the secondary Regions to take read/write responsibilities. An Aurora cluster can recover in less than 1 minute, even in the event of a complete Regional outage. This provides your application with an effective recovery point objective (RPO) of 1 second and a recovery time objective (RTO) of less than 1 minute, providing a strong foundation for a global business continuity plan.



Features

Sub-second data access in any Region

With Aurora Global Database, you can more easily scale database reads across the world and place your applications close to your users. Your applications enjoy quick data access regardless of the number and location of secondary Regions, with typical cross-Region replication latencies below 1 second. You can achieve further scalability by creating up to 16 database instances in each Region, which all stay continuously up to date.

Extending your database to additional Regions has no impact on performance. Cross-Region replication uses dedicated infrastructure in the Aurora storage layer, keeping database resources in the primary and secondary Regions fully available to serve application needs.

Cross-Region disaster recovery

If your primary Region suffers a performance degradation or outage, you can promote one of the secondary Regions to take read/write responsibilities. An Aurora cluster can recover in less than 1 minute, even in the event of a complete Regional outage. This provides your application with an effective recovery point objective (RPO) of 1 second and a recovery time objective (RTO) of less than 1 minute, providing a strong foundation for a global business continuity plan.

via - https://aws.amazon.com/rds/aurora/global-database/

Incorrect options:

Set up an Amazon Aurora serverless Database cluster

Set up an Amazon Aurora provisioned Database cluster

Both these options work in a single AWS Region, so these options are incorrect.

Set up an Amazon Aurora multi-master Database cluster - AWS does not offer the multi-master feature in a Aurora database cluster, so this option acts as a distractor.

Reference:

https://aws.amazon.com/rds/aurora/global-database/

Domain

Design High-Performing Architectures

Question 11 **Correct**

A media production studio is building a content rendering and editing platform on AWS. The editing workstations and rendering tools require access to shared files over the SMB (Server Message Block) protocol. The studio wants a managed storage solution that is simple to set up, integrates easily with SMB clients, and minimizes ongoing operational tasks.

Which solution will best meet the requirements with the LEAST administrative overhead?

Your answer is correct

Provision an Amazon FSx for Windows File Server file system. Mount the file system using the SMB protocol on the media servers

Set up an AWS Storage Gateway Volume Gateway in cached volume mode. Attach the volume as an iSCSI device to the application server and configure a file system with SMB sharing enabled

Use Amazon S3 with Transfer Acceleration enabled. Configure the application to upload and download files over HTTPS using signed URLs

Launch an Amazon EC2 Windows instance and manually configure a Windows file share. Use this instance to serve SMB access to application clients

Overall explanation

Correct option:

Provision an Amazon FSx for Windows File Server file system. Mount the file system using the SMB protocol on the media servers

Amazon FSx for Windows File Server is the most appropriate solution for workloads that require SMB (Server Message Block) protocol access with minimal administrative effort. FSx is a fully managed file storage service built on Windows Server, natively supporting SMB protocol, NTFS file systems, DFS namespaces, and Windows ACLs. This makes it ideal for applications like media editing platforms where Windows-based tools need seamless access to shared storage. Unlike

self-managed EC2-based file servers, FSx handles patching, backups, availability, and performance optimization automatically. It integrates easily with AWS Managed Microsoft AD or self-managed AD, enabling fine-grained access control through Active Directory groups and users. Its scalability, native SMB support, and hands-off management model ensure the least operational overhead while meeting all the technical and security requirements of the use case.

Incorrect options:

Use Amazon S3 with Transfer Acceleration enabled. Configure the application to upload and download files over HTTPS using signed URLs - While Amazon S3 is highly durable and accessible globally, it does not support SMB protocol, nor does it act as a traditional file system for concurrent file access by multiple clients. S3 is best suited for object-based storage, not file sharing. Transfer Acceleration may improve upload/download performance, but the manual handling of objects and URLs introduces high operational complexity and doesn't meet the requirement for seamless SMB-based access.

Launch an Amazon EC2 Windows instance and manually configure a Windows file share. Use this instance to serve SMB access to application clients - Although EC2 Windows instances can support SMB by configuring a file server role, this is a self-managed solution that requires the studio to handle OS maintenance, backups, patching, fault tolerance, and scaling. Compared to Amazon FSx, this approach adds significant administrative overhead and is not as resilient or cost-effective. It also lacks built-in integration with AWS-managed features like scheduled backups or automatic failover.

Set up an AWS Storage Gateway Volume Gateway in cached volume mode. Attach the volume as an iSCSI device to the application server and configure a file system with SMB sharing enabled - While AWS Storage Gateway Volume Gateway provides block storage access over iSCSI, it does not natively support the SMB protocol required by the application. To enable SMB access, you would need to set up a file system manually on the connected server and configure SMB sharing at the OS level, essentially turning the application server into a self-managed file server. This not only increases administrative overhead but also places the burden of maintenance, backups, and patching on the user, which contradicts the requirement of a low-maintenance solution. Additionally, Volume Gateway is best suited for hybrid on-premises-to-cloud block storage use cases, not fully cloud-based file sharing with SMB.

References:

https://docs.aws.amazon.com/fsx/latest/WindowsGuide/what-is.html

https://docs.aws.amazon.com/AmazonS3/latest/userguide/transfer-acceleration.html

https://docs.aws.amazon.com/storagegateway/latest/vgw/WhatIsStorageGateway.html

Domain

Design High-Performing Architectures

Question 12 Correct

What does this IAM policy do?

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "Mystery Policy",
      "Action": [
        "ec2:RunInstances"
      ],
      "Effect": "Allow",
      "Resource": "*",
      "Condition": {
        "StringEquals": {
          "aws:RequestedRegion": "eu-west-1"
        }
      }
    }
  ]
}
```

It allows running Amazon EC2 instances in the `eu-west-1` region, when the API call is made from the `eu-west-1` region

It allows running Amazon EC2 instances in any region when the API call is originating from the `eu-west-1` region

Your answer is correct

It allows running Amazon EC2 instances only in the `eu-west-1` region, and the API call can be made from anywhere in the world

It allows running Amazon EC2 instances anywhere but in the `eu-west-1` region

Overall explanation

Correct option:

It allows running Amazon EC2 instances only in the `eu-west-1` region, and the API call can be made from anywhere in the world

You manage access in AWS by creating policies and attaching them to IAM identities (users, groups of users, or roles) or AWS resources. A policy is an object in AWS that, when associated with an identity or resource, defines their permissions. AWS evaluates these policies when an IAM principal (user or role) makes a request. Permissions in the policies determine whether the request is allowed or denied. Most policies are stored in AWS as JSON documents. AWS supports six types of policies: identity-based policies, resource-based policies, permissions boundaries, Organizations service control policy (SCPs), access control lists (ACLs), and session policies.

You can use the `aws:RequestedRegion` key to compare the AWS Region that was called in the request with the Region that you specify in the policy. You can use this global condition key to control which Regions can be requested.

`aws:RequestedRegion` represents the target of the API call. So in this example, we can only launch an Amazon EC2 instance in `eu-west-1`, and we can do this API call from anywhere.

Incorrect options:

It allows running Amazon EC2 instances anywhere but in the `eu-west-1` region

It allows running Amazon EC2 instances in any region when the API call is originating from the `eu-west-1` region

It allows running Amazon EC2 instances in the `eu-west-1` region, when the API call is made from the `eu-west-1` region

These three options contradict the earlier details provided in the explanation. To summarize, `aws:RequestedRegion` represents the target of the API call. So, we can only launch an

Amazon EC2 instance in `eu-west-1` region and we can do this API call from anywhere. Hence these options are incorrect.

References:

https://docs.aws.amazon.com/IAM/latest/UserGuide/access_policies.html

https://docs.aws.amazon.com/IAM/latest/UserGuide/reference_policies_condition-keys.html

Domain

Design Secure Architectures

Question 13 Correct

Which of the following IAM policies provides read-only access to the Amazon S3 bucket `mybucket` and its content?

Your answer is correct

```
{
   "Version":"2012-10-17",
   "Statement":[
      {
         "Effect":"Allow",
         "Action":[
            "s3:ListBucket"
         ],
         "Resource":"arn:aws:s3:::mybucket"
      },
      {
         "Effect":"Allow",
         "Action":[
            "s3:GetObject"
         ],
         "Resource":"arn:aws:s3:::mybucket/*"
      }
   ]
}
```

```
{
   "Version":"2012-10-17",
   "Statement":[
      {
         "Effect":"Allow",
         "Action":[
            "s3:ListBucket",
            "s3:GetObject"
```

```
            ],
            "Resource":"arn:aws:s3:::mybucket/*"
        }
    ]
}
```

```
{
    "Version":"2012-10-17",
    "Statement":[
        {
            "Effect":"Allow",
            "Action":[
                "s3:ListBucket"
            ],
            "Resource":"arn:aws:s3:::mybucket/*"
        },
        {
            "Effect":"Allow",
            "Action":[
                "s3:GetObject"
            ],
            "Resource":"arn:aws:s3:::mybucket"
        }
    ]
}
```

```
{
    "Version":"2012-10-17",
    "Statement":[
        {
            "Effect":"Allow",
            "Action":[
                "s3:ListBucket",
                "s3:GetObject"
            ],
            "Resource":"arn:aws:s3:::mybucket"
        }
    ]
}
```

Overall explanation

Correct option:

```
{
    "Version":"2012-10-17",
    "Statement":[
```

```
      {
         "Effect":"Allow",
         "Action":[
            "s3:ListBucket"
         ],
         "Resource":"arn:aws:s3:::mybucket"
      },
      {
         "Effect":"Allow",
         "Action":[
            "s3:GetObject"
         ],
         "Resource":"arn:aws:s3:::mybucket/*"
      }
   ]
}
```

You manage access in AWS by creating policies and attaching them to IAM identities (users, groups of users, or roles) or AWS resources. A policy is an object in AWS that, when associated with an identity or resource, defines their permissions. AWS evaluates these policies when an IAM principal (user or role) makes a request. Permissions in the policies determine whether the request is allowed or denied. Most policies are stored in AWS as JSON documents. AWS supports six types of policies: identity-based policies, resource-based policies, permissions boundaries, service control policy (SCP)of AWS Organizations, access control list (ACL), and session policies.

`s3:ListBucket` is applied to buckets, so the ARN is in the form `"Resource":"arn:aws:s3:::mybucket"`, without a trailing `/` `s3:GetObject` is applied to objects within the bucket, so the ARN is in the form `"Resource":"arn:aws:s3:::mybucket/*"`, with a trailing `/*` to indicate all objects within the bucket `mybucket`

Therefore, this is the correct option.

Incorrect options:

```
{
   "Version":"2012-10-17",
   "Statement":[
      {
         "Effect":"Allow",
         "Action":[
            "s3:ListBucket",
            "s3:GetObject"
         ],
         "Resource":"arn:aws:s3:::mybucket"
      }
   ]
}
```

This option is incorrect as it provides read-only access only to the bucket, not its contents.

```
{
   "Version":"2012-10-17",
   "Statement":[
      {
         "Effect":"Allow",
         "Action":[
            "s3:ListBucket",
            "s3:GetObject"
         ],
         "Resource":"arn:aws:s3:::mybucket/*"
      }
   ]
}
```

This option is incorrect as it provides read-only access only to the objects within the bucket and it does not provide listBucket permissions to the bucket itself.

```
{
   "Version":"2012-10-17",
   "Statement":[
      {
         "Effect":"Allow",
         "Action":[
            "s3:ListBucket"
         ],
         "Resource":"arn:aws:s3:::mybucket/*"
      },
      {
         "Effect":"Allow",
         "Action":[
            "s3:GetObject"
         ],
         "Resource":"arn:aws:s3:::mybucket"
      }
   ]
}
```

This option is incorrect as it provides listing access only to the bucket contents.

References:

https://aws.amazon.com/blogs/security/writing-iam-policies-how-to-grant-access-to-an-amazon-s3-bucket/

https://docs.aws.amazon.com/IAM/latest/UserGuide/access_policies.html

Domain

Design Secure Architectures

Question 14 Incorrect

An enterprise is building a secure business intelligence API using Amazon API Gateway to serve internal users with confidential analytics data. The API must be accessible only from a set of trusted IP addresses that are part of the organization's internal network ranges. No external IP traffic should be able to invoke the API. A solutions architect must design this access control mechanism with the least operational complexity.

What should the architect do to meet these requirements?

Correct answer

Create a resource policy for the API Gateway API that explicitly denies access to all IP addresses except those listed in an allow list

Deploy the API Gateway resource to an on-premises server using AWS Outposts. Apply host-based firewall rules to filter allowed IPs

Deploy the API Gateway as a regional API in a public subnet and associate the subnet with a security group that permits inbound traffic only from trusted IP ranges

Your answer is incorrect

Modify the security group that is attached to API Gateway to allow only traffic from specific IP addresses

Overall explanation

Correct option:

Create a resource policy for the API Gateway API that explicitly denies access to all IP addresses except those listed in an allow list

This is the most appropriate and secure solution to restrict access to an Amazon API Gateway endpoint based on IP addresses. API Gateway supports resource policies, which are IAM-style JSON policies that you attach directly to your REST or HTTP APIs. These policies can use IpAddress and NotIpAddress conditions to enforce fine-grained network controls. By configuring a policy that explicitly denies all IPs except for the trusted internal IP ranges, the company can ensure that only requests originating from its internal network are allowed to invoke the API. This approach provides centralized, declarative access control without the need for additional infrastructure changes, and it aligns with AWS best practices for managing access to public APIs that carry sensitive data.

Incorrect options:

Deploy the API Gateway as a regional API in a public subnet and associate the subnet with a security group that permits inbound traffic only from trusted IP ranges - API Gateway is a fully managed service and does not run inside user-created subnets — public or private — so you cannot associate it with a subnet or a security group. Security groups are used for Amazon VPC-based resources such as EC2 instances, not API Gateway endpoints. While you can use regional APIs to serve clients directly, IP-based access control must be implemented using API Gateway resource policies, not VPC security groups or subnet-level controls. This makes the solution invalid for restricting access to specific IPs.

Deploy the API Gateway resource to an on-premises server using AWS Outposts. Apply host-based firewall rules to filter allowed IPs - Deploying the API to an on-premises server via AWS Outposts is not only unnecessary but also misaligned with the use of API Gateway, which is a fully managed AWS service that resides in the cloud. Using Outposts would dramatically increase cost and operational complexity and would not apply to the native API Gateway access control mechanisms such as resource policies or usage plans.

Modify the security group that is attached to API Gateway to allow only traffic from specific IP addresses - API Gateway does not use security groups because it is a managed service that is not deployed into a VPC by default. Security groups apply to resources like EC2 instances, RDS databases, or Lambda functions within a VPC. Therefore, this option is not valid for controlling inbound access to an API Gateway endpoint.

References:

https://docs.aws.amazon.com/apigateway/latest/developerguide/apigateway-resource-policies.html

https://docs.aws.amazon.com/outposts/latest/userguide/what-is-outposts.html

Domain

Design Secure Architectures

Question 15 Correct

You are establishing a monitoring solution for desktop systems, that will be sending telemetry data into AWS every 1 minute. Data for each system must be processed in order, independently, and you would like to scale the number of consumers to be possibly equal to the number of desktop systems that are being monitored.

What do you recommend?

Your answer is correct

Use an Amazon Simple Queue Service (Amazon SQS) FIFO (First-In-First-Out) queue, and make sure the telemetry data is sent with a Group ID attribute representing the value of the Desktop ID

Use an Amazon Simple Queue Service (Amazon SQS) FIFO (First-In-First-Out) queue, and send the telemetry data as is

Use an Amazon Kinesis Data Stream, and send the telemetry data with a Partition ID that uses the value of the Desktop ID

Use an Amazon Simple Queue Service (Amazon SQS) standard queue, and send the telemetry data as is

Overall explanation

Correct option:

Use an Amazon Simple Queue Service (Amazon SQS) FIFO (First-In-First-Out) queue, and make sure the telemetry data is sent with a Group ID attribute representing the value of the Desktop ID

Amazon Simple Queue Service (SQS) is a fully managed message queuing service that enables you to decouple and scale microservices, distributed systems, and serverless applications. SQS offers two types of message queues. Standard queues offer maximum throughput, best-effort ordering, and at-least-once delivery. SQS FIFO queues are designed to guarantee that messages are processed exactly once, in the exact order that they are sent.

We, therefore, need to use an SQS FIFO queue. If we don't specify a GroupID, then all the messages are in absolute order, but we can only have 1 consumer at most. To allow for multiple consumers to read data for each Desktop application, and to scale the number of consumers, we should use the "Group ID" attribute. So this is the correct option.

Incorrect options:

Use an Amazon Simple Queue Service (Amazon SQS) FIFO (First-In-First-Out) queue, and send the telemetry data as is - This is incorrect because if we send the telemetry data as is then we will not be able to scale the number of consumers to be equal to the number of desktop systems. In this case, each message will have its consumer. So we should use the "Group ID" attribute so that multiple consumers can read data for each Desktop application.

Use an Amazon Simple Queue Service (Amazon SQS) standard queue, and send the telemetry data as is - An Amazon SQS standard queue has no ordering capability so that's ruled out.

Use an Amazon Kinesis Data Stream, and send the telemetry data with a Partition ID that uses the value of the Desktop ID - Amazon Kinesis Data Streams (KDS) is a massively scalable and durable real-time data streaming service. KDS can continuously capture gigabytes of data per second from hundreds of thousands of sources such as website clickstreams, database event streams, financial transactions, social media feeds, IT logs, and location-tracking events. A Kinesis Data Stream would work and would give us the data for each desktop application within shards, but we can only have as many consumers as shards in Kinesis (which is in practice, much less than the number of producers).

References:

https://aws.amazon.com/blogs/compute/solving-complex-ordering-challenges-with-amazon-sqs-fifo-queues/

https://aws.amazon.com/sqs/faqs/

https://aws.amazon.com/kinesis/data-streams/faqs/

Domain

Design High-Performing Architectures

Question 16 Correct

A Hollywood studio is planning a series of promotional events leading up to the launch of the trailer of its next sci-fi thriller. The executives at the studio want to create a static website with lots of animations in line with the theme of the movie. The studio has hired you as a solutions architect to build a scalable serverless solution.

Which of the following represents the MOST cost-optimal and high-performance solution?

Your answer is correct

Build the website as a static website hosted on Amazon S3. Create an Amazon CloudFront distribution with Amazon S3 as the origin. Use Amazon Route 53 to create an alias record that points to your Amazon CloudFront distribution

Host the website on an Amazon EC2 instance. Create a Amazon CloudFront distribution with the Amazon EC2 instance as the custom origin

Host the website on AWS Lambda. Create an Amazon CloudFront distribution with Lambda as the origin

Host the website on an instance in the studio's on-premises data center. Create an Amazon CloudFront distribution with this instance as the custom origin

Overall explanation

Correct option:

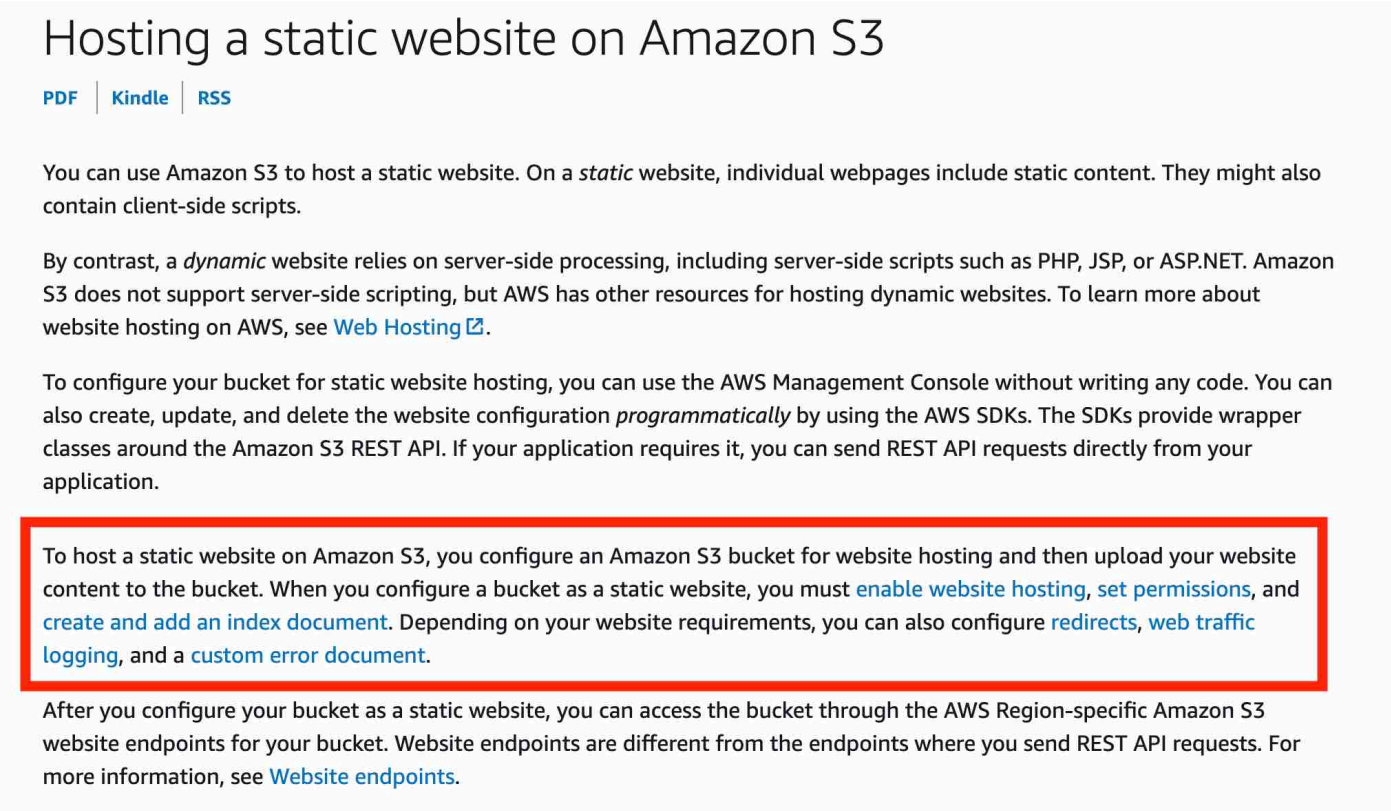
Build the website as a static website hosted on Amazon S3. Create an Amazon CloudFront distribution with Amazon S3 as the origin. Use Amazon Route 53 to create an alias record that points to your Amazon CloudFront distribution

You can use Amazon S3 to host a static website. On a static website, individual web pages include static content. They might also contain client-side scripts. To host a static website on Amazon S3, you configure an Amazon S3 bucket for website hosting and then upload your website content to the bucket.

Amazon CloudFront is a fast content delivery network (CDN) service that securely delivers data, videos, applications, and APIs to customers globally with low latency, high transfer speeds, all within a developer-friendly environment.

You can use Amazon CloudFront to improve the performance of your website. CloudFront makes your website files (such as HTML, images, and video) available from data centers around the world (called edge locations). When a visitor requests a file from your website, CloudFront automatically redirects the request to a copy of the file at the nearest edge location. This results in faster download times than if the visitor had requested the content from a data center that is located farther away. Therefore, this option is correct.

Hosting a static website on Amazon S3:



Hosting a static website on Amazon S3

PDF | Kindle | RSS

You can use Amazon S3 to host a static website. On a *static* website, individual webpages include static content. They might also contain client-side scripts.

By contrast, a *dynamic* website relies on server-side processing, including server-side scripts such as PHP, JSP, or ASP.NET. Amazon S3 does not support server-side scripting, but AWS has other resources for hosting dynamic websites. To learn more about website hosting on AWS, see Web Hosting.

To configure your bucket for static website hosting, you can use the AWS Management Console without writing any code. You can also create, update, and delete the website configuration *programmatically* by using the AWS SDKs. The SDKs provide wrapper classes around the Amazon S3 REST API. If your application requires it, you can send REST API requests directly from your application.

To host a static website on Amazon S3, you configure an Amazon S3 bucket for website hosting and then upload your website content to the bucket. When you configure a bucket as a static website, you must enable website hosting, set permissions, and create and add an index document. Depending on your website requirements, you can also configure redirects, web traffic logging, and a custom error document.

After you configure your bucket as a static website, you can access the bucket through the AWS Region-specific Amazon S3 website endpoints for your bucket. Website endpoints are different from the endpoints where you send REST API requests. For more information, see Website endpoints.

via - https://docs.aws.amazon.com/AmazonS3/latest/dev/WebsiteHosting.html

Incorrect options:

Host the website on AWS Lambda. Create an Amazon CloudFront distribution with Lambda as the origin

With AWS Lambda, you can run code without provisioning or managing servers. You can't host a website on Lambda. Also, you can't have CloudFront in front of Lambda. So this option is incorrect.

Host the website on an Amazon EC2 instance. Create a Amazon CloudFront distribution with the Amazon EC2 instance as the custom origin

Host the website on an instance in the studio's on-premises data center. Create an Amazon CloudFront distribution with this instance as the custom origin

Hosting the website on an Amazon EC2 instance or a data-center specific instance is ruled out as the studio wants a serverless solution. So both these options are incorrect.

References:

https://docs.aws.amazon.com/AmazonS3/latest/dev/WebsiteHosting.html

https://docs.aws.amazon.com/AmazonS3/latest/dev/website-hosting-custom-domain-walkthrough.html

https://docs.aws.amazon.com/AmazonS3/latest/dev/website-hosting-cloudfront-walkthrough.html

Domain

Design High-Performing Architectures

Question 17 **Incorrect**

A social media application is hosted on an Amazon EC2 fleet running behind an Application Load Balancer. The application traffic is fronted by an Amazon CloudFront distribution. The engineering team wants to decouple the user authentication process for the application, so that the application servers can just focus on the business logic.

As a Solutions Architect, which of the following solutions would you recommend to the development team so that it requires minimal development effort?

Use Amazon Cognito Authentication via Cognito Identity Pools for your Amazon CloudFront distribution

Use Amazon Cognito Authentication via Cognito Identity Pools for your Application Load Balancer

Your answer is incorrect

Use Amazon Cognito Authentication via Cognito User Pools for your Amazon CloudFront distribution

Correct answer

Use Amazon Cognito Authentication via Cognito User Pools for your Application Load Balancer

Overall explanation

Correct option:

Use Amazon Cognito Authentication via Cognito User Pools for your Application Load Balancer

Application Load Balancer can be used to securely authenticate users for accessing your applications. This enables you to offload the work of authenticating users to your load balancer so that your applications can focus on their business logic. You can use Cognito User Pools to authenticate users through well-known social IdPs, such as Amazon, Facebook, or Google, through the user pools supported by Amazon Cognito or through corporate identities, using SAML, LDAP, or Microsoft AD, through the user pools supported by Amazon Cognito. You configure user authentication by creating an authenticate action for one or more listener rules.

Authenticate users using an Application Load Balancer

PDF | Kindle | RSS

You can configure an Application Load Balancer to securely authenticate users as they access your applications. This enables you to offload the work of authenticating users to your load balancer so that your applications can focus on their business logic.

The following use cases are supported:

- Authenticate users through an identity provider (IdP) that is OpenID Connect (OIDC) compliant.
- Authenticate users through well-known social IdPs, such as Amazon, Facebook, or Google, through the user pools supported by Amazon Cognito.
- Authenticate users through corporate identities, using SAML, LDAP, or Microsoft AD, through the user pools supported by Amazon Cognito.

---

via - https://docs.aws.amazon.com/elasticloadbalancing/latest/application/listener-authenticate-users.html

Exam Alert:

Please review the following note to understand the differences between Amazon Cognito User Pools and Amazon Cognito Identity Pools:

Features of Amazon Cognito

User pools

A user pool is a user directory in Amazon Cognito. With a user pool, your users can sign in to your web or mobile app through Amazon Cognito, or federate through a third-party identity provider (IdP). Whether your users sign in directly or through a third party, all members of the user pool have a directory profile that you can access through an SDK.

User pools provide:

- Sign-up and sign-in services.
- A built-in, customizable web UI to sign in users.
- Social sign-in with Facebook, Google, Login with Amazon, and Sign in with Apple, and through SAML and OIDC identity providers from your user pool.
- User directory management and user profiles.
- Security features such as multi-factor authentication (MFA), checks for compromised credentials, account takeover protection, and phone and email verification.
- Customized workflows and user migration through AWS Lambda triggers.

For more information about user pools, see Getting Started with User Pools and the Amazon Cognito User Pools API Reference.

Identity pools

With an identity pool, your users can obtain temporary AWS credentials to access AWS services, such as Amazon S3 and DynamoDB. Identity pools support anonymous guest users, as well as the following identity providers that you can use to authenticate users for identity pools:

- Amazon Cognito user pools
- Social sign-in with Facebook, Google, Login with Amazon, and Sign in with Apple
- OpenID Connect (OIDC) providers
- SAML identity providers
- Developer authenticated identities

To save user profile information, your identity pool needs to be integrated with a user pool.

For more information about identity pools, see Getting Started with Amazon Cognito Identity Pools (Federated Identities) and the Amazon Cognito Identity Pools API Reference.

via - https://docs.aws.amazon.com/cognito/latest/developerguide/what-is-amazon-cognito.html

Incorrect options:

Use Amazon Cognito Authentication via Cognito Identity Pools for your Application Load Balancer - There is no such thing as using Amazon Cognito Authentication via Cognito Identity Pools for managing user authentication for the application. Application-specific user authentication can be provided via Cognito User Pools. Amazon Cognito identity pools provide temporary AWS credentials for users who are guests (unauthenticated) and for users who have been authenticated and received a token.

Use Amazon Cognito Authentication via Cognito User Pools for your Amazon CloudFront distribution - You cannot directly integrate Cognito User Pools with CloudFront distribution as you have to create a separate AWS Lambda@Edge function to accomplish the authentication via Cognito User Pools. This involves additional development effort, so this option is not the best fit for the given use-case.

Use Amazon Cognito Authentication via Cognito Identity Pools for your Amazon CloudFront distribution - You cannot use Cognito Identity Pools for managing user authentication, so this option is not correct.

References:

https://docs.aws.amazon.com/elasticloadbalancing/latest/application/listener-authenticate-users.html

https://docs.aws.amazon.com/cognito/latest/developerguide/cognito-user-identity-pools.html

https://aws.amazon.com/blogs/networking-and-content-delivery/authorizationedge-using-cookies-protect-your-amazon-cloudfront-content-from-being-downloaded-by-unauthenticated-users/

Domain

Design Secure Architectures

Question 18 **Correct**

What does this IAM policy do?

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "Mystery Policy",
      "Action": [
        "ec2:RunInstances"
      ],
      "Effect": "Allow",
      "Resource": "*",
      "Condition": {
        "IpAddress": {
          "aws:SourceIp": "34.50.31.0/24"
        }
      }
    }
  ]
}
```

It allows starting an Amazon EC2 instance only when they have a Public IP within the `34.50.31.0/24` CIDR block

It allows starting an Amazon EC2 instance only when they have an Elastic IP within the `34.50.31.0/24` CIDR block

Your answer is correct

It allows starting an Amazon EC2 instance only when the IP where the call originates is within the `34.50.31.0/24` CIDR block

It allows starting an Amazon EC2 instance only when they have a Private IP within the `34.50.31.0/24` CIDR block

Overall explanation

Correct option:

It allows starting an Amazon EC2 instance only when the IP where the call originates is within the `34.50.31.0/24` CIDR block

You manage access in AWS by creating policies and attaching them to IAM identities (users, groups of users, or roles) or AWS resources. A policy is an object in AWS that, when associated with an identity or resource, defines their permissions. AWS evaluates these policies when an IAM principal (user or role) makes a request. Permissions in the policies determine whether the request is allowed or denied. Most policies are stored in AWS as JSON documents. AWS supports six types of policies: identity-based policies, resource-based policies, permissions boundaries, Organizations service control policy (SCPs), access control lists (ACLs), and session policies.

Consider the following snippet from the given policy document:

```
"Condition": {
  "IpAddress": {
    "aws:SourceIp": "34.50.31.0/24"
  }
}
```

The `aws:SourceIP` in this condition always represents the IP of the caller of the API. That is very helpful if you want to restrict access to certain AWS API for example from the public IP of your on-premises infrastructure.

Please see this overview of Elastic vs Public vs Private IP addresses:

elastic IP address (EIP) - An elastic IP address (EIP) is a static IPv4 address designed for dynamic cloud computing. An Elastic IP address is associated with your AWS account. With an Elastic IP address, you can mask the failure of an instance or software by rapidly remapping the address to another instance in your account.

Private IP address - A private IPv4 address is an IP address that's not reachable over the Internet. You can use private IPv4 addresses for communication between instances in the same VPC.

Public IP address - A public IP address is an IPv4 address that's reachable from the Internet. You can use public addresses for communication between your instances and the Internet.

Please note `34.50.31.0/24` is a public IP range, not a private IP range. Private IP ranges are: 192.168.0.0 - 192.168.255.255 (65,536 IP addresses) 172.16.0.0 - 172.31.255.255 (1,048,576 IP addresses) 10.0.0.0 - 10.255.255.255 (16,777,216 IP addresses)

Incorrect options:

It allows starting an Amazon EC2 instance only when they have a Public IP within the `34.50.31.0/24` CIDR block

It allows starting an Amazon EC2 instance only when they have an Elastic IP within the `34.50.31.0/24` CIDR block

It allows starting an Amazon EC2 instance only when they have a Private IP within the `34.50.31.0/24` CIDR block

Each of these three options suggests that the IP addresses of the Amazon EC2 instances must belong to the `34.50.31.0/24` CIDR block for the EC2 instances to start. Actually, the policy states that the AMazon EC2 instance should start only when the IP where the call originates is within the `34.50.31.0/24` CIDR block. Hence these options are incorrect.

References:

https://docs.aws.amazon.com/IAM/latest/UserGuide/access_policies.html

https://aws.amazon.com/premiumsupport/knowledge-center/iam-restrict-calls-ip-addresses/

https://docs.aws.amazon.com/AWSEC2/latest/UserGuide/using-instance-addressing.html

Domain

Design Secure Architectures

Question 19 Correct

An IT company is working on a client project to build a Supply Chain Management application. The web-tier of the application runs on an Amazon EC2 instance and the database tier is on Amazon RDS MySQL. For beta testing, all the resources are currently deployed in a single Availability Zone (AZ). The development team wants to improve application availability before the go-live.

Given that all end users of the web application would be located in the US, which of the following would be the MOST resource-efficient solution?

Deploy the web-tier Amazon EC2 instances in two regions, behind an Elastic Load Balancer. Deploy the Amazon RDS MySQL database in read replica configuration

Deploy the web-tier Amazon EC2 instances in two regions, behind an Elastic Load Balancer. Deploy the Amazon RDS MySQL database in Multi-AZ configuration

Your answer is correct

Deploy the web-tier Amazon EC2 instances in two Availability Zones (AZs), behind an Elastic Load Balancer. Deploy the Amazon RDS MySQL database in Multi-AZ configuration

Deploy the web-tier Amazon EC2 instances in two Availability Zones (AZs), behind an Elastic Load Balancer. Deploy the Amazon RDS MySQL database in read replica configuration

Overall explanation

Correct option:

Deploy the web-tier Amazon EC2 instances in two Availability Zones (AZs), behind an Elastic Load Balancer. Deploy the Amazon RDS MySQL database in Multi-AZ configuration

Elastic Load Balancing automatically distributes incoming application traffic across multiple targets, such as Amazon EC2 instances, containers, IP addresses, and Lambda functions. It can handle the varying load of your application traffic in a single Availability Zone or across multiple Availability Zones. Therefore, deploying the web-tier Amazon EC2 instances in two Availability Zones (AZs), behind an Elastic Load Balancer would improve the availability of the application.

Amazon RDS Multi-AZ deployments provide enhanced availability and durability for RDS database (DB) instances, making them a natural fit for production database workloads. When you provision a Multi-AZ DB Instance, Amazon RDS automatically creates a primary DB Instance and synchronously replicates the data to a standby instance in a different Availability Zone (AZ). Each Availability Zone (AZ) runs on its own physically distinct, independent infrastructure, and is engineered to be highly reliable. Deploying the Amazon RDS MySQL database in Multi-AZ configuration would improve availability and hence this is the correct option.

Incorrect options:

Deploy the web-tier Amazon EC2 instances in two Availability Zones (AZs), behind an Elastic Load Balancer. Deploy the Amazon RDS MySQL database in read replica configuration

Deploy the web-tier Amazon EC2 instances in two regions, behind an Elastic Load Balancer. Deploy the Amazon RDS MySQL database in read replica configuration

Amazon RDS Read Replicas provide enhanced performance and durability for RDS database (DB) instances. They make it easy to elastically scale out beyond the capacity constraints of a single DB instance for read-heavy database workloads. Read replicas are meant to address scalability issues. You cannot use read replicas for improving availability, so both these options are incorrect.

Exam Alert:

Please review this comparison vis-a-vis Multi-AZ vs Read Replica for Amazon RDS:

Multi-AZ deployments, multi-region deployments, and read replicas

Amazon RDS Multi-AZ deployments complement multi-region deployments and read replicas. While all three features increase availability and durability by maintaining additional copies of your data, there are differences between them:

Multi-AZ deployments	Multi-Region deployments	Read replicas
Main purpose is high availability	Main purpose is disaster recovery and local performance	Main purpose is scalability
Non-Aurora: synchronous replication; Aurora: asynchronous replication	Asynchronous replication	Asynchronous replication
Non-Aurora: only the primary instance is active; Aurora: all instances are active	All regions are accessible and can be used for reads	All read replicas are accessible and can be used for readscaling
Non-Aurora: automated backups are taken from standby; Aurora: automated backups are taken from shared storage layer	Automated backups can be taken in each region	No backups configured by default
Always span at least two Availability Zones within a single region	Each region can have a Multi-AZ deployment	Can be within an Availability Zone, Cross-AZ, or Cross-Region
Non-Aurora: database engine version upgrades happen on primary; Aurora: all instances are updated together	Non-Aurora: database engine version upgrade is independent in each region; Aurora: all instances are updated together	Non-Aurora: database engine version upgrade is independent from source instance; Aurora: all instances are updated together
Automatic failover to standby (non-Aurora) or read replica (Aurora) when a problem is detected	Aurora allows promotion of a secondary region to be the master	Can be manually promoted to a standalone database instance (non-Aurora) or to be the primary instance (Aurora)

via - https://aws.amazon.com/rds/features/multi-az/

Deploy the web-tier Amazon EC2 instances in two regions, behind an Elastic Load Balancer. Deploy the Amazon RDS MySQL database in Multi-AZ configuration - As Elastic Load Balancing does not work across regions, so this option is incorrect.

Reference:

https://aws.amazon.com/rds/features/multi-az/

Domain

Design Resilient Architectures

Question 20 Correct

An HTTP application is deployed on an Auto Scaling Group, is accessible from an Application Load Balancer (ALB) that provides HTTPS termination, and accesses a PostgreSQL database managed by Amazon RDS.

How should you configure the security groups? (Select three)

Your selection is correct

The security group of the Application Load Balancer should have an inbound rule from anywhere on port 443

Your selection is correct

The security group of the Amazon EC2 instances should have an inbound rule from the security group of the Application Load Balancer on port 80

The security group of the Application Load Balancer should have an inbound rule from anywhere on port 80

The security group of Amazon RDS should have an inbound rule from the security group of the Amazon EC2 instances in the Auto Scaling group on port 80

The security group of the Amazon EC2 instances should have an inbound rule from the security group of the Amazon RDS database on port 5432

Your selection is correct

The security group of Amazon RDS should have an inbound rule from the security group of the Amazon EC2 instances in the Auto Scaling group on port 5432

Overall explanation

Correct options:

The security group of Amazon RDS should have an inbound rule from the security group of the Amazon EC2 instances in the Auto Scaling group on port 5432

The security group of the Amazon EC2 instances should have an inbound rule from the security group of the Application Load Balancer on port 80

The security group of the Application Load Balancer should have an inbound rule from anywhere on port 443

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

The following are the characteristics of security group rules: 1. By default, security groups allow all outbound traffic. 2. Security group rules are always permissive; you can't create rules that deny access. 3. Security groups are stateful

PostgreSQL port = 5432 HTTP port = 80 HTTPS port = 443

The traffic goes like this : The client sends an HTTPS request to ALB on port 443. This is handled by the rule - "The security group of the Application Load Balancer should have an inbound rule from anywhere on port 443"

The Application Load Balancer then forwards the request to one of the Amazon EC2 instances. This is handled by the rule - "The security group of the Amazon EC2 instances should have an inbound rule from the security group of the Application Load Balancer on port 80"

The Amazon EC2 instance further accesses the PostgreSQL database managed by Amazon RDS on port 5432. This is handled by the rule - "The security group of Amazon RDS should have an inbound rule from the security group of the Amazon EC2 instances in the Auto Scaling group on port 5432"

Incorrect options:

The security group of the Application Load Balancer should have an inbound rule from anywhere on port 80 - The client sends an HTTPS request to ALB on port 443 and not on port 80, so this is incorrect.

The security group of the Amazon EC2 instances should have an inbound rule from the security group of the Amazon RDS database on port 5432 - The security group of the Amazon EC2 instances should have an inbound rule from the security group of the Application Load

Balancer and not from the security group of the Amazon RDS database, so this option is incorrect.

The security group of Amazon RDS should have an inbound rule from the security group of the Amazon EC2 instances in the Auto Scaling group on port 80 - The Amazon EC2 instance further accesses the PostgreSQL database managed by Amazon RDS on port 5432 and not on port 80, so this option is incorrect.

Reference:

https://docs.aws.amazon.com/AWSEC2/latest/UserGuide/ec2-security-groups.html

Domain

Design Secure Architectures

Question 21 Correct

A SaaS company is modernizing one of its legacy web applications by migrating it to AWS. The company aims to improve the availability of the application during both normal and peak traffic periods. Additionally, the company wants to implement protection against common web exploits and malicious traffic. The architecture must be scalable and integrate AWS WAF to secure incoming traffic.

Which solution will best meet these requirements with high availability and minimal configuration complexity?

Create an Auto Scaling group with EC2 instances in multiple Availability Zones. Attach a Network Load Balancer (NLB) to distribute incoming traffic. Integrate AWS WAF directly with the Auto Scaling group for traffic filtering

Launch two EC2 instances in separate Availability Zones and register them as targets of an Application Load Balancer. Associate the ALB with AWS WAF to filter incoming traffic

Your answer is correct

Deploy the application on multiple Amazon EC2 instances in an Auto Scaling group that spans two Availability Zones. Place an Application Load Balancer (ALB) in front of the group. Associate AWS WAF with the ALB

Launch EC2 instances in a single Availability Zone and configure AWS Global Accelerator to route traffic to the instances. Attach AWS WAF to Global Accelerator for application protection

Overall explanation

Correct option:

Deploy the application on multiple Amazon EC2 instances in an Auto Scaling group that spans two Availability Zones. Place an Application Load Balancer (ALB) in front of the group. Associate AWS WAF with the ALB

This solution provides a highly available, scalable, and secure architecture with minimal operational complexity. By using an Auto Scaling group across multiple Availability Zones, the application can automatically scale out during high traffic and maintain fault tolerance in case one AZ fails. Placing an Application Load Balancer (ALB) in front of the EC2 instances distributes incoming requests evenly and supports layer 7 routing, which is necessary for web application traffic. Most importantly, AWS WAF integrates directly with ALB, allowing the company to define and enforce web access control rules that protect against common threats such as SQL injection and cross-site scripting. This setup ensures that the application is resilient, responsive to load, and protected from malicious web activity, all while minimizing ongoing management overhead.

Incorrect options:

Launch EC2 instances in a single Availability Zone and configure AWS Global Accelerator to route traffic to the instances. Attach AWS WAF to Global Accelerator for application protection - While AWS Global Accelerator supports integration with AWS WAF and can reduce global latency, this solution does not address high availability adequately because it deploys EC2 instances in a single Availability Zone. This creates a single point of failure and doesn't offer automatic scaling. Moreover, Global Accelerator is best suited for global user performance optimization, not for replacing the load balancing and scaling functionality of ALB and Auto Scaling groups.

Launch two EC2 instances in separate Availability Zones and register them as targets of an Application Load Balancer. Associate the ALB with AWS WAF to filter incoming traffic - While

this setup does provide some availability by using two Availability Zones and integrates WAF correctly via ALB, it lacks scalability. Without an Auto Scaling group, the system cannot respond to traffic spikes or recover automatically from instance failure. Manual instance management increases operational overhead and limits elasticity.

Create an Auto Scaling group with EC2 instances in multiple Availability Zones. Attach a Network Load Balancer (NLB) to distribute incoming traffic. Integrate AWS WAF directly with the Auto Scaling group for traffic filtering - While this solution achieves high availability and scalability, AWS WAF cannot be directly associated with an Auto Scaling group or a Network Load Balancer. WAF only supports integration with Application Load Balancer (ALB), API Gateway, App Runner, CloudFront, and Global Accelerator. Since NLB operates at Layer 4 and AWS WAF protects Layer 7, this design fails to implement WAF protection, which is a core requirement.

References:

https://docs.aws.amazon.com/waf/latest/developerguide/how-aws-waf-works.html

https://docs.aws.amazon.com/elasticloadbalancing/latest/application/introduction.html

https://docs.aws.amazon.com/global-accelerator/latest/dg/what-is-global-accelerator.html

Domain

Design Secure Architectures

Question 22 **Correct**

An IT company has an Access Control Management (ACM) application that uses Amazon RDS for MySQL but is running into performance issues despite using Read Replicas. The company has hired you as a solutions architect to address these performance-related challenges without moving away from the underlying relational database schema. The company has branch offices across the world, and it needs the solution to work on a global scale.

Which of the following will you recommend as the MOST cost-effective and high-performance solution?

Use Amazon DynamoDB Global Tables to provide fast, local, read and write performance in each region

Spin up a Amazon Redshift cluster in each AWS region. Migrate the existing data into Redshift clusters

Spin up Amazon EC2 instances in each AWS region, install MySQL databases and migrate the existing data into these new databases

Your answer is correct

Use Amazon Aurora Global Database to enable fast local reads with low latency in each region

Overall explanation

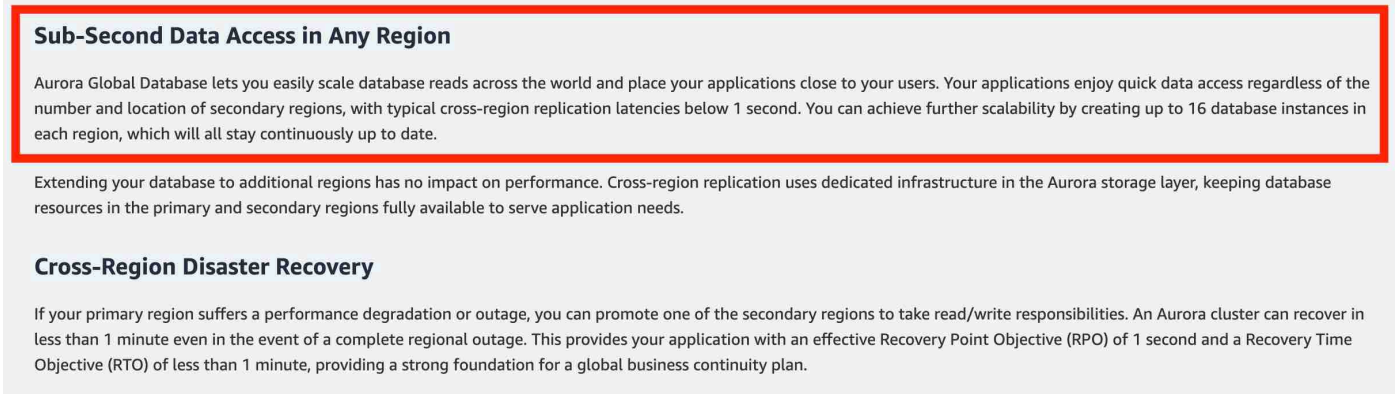
Correct option:

Use Amazon Aurora Global Database to enable fast local reads with low latency in each region

Amazon Aurora is a MySQL and PostgreSQL-compatible relational database built for the cloud, that combines the performance and availability of traditional enterprise databases with the simplicity and cost-effectiveness of open source databases. Amazon Aurora features a distributed, fault-tolerant, self-healing storage system that auto-scales up to 64TB per database instance. Aurora is not an in-memory database.

Amazon Aurora Global Database is designed for globally distributed applications, allowing a single Amazon Aurora database to span multiple AWS regions. It replicates your data with no impact on database performance, enables fast local reads with low latency in each region, and provides disaster recovery from region-wide outages. Amazon Aurora Global Database is the correct choice for the given use-case.

Amazon Aurora Global Database Features:



via - https://aws.amazon.com/rds/aurora/global-database/

Incorrect options:

Use Amazon DynamoDB Global Tables to provide fast, local, read and write performance in each region - Amazon DynamoDB is a key-value and document database that delivers single-digit millisecond performance at any scale. It's a fully managed, multi-region, multi-master, durable database with built-in security, backup and restore, and in-memory caching for internet-scale applications.

Global Tables builds upon DynamoDB's global footprint to provide you with a fully managed, multi-region, and multi-master database that provides fast, local, read, and write performance for massively scaled, global applications. Global Tables replicates your Amazon DynamoDB tables automatically across your choice of AWS regions. Given that the use-case wants you to continue with the underlying schema of the relational database, DynamoDB is not the right choice as it's a NoSQL database.

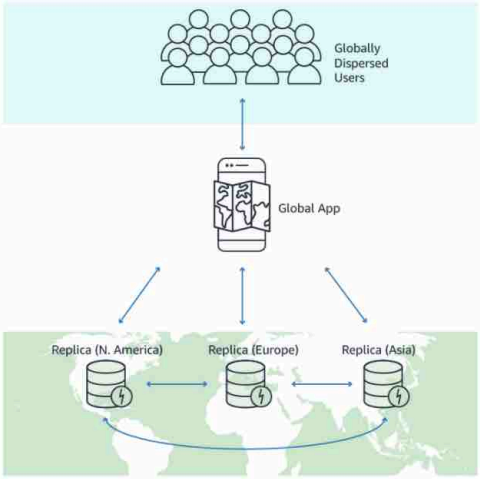
Amazon DynamoDB Global Tables Overview:

Global Tables

Multi-Region, Multi-Master tables for fast local performance for globally distributed apps

Global Tables builds upon DynamoDB's global footprint to provide you with a fully managed, multi-region, and multi-master database that provides fast, local, read and write performance for massively scaled, global applications. Global Tables replicates your Amazon DynamoDB tables automatically across your choice of AWS regions.

Global Tables eliminates the difficult work of replicating data between regions and resolving update conflicts, enabling you to focus on your application's business logic. In addition, Global Tables enables your applications to stay highly available even in the unlikely event of isolation or degradation of an entire region.



Global Tables

You can setup Global Tables with just a few clicks in the AWS Management Console. No application changes are required because Global Tables use existing DynamoDB APIs. There are no upfront costs or commitments for using Global Tables, and you pay only for the resources provisioned. Learn more about setting up Global Tables in the DynamoDB Developer Guide.

via - https://aws.amazon.com/dynamodb/global-tables/

Spin up a Amazon Redshift cluster in each AWS region. Migrate the existing data into Redshift clusters - Amazon Redshift is a fully-managed petabyte-scale cloud-based data warehouse product designed for large scale data set storage and analysis. Amazon Redshift is not suited to be used as a transactional relational database, so this option is not correct.

Spin up Amazon EC2 instances in each AWS region, install MySQL databases and migrate the existing data into these new databases - Setting up Amazon EC2 instances in multiple regions with manually managed MySQL databases represents a maintenance nightmare and is not the correct choice for this use-case.

References:

https://aws.amazon.com/rds/aurora/global-database/

https://aws.amazon.com/dynamodb/global-tables/

Domain

Design High-Performing Architectures

Question 23 **Incorrect**

A video conferencing platform serves users worldwide through a globally distributed deployment of Amazon EC2 instances behind Network Load Balancers (NLBs) in several AWS Regions. The platform's architecture currently allows clients to connect to any Region via public endpoints, depending on how DNS resolves. However, users in regions far from the load balancers frequently experience high latency and slow connection times, especially during session initiation. The company wants to optimize the experience for global users by reducing end-to-end latency and load time while keeping the existing NLBs and EC2-based application infrastructure in place.

Which solution will best meet these requirements?

Deploy Amazon CloudFront with HTTP caching enabled in front of the NLBs. Use CloudFront edge locations to serve user requests faster and reduce the load on the backend EC2 instances

Replace all Network Load Balancers (NLBs) with Application Load Balancers (ALBs) in each Region. Register the EC2 instances as targets behind the ALBs and use cross-zone load balancing for latency distribution

Correct answer

Deploy a standard accelerator in AWS Global Accelerator and register the existing regional NLBs as endpoints. Use the accelerator to route user requests through AWS's global edge network to the closest healthy Regional NLB

Your answer is incorrect

Configure Amazon Route 53 with latency-based routing policies to direct users to the Region with the lowest response time. Use health checks to fail over to another Region if a specific NLB becomes unhealthy

Overall explanation

Correct option:

Deploy a standard accelerator in AWS Global Accelerator and register the existing regional NLBs as endpoints. Use the accelerator to route user requests through AWS's global edge network to the closest healthy Regional NLB

AWS Global Accelerator provides static anycast IP addresses that route incoming user traffic through the AWS global edge network to the closest AWS Regional endpoint (in this case, the NLBs). It supports TCP and UDP, making it ideal for real-time applications. Global Accelerator reduces latency by using optimized AWS routes instead of the unpredictable public internet. By configuring existing NLBs as endpoints, the company avoids modifying its core infrastructure while dramatically improving global performance.

How AWS Global Accelerator works:

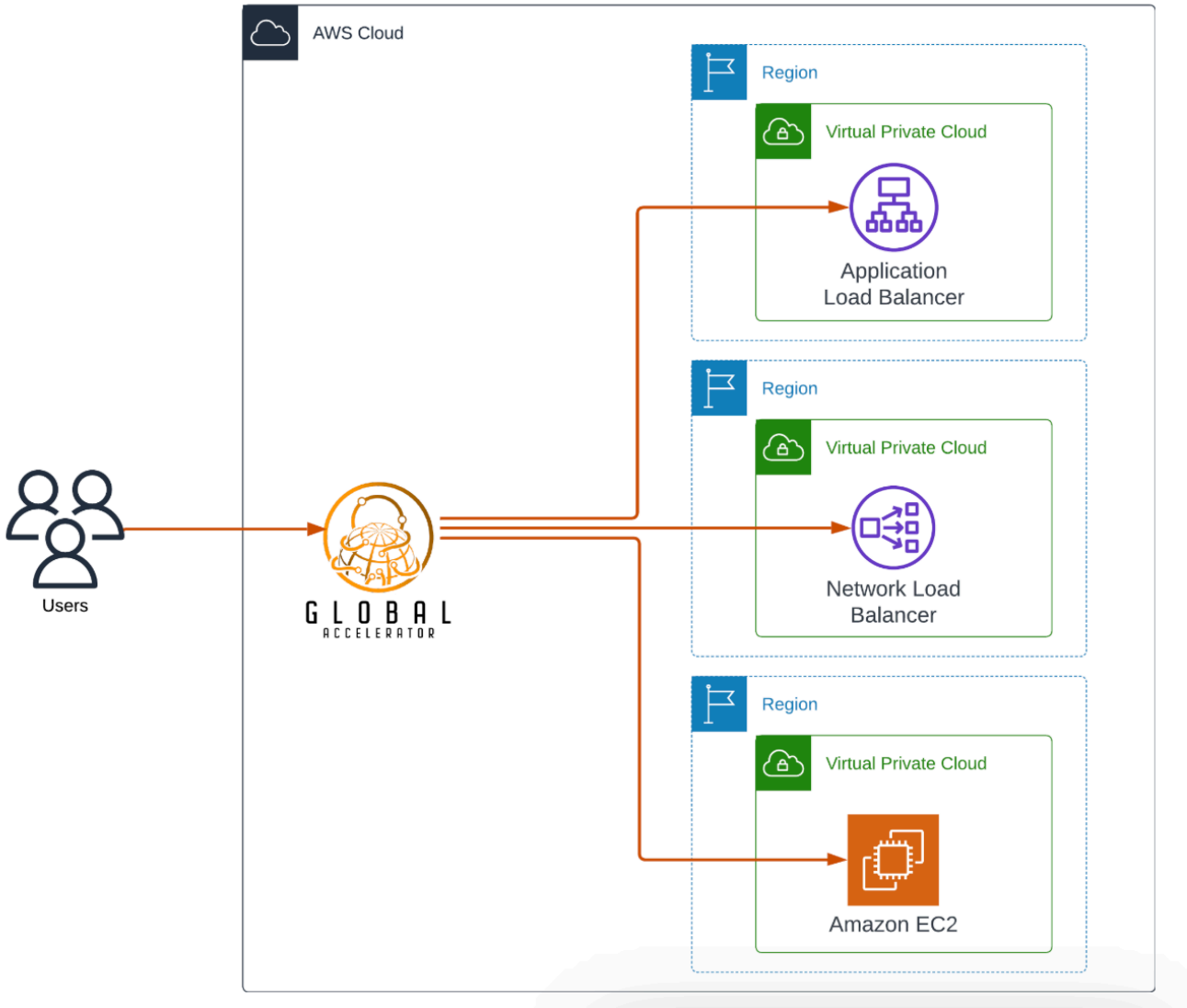
Overview

AWS Global Accelerator is a networking service that helps you improve the internet client experience for your applications by leveraging the AWS Global Network infrastructure. Depending on your needs, you can use AWS Global Accelerator for several deployment use cases, including A/B testing, blue-green deployments, API acceleration, or live video ingestion.

When you create an accelerator, you get two global static IP addresses for each IP stack (for example, two IPv4 and two IPv6 addresses for a dual-stack accelerator). The static IP addresses are anycast routed from the AWS edge network. Clients use the static IP addresses to access your applications, and traffic enters the AWS network at the nearest Global Accelerator point of presence (POP).

Traffic for standard accelerators is routed to the optimal regional endpoint, based on health, client location, and configured policies. Global Accelerator also implements automatic routing optimizations to keep packet loss, jitter, and latency consistently low for your applications.

The following diagram (Figure 1) provides a high-level overview of how Global Accelerator works.



via - https://aws.amazon.com/blogs/networking-and-content-delivery/configuring-client-ip-address-preservation-with-a-network-load-balancer-in-aws-global-accelerator/

Incorrect options:

Deploy Amazon CloudFront with HTTP caching enabled in front of the NLBs. Use CloudFront edge locations to serve user requests faster and reduce the load on the backend EC2 instances - Amazon CloudFront is a global CDN that excels at caching static and HTTP-based content, but it does not support TCP/UDP protocols that are typically used with Network Load Balancers or gaming and real-time applications. Since NLBs operate at Layer 4 (TCP/UDP) and

CloudFront works at Layer 7 (HTTP/HTTPS), this integration is not possible. CloudFront is not suitable for accelerating TCP traffic to NLBs.

Replace all Network Load Balancers (NLBs) with Application Load Balancers (ALBs) in each Region. Register the EC2 instances as targets behind the ALBs and use cross-zone load balancing for latency distribution - Application Load Balancers (ALBs) operate at Layer 7 and are designed for HTTP/HTTPS workloads, while NLBs operate at Layer 4 and are better suited for real-time, low-latency TCP/UDP traffic, such as gaming or video conferencing. Replacing NLBs with ALBs may break protocol compatibility and reduce performance for non-HTTP workloads. Cross-zone load balancing does not address global latency since it works only within a Region, not across Regions.

Configure Amazon Route 53 with latency-based routing policies to direct users to the Region with the lowest response time. Use health checks to fail over to another Region if a specific NLB becomes unhealthy - Latency-based routing in Route 53 can help direct DNS queries to the Region with the lowest latency, but it does not guarantee global network optimization. It also relies on public DNS resolution and internet paths, which may not always route users optimally. Additionally, Route 53 does not provide the transport-level acceleration or failover performance that Global Accelerator offers.

References:

https://docs.aws.amazon.com/global-accelerator/latest/dg/introduction-how-it-works.html

https://aws.amazon.com/blogs/networking-and-content-delivery/configuring-client-ip-address-preservation-with-a-network-load-balancer-in-aws-global-accelerator/

https://docs.aws.amazon.com/AmazonCloudFront/latest/DeveloperGuide/Introduction.html

https://docs.aws.amazon.com/Route53/latest/DeveloperGuide/routing-policy.html

Domain

Design High-Performing Architectures

Question 24 Correct

A financial services company runs a Kubernetes-based microservices application in its on-premises data center. The application uses the Advanced Message Queuing Protocol (AMQP) to interact with a message queue. The company is experiencing rapid growth and its on-prem infrastructure cannot scale fast enough. The company wants to migrate the application to AWS with minimal code changes and reduce infrastructure management overhead. The messaging component must continue using AMQP, and the solution should offer high scalability and low operational effort.

Which combination of options will together meet these requirements? (Select two)

Your selection is correct

Replace the current messaging system with Amazon MQ, a fully managed broker that supports AMQP natively. Integrate the application with the Amazon MQ endpoint without modifying the existing message format

Use Amazon Simple Queue Service (Amazon SQS) as the replacement for the AMQP message broker. Refactor the application to use SQS SDKs and polling logic

Run the application on Amazon EC2 Auto Scaling groups and use a self-hosted RabbitMQ instance on EC2 to preserve AMQP compatibility

Your selection is correct

Deploy the containerized application to Amazon Elastic Kubernetes Service (Amazon EKS) using AWS Fargate to avoid managing EC2 nodes

Deploy the application to Amazon ECS on EC2, and integrate the messaging workflow using Amazon SNS for asynchronous pub/sub delivery

Overall explanation

Correct options:

Deploy the containerized application to Amazon Elastic Kubernetes Service (Amazon EKS) using AWS Fargate to avoid managing EC2 nodes

Amazon EKS provides a Kubernetes-compatible, managed container orchestration service, which allows the company to reuse its existing Kubernetes manifests, Helm charts, and service configurations. By deploying workloads using AWS Fargate, the company avoids provisioning and managing EC2 worker nodes, aligning with the goal of reducing infrastructure overhead while maintaining scalability and agility.

Replace the current messaging system with Amazon MQ, a fully managed broker that supports AMQP natively. Integrate the application with the Amazon MQ endpoint without modifying the existing message format

Amazon MQ is a fully managed message broker service that supports AMQP (as well as MQTT, STOMP, and JMS). This allows the company to retain its existing messaging code and AMQP configuration without any major changes. Amazon MQ handles availability, durability, and patching, reducing the operational burden significantly.

Incorrect options:

Use Amazon Simple Queue Service (Amazon SQS) as the replacement for the AMQP message broker. Refactor the application to use SQS SDKs and polling logic - Amazon SQS does not support AMQP. To use SQS, the company would need to rewrite the messaging logic in the application to conform to SQS's polling-based message model and SDKs. This introduces significant rework and goes against the requirement of minimal code changes.

Deploy the application to Amazon ECS on EC2, and integrate the messaging workflow using Amazon SNS for asynchronous pub/sub delivery - Although ECS supports containerized workloads, using EC2 launch type introduces infrastructure management responsibilities (e.g., patching, scaling, monitoring). Additionally, Amazon SNS does not support AMQP and is intended for pub/sub, not point-to-point messaging. This option fails to meet both the protocol and operational efficiency requirements.

Run the application on Amazon EC2 Auto Scaling groups and use a self-hosted RabbitMQ instance on EC2 to preserve AMQP compatibility - Hosting RabbitMQ on EC2 can preserve AMQP compatibility, but it comes with the burden of managing the message broker, including patching, scaling, monitoring, and failover logic. This conflicts with the requirement of minimizing operational overhead, especially when a fully managed AMQP service (Amazon MQ) is available.

References:

https://docs.aws.amazon.com/eks/latest/userguide/fargate.html

https://docs.aws.amazon.com/amazon-mq/latest/developer-guide/welcome.html

https://docs.aws.amazon.com/AWSSimpleQueueService/latest/SQSDeveloperGuide/welcome.html

https://docs.aws.amazon.com/sns/latest/dg/welcome.html

Domain

Design Resilient Architectures

Question 25 Correct

A security consultant is designing a solution for a company that wants to provide developers with individual AWS accounts through AWS Organizations, while also maintaining standard security controls. Since the individual developers will have AWS account root user-level access to their own accounts, the consultant wants to ensure that the mandatory AWS CloudTrail configuration that is applied to new developer accounts is not modified.

Which of the following actions meets the given requirements?

Your answer is correct

Set up a service control policy (SCP) that prohibits changes to AWS CloudTrail, and attach it to the developer accounts

Set up an IAM policy that prohibits changes to AWS CloudTrail and attach it to the root user

Configure a new trail in AWS CloudTrail from within the developer accounts with the organization trails option enabled

Set up a service-linked role for AWS CloudTrail with a policy condition that allows changes only from an Amazon Resource Name (ARN) in the master account

Overall explanation

Correct option:

Set up a service control policy (SCP) that prohibits changes to AWS CloudTrail, and attach it to the developer accounts

Service control policy (SCP) is a type of organization policy that you can use to manage permissions in your organization. SCPs offer central control over the maximum available permissions for all accounts in your organization. SCPs help you to ensure your accounts stay within your organization's access control guidelines.

An SCP restricts permissions for IAM users and roles in member accounts, including the member account's root user. Any account has only those permissions permitted by every parent above it. If a permission is blocked at any level above the account, either implicitly (by not being included in an Allow policy statement) or explicitly (by being included in a Deny policy statement), a user or role in the affected account can't use that permission, even if the account administrator attaches the AdministratorAccess IAM policy with / permissions to the user.

SCPs don't affect users or roles in the management account. They affect only the member accounts in your organization.

Incorrect options:

Configure a new trail in AWS CloudTrail from within the developer accounts with the organization trails option enabled - Configuring each developer account individually is not a viable solution to start with. In addition, any configuration changes can be undone by the user once they are logged into their individual accounts as root users.

Set up an IAM policy that prohibits changes to AWS CloudTrail and attach it to the root user - The root user can modify this IAM policy itself, so this option is not correct.

Set up a service-linked role for AWS CloudTrail with a policy condition that allows changes only from an Amazon Resource Name (ARN) in the master account - A service-linked role is a unique type of IAM role that is linked directly to an AWS service. Service-linked roles are predefined by the service and include all the permissions that the service requires to call other AWS services on your behalf. The linked service also defines how you create, modify, and delete a service-linked role.

The linked service defines the permissions of its service-linked roles, and unless defined otherwise, only that service can assume the roles. The defined permissions include the trust policy and the permissions policy, and that permissions policy cannot be attached to any other entity such as the ARN in the master account.

Reference:

https://docs.aws.amazon.com/organizations/latest/userguide/orgs_manage_policies_scps.html

Domain

Design Secure Architectures

Question 26 **Correct**

A financial services company has developed its flagship application on AWS Cloud with data security requirements such that the encryption key must be stored in a custom application running on-premises. The company wants to offload the data storage as well as the encryption process to Amazon S3 but continue to use the existing encryption key.

Which of the following Amazon S3 encryption options allows the company to leverage Amazon S3 for storing data with given constraints?

Server-Side Encryption with AWS Key Management Service (AWS KMS) keys (SSE-KMS)

Your answer is correct

Server-Side Encryption with Customer-Provided Keys (SSE-C)

Client-Side Encryption with data encryption is done on the client-side before sending it to Amazon S3

Server-Side Encryption with Amazon S3 managed keys (SSE-S3)

Overall explanation

Correct option:

Server-Side Encryption with Customer-Provided Keys (SSE-C)

You have the following options for protecting data at rest in Amazon S3:

Server-Side Encryption – Request Amazon S3 to encrypt your object before saving it on disks in its data centers and then decrypt it when you download the objects.

Client-Side Encryption – Encrypt data client-side and upload the encrypted data to Amazon S3. In this case, you manage the encryption process, the encryption keys, and related tools.

For the given use-case, the company wants to manage the encryption keys via its custom application and let Amazon S3 manage the encryption, therefore you must use Server-Side Encryption with Customer-Provided Keys (SSE-C).

Please review these three options for Server Side Encryption on Amazon S3:

Protecting data with server-side encryption

PDF | RSS

Important

Amazon S3 now applies server-side encryption with Amazon S3 managed keys (SSE-S3) as the base level of encryption for every bucket in Amazon S3. Starting January 5, 2023, all new object uploads to Amazon S3 are automatically encrypted at no additional cost and with no impact on performance. The automatic encryption status for S3 bucket default encryption configuration and for new object uploads is available in AWS CloudTrail logs, S3 Inventory, S3 Storage Lens, the Amazon S3 console, and as an additional Amazon S3 API response header in the AWS Command Line Interface and AWS SDKs. For more information, see Default encryption FAQ.

Server-side encryption is the encryption of data at its destination by the application or service that receives it. Amazon S3 encrypts your data at the object level as it writes it to disks in AWS data centers and decrypts it for you when you access it. As long as you authenticate your request and you have access permissions, there is no difference in the way you access encrypted or unencrypted objects. For example, if you share your objects by using a presigned URL, that URL works the same way for both encrypted and unencrypted objects. Additionally, when you list objects in your bucket, the list API operations return a list of all objects, regardless of whether they are encrypted.

All Amazon S3 buckets have encryption configured by default, and all new objects that are uploaded to an S3 bucket are automatically encrypted at rest. Server-side encryption with Amazon S3 managed keys (SSE-S3) is the default encryption configuration for every bucket in Amazon S3. To use a different type of encryption, you can either specify the type of server-side encryption to use in your S3 `PUT` requests, or you can set the default encryption configuration in the destination bucket.

If you want to specify a different encryption type in your `PUT` requests, you can use server-side encryption with AWS Key Management Service (AWS KMS) keys (SSE-KMS), dual-layer server-side encryption with AWS KMS keys (DSSE-KMS), or server-side encryption with customer-provided keys (SSE-C). If you want to set a different default encryption configuration in the destination bucket, you can use SSE-KMS or DSSE-KMS.

Note

You can't apply different types of server-side encryption to the same object simultaneously.

If you need to encrypt your existing objects, use S3 Batch Operations and S3 Inventory. For more information, see Encrypting objects with Amazon S3 Batch Operations and Performing large-scale batch operations on Amazon S3 objects.

You have four mutually exclusive options for server-side encryption, depending on how you choose to manage the encryption keys and the number of encryption layers that you want to apply.

Server-side encryption with Amazon S3 managed keys (SSE-S3)

All Amazon S3 buckets have encryption configured by default. The default option for server-side encryption is with Amazon S3 managed keys (SSE-S3). Each object is encrypted with a unique key. As an additional safeguard, SSE-S3 encrypts the key itself with a root key that it regularly rotates. SSE-S3 uses one of the strongest block ciphers available, 256-bit Advanced Encryption Standard (AES-256), to encrypt your data. For more information, see Using server-side encryption with Amazon S3 managed keys (SSE-S3).

Server-side encryption with AWS Key Management Service (AWS KMS) keys (SSE-KMS)

Server-side encryption with AWS KMS keys (SSE-KMS) is provided through an integration of the AWS KMS service with Amazon S3. With AWS KMS, you have more control over your keys. For example, you can view separate keys, edit control policies, and follow the keys in AWS CloudTrail. Additionally, you can create and manage customer managed keys or use AWS managed keys that are unique to you, your service, and your Region. For more information, see Using server-side encryption with AWS KMS keys (SSE-KMS).

Dual-layer server-side encryption with AWS Key Management Service (AWS KMS) keys (DSSE-KMS)

Dual-layer server-side encryption with AWS KMS keys (DSSE-KMS) is similar to SSE-KMS, but DSSE-KMS applies two individual layers of object-level encryption instead of one layer. Because both layers of encryption are applied to an object on the server side, you can use a wide range of AWS services and tools to analyze data in S3 while using an encryption method that can satisfy your compliance requirements. For more information, see Using dual-layer server-side encryption with AWS KMS keys (DSSE-KMS).

Server-side encryption with customer-provided keys (SSE-C)

With server-side encryption with customer-provided keys (SSE-C), you manage the encryption keys, and Amazon S3 manages the encryption as it writes to disks and the decryption when you access your objects. For more information, see Using server-side encryption with customer-provided keys (SSE-C).

via - https://docs.aws.amazon.com/AmazonS3/latest/dev/serv-side-encryption.html

Incorrect options:

Server-Side Encryption with Amazon S3 managed keys (SSE-S3) - When you use Server-Side Encryption with Amazon S3-Managed Keys (SSE-S3), each object is encrypted with a unique key. As an additional safeguard, it encrypts the key itself with a master key that it regularly rotates. So this option is incorrect.

Server-Side Encryption with AWS Key Management Service (AWS KMS) keys (SSE-KMS) - Unless you specify otherwise, buckets use SSE-S3 by default to encrypt objects. However, you can choose to configure buckets to use server-side encryption with AWS Key Management Service (AWS KMS) keys (SSE-KMS) instead. Amazon S3 uses server-side encryption with AWS KMS (SSE-KMS) to encrypt your S3 object data. Also, when SSE-KMS is requested for the object, the S3 checksum as part of the object's metadata, is stored in encrypted form.

Client-Side Encryption with data encryption is done on the client-side before sending it to Amazon S3 - You can encrypt the data client-side and upload the encrypted data to Amazon S3. In this case, you manage the encryption process, the encryption keys, and related tools.

Reference:

https://docs.aws.amazon.com/AmazonS3/latest/dev/serv-side-encryption.html

Domain

Design Secure Architectures

Question 27 Incorrect

An application is currently hosted on four Amazon EC2 instances (behind Application Load Balancer) deployed in a single Availability Zone (AZ). To maintain an acceptable level of end-user experience, the application needs at least 4 instances to be always available.

As a solutions architect, which of the following would you recommend so that the application achieves high availability with MINIMUM cost?

Deploy the instances in two Availability Zones (AZs). Launch four instances in each Availability Zone (AZ)

Your answer is incorrect

Deploy the instances in two Availability Zones (AZs). Launch two instances in each Availability Zone (AZ)

Deploy the instances in one Availability Zones. Launch two instances in the Availability Zone (AZ)

Correct answer

Deploy the instances in three Availability Zones (AZs). Launch two instances in each Availability Zone (AZ)

Overall explanation

Correct option:

Deploy the instances in three Availability Zones (AZs). Launch two instances in each Availability Zone (AZ)

The correct option is to deploy the instances in three Availability Zones (AZs) and launch two instances in each Availability Zone (AZ). Even if one of the AZs goes out of service, still we shall have 4 instances available and the application can maintain an acceptable level of end-user experience. Therefore, we can achieve high availability with just 6 instances in this case.

Incorrect options:

Deploy the instances in two Availability Zones (AZs). Launch two instances in each Availability Zone (AZ) - When we launch two instances in two AZs, we run the risk of falling below the minimum acceptable threshold of 4 instances if one of the AZs fails. So this option is ruled out.

Deploy the instances in two Availability Zones (AZs). Launch four instances in each Availability Zone (AZ) - When we launch four instances in two AZs, we have to bear costs for 8 instances which is NOT cost-optimal. So this option is ruled out.

Deploy the instances in one Availability Zones. Launch two instances in the Availability Zone (AZ) - We can't have just two instances in a single AZ as that is below the minimum acceptable threshold of 4 instances.

Domain

Design Cost-Optimized Architectures

Question 28 **Correct**

A media company has created an AWS Direct Connect connection for migrating its flagship application to the AWS Cloud. The on-premises application writes hundreds of video files into a mounted NFS file system daily. Post-migration, the company will host the application on an Amazon EC2 instance with a mounted Amazon Elastic File System (Amazon EFS) file system. Before the migration cutover, the company must build a process that will replicate the newly created on-premises video files to the Amazon EFS file system.

Which of the following represents the MOST operationally efficient way to meet this requirement?

Your answer is correct

Configure an AWS DataSync agent on the on-premises server that has access to the NFS file system. Transfer data over the AWS Direct Connect connection to an AWS PrivateLink interface VPC endpoint for Amazon EFS by using a private VIF. Set up an AWS DataSync scheduled task to send the video files to the Amazon EFS file system every 24 hours

Configure an AWS DataSync agent on the on-premises server that has access to the NFS file system. Transfer data over the AWS Direct Connect connection to an Amazon S3 bucket by using a VPC gateway endpoint for Amazon S3. Set up an AWS Lambda function to process event notifications from Amazon S3 and copy the video files from Amazon S3 to the Amazon EFS file system

Configure an AWS DataSync agent on the on-premises server that has access to the NFS file system. Transfer data over the AWS Direct Connect connection to an AWS VPC peering endpoint for Amazon EFS by using a private VIF. Set up an AWS DataSync scheduled task to send the video files to the Amazon EFS file system every 24 hours

Configure an AWS DataSync agent on the on-premises server that has access to the NFS file system. Transfer data over the AWS Direct Connect connection to an Amazon S3 bucket by using public VIF. Set up an AWS Lambda function to process

event notifications from Amazon S3 and copy the video files from Amazon S3 to the Amazon EFS file system

Overall explanation

Correct option:

Configure an AWS DataSync agent on the on-premises server that has access to the NFS file system. Transfer data over the AWS Direct Connect connection to an AWS PrivateLink interface VPC endpoint for Amazon EFS by using a private VIF. Set up an AWS DataSync scheduled task to send the video files to the Amazon EFS file system every 24 hours

AWS DataSync is an online data transfer service that simplifies, automates, and accelerates copying large amounts of data between on-premises storage systems and AWS Storage services, as well as between AWS Storage services.

You can use AWS DataSync to migrate data located on-premises, at the edge, or in other clouds to Amazon S3, Amazon EFS, Amazon FSx for Windows File Server, Amazon FSx for Lustre, Amazon FSx for OpenZFS, and Amazon FSx for NetApp ONTAP.

AWS DataSync:

via - https://aws.amazon.com/datasync/

To establish a private connection between your virtual private cloud (VPC) and the Amazon EFS API, you can create an interface VPC endpoint. You can also access the interface VPC endpoint from on-premises environments or other VPCs using AWS VPN, AWS Direct Connect, or VPC peering.

AWS Direct Connect provides three types of virtual interfaces: public, private, and transit.

AWS Direct Connect VIFs:

Which type of Direct Connect virtual interface should I use to connect different AWS resources?

Last updated: 2022-01-21

AWS Direct Connect provides three types of virtual interfaces: public, private, and transit. How do I determine which type I should use to connect different AWS resources?

Resolution

Use the following Direct Connect virtual interface based on your use case.

Public virtual interface

To connect to AWS resources that are reachable by a public IP address such as an Amazon Simple Storage Service (Amazon S3) bucket or AWS public endpoints, use a public virtual interface. With a public virtual interface, you can:

- Connect to all AWS public IP addresses globally.
- Create public virtual interfaces in any Direct Connect location to receive Amazon's global IP routes.
- Access publicly routable Amazon services in any AWS Region (except the AWS China Region).

Private virtual interface

To connect to your resources hosted in an Amazon Virtual Private Cloud (Amazon VPC) using their private IP addresses, use a private virtual interface. With a private virtual interface, you can:

- Connect VPC resources such as Amazon Elastic Compute Cloud (Amazon EC2) instances or load balancers on your private IP address or endpoint.
- Connect a private virtual interface to a Direct Connect gateway. Then, associate the Direct Connect gateway with one or more virtual private gateways in any AWS Region (except the AWS China Region).
- Connect to multiple Amazon VPCs in any AWS Region (except the AWS China Region), because a virtual private gateway is associated with a single VPC.

Note: For a private virtual interface, AWS advertises the VPC CIDR only over the Border Gateway Protocol (BGP) neighbor. AWS can't advertise or suppress specific subnet blocks in the Amazon VPC for a private virtual interface.

Transit virtual interface

To connect to your resources hosted in an Amazon VPC (using their private IP addresses) through a transit gateway, use a transit virtual interface. With a transit virtual interface, you can:

- Connect multiple Amazon VPCs in the same or different AWS account using Direct Connect.
- Associate up to three transit gateways in any AWS Region when you use a transit virtual interface to connect to a Direct Connect gateway.
- Attach Amazon VPCs in the same AWS Region to the transit gateway. Then, access multiple VPCs in different AWS accounts in the same AWS Region using a transit virtual interface.

Note: For transit virtual interface, AWS advertises only routes that you specify in the allowed prefixes list on the Direct Connect gateway. For a list of all AWS Regions that offer Direct Connect support for AWS Transit Gateway, see AWS Transit Gateway support.

via - https://aws.amazon.com/premiumsupport/knowledge-center/public-private-interface-dx/

For the given use case, you can send data over the Direct Connect connection to an AWS PrivateLink interface VPC endpoint for Amazon EFS by using a private VIF.

Using task scheduling in AWS DataSync, you can periodically execute a transfer task from your source storage system to the destination. You can use the DataSync scheduled task to send the video files to the Amazon EFS file system every 24 hours.

Incorrect options:

Configure an AWS DataSync agent on the on-premises server that has access to the NFS file system. Transfer data over the AWS Direct Connect connection to an AWS VPC peering endpoint for Amazon EFS by using a private VIF. Set up an AWS DataSync scheduled task to send the video files to the Amazon EFS file system every 24 hours - A VPC peering connection is a networking connection between two VPCs that enables you to route traffic between them privately. You cannot use VPC peering to transfer data over the Direct Connect connection from the on-premises systems to AWS. So this option is incorrect.

Configure an AWS DataSync agent on the on-premises server that has access to the NFS file system. Transfer data over the AWS Direct Connect connection to an Amazon S3 bucket by using public VIF. Set up an AWS Lambda function to process event notifications from Amazon S3 and copy the video files from Amazon S3 to the Amazon EFS file system - You can use a public virtual interface to connect to AWS resources that are reachable by a public IP address such as an Amazon Simple Storage Service (Amazon S3) bucket or AWS public endpoints. Although it is theoretically possible to set up this solution, however, it is not the most operationally efficient solution, since it involves sending data via AWS DataSync to Amazon S3 and then in turn using an AWS Lambda function to finally send data to Amazon EFS.

Configure an AWS DataSync agent on the on-premises server that has access to the NFS file system. Transfer data over the AWS Direct Connect connection to an Amazon S3 bucket by using a VPC gateway endpoint for Amazon S3. Set up an AWS Lambda function to process event notifications from Amazon S3 and copy the video files from Amazon S3 to the Amazon EFS file system - You can access Amazon S3 from your VPC using gateway VPC endpoints. You cannot use the Amazon S3 gateway endpoint to transfer data over the AWS Direct Connect connection from the on-premises systems to Amazon S3. So this option is incorrect.

References:

https://aws.amazon.com/datasync/

https://aws.amazon.com/blogs/storage/transferring-files-from-on-premises-to-aws-and-back-without-leaving-your-vpc-using-aws-datasync/

https://docs.aws.amazon.com/efs/latest/ug/efs-vpc-endpoints.html

https://aws.amazon.com/datasync/faqs/

https://aws.amazon.com/premiumsupport/knowledge-center/public-private-interface-dx/

https://docs.aws.amazon.com/datasync/latest/userguide/task-scheduling.html

Domain

Design High-Performing Architectures

Question 29 **Incorrect**

You have a team of developers in your company, and you would like to ensure they can quickly experiment with AWS Managed Policies by attaching them to their accounts, but you would like to prevent them from doing an escalation of privileges, by granting themselves the `AdministratorAccess` managed policy. How should you proceed?

Attach an IAM policy to your developers, that prevents them from attaching the `AdministratorAccess` policy

Your answer is incorrect

Put the developers into an IAM group, and then define an IAM permission boundary on the group that will restrict the managed policies they can attach to themselves

Correct answer

For each developer, define an IAM permission boundary that will restrict the managed policies they can attach to themselves

Create a Service Control Policy (SCP) on your AWS account that restricts developers from attaching themselves the `AdministratorAccess` policy

Overall explanation

Correct option:

For each developer, define an IAM permission boundary that will restrict the managed policies they can attach to themselves

AWS supports permissions boundaries for IAM entities (users or roles). A permissions boundary is an advanced feature for using a managed policy to set the maximum permissions that an identity-based policy can grant to an IAM entity. An entity's permissions boundary allows it to perform only the actions that are allowed by both its identity-based policies and its permissions boundaries. Here we have to use an IAM permission boundary. They can only be applied to roles or users, not IAM groups.

Permissions boundaries for IAM entities:

Evaluating Effective Permissions with Boundaries

The permissions boundary for an IAM entity (user or role) sets the maximum permissions that the entity can have. This can change the effective permissions for that user or role. The effective permissions for an entity are the permissions that are granted by all the policies that affect the user or role. Within an account, the permissions for an entity can be affected by identity-based policies, resource-based policies, permissions boundaries, Organizations SCPs, or session policies. For more information about the different types of policies, see Policies and Permissions.

If any one of these policy types explicitly denies access for an operation, then the request is denied. The permissions granted to an entity by multiple permissions types are more complex. For more details about how AWS evaluates policies, see Policy Evaluation Logic.

Identity-based policies with boundaries – Identity-based policies are inline or managed policies that are attached to a user, group of users, or role. Identity-based policies grant permission to the entity, and permissions boundaries limit those permissions. The effective permissions are the intersection of both policy types. An explicit deny in either of these policies overrides the allow.

via - https://docs.aws.amazon.com/IAM/latest/UserGuide/access_policies_boundaries.html

Incorrect options:

Create a Service Control Policy (SCP) on your AWS account that restricts developers from attaching themselves the `AdministratorAccess` policy - Service control policy (SCP) is one type of policy that you can use to manage your organization. SCPs offer central control over the maximum available permissions for all accounts in your organization, allowing you to ensure your accounts stay within your organization's access control guidelines. SCPs are available only in an organization that has all features enabled. SCPs aren't available if your organization has enabled only the consolidated billing features. Attaching an SCP to an AWS Organizations entity (root, OU, or account) defines a guardrail for what actions the principals can perform. If you consider this option, since AWS Organizations is not mentioned in this question, so we can't apply an SCP.

Attach an IAM policy to your developers, that prevents them from attaching the `AdministratorAccess` policy - This option is incorrect as the developers can remove this policy from themselves and escalate their privileges.

Put the developers into an IAM group, and then define an IAM permission boundary on the group that will restrict the managed policies they can attach to themselves - IAM permission boundary can only be applied to roles or users, not IAM groups. Hence this option is incorrect.

References:

https://docs.aws.amazon.com/IAM/latest/UserGuide/access_policies_boundaries.html

https://docs.aws.amazon.com/organizations/latest/userguide/orgs_manage_policies_scp.html

https://docs.aws.amazon.com/IAM/latest/UserGuide/access_policies_boundaries.html

Domain

Design Secure Architectures

Question 30 Incorrect

A retail company wants to share sensitive accounting data that is stored in an Amazon RDS database instance with an external auditor. The auditor has its own AWS account and needs its own copy of the database.

Which of the following would you recommend to securely share the database with the auditor?

Export the database contents to text files, store the files in Amazon S3, and create a new IAM user for the auditor with access to that bucket

Create a snapshot of the database in Amazon S3 and assign an IAM role to the auditor to grant access to the object in that bucket

Your answer is incorrect

Set up a read replica of the database and configure IAM standard database authentication to grant the auditor access

Correct answer

Create an encrypted snapshot of the database, share the snapshot, and allow access to the AWS Key Management Service (AWS KMS) encryption key

Overall explanation

Correct option:

Create an encrypted snapshot of the database, share the snapshot, and allow access to the AWS Key Management Service (AWS KMS) encryption key

You can share the AWS Key Management Service (AWS KMS) key that was used to encrypt the snapshot with any accounts that you want to be able to access the snapshot. You can share AWS KMS Key with another AWS account by adding the other account to the AWS KMS key policy.

Making an encrypted snapshot of the database will give the auditor a copy of the database, as required for the given use case.

Incorrect options:

Create a snapshot of the database in Amazon S3 and assign an IAM role to the auditor to grant access to the object in that bucket - Amazon RDS stores the DB snapshots in the Amazon S3 bucket belonging to the same AWS region where the Amazon RDS instance is located. Amazon RDS stores these on your behalf and you do not have direct access to these snapshots in Amazon S3, so it's not possible to grant access to the snapshot objects in Amazon S3.

Export the database contents to text files, store the files in Amazon S3, and create a new IAM user for the auditor with access to that bucket - This solution is feasible though not optimal. It requires a lot of unnecessary work and is difficult to audit when such bulk data is exported into text files.

Set up a read replica of the database and configure IAM standard database authentication to grant the auditor access - Read Replicas make it easy to elastically scale out beyond the capacity constraints of a single DB instance for read-heavy database workloads. Creating Read Replicas for audit purposes is overkill. Also, the question mentions that the auditor needs to have their own copy of the database, which is not possible with replicas.

Reference:

https://docs.aws.amazon.com/AmazonRDS/latest/UserGuide/USER_ShareSnapshot.html

Domain

Design Secure Architectures

Question 31 **Incorrect**

You would like to use AWS Snowball to move on-premises backups into a long term archival tier on AWS. Which solution provides the MOST cost savings?

Correct answer

Create an AWS Snowball job and target an Amazon S3 bucket. Create a lifecycle policy to transition this data to Amazon S3 Glacier Deep Archive on the same day

Create an AWS Snowball job and target a Amazon S3 Glacier Vault

Your answer is incorrect

Create a AWS Snowball job and target an Amazon S3 Glacier Deep Archive Vault

Create an AWS Snowball job and target an Amazon S3 bucket. Create a lifecycle policy to transition this data to Amazon S3 Glacier on the same day

Overall explanation

Correct option:

Create an AWS Snowball job and target an Amazon S3 bucket. Create a lifecycle policy to transition this data to Amazon S3 Glacier Deep Archive on the same day

AWS Snowball, a part of the AWS Snow Family, is a data migration and edge computing device that comes in two options. Snowball Edge Storage Optimized devices provide both block storage and Amazon S3-compatible object storage, and 40 vCPUs. They are well suited for local storage

and large scale data transfer. AWS Snowball Edge Compute Optimized devices provide 52 vCPUs, block and object storage, and an optional GPU for use cases like advanced machine learning and full-motion video analysis in disconnected environments.

AWS Snowball Edge Storage Optimized is the optimal choice if you need to securely and quickly transfer dozens of terabytes to petabytes of data to AWS. It provides up to 80 terabytes of usable HDD storage, 40 vCPUs, 1 terabyte of SATA SSD storage, and up to 40 gigabytes network connectivity to address large scale data transfer and pre-processing use cases.

The original AWS Snowball devices were transitioned out of service and AWS Snowball Edge Storage Optimized are now the primary devices used for data transfer. You may see the AWS Snowball device on the exam, just remember that the original AWS Snowball device had 80 terabytes of storage space.

For this scenario, you will want to minimize the time spent in Amazon S3 Standard for all files to avoid unintended Amazon S3 Standard storage charges. To do this, AWS recommends using a zero-day lifecycle policy. From a cost perspective, when using a zero-day lifecycle policy, you are only charged Amazon S3 Glacier Deep Archive rates. When billed, the lifecycle policy is accounted for first, and if the destination is Amazon S3 Glacier Deep Archive, you are charged Amazon S3 Glacier Deep Archive rates for the transferred files.

You can't move data directly from AWS Snowball into Amazon S3 Glacier, you need to go through Amazon S3 first, and then use a lifecycle policy. So this option is correct.

Incorrect options:

Create an AWS Snowball job and target a Amazon S3 Glacier Vault

Create a AWS Snowball job and target an Amazon S3 Glacier Deep Archive Vault

Amazon S3 Glacier and S3 Glacier Deep Archive are a secure, durable, and extremely low-cost Amazon S3 cloud storage classes for data archiving and long-term backup. They are designed to deliver 99.999999999% durability and provide comprehensive security and compliance capabilities that can help meet even the most stringent regulatory requirements. Finally, Amazon S3 Glacier Deep Archive provides more cost savings than Amazon S3 Glacier.

Both these options are incorrect as you can't move data directly from AWS Snowball into a Amazon S3 Glacier Vault or a Glacier Deep Archive Vault. You need to go through Amazon S3 first and then use a lifecycle policy.

Create an AWS Snowball job and target an Amazon S3 bucket. Create a lifecycle policy to transition this data to Amazon S3 Glacier on the same day - As Amazon S3 Glacier Deep Archive provides more cost savings than Amazon S3 Glacier, you should use Amazon S3 Glacier Deep Archive for long term archival for this use-case.

References:

https://aws.amazon.com/snowball/features/

https://aws.amazon.com/glacier/

Domain

Design Cost-Optimized Architectures

Question 32 Correct

A silicon valley based startup has a two-tier architecture using Amazon EC2 instances for its flagship application. The web servers (listening on port 443), which have been assigned security group A, are in public subnets across two Availability Zones (AZs) and the MSSQL based database instances (listening on port 1433), which have been assigned security group B, are in two private subnets across two Availability Zones (AZs). The DevOps team wants to review the security configurations of the application architecture.

As a solutions architect, which of the following options would you select as the MOST secure configuration? (Select two)

For security group A: Add an inbound rule that allows traffic from all sources on port 443. Add an outbound rule with the destination as security group B on port 443

For security group B: Add an inbound rule that allows traffic only from security group A on port 443

Your selection is correct

For security group B: Add an inbound rule that allows traffic only from security group A on port 1433

Your selection is correct

For security group A: Add an inbound rule that allows traffic from all sources on port 443. Add an outbound rule with the destination as security group B on port 1433

For security group B: Add an inbound rule that allows traffic only from all sources on port 1433

Overall explanation

Correct options:

For security group A: Add an inbound rule that allows traffic from all sources on port 443. Add an outbound rule with the destination as security group B on port 1433

For security group B: Add an inbound rule that allows traffic only from security group A on port 1433

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

The following are the characteristics of security group rules:

By default, security groups allow all outbound traffic.

Security group rules are always permissive; you can't create rules that deny access.

Security groups are stateful

The MOST secure configuration for the given use case is:

For security group A: Add an inbound rule that allows traffic from all sources on port 443. Add an outbound rule with the destination as security group B on port 1433

The above rules make sure that web servers are listening for traffic on all sources on the HTTPS protocol on port 443. The web servers only allow outbound traffic to MSSQL servers in Security Group B on port 1433.

For security group B: Add an inbound rule that allows traffic only from security group A on port 1433. The above rule makes sure that the MSSQL servers only accept traffic from web servers in security group A on port 1433.

Therefore, both of these options are correct.

Incorrect options:

For security group A: Add an inbound rule that allows traffic from all sources on port 443. Add an outbound rule with the destination as security group B on port 443 - As the MSSQL based database instances are listening on port 1433, therefore for security group A, the outbound rule should be added on port 443 with the destination as security group B.

For security group B: Add an inbound rule that allows traffic only from all sources on port 1433 - The inbound rule should allow traffic only from security group A on port 1433. Allowing traffic from all sources will compromise security.

For security group B: Add an inbound rule that allows traffic only from security group A on port 443 - The inbound rule should allow traffic only from security group A on port 1433 because the MSSQL based database instances are listening on port 1433.

Reference:

https://docs.aws.amazon.com/AWSEC2/latest/UserGuide/ec2-security-groups.html

Domain

Design Secure Architectures

Question 33 **Correct**

Amazon EC2 Auto Scaling needs to terminate an instance from Availability Zone (AZ) `us-east-1a` as it has the most number of instances amongst the Availability Zone (AZs) being used currently. There are 4 instances in the Availability Zone (AZ) `us-east-1a` like so: Instance A has the oldest launch template, Instance B has the oldest launch configuration, Instance C has the newest launch configuration and Instance D is closest to the next billing hour.

Which of the following instances would be terminated per the default termination policy?

Instance D

Instance C

Instance A

Your answer is correct

Instance B

Overall explanation

Correct option:

Instance B

Per the default termination policy, the first priority is given to any allocation strategy for On-Demand vs Spot instances. As no such information has been provided for the given use-case, so this criterion can be ignored. The next priority is to consider any instance with the oldest launch template unless there is an instance that uses a launch configuration. So this rules out Instance A. Next, you need to consider any instance which has the oldest launch configuration. This implies Instance B will be selected for termination and Instance C will also be ruled out as it has the newest launch configuration. Instance D, which is closest to the next billing hour, is not selected as this criterion is last in the order of priority.

Please see this note for a deep-dive on the default termination policy:

Default termination policy

The default termination policy is designed to help ensure that your instances span Availability Zones evenly for high availability. The default policy is kept generic and flexible to cover a range of scenarios.

Before Amazon EC2 Auto Scaling selects an instance to terminate, it first determines which Availability Zones have the most instances, and at least one instance that is not protected from scale in.

Within the selected Availability Zone, the default termination policy behavior is as follows:

1. Determine which instances to terminate so as to align the remaining instances to the allocation strategy for the On-Demand or Spot Instance that is terminating. This only applies to an Auto Scaling group that specifies a mixed instances policy, which uses allocation strategies.

   For example, after your instances launch, you change the priority order of your preferred instance types. When a scale-in event occurs, Amazon EC2 Auto Scaling tries to gradually shift the On-Demand Instances away from instance types that are lower priority.
2. Determine whether any of the instances use the oldest launch template or configuration:
   a. [For Auto Scaling groups that use a launch template]

      Determine whether any of the instances use the oldest launch template unless there are instances that use a launch configuration. Amazon EC2 Auto Scaling terminates instances that use a launch configuration before instances that use a launch template.
   b. [For Auto Scaling groups that use a launch configuration]

      Determine whether any of the instances use the oldest launch configuration.
3. After applying all of the above criteria, if there are multiple unprotected instances to terminate, determine which instances are closest to the next billing hour. If there are multiple unprotected instances closest to the next billing hour, terminate one of these instances at random.

   Note that terminating the instance closest to the next billing hour helps you maximize the use of your instances that have an hourly charge. Alternatively, if your Auto Scaling group uses Amazon Linux or Ubuntu, your EC2 usage is billed in one-second increments. For more information, see Amazon EC2 pricing.

via - https://docs.aws.amazon.com/autoscaling/ec2/userguide/as-instance-termination.html

Incorrect options:

Instance A

Instance C

Instance D

These three options contradict the explanation provided above, so these options are incorrect.

Reference:

https://docs.aws.amazon.com/autoscaling/ec2/userguide/as-instance-termination.html

Domain

Design Cost-Optimized Architectures

Question 34 Correct

A company is looking at storing their less frequently accessed files on AWS that can be concurrently accessed by hundreds of Amazon EC2 instances. The company needs the most cost-effective file storage service that provides immediate access to data whenever needed.

Which of the following options represents the best solution for the given requirements?

Amazon Elastic Block Store (EBS)

Your answer is correct

Amazon Elastic File System (EFS) Standard–IA storage class

Amazon Elastic File System (EFS) Standard storage class

Amazon S3 Standard-Infrequent Access (S3 Standard-IA) storage class

Overall explanation

Correct option:

Amazon Elastic File System (EFS) Standard–IA storage class - Amazon EFS is a file storage service for use with Amazon compute (EC2, containers, serverless) and on-premises servers. Amazon EFS provides a file system interface, file system access semantics (such as strong consistency and file locking), and concurrently accessible storage for up to thousands of Amazon EC2 instances.

The Amazon S3 Standard–IA storage class reduces storage costs for files that are not accessed every day. It does this without sacrificing the high availability, high durability, elasticity, and POSIX file system access that Amazon EFS provides. AWS recommends Standard-IA storage if you need

your full dataset to be readily accessible and want to automatically save on storage costs for files that are less frequently accessed.

Incorrect options:

Amazon S3 Standard-Infrequent Access (S3 Standard-IA) storage class - Amazon S3 is an object storage service. Amazon S3 makes data available through an Internet API that can be accessed anywhere. It is not a file storage service, as is needed in the use case.

Amazon Elastic File System (EFS) Standard storage class - Amazon EFS Standard storage classes are ideal for workloads that require the highest levels of durability and availability. The Amazon EFS Standard storage class is used for frequently accessed files. It is the storage class to which customer data is initially written for Standard storage classes. The company is also looking at cutting costs by optimally storing the infrequently accessed data. Hence, Amazon EFS standard storage class is not the right solution for the given use case.

Amazon Elastic Block Store (EBS) - Amazon EBS is a block-level storage service for use with Amazon EC2. Amazon EBS can deliver performance for workloads that require the lowest latency access to data from a single Amazon EC2 instance. Amazon EBS volume cannot be accessed by hundreds of Amazon EC2 instances concurrently. It is not a file storage service, as is needed in the use case.

Reference:

https://docs.aws.amazon.com/efs/latest/ug/storage-classes.html

Domain

Design Cost-Optimized Architectures

Question 35 **Incorrect**

An IT company wants to optimize the costs incurred on its fleet of 100 Amazon EC2 instances for the next year. Based on historical analyses, the engineering team observed that 70 of these instances handle the compute services of its flagship application and need to be always available. The other 30 instances are used to handle batch jobs that can afford a delay in processing.

As a solutions architect, which of the following would you recommend as the MOST cost-optimal solution?

Correct answer

Purchase 70 reserved instances (RIs) and 30 spot instances

Purchase 70 on-demand instances and 30 spot instances

Purchase 70 on-demand instances and 30 reserved instances

Your answer is incorrect

Purchase 70 reserved instances and 30 on-demand instances

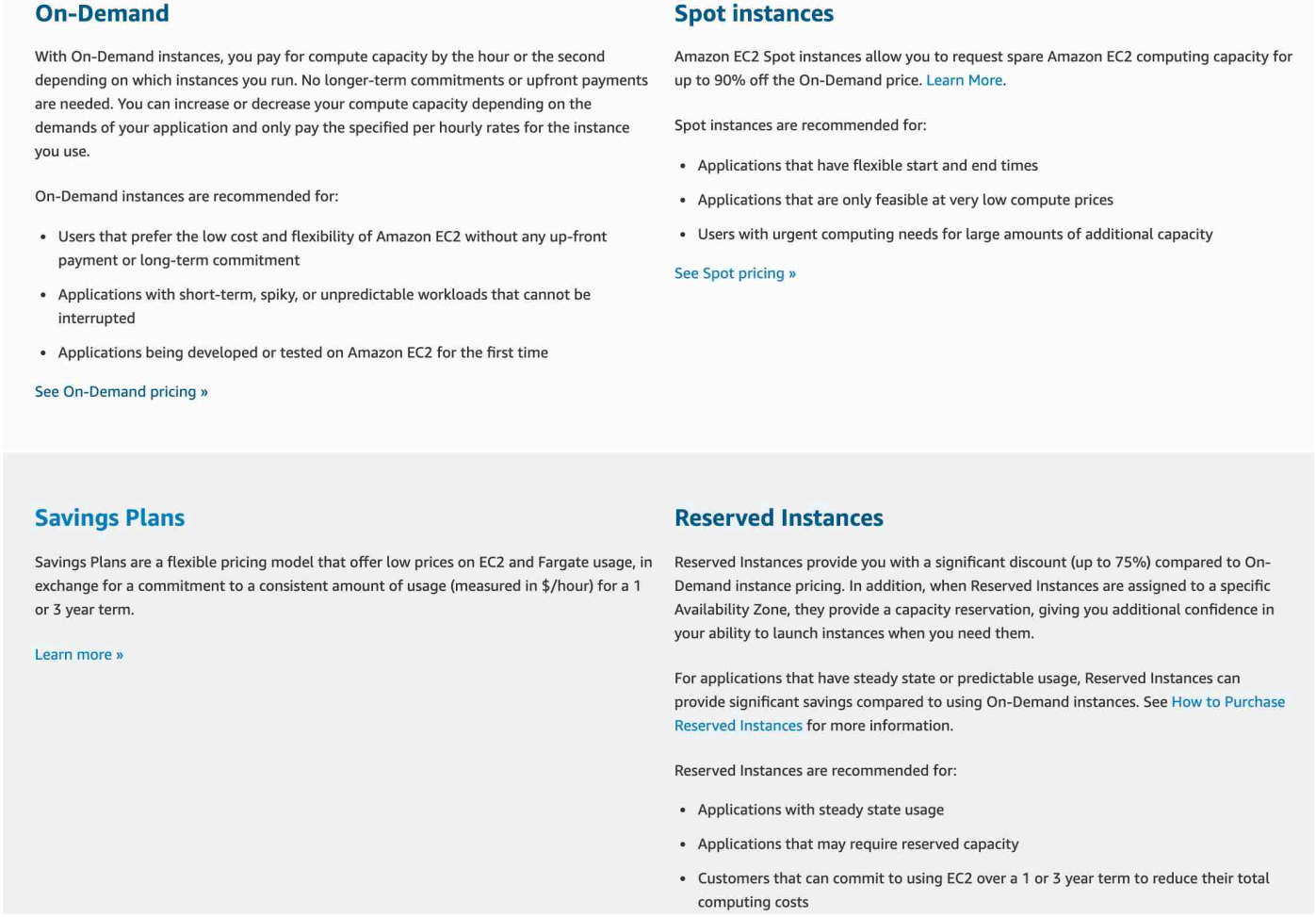
Overall explanation

Correct option:

Purchase 70 reserved instances (RIs) and 30 spot instances

As 70 instances need to be always available, these can be purchased as reserved instances for a one-year duration. The other 30 instances responsible for the batch job can be purchased as spot instances. Even if some of the spot instances are interrupted, other spot instances can continue with the job.

Please see this detailed overview of various types of Amazon EC2 instances from a pricing perspective:



via - https://aws.amazon.com/ec2/pricing/

Incorrect options:

Purchase 70 on-demand instances and 30 spot instances

Purchase 70 on-demand instances and 30 reserved instances

Purchasing 70 on-demand instances would be costlier than 70 reserved instances, so these two options are ruled out.

Purchase 70 reserved instances and 30 on-demand instances - Purchasing 30 instances as on-demand instances to handle the batch jobs would not be cost-optimal as these instances don't need to be always available. Spot instances are better at handling such batch jobs. So this option is not correct.

Reference:

https://aws.amazon.com/ec2/pricing/

Domain

Design Cost-Optimized Architectures

Question 36 Incorrect

A wildlife research organization uses IoT-based motion sensors attached to thousands of migrating animals to monitor their movement across regions. Every few minutes, a sensor checks for significant movement and sends updated location data to a backend application running on Amazon EC2 instances spread across multiple Availability Zones in a single AWS Region. Recently, an unexpected surge in motion data overwhelmed the application, leading to lost location records with no mechanism to replay missed data. A solutions architect must redesign the ingestion mechanism to prevent future data loss and to minimize operational overhead.

What should the solutions architect do to meet these requirements?

Set up a containerized service using Amazon ECS with an internal queue built into the application layer. Configure the motion sensors to send location updates directly to the container endpoints

Correct answer

Create an Amazon Simple Queue Service (Amazon SQS) queue to buffer the incoming location data. Configure the backend application to poll the queue and process messages

Deploy an Amazon Data Firehose delivery stream to collect the motion data. Configure it to deliver data to an S3 bucket where the application scans and processes the files periodically

Your answer is incorrect

Implement an AWS IoT Core rule to route location updates directly from each sensor to Amazon SNS. Configure the application to poll the SNS topic for new messages

Overall explanation

Correct option:

Create an Amazon Simple Queue Service (Amazon SQS) queue to buffer the incoming location data. Configure the backend application to poll the queue and process messages

Using Amazon Simple Queue Service (Amazon SQS) is the most appropriate and cost-effective solution for this scenario because it introduces a buffering layer between data ingestion and processing, which helps prevent data loss during traffic spikes. When the sensor detects movement and sends data, the message is stored in the SQS queue, allowing the backend EC2 application to process the data at its own pace. This asynchronous decoupling ensures high availability and scalability while significantly reducing the risk of dropped messages. SQS automatically scales to handle large volumes of messages and provides at-least-once delivery, making it resilient even during processing delays. Because SQS is a fully managed serverless service, it requires minimal operational overhead—no need to provision or manage servers, and built-in retry and dead-letter queue capabilities improve reliability further.

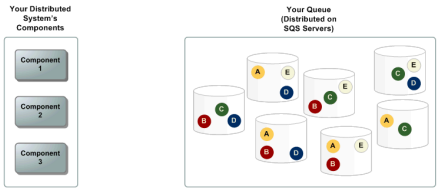
Basic Amazon SQS architecture

This section describes the components of a distributed messaging system and explains the lifecycle of an Amazon SQS message.

Distributed queues

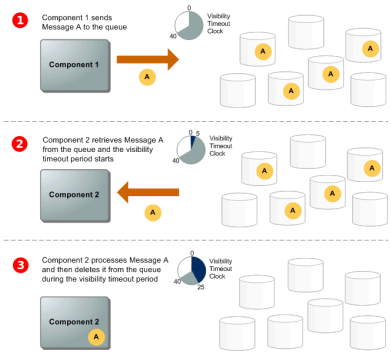
There are three main parts in a distributed messaging system: the **components of your distributed system**, your **queue** (distributed on Amazon SQS servers), and the **messages in the queue**.

In the following scenario, your system has several *producers* (components that send messages to the queue) and *consumers* (components that receive messages from the queue). The queue (which holds messages A through E) redundantly stores the messages across multiple Amazon SQS servers.



Message lifecycle

The following scenario describes the lifecycle of an Amazon SQS message in a queue, from creation to deletion.



1. A producer (Component 1) sends message A to a queue, and the message is distributed across the Amazon SQS servers redundantly.
2. When a consumer (Component 2) is ready to process messages, it consumes messages from the queue, and message A is returned. While message A is being processed, it remains in the queue and isn't returned to subsequent receive requests for the duration of the visibility timeout.
3. The consumer (Component 2) deletes message A from the queue to prevent the message from being received and processed again when the visibility timeout expires.

via - https://docs.aws.amazon.com/AWSSimpleQueueService/latest/SQSDeveloperGuide/welcome.html

Incorrect options:

Set up a containerized service using Amazon ECS with an internal queue built into the application layer. Configure the motion sensors to send location updates directly to the container endpoints - While using Amazon ECS with a custom-built queue may offer control, this solution introduces unnecessary complexity and operational overhead. The internal queue would need to handle scalability, durability, and retry logic, all of which are natively handled by

AWS messaging services like SQS. Additionally, receiving traffic directly from thousands of sensors could overload the container endpoints, making this design less resilient to traffic bursts.

Deploy an Amazon Data Firehose delivery stream to collect the motion data. Configure it to deliver data to an S3 bucket where the application scans and processes the files periodically - Amazon Data Firehose is well-suited for streaming analytics and data lake ingestion but is not optimized for event-by-event processing with immediate responsiveness. Delivering data to Amazon S3 and then scanning it introduces latency and complexity in managing the processing loop. It also lacks a built-in retry mechanism for per-event failures unless additional services like Lambda are introduced, which increases operational overhead.

Implement an AWS IoT Core rule to route location updates directly from each sensor to Amazon SNS. Configure the application to poll the SNS topic for new messages - While AWS IoT Core is a suitable service for handling device-to-cloud communication, routing messages through Amazon SNS introduces architectural limitations for this use case. SNS is designed for push-based, fan-out notification patterns—not for buffering or decoupled message processing. Moreover, applications cannot poll an SNS topic, as SNS pushes messages to subscribers rather than allowing pull-based consumption. Attempting to use SNS as a message queue violates its design pattern, increasing the likelihood of message loss if the subscriber application is overwhelmed or unavailable.

References:

https://docs.aws.amazon.com/AWSSimpleQueueService/latest/SQSDeveloperGuide/welcome.html

https://docs.aws.amazon.com/AmazonECS/latest/developerguide/Welcome.html

https://docs.aws.amazon.com/firehose/latest/dev/what-is-this-service.html

https://docs.aws.amazon.com/iot/latest/developerguide/iot-rules.html

Domain

Design Resilient Architectures

Question 37 **Incorrect**

A retail company wants to rollout and test a blue-green deployment for its global application in the next 48 hours. Most of the customers use mobile phones which are prone to Domain Name System (DNS) caching. The company has only two days left for the annual Thanksgiving sale to commence.

As a Solutions Architect, which of the following options would you recommend to test the deployment on as many users as possible in the given time frame?

Use AWS CodeDeploy deployment options to choose the right deployment

Use Elastic Load Balancing (ELB) to distribute traffic across deployments

Correct answer

Use AWS Global Accelerator to distribute a portion of traffic to a particular deployment

Your answer is incorrect

Use Amazon Route 53 weighted routing to spread traffic across different deployments

Overall explanation

Correct option:

Blue/green deployment is a technique for releasing applications by shifting traffic between two identical environments running different versions of the application: "Blue" is the currently running version and "green" the new version. This type of deployment allows you to test features in the green environment without impacting the currently running version of your application. When you're satisfied that the green version is working properly, you can gradually reroute the traffic from the old blue environment to the new green environment. Blue/green deployments can mitigate common risks associated with deploying software, such as downtime and rollback capability.

Use AWS Global Accelerator to distribute a portion of traffic to a particular deployment

AWS Global Accelerator is a network layer service that directs traffic to optimal endpoints over the AWS global network, this improves the availability and performance of your internet applications. It provides two static anycast IP addresses that act as a fixed entry point to your application endpoints in a single or multiple AWS Regions, such as your Application Load Balancers, Network Load Balancers, Elastic IP addresses or Amazon EC2 instances, in a single or in multiple AWS regions.

AWS Global Accelerator uses endpoint weights to determine the proportion of traffic that is directed to endpoints in an endpoint group, and traffic dials to control the percentage of traffic that is directed to an endpoint group (an AWS region where your application is deployed).

While relying on the DNS service is a great option for blue/green deployments, it may not fit use-cases that require a fast and controlled transition of the traffic. Some client devices and internet resolvers cache DNS answers for long periods; this DNS feature improves the efficiency of the DNS service as it reduces the DNS traffic across the Internet, and serves as a resiliency technique by preventing authoritative name-server overloads. The downside of this in blue/green deployments is that you don't know how long it will take before all of your users receive updated IP addresses when you update a record, change your routing preference or when there is an application failure.

With AWS Global Accelerator, you can shift traffic gradually or all at once between the blue and the green environment and vice-versa without being subject to DNS caching on client devices and internet resolvers, traffic dials and endpoint weights changes are effective within seconds.

Incorrect options:

Use Amazon Route 53 weighted routing to spread traffic across different deployments - Weighted routing lets you associate multiple resources with a single domain name (example.com) or subdomain name (acme.example.com) and choose how much traffic is routed to each resource. This can be useful for a variety of purposes, including load balancing and testing new versions of the software. As discussed earlier, DNS caching is a negative behavior for this use case and hence Amazon Route 53 is not a good option.

Use Elastic Load Balancing (ELB) to distribute traffic across deployments - Elastic Load Balancing (ELB) can distribute traffic across healthy instances. You can also use the Application Load Balancers weighted target groups feature for blue/green deployments as it does not rely on the DNS service. In addition you don't need to create new ALBs for the green environment. As the use-case refers to a global application, so this option cannot be used for a multi-Region solution which is needed for the given requirement.

Use AWS CodeDeploy deployment options to choose the right deployment - In AWS CodeDeploy, a deployment is the process, and the components involved in the process, of installing content on one or more instances. This content can consist of code, web and configuration files, executables, packages, scripts, and so on. AWS CodeDeploy deploys content

that is stored in a source repository, according to the configuration rules you specify. Blue/Green deployment is one of the deployment types that CodeDeploy supports. CodeDeploy is not meant to distribute traffic across instances, so this option is incorrect.

References:

https://aws.amazon.com/blogs/networking-and-content-delivery/using-aws-global-accelerator-to-achieve-blue-green-deployments

https://docs.aws.amazon.com/codedeploy/latest/userguide/deployments.html

https://docs.aws.amazon.com/Route53/latest/DeveloperGuide/routing-policy.html#routing-policy-weighted

Domain

Design Resilient Architectures

Question 38 Incorrect

A multinational logistics company is migrating its core systems to AWS. As part of this migration, the company has built an Amazon S3–based data lake to ingest and analyze supply chain data from external carriers and vendors. While some vendors have adopted the company's modern REST-based APIs for S3 uploads, others operate legacy systems that rely exclusively on SFTP for file transfers. These vendors are unable or unwilling to modify their workflows to support S3 APIs. The company wants to provide these vendors with an SFTP-compatible solution that allows direct uploads to Amazon S3, and must use fully managed AWS services to avoid managing any infrastructure. It must also support identity federation so that internal teams can map vendor access securely to specific S3 buckets or prefixes.

Which combination of options will provide a scalable and low-maintenance solution for this use case? (Select two)

Set up an Amazon EC2 instance with a custom SFTP server using OpenSSH. Configure cron jobs to upload received files to S3. Use Amazon CloudWatch to monitor EC2 health and disk usage

Your selection is correct

Configure Amazon S3 bucket policies to use IAM role-based access control for each vendor. Combine this with Transfer Family identity provider integration using Amazon Cognito or a custom identity provider for fine-grained permissions

Your selection is incorrect

Use AWS Transfer Family with SFTP for file uploads. Integrate the SFTP access control with Amazon Route 53 private hosted zones to create vendor-specific upload subdomains pointing to the SFTP endpoint

Use Amazon AppFlow to extract data from the legacy vendor systems and transform it into S3-compliant uploads. Schedule batch sync jobs to trigger every hour and send logs to CloudWatch for audit purposes

Your selection is correct

Deploy a fully managed AWS Transfer Family endpoint with SFTP enabled. Configure it to store uploaded files directly in an Amazon S3 bucket. Set up IAM roles mapped to each vendor for secure bucket or prefix access

Overall explanation

Correct options:

Deploy a fully managed AWS Transfer Family endpoint with SFTP enabled. Configure it to store uploaded files directly in an Amazon S3 bucket. Set up IAM roles mapped to each vendor for secure bucket or prefix access

To meet the needs of vendors that rely on legacy SFTP-based systems, the company can deploy a fully managed AWS Transfer Family endpoint with SFTP enabled. This service allows vendors to use their existing SFTP clients to upload files, while the company seamlessly stores those files in Amazon S3. To ensure secure and isolated access for each vendor, AWS Transfer Family supports mapping individual SFTP users to specific IAM roles. Each IAM role is configured with permissions that restrict access to a designated S3 bucket or prefix (e.g., s3://company-ingest/vendor1/),

enforcing least-privilege access and preventing data leakage between vendors. This role-based model also allows for auditing and logging through AWS CloudTrail, giving the company visibility into who is uploading what and when.

Configure Amazon S3 bucket policies to use IAM role-based access control for each vendor. Combine this with Transfer Family identity provider integration using Amazon Cognito or a custom identity provider for fine-grained permissions

To extend this solution with greater flexibility and centralized identity control, the company can integrate AWS Transfer Family with an identity provider, such as Amazon Cognito, an external SAML 2.0 provider, or a custom identity system. This enables federated authentication, where users authenticate through the identity provider and are then dynamically mapped to IAM roles based on attributes like username or group membership. These IAM roles enforce fine-grained S3 access through policies that can include conditions such as IP-based restrictions, encryption requirements, or access to specific folders. This identity federation model ensures scalable, secure, and compliant user management, allowing the company to onboard and manage vendor users without maintaining custom scripts or SFTP server infrastructure. By combining AWS Transfer Family for SFTP access and IAM roles for access control via identity provider integration, the company achieves a low-maintenance, secure, and scalable solution for supporting legacy SFTP uploads into their S3-based data lake.

Incorrect options:

Set up an Amazon EC2 instance with a custom SFTP server using OpenSSH. Configure cron jobs to upload received files to S3. Use Amazon CloudWatch to monitor EC2 health and disk usage - Hosting your own SFTP server on Amazon EC2 introduces significant operational overhead, including patching, security hardening, high availability, backup management, and scaling. Although this approach is technically feasible, it does not meet the requirement of using fully managed AWS services. It also creates long-term maintenance burden for a simple use case.

Use AWS Transfer Family with SFTP for file uploads. Integrate the SFTP access control with Amazon Route 53 private hosted zones to create vendor-specific upload subdomains pointing to the SFTP endpoint - Amazon Route 53 private hosted zones are used for internal DNS resolution and do not directly support public subdomain routing for SFTP endpoints. AWS Transfer Family already provides custom hostnames or AWS-hosted endpoints, and subdomain mapping adds complexity without functional gain. Route 53 is unnecessary here and does not simplify or enhance the solution.

Use Amazon AppFlow to extract data from the legacy vendor systems and transform it into S3-compliant uploads. Schedule batch sync jobs to trigger every hour and send logs to CloudWatch for audit purposes - Amazon AppFlow is a SaaS integration service designed for API-based applications like Salesforce, Zendesk, or Google Analytics. It does not support SFTP-based integrations or custom connectors for legacy systems. AppFlow is not appropriate for handling direct file uploads from vendors using legacy SFTP software.

References:

https://docs.aws.amazon.com/transfer/latest/userguide/what-is-aws-transfer-family.html

https://aws.amazon.com/blogs/storage/architecting-secure-and-compliant-managed-file-transfers-with-aws-transfer-family-sftp-connectors-and-pgp-encryption/

https://docs.aws.amazon.com/appflow/latest/userguide/what-is-appflow.html

https://docs.aws.amazon.com/Route53/latest/DeveloperGuide/hosted-zones-private.html

Domain

Design Secure Architectures

Question 39 Incorrect

A media publishing company is migrating its legacy content management application to AWS. Currently, the application and its MySQL database run on a single on-premises virtual machine, which creates a single point of failure and limits scalability. As traffic has increased due to growing reader engagement and video uploads, the company needs to redesign the solution to ensure automatic scaling, high availability, and separation of application and database layers. The company wants to continue using a MySQL-compatible engine and needs a cost-effective, managed solution that minimizes operational overhead.

Which AWS architecture will best fulfill these requirements?

Deploy the application to EC2 instances registered in a Network Load Balancer target group. Use Amazon ElastiCache for Redis as the database and configure it with Redis Streams for persistent storage

Your answer is incorrect

Host the application on EC2 instances that are part of a target group for an Application Load Balancer. Create an Amazon RDS for MySQL Multi-AZ DB instance to provide high availability and automatic failover for the database

Containerize the application and deploy it to Amazon ECS with EC2 launch type behind an Application Load Balancer. Use Amazon Neptune to store structured relational data with SQL-like queries

Correct answer

Migrate the application to Amazon EC2 instances in an Auto Scaling group behind an Application Load Balancer. Use Amazon Aurora Serverless v2 for MySQL to manage the database layer with auto-scaling and built-in high availability

Overall explanation

Correct option:

Migrate the application to Amazon EC2 instances in an Auto Scaling group behind an Application Load Balancer. Use Amazon Aurora Serverless v2 for MySQL to manage the database layer with auto-scaling and built-in high availability

This architecture provides automatic horizontal scaling for the application via an Auto Scaling group and managed, serverless scaling for the database layer with Amazon Aurora Serverless v2 for MySQL. Aurora Serverless v2 automatically adjusts capacity based on workload, and offers multi-AZ high availability, backups, and failover—all without requiring the company to manage the underlying database servers. It supports MySQL compatibility and is a cost-effective alternative to provisioned RDS when workloads are variable.

Read Scalability with Amazon Aurora Serverless v2

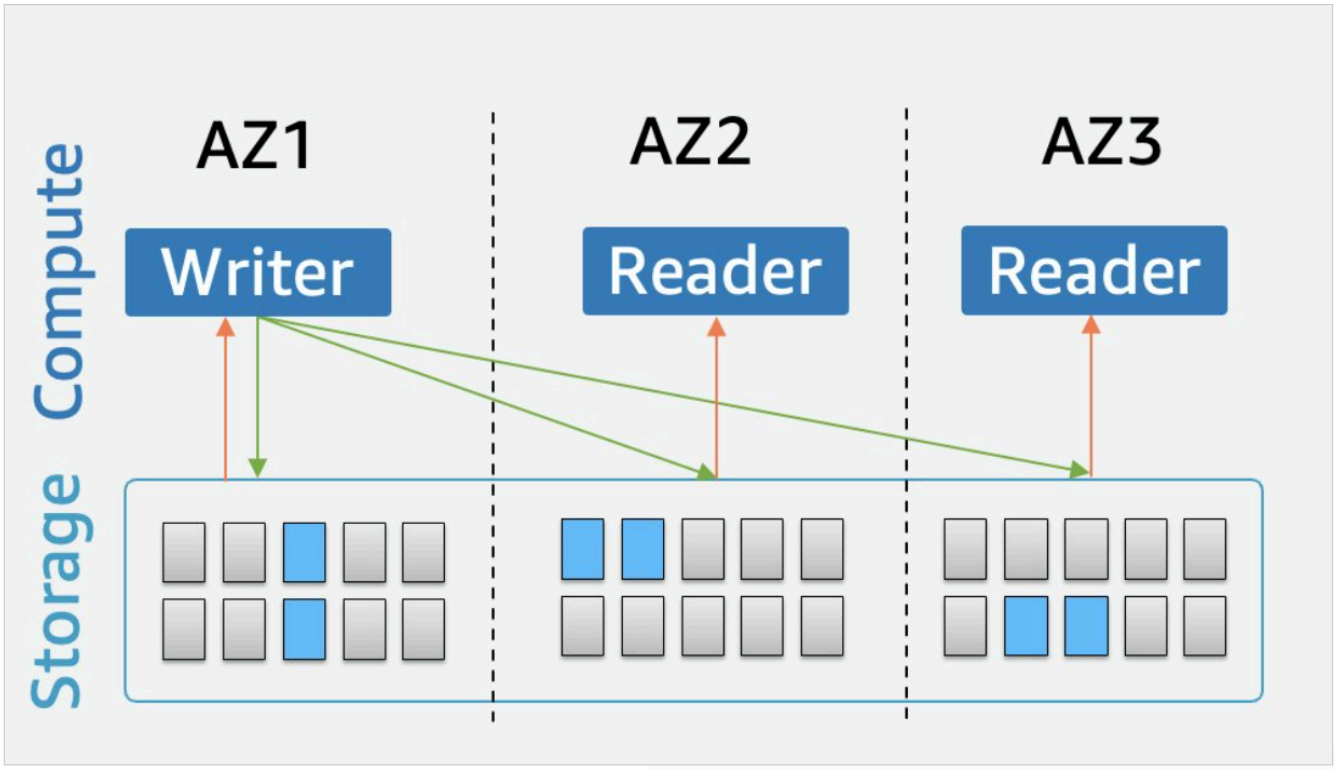
by Neha Gupta and Marie Yap | on 02 MAR 2023 | in Amazon Aurora, Intermediate (200), Serverless | Permalink | Comments | Share

Amazon Aurora Serverless is an on-demand, auto scaling configuration for Aurora. It scales the capacity up or down based on your application's needs. It enables you to run your database in the cloud without managing any database capacity. Aurora Serverless v2 supports Amazon Aurora Global Database , Multi AZ, AWS Identity and Access Management (IAM) database authentication, Amazon RDS Performance Insights, and Amazon RDS Proxy. Aurora Serverless v2 is available for the MySQL 8.0- and PostgreSQL 13+ compatible editions of Amazon Aurora.
In the post Scaling your Amazon RDS instance vertically and horizontally, we saw how to scale the non-serverless RDMS. In this post, we discuss use cases and best practices for using the scaling features of Aurora Serverless. There are two versions of Aurora Serverless: v1 and v2. We will focus on v2 for this post, which was released in April 2022.

Overview

Aurora Serverless v2 is suitable when you have a variable, unpredictable, spiky, demanding or multi-tenant application workload and it's not feasible to manually manage the database capacity. It supports all manners of workloads including the most demanding, business-critical environments that require high scale and high availability.

Aurora Serverless v2 is built on top of the same architecture of Aurora provisioned cluster with a decoupled storage and compute layer. The purpose-built storage layer is distributed across three availability zones, maintains six copies of your data, and is distributed across hundreds to thousands of storage nodes depending on the size of your database. It allows the separation of storage and compute, which is crucial for enabling a serverless database.



via - https://docs.aws.amazon.com/AmazonRDS/latest/AuroraUserGuide/aurora-serverless-v2.how-it-works.html

Incorrect options:

Deploy the application to EC2 instances registered in a Network Load Balancer target group. Use Amazon ElastiCache for Redis as the database and configure it with Redis Streams for persistent storage - ElastiCache is a key-value store designed for caching, not as a replacement for a relational MySQL database. Redis does not support relational joins, transactions, or ACID

compliance in the way MySQL does. Using it as a persistent store for structured application data would require significant re-architecture and compromises consistency.

Containerize the application and deploy it to Amazon ECS with EC2 launch type behind an Application Load Balancer. Use Amazon Neptune to store structured relational data with SQL-like queries - Amazon Neptune is a graph database designed for workloads that require graph models like social networks, recommendation engines, and knowledge graphs. It does not support SQL or relational schemas and is not compatible with MySQL. Using Neptune in place of a MySQL database would require entirely rearchitecting the data model and application queries.

Host the application on EC2 instances that are part of a target group for an Application Load Balancer. Create an Amazon RDS for MySQL Multi-AZ DB instance to provide high availability and automatic failover for the database - While Amazon RDS for MySQL with Multi-AZ offers high availability, it does not support automatic compute scaling for the database layer. If the workload grows, manual intervention is required to resize the instance or configure read replicas for performance. Also, RDS Multi-AZ provides failover, but not the instantaneous, granular auto-scaling offered by Aurora Serverless v2.

References:

https://docs.aws.amazon.com/AmazonRDS/latest/AuroraUserGuide/aurora-serverless-v2.how-it-works.html

https://aws.amazon.com/blogs/database/read-scalability-with-amazon-aurora-serverless-v2/

https://aws.amazon.com/blogs/database/introducing-scaling-to-0-capacity-with-amazon-aurora-serverless-v2/

https://docs.aws.amazon.com/neptune/latest/userguide/intro.html

https://docs.aws.amazon.com/AmazonElastiCache/latest/dg/WhatIs.html

https://docs.aws.amazon.com/AmazonRDS/latest/UserGuide/CHAP_MySQL.html

Domain

Design Resilient Architectures

Question 40 **Correct**

A silicon valley based startup has a content management application with the web-tier running on Amazon EC2 instances and the database tier running on Amazon Aurora. Currently, the entire infrastructure is located in `us-east-1` region. The startup has 90% of its customers in the US and Europe. The engineering team is getting reports of deteriorated application performance from customers in Europe with high application load time.

As a solutions architect, which of the following would you recommend addressing these performance issues? (Select two)

Your selection is correct

Create Amazon Aurora read replicas in the `eu-west-1` region

Your selection is correct

Setup another fleet of Amazon EC2 instances for the web tier in the `eu-west-1` region. Enable latency routing policy in Amazon Route 53

Create Amazon Aurora Multi-AZ standby instance in the `eu-west-1` region

Setup another fleet of Amazon EC2 instances for the web tier in the `eu-west-1` region. Enable failover routing policy in Amazon Route 53

Setup another fleet of Amazon EC2 instances for the web tier in the `eu-west-1` region. Enable geolocation routing policy in Amazon Route 53

Overall explanation

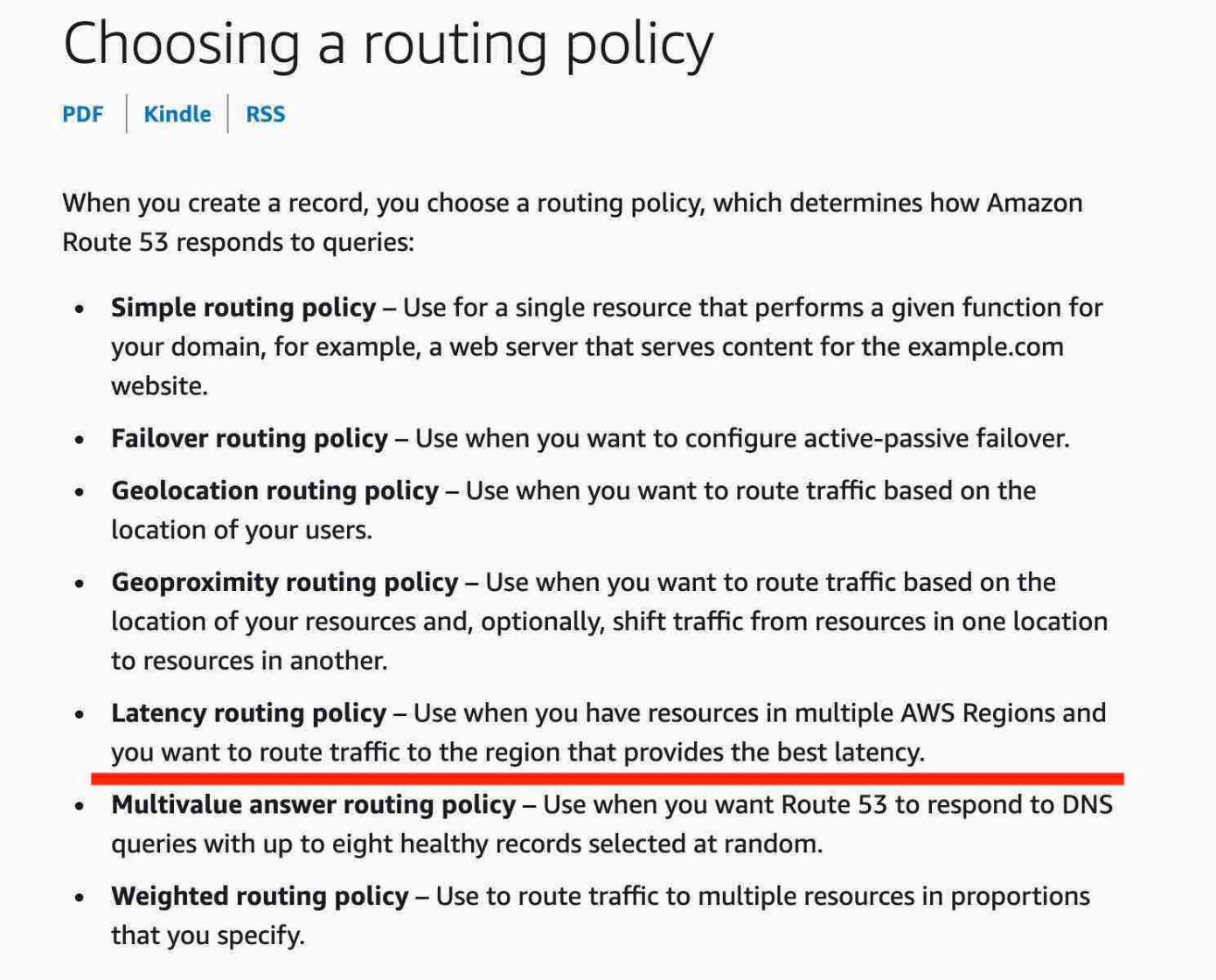
Correct options:

Setup another fleet of Amazon EC2 instances for the web tier in the `eu-west-1` region. Enable latency routing policy in Amazon Route 53

Amazon Route 53 is a highly available and scalable cloud Domain Name System (DNS) web service. Use latency based routing when you have resources in multiple AWS Regions and you want to route traffic to the region that provides the lowest latency. To use latency-based routing, you create latency records for your resources in multiple AWS Regions. When Amazon Route 53 receives a DNS query for your domain or subdomain (example.com or acme.example.com), it determines which AWS Regions you've created latency records for, determines which region gives the user the lowest latency, and then selects a latency record for that region. Route 53 responds with the value from the selected record, such as the IP address for a web server.

As customers in Europe are facing performance issues with high application load time, you can use latency based routing to reduce the latency. Hence this is the correct option.

Amazon Route 53 Routing Policy Overview:



Choosing a routing policy

PDF | Kindle | RSS

When you create a record, you choose a routing policy, which determines how Amazon Route 53 responds to queries:

- **Simple routing policy** – Use for a single resource that performs a given function for your domain, for example, a web server that serves content for the example.com website.
- **Failover routing policy** – Use when you want to configure active-passive failover.
- **Geolocation routing policy** – Use when you want to route traffic based on the location of your users.
- **Geoproximity routing policy** – Use when you want to route traffic based on the location of your resources and, optionally, shift traffic from resources in one location to resources in another.
- **Latency routing policy** – Use when you have resources in multiple AWS Regions and you want to route traffic to the region that provides the best latency.
- **Multivalue answer routing policy** – Use when you want Route 53 to respond to DNS queries with up to eight healthy records selected at random.
- **Weighted routing policy** – Use to route traffic to multiple resources in proportions that you specify.

via - https://docs.aws.amazon.com/Route53/latest/DeveloperGuide/routing-policy.html

Create Amazon Aurora read replicas in the `eu-west-1` region

Amazon Aurora is a MySQL and PostgreSQL-compatible relational database built for the cloud, that combines the performance and availability of traditional enterprise databases with the

simplicity and cost-effectiveness of open source databases. Amazon Aurora features a distributed, fault-tolerant, self-healing storage system that auto-scales up to 64TB per database instance.

Amazon Aurora read replicas can be used to scale out reads across regions. This will improve the application performance for users in Europe. Therefore, this is also a correct option for the given use-case.

Incorrect options:

Setup another fleet of Amazon EC2 instances for the web tier in the `eu-west-1` region. Enable geolocation routing policy in Amazon Route 53 - Geolocation routing lets you choose the resources that serve your traffic based on the geographic location of your users, meaning the location that DNS queries originate from. For example, you might want all queries from Europe to be routed to an ELB load balancer in the Frankfurt region. You can also use geolocation routing to restrict the distribution of content to only the locations in which you have distribution rights. You cannot use geolocation routing to reduce latency, hence this option is incorrect.

Setup another fleet of Amazon EC2 instances for the web tier in the `eu-west-1` region. Enable failover routing policy in Amazon Route 53 - Failover routing lets you route traffic to a resource when the resource is healthy or to a different resource when the first resource is unhealthy. The primary and secondary records can route traffic to anything from an Amazon S3 bucket that is configured as a website to a complex tree of records. You cannot use failover routing to reduce latency, hence this option is incorrect.

Create Amazon Aurora Multi-AZ standby instance in the `eu-west-1` region - Amazon Aurora Multi-AZ enhances the availability and durability for the database, it does not help in read scaling, so it is not a correct option for the given use-case.

References:

https://docs.aws.amazon.com/Route53/latest/DeveloperGuide/routing-policy.html

https://aws.amazon.com/blogs/aws/new-cross-region-read-replicas-for-amazon-aurora/

Domain

Design High-Performing Architectures

Question 41 **Incorrect**

The engineering team at a logistics company has noticed that the Auto Scaling group (ASG) is not terminating an unhealthy Amazon EC2 instance.

As a Solutions Architect, which of the following options would you suggest to troubleshoot the issue? (Select three)

A custom health check might have failed. The Auto Scaling group (ASG) does not terminate instances that are set unhealthy by custom checks

Your selection is correct

The health check grace period for the instance has not expired

A user might have updated the configuration of the Auto Scaling group (ASG) and increased the minimum number of instances forcing ASG to keep all instances alive

The Amazon EC2 instance could be a spot instance type, which cannot be terminated by the Auto Scaling group (ASG)

Correct selection

The instance has failed the Elastic Load Balancing (ELB) health check status

Your selection is correct

The instance maybe in Impaired status

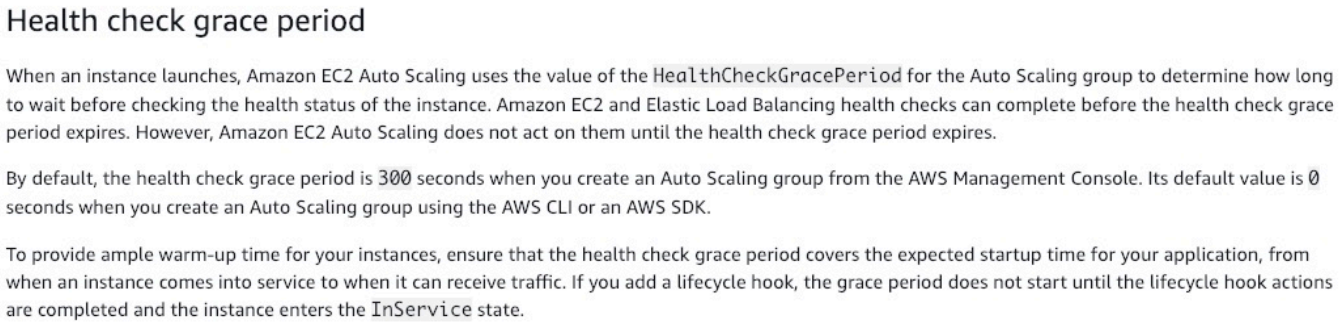
Overall explanation

Correct options:

The health check grace period for the instance has not expired

Amazon EC2 Auto Scaling doesn't terminate an instance that came into service based on Amazon EC2 status checks and Elastic Load Balancing (ELB) health checks until the health check grace period expires.

More on Health check grace period:



Health check grace period

When an instance launches, Amazon EC2 Auto Scaling uses the value of the `HealthCheckGracePeriod` for the Auto Scaling group to determine how long to wait before checking the health status of the instance. Amazon EC2 and Elastic Load Balancing health checks can complete before the health check grace period expires. However, Amazon EC2 Auto Scaling does not act on them until the health check grace period expires.

By default, the health check grace period is `300` seconds when you create an Auto Scaling group from the AWS Management Console. Its default value is `0` seconds when you create an Auto Scaling group using the AWS CLI or an AWS SDK.

To provide ample warm-up time for your instances, ensure that the health check grace period covers the expected startup time for your application, from when an instance comes into service to when it can receive traffic. If you add a lifecycle hook, the grace period does not start until the lifecycle hook actions are completed and the instance enters the `InService` state.

via - https://docs.aws.amazon.com/autoscaling/ec2/userguide/healthcheck.html#health-check-grace-period

The instance maybe in Impaired status

Amazon EC2 Auto Scaling does not immediately terminate instances with an Impaired status. Instead, Amazon EC2 Auto Scaling waits a few minutes for the instance to recover. Amazon EC2 Auto Scaling might also delay or not terminate instances that fail to report data for status checks. This usually happens when there is insufficient data for the status check metrics in Amazon CloudWatch.

The instance has failed the Elastic Load Balancing (ELB) health check status

By default, Amazon EC2 Auto Scaling doesn't use the results of ELB health checks to determine an instance's health status when the group's health check configuration is set to EC2. As a result, Amazon EC2 Auto Scaling doesn't terminate instances that fail ELB health checks. If an instance's status is OutofService on the ELB console, but the instance's status is Healthy on the Amazon EC2 Auto Scaling console, confirm that the health check type is set to ELB.

Incorrect options:

The Amazon EC2 instance could be a spot instance type, which cannot be terminated by the Auto Scaling group (ASG) - This is an incorrect statement. Amazon EC2 Auto Scaling terminates Spot instances when capacity is no longer available or the Spot price exceeds your maximum price.

A user might have updated the configuration of the Auto Scaling group (ASG) and increased the minimum number of instances forcing ASG to keep all instances alive - This statement is incorrect. If the configuration is updated and ASG needs more number of instances, ASG will launch new, healthy instances and does not keep unhealthy ones alive.

A custom health check might have failed. The Auto Scaling group (ASG) does not terminate instances that are set unhealthy by custom checks - This statement is incorrect. You can define custom health checks in Amazon EC2 Auto Scaling. When a custom health check determines that an instance is unhealthy, the check manually triggers SetInstanceHealth and then sets the instance's state to Unhealthy. Amazon EC2 Auto Scaling then terminates the unhealthy instance.

References:

https://aws.amazon.com/premiumsupport/knowledge-center/auto-scaling-terminate-instance/

https://aws.amazon.com/premiumsupport/knowledge-center/auto-scaling-instance-how-terminated/

Domain

Design Resilient Architectures

Question 42 **Incorrect**

A Machine Learning research group uses a proprietary computer vision application hosted on an Amazon EC2 instance. Every time the instance needs to be stopped and started again, the application takes about 3 minutes to start as some auxiliary software programs need to be executed so that the application can function. The research group would like to minimize the application boostrap time whenever the system needs to be stopped and then started at a later point in time.

As a solutions architect, which of the following solutions would you recommend for this use-case?

Use Amazon EC2 User-Data

Correct answer

Use Amazon EC2 Instance Hibernate

Use Amazon EC2 Meta-Data

Your answer is incorrect

Create an Amazon Machine Image (AMI) and launch your Amazon EC2 instances from that

Overall explanation

Correct option:

Use Amazon EC2 Instance Hibernate

When you hibernate an instance, AWS signals the operating system to perform hibernation (suspend-to-disk). Hibernation saves the contents from the instance memory (RAM) to your Amazon EBS root volume. AWS then persists the instance's Amazon EBS root volume and any attached Amazon EBS data volumes.

When you start your instance:

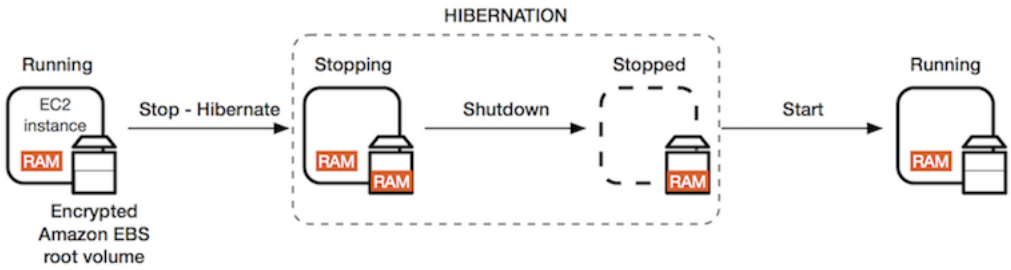
The Amazon EBS root volume is restored to its previous state

The RAM contents are reloaded

The processes that were previously running on the instance are resumed

Previously attached data volumes are reattached and the instance retains its instance ID

Overview of Amazon EC2 hibernation:



via - https://docs.aws.amazon.com/AWSEC2/latest/UserGuide/Hibernate.html

By using Amazon EC2 hibernate, we have the capability to resume it at any point of time, with the application already launched, thus helping us cut the 3 minutes start time.

Incorrect options:

Use Amazon EC2 User-Data - Amazon EC2 instance user data is the data that you specified in the form of a configuration script while launching your instance. Here, the problem is that the application takes 3 minutes to launch, no matter what. EC2 user data won't help us because it's just here to help us execute a list of commands, not speed them up.

Use Amazon EC2 Meta-Data - Amazon EC2 instance metadata is data about your instance that you can use to configure or manage the running instance. Instance metadata is divided into categories, for example, host name, events, and security groups. The EC2 meta-data is a distractor and can only help us determine some metadata attributes on our EC2 instances.

Create an Amazon Machine Image (AMI) and launch your Amazon EC2 instances from that - An Amazon Machine Image (AMI) provides the information required to launch an instance. You must specify an AMI when you launch an instance. You can launch multiple instances from a single AMI when you need multiple instances with the same configuration. You can use different AMIs to launch instances when you need instances with different configurations.

Creating an AMI may help with all the system dependencies, but it won't help us with speeding up the application start time.

References:

https://docs.aws.amazon.com/AWSEC2/latest/UserGuide/Hibernate.html

https://docs.aws.amazon.com/AWSEC2/latest/UserGuide/ec2-instance-metadata.html

https://docs.aws.amazon.com/AWSEC2/latest/UserGuide/AMIs.html

Domain

Design High-Performing Architectures

Question 43 Incorrect

A mobile app allows users to submit photos, which are stored in an Amazon S3 bucket. Currently, a batch of Amazon EC2 Spot Instances is launched nightly to process all the day's uploads. Each photo requires approximately 3 minutes and 512 MB of memory to process. To improve responsiveness and minimize costs, the company wants to shift to near real-time image processing that begins as soon as an image is uploaded.

Which solution will provide the MOST cost-effective and scalable architecture to meet these new requirements?

Correct answer

Configure Amazon S3 to send event notifications to an Amazon SQS queue each time a photo is uploaded. Set up an AWS Lambda function to poll the queue and process images asynchronously

Your answer is incorrect

Set up Amazon S3 to push events to an Amazon SQS queue. Launch a single EC2 Reserved Instance that continuously polls the queue and processes each image upon receipt

Enable S3 event notifications to invoke an Amazon EventBridge rule. Configure an AWS Step Functions workflow to initiate an Fargate task in Amazon ECS to process the image

Configure S3 to trigger an AWS App Runner service directly. Deploy a containerized image-processing application to App Runner to automatically process each upload

Overall explanation

Correct option:

Configure Amazon S3 to send event notifications to an Amazon SQS queue each time a photo is uploaded. Set up an AWS Lambda function to poll the queue and process images asynchronously

This is the most cost-effective solution with a serverless approach for near real-time processing. S3 event notifications can be configured to send metadata about each uploaded object to Amazon SQS, which acts as a durable buffer. AWS Lambda functions are triggered by new messages in the queue and provide a pay-per-use model, which aligns perfectly with sporadic workloads. Since each image requires only 512 MB and 2 minutes of processing, it easily fits within Lambda's runtime and memory limits. This solution eliminates the need for managing EC2 instances, greatly reducing operational overhead.

Incorrect options:

Set up Amazon S3 to push events to an Amazon SQS queue. Launch a single EC2 Reserved Instance that continuously polls the queue and processes each image upon receipt - While this approach does support real-time processing, it relies on a long-running EC2 instance, which introduces ongoing compute costs even during idle periods. A Reserved Instance offers cost savings over On-Demand, but it is not event-driven or elastic, and the instance may still be underutilized or overwhelmed depending on image upload rates. Compared to Lambda's on-demand scaling and granular billing, this solution incurs higher costs and more administrative overhead.

Enable S3 event notifications to invoke an Amazon EventBridge rule. Configure an AWS Step Functions workflow to initiate an Fargate task in Amazon ECS to process the image - While technically feasible, this solution is more complex and expensive than needed. Step Functions and Fargate tasks introduce multiple managed components, which adds to operational complexity and cost. This design may be better suited for multi-step processing workflows or heavier compute tasks. For lightweight and short-lived operations like these, Lambda offers simpler orchestration and lower cost without requiring container orchestration or task definition management.

Configure S3 to trigger an AWS App Runner service directly. Deploy a containerized image-processing application to App Runner to automatically process each upload - AWS App Runner is a good fit for container-based web services, but it is not designed for direct, event-driven invocation from Amazon S3. While you can deploy stateless apps easily, you'd need an intermediate component like SQS or EventBridge to bridge the event and container invocation. Additionally, App Runner is less cost-efficient for short-lived background tasks, as it keeps container instances warm for performance, leading to unnecessary compute charges for infrequent workloads.

References:

https://docs.aws.amazon.com/AmazonS3/latest/userguide/EventNotifications.html

https://docs.aws.amazon.com/lambda/latest/dg/gettingstarted-limits.html

https://docs.aws.amazon.com/lambda/latest/dg/with-s3-example.html

https://docs.aws.amazon.com/apprunner/latest/dg/what-is-apprunner.html

Domain

Design Cost-Optimized Architectures

Question 44 **Correct**

An analytics company wants to improve the performance of its big data processing workflows running on Amazon Elastic File System (Amazon EFS). Which of the following performance modes should be used for Amazon EFS to address this requirement?

Your answer is correct

Max I/O

Bursting Throughput

General Purpose

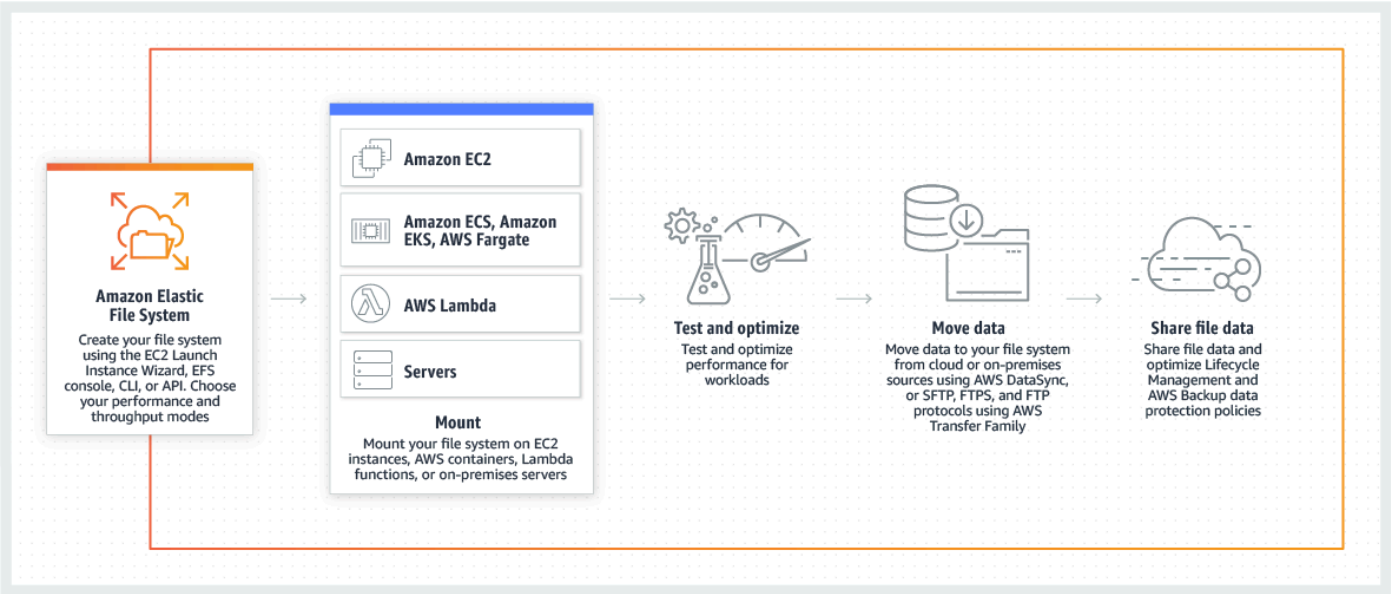
Provisioned Throughput

Overall explanation

Correct option:

Max I/O

How Amazon EFS Works:



via - https://aws.amazon.com/efs/

Max I/O performance mode is used to scale to higher levels of aggregate throughput and operations per second. This scaling is done with a tradeoff of slightly higher latencies for file metadata operations. Highly parallelized applications and workloads, such as big data analysis, media processing, and genomic analysis, can benefit from this mode.

Performance modes

Amazon EFS offers two performance modes, General Purpose and Max I/O.

- **General Purpose mode** has the lowest per-operation latency and is the default performance mode for file systems. One Zone file systems always use the General Purpose performance mode. For faster performance, we recommend always using General Purpose performance mode.
- **Max I/O mode** is a previous generation performance type that is designed for highly parallelized workloads that can tolerate higher latencies than the General Purpose mode. Max I/O mode is not supported for One Zone file systems or file systems that use Elastic throughput.

> **Important**
> Due to the higher per-operation latencies with Max I/O, we recommend using General Purpose performance mode for all file systems.

To help ensure that your workload stays within the IOPS limit available to file systems using General Purpose performance mode, you can monitor the `PercentIOLimit` CloudWatch metric. For more information, see Amazon CloudWatch metrics for Amazon EFS.

Applications can scale their IOPS elastically up to the limit associated with the performance mode. You are not billed separately for IOPS; they are included in a file system's throughput accounting. Every Network File System (NFS) request is accounted for as 4 kilobyte (KB) of throughput, or its actual request and response size, whichever is larger.

via - https://docs.aws.amazon.com/efs/latest/ug/performance.html

Incorrect options:

Provisioned Throughput

Bursting Throughput

These two options have been added as distractors as these refer to the throughput mode of Amazon EFS and not the performance mode. There are two throughput modes to choose from for your file system, Bursting Throughput and Provisioned Throughput. With Bursting Throughput mode, throughput on Amazon EFS scales as the size of your file system in the standard storage class grows. With Provisioned Throughput mode, you can instantly provision the throughput of your file system (in MiB/s) independent of the amount of data stored.

General Purpose - General Purpose performance mode is ideal for latency-sensitive use cases, like web serving environments, content management systems, home directories, and general file serving. If you don't choose a performance mode when you create your file system, Amazon EFS selects the General Purpose mode for you by default.

References:

https://docs.aws.amazon.com/efs/latest/ug/performance.html

https://aws.amazon.com/efs/

Domain

Design High-Performing Architectures

Question 45 Incorrect

An e-commerce company operates multiple AWS accounts and has interconnected these accounts in a hub-and-spoke style using the AWS Transit Gateway. Amazon Virtual Private Cloud (Amazon VPCs) have been provisioned across these AWS accounts to facilitate network isolation.

Which of the following solutions would reduce both the administrative overhead and the costs while providing shared access to services required by workloads in each of the VPCs?

Use Transit VPC to reduce cost and share the resources across Amazon Virtual Private Cloud (Amazon VPCs)

Your answer is incorrect

Use VPCs connected with AWS Direct Connect

Correct answer

Build a shared services Amazon Virtual Private Cloud (Amazon VPC)

Use Fully meshed VPC Peering connection

Overall explanation

Correct option:

Build a shared services Amazon Virtual Private Cloud (Amazon VPC)

Consider an organization that has built a hub-and-spoke network with AWS Transit Gateway. VPCs have been provisioned into multiple AWS accounts, perhaps to facilitate network isolation or to enable delegated network administration. When deploying distributed architectures such as this, a popular approach is to build a "shared services VPC, which provides access to services required by workloads in each of the VPCs. This might include directory services or VPC endpoints. Sharing resources from a central location instead of building them in each VPC may reduce administrative overhead and cost.

Centralized VPC Endpoints (multiple VPCs):

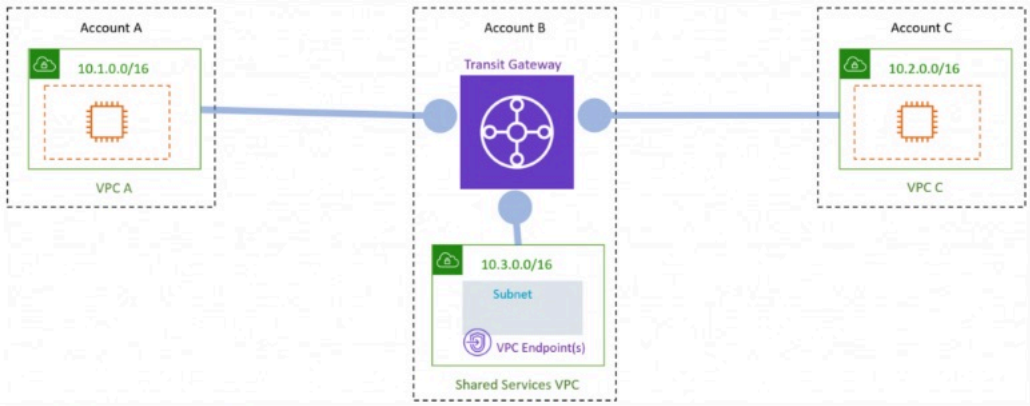


Figure 3: Centralized VPC Endpoints (multiple VPCs)

via - https://aws.amazon.com/blogs/architecture/reduce-cost-and-increase-security-with-amazon-vpc-endpoints/

A VPC endpoint allows you to privately connect your VPC to supported AWS services without requiring an Internet gateway, NAT device, VPN connection, or AWS Direct Connect connection. Endpoints are virtual devices that are horizontally scaled, redundant, and highly available VPC components. They allow communication between instances in your VPC and services without imposing availability risks or bandwidth constraints on your network traffic.

VPC endpoints enable you to reduce data transfer charges resulting from network communication between private VPC resources (such as Amazon Elastic Cloud Compute—or EC2—instances) and AWS Services (such as Amazon Quantum Ledger Database, or QLDB). Without VPC endpoints configured, communications that originate from within a VPC destined for public AWS services must egress AWS to the public Internet in order to access AWS services. This network path incurs outbound data transfer charges. Data transfer charges for traffic egressing from Amazon EC2 to the Internet vary based on volume. With VPC endpoints configured, communication between your VPC and the associated AWS service does not leave the Amazon network. If your workload requires you to transfer significant volumes of data between your VPC and AWS, you can reduce costs by leveraging VPC endpoints.

Incorrect options:

Use Transit VPC to reduce cost and share the resources across Amazon Virtual Private Cloud (Amazon VPCs) - Transit VPC uses customer-managed Amazon Elastic Compute Cloud (Amazon EC2) VPN instances in a dedicated transit VPC with an Internet gateway. This design requires the customer to deploy, configure, and manage EC2-based VPN appliances, which will result in additional EC2, and potentially third-party product and licensing charges. Note that this design will generate additional data transfer charges for traffic traversing the transit VPC: data is charged when it is sent from a spoke VPC to the transit VPC, and again from the transit VPC to the on-premises network or a different AWS Region. Transit VPC is not the right choice here.

More on Transit VPC:

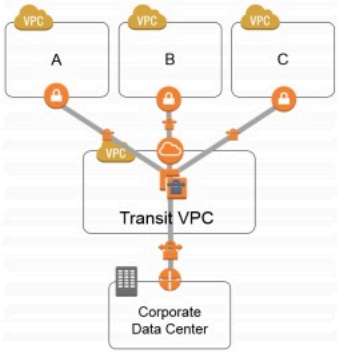
Transit VPC

This approach uses customer-managed Amazon Elastic Compute Cloud (Amazon EC2) VPN instances in a dedicated transit VPC with an Internet gateway. The EC2 instances initiate the VPN connections and route traffic between multiple VPCs and shared-services VPCs. The spoke VPCs can leverage VPC peering to circumvent the transit VPC, providing more scalable, direct access between VPCs.

This design requires the customer to deploy, configure, and manage EC2-based VPN appliances, which will result in additional EC2, and potentially third-party product and licensing charges. Therefore, it is best suited for customers who have already implemented a transit VPC and want to leverage it to manage more advanced connection types, such as inter-region connectivity, or multi-VPC connectivity to on-premises resources.

Note that this design will generate additional data transfer charges for traffic traversing the transit VPC: data is charged when it is sent from a spoke VPC to the transit VPC, and again from the transit VPC to the on-premises network or a different AWS Region.

For additional details on this solution, see the Multiple-VPC VPN Connection Sharing Solution Brief.



via - https://d0.awsstatic.com/aws-answers/AWS_Single_Region_Multi_VPC_Connectivity.pdf

Use Fully meshed VPC Peering connection - This approach creates multiple peering connections to facilitate the sharing of information between resources in different VPCs. This design connects multiple VPCs in a fully meshed configuration, with peering connections between each pair of VPCs. With this configuration, each VPC has access to the resources in all other VPCs. Each peering connection requires modifications to all the other VPCs' route tables and, as the number of VPCs grows, this can be difficult to maintain. And keep in mind that AWS recommends a maximum of 125 peering connections per VPC. It's complex to manage and isn't a right fit for the current scenario.

More on Fully meshed VPC Peers:

Configuration Details

This design connects multiple VPCs in a fully meshed configuration, with peering connections between each pair of VPCs. With this configuration, each VPC has access to the resources in all other VPCs.

To enable the flow of traffic between VPCs, each VPC route table must contain entries that point to the IP address ranges of all the other VPCs in the fully meshed configuration. This design is more complicated to set up than a partially meshed configuration, but it enables communication across all VPCs in the system.

Considerations

Each peering connection requires modifications to all the other VPCs' route tables and, as the number of VPCs grows, this can be difficult to maintain. And keep in mind that AWS recommends a maximum of 125 peering connections per VPC.

Customers can create VPC peering connections between VPCs in the same account, or with VPCs in a different AWS account, as long as the VPCs are in the same region.

via - https://d0.awsstatic.com/aws-answers/AWS_Single_Region_Multi_VPC_Connectivity.pdf

Use VPCs connected with AWS Direct Connect - This approach is a good alternative for customers who need to connect a high number of VPCs to a central VPC or on-premises resources, or who already have an AWS Direct Connect connection in place. This design also offers customers the ability to incorporate transitive routing into their network design. For example, if VPC A and VPC B are both connected to an on-premises network using AWS Direct Connect connections, then the two VPCs can be connected to each other via AWS Direct Connect. AWS Direct Connect requires physical cables and takes about a month for setting up, this is not an ideal solution for the given scenario.

References:

https://aws.amazon.com/blogs/architecture/reduce-cost-and-increase-security-with-amazon-vpc-endpoints/

https://d0.awsstatic.com/aws-answers/AWS_Single_Region_Multi_VPC_Connectivity.pdf

Domain

Design Secure Architectures

Question 46 Incorrect

A company has recently launched a new mobile gaming application that the users are adopting rapidly. The company uses Amazon RDS MySQL as the database. The engineering team wants an urgent solution to this issue where the rapidly increasing workload might exceed the available database storage.

As a solutions architect, which of the following solutions would you recommend so that it requires minimum development and systems administration effort to address this requirement?

Create read replica for Amazon RDS MySQL

Your answer is incorrect

Migrate RDS MySQL database to Amazon Aurora which offers storage auto-scaling

Correct answer

Enable storage auto-scaling for Amazon RDS MySQL

Migrate Amazon RDS MySQL database to Amazon DynamoDB which automatically allocates storage space when required

Overall explanation

Correct option:

Enable storage auto-scaling for Amazon RDS MySQL

If your workload is unpredictable, you can enable storage autoscaling for an Amazon RDS DB instance. With storage autoscaling enabled, when Amazon RDS detects that you are running out of free database space it automatically scales up your storage. Amazon RDS starts a storage modification for an autoscaling-enabled DB instance when these factors apply:

Free available space is less than 10 percent of the allocated storage.

The low-storage condition lasts at least five minutes.

At least six hours have passed since the last storage modification.

The maximum storage threshold is the limit that you set for autoscaling the DB instance. You can't set the maximum storage threshold for autoscaling-enabled instances to a value greater than the maximum allocated storage.

Incorrect options:

Migrate RDS MySQL database to Amazon Aurora which offers storage auto-scaling - Although Aurora offers automatic storage scaling, this option is ruled out since it involves significant systems administration effort to migrate from Amazon RDS MySQL to Aurora. It is much easier to just enable storage auto-scaling for Amazon RDS MySQL.

Migrate Amazon RDS MySQL database to Amazon DynamoDB which automatically allocates storage space when required - This option is ruled out since Amazon DynamoDB is a NoSQL database which implies significant development effort to change the application logic to connect and query data from the underlying database. It is much easier to just enable storage auto-scaling for Amazon RDS MySQL.

Create read replica for Amazon RDS MySQL - Read replicas make it easy to take advantage of supported engines' built-in replication functionality to elastically scale out beyond the capacity constraints of a single DB instance for read-heavy database workloads. You can create multiple read replicas for a given source DB Instance and distribute your application's read traffic amongst them. This option acts as a distractor as read replicas cannot help to automatically scale storage for the primary database.

Reference:

https://docs.aws.amazon.com/AmazonRDS/latest/UserGuide/USER_PIOPS.StorageTypes.html

Domain

Design Resilient Architectures

Question 47 **Correct**

An enterprise uses a centralized Amazon S3 bucket to store logs and reports generated by multiple analytics services. Each service writes to and reads from a dedicated prefix (folder path) in the bucket. The company wants to enforce fine-grained access control so that each service can access only its own prefix, without being able to see or modify other services' data. The solution must support scalable and maintainable permissions management with minimal operational overhead.

Which approach will best meet these requirements?

Create separate IAM users for each service. Manually assign inline IAM policies to grant read/write permissions to the S3 bucket. Reference specific object names in the policy for each user

Create a single S3 bucket policy that lists all object ARNs under each prefix and grants permissions accordingly. Use resource-level permissions to restrict access to individual services

Deploy Amazon Macie to classify the objects in the bucket by prefix and apply automated object-level access policies to each object based on service tags

Your answer is correct

Configure individual S3 access points for each analytics service. Attach access point policies that restrict access to only the relevant prefix in the S3 bucket

Overall explanation

Correct option:

Configure individual S3 access points for each analytics service. Attach access point policies that restrict access to only the relevant prefix in the S3 bucket

Amazon S3 Access Points provide a scalable, manageable solution for managing permissions on shared buckets. By creating a dedicated access point for each service, and setting access point-level policies that scope access down to specific prefixes within the bucket, the company can enforce fine-grained, isolated access per application. This approach avoids complexity in the bucket policy and eliminates the need for per-object permissions management. Access points are ideal for environments with multiple applications or teams sharing a common S3 bucket.

Amazon S3 Access Points simplify managing and securing data access at scale for applications using shared datasets on S3. Customers can create unique hostnames using access points to enforce distinct and secure permissions and network controls for any request that they make through the Access Point.

Large organizations with private Amazon S3 buckets using S3 Access Points as their solution for complying with data-sharing requirements across departments need a way to simplify access management while adhering to strict security standards. In our example, a customer has three different organizational departments: Finance, Marketing, and Operations. Each department has different needs for the data in the S3 bucket, while also requiring different controls.

Organizational and departmental requirements in this example:

- An Amazon S3 bucket that holds the restricted data that different departments must securely access and manage. The S3 bucket should not be public, and it should only be accessible from within the VPC. Furthermore, the VPC should have no outbound or inbound internet access.
- The Finance department uses **application role 1,** and this role should enable members of this department to upload data to the S3 bucket if the prefix matches /Application1 or /Application3. In this scenario, this role ensures that members of this department could take no other actions, like download or delete.
- The Marketing department uses **application role 2,** and this role should enable members of this department to download objects from the S3 bucket. In this scenario, this role ensures that this department could take no other actions, like upload or delete.
- The Operations department uses **application role 3,** and this role should enable members of this department to download objects if the prefix matches /Application3. This role also permits members of this department to delete objects in any of the application folders inside the S3 bucket.

For this solution, we use the following services to securely grant access to shared data in S3 buckets to three different IAM roles.

- Amazon EC2: An Amazon EC2 instance to assume IAM roles to demonstrate the security boundaries application to IAM roles.
- Amazon S3: S3 bucket stores data, and access is restricted to the VPC by S3 Access Points, with no other access.
- Amazon S3 Access Points: The S3 bucket has three Access Points tied to different IAM roles.
- AWS IAM: IAM roles govern the access to S3 Access Points and operations possible on the Access Points.
- AWS Systems Manager Session Manager: Session Manager to log in to the Amazon EC2 instance to test different IAM roles.
- Amazon VPC: Our VPC has two private subnets and has no internet or NAT Gateway. VPC can reach to AWS services like Amazon S3 and Systems Manager via VPC endpoints.

via - https://aws.amazon.com/blogs/storage/securing-data-in-a-virtual-private-cloud-using-amazon-s3-access-points/

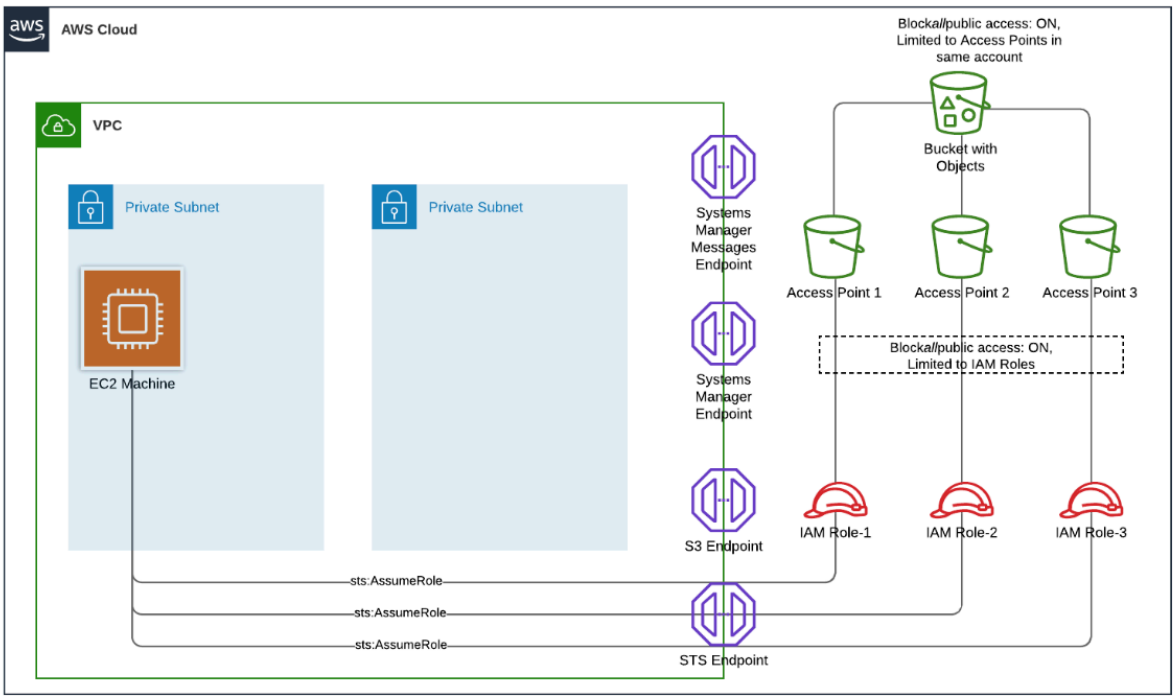


Figure 1: Securing data in a virtual private cloud using Amazon S3 Access Points

Here is how the process works, as shown in *Figure 1*:

1. The solution uses S3 buckets that are only accessible within the VPC. This allows no public access to S3 buckets, and you cannot access data from outside the VPC, resulting in tight security. To restrict data access, S3 bucket has Block Public Access set to on and restricted to Access Points in the AWS account.
2. The VPC has no internet gateway or network address translation (NAT) gateway attached to it restricting inbound and outbound internet access.
3. Amazon S3 Access Points enable fine grain controls of what operations users can perform on objects in the S3 bucket. They have policies restricting access to IAM roles and possible operations like **s3:GetObject.** In this solution, there are three Access Points, one for each of the organization's departments. Check the documentation on configuring IAM policies for using access points for more details on how S3 access points support IAM.
4. As we can only access the Amazon S3 objects from within the VPC, we must stand up an EC2 instance so that we can assume IAM roles to access the S3 objects using Access Points. The Amazon EC2 instance in the private subnet has IAM permissions to assume those roles.
5. Each Amazon S3 Access Point has its own policies and allows access to different IAM roles as per the policies. You control data access using S3 Access Point policies and IAM roles.

via - https://aws.amazon.com/blogs/storage/securing-data-in-a-virtual-private-cloud-using-amazon-s3-access-points/

Incorrect options:

Create separate IAM users for each service. Manually assign inline IAM policies to grant read/write permissions to the S3 bucket. Reference specific object names in the policy for each user - While IAM policies can restrict access to certain prefixes within an S3 bucket, using inline policies tied to IAM users introduces high operational complexity. Over time, as object names change or services evolve, these policies would require constant manual updates. Moreover, hardcoding object names reduces scalability and violates best practices for managing data access at scale. This method lacks the flexibility and maintainability that access points offer.

Deploy Amazon Macie to classify the objects in the bucket by prefix and apply automated object-level access policies to each object based on service tags - Amazon Macie is designed

for data discovery and classification, especially for identifying sensitive data like PII—not for managing object-level access controls. While Macie can label data and notify users of risks, it cannot automatically enforce S3 permissions or assign per-prefix policies. Using Macie for this purpose would introduce unnecessary cost and complexity without addressing the core access isolation requirement.

Create a single S3 bucket policy that lists all object ARNs under each prefix and grants permissions accordingly. Use resource-level permissions to restrict access to individual services - Although it is possible to use a single S3 bucket policy with resource-level permissions, this approach does not scale well. As the number of objects and services grows, maintaining a flat bucket policy that enumerates specific ARNs becomes cumbersome and error-prone. Any change to the policy would require full redeployment, making this solution difficult to manage over time. In contrast, access points offer isolated, easily maintainable policy boundaries per service.

References:

https://docs.aws.amazon.com/AmazonS3/latest/userguide/access-points.html

https://aws.amazon.com/blogs/storage/securing-data-in-a-virtual-private-cloud-using-amazon-s3-access-points/

https://docs.aws.amazon.com/macie/latest/user/what-is-macie.html

Domain

Design Secure Architectures

Question 48 **Incorrect**

"An enterprise organization is expanding its cloud footprint and needs to centralize its security event data from various AWS accounts and services. The goal is to evaluate security posture across all environments and improve threat detection and response — without requiring significant custom code or manual integration.

Which solution will fulfill these needs with the least development effort?

Use Amazon Athena with predefined SQL queries to scan security logs stored in multiple S3 buckets. Visualize the findings by exporting results to an Amazon QuickSight dashboard

Correct answer

Use Amazon Security Lake to create a centralized data lake that automatically collects security-related logs and events from AWS services and third-party sources. Store the data in an Amazon S3 bucket managed by Security Lake

Your answer is incorrect

Deploy a custom Lambda function to aggregate security logs from multiple AWS accounts. Format the data into CSV files and upload them to a central S3 bucket for analysis

Set up a data lake using AWS Lake Formation to collect and organize security event logs. Use AWS Glue to perform ETL operations and standardize the log formats for centralized analysis

Overall explanation

Correct option:

Use Amazon Security Lake to create a centralized data lake that automatically collects security-related logs and events from AWS services and third-party sources. Store the data in an Amazon S3 bucket managed by Security Lake

Amazon Security Lake is a fully managed, purpose-built service designed to automatically collect, normalize, and centralize security-related data from various AWS accounts, Regions, services, and even third-party sources. It stores this data in Amazon S3 buckets and formats it using the Open Cybersecurity Schema Framework (OCSF), which enhances compatibility with multiple analytics tools. Security Lake eliminates the need to build custom ETL pipelines or configure cross-service log ingestion manually, significantly reducing development effort. It also integrates natively with AWS services like CloudTrail, VPC Flow Logs, GuardDuty, and AWS Config, providing a single authoritative view of security data across the organization. With built-in support for log partitioning, retention, and access management, it delivers both centralization and scalability with minimal operational overhead. The managed nature of Security Lake means there's minimal setup or custom coding, making it the lowest-effort and most scalable solution.

What is Amazon Security Lake?

PDF | RSS | Focus mode | Summarize page

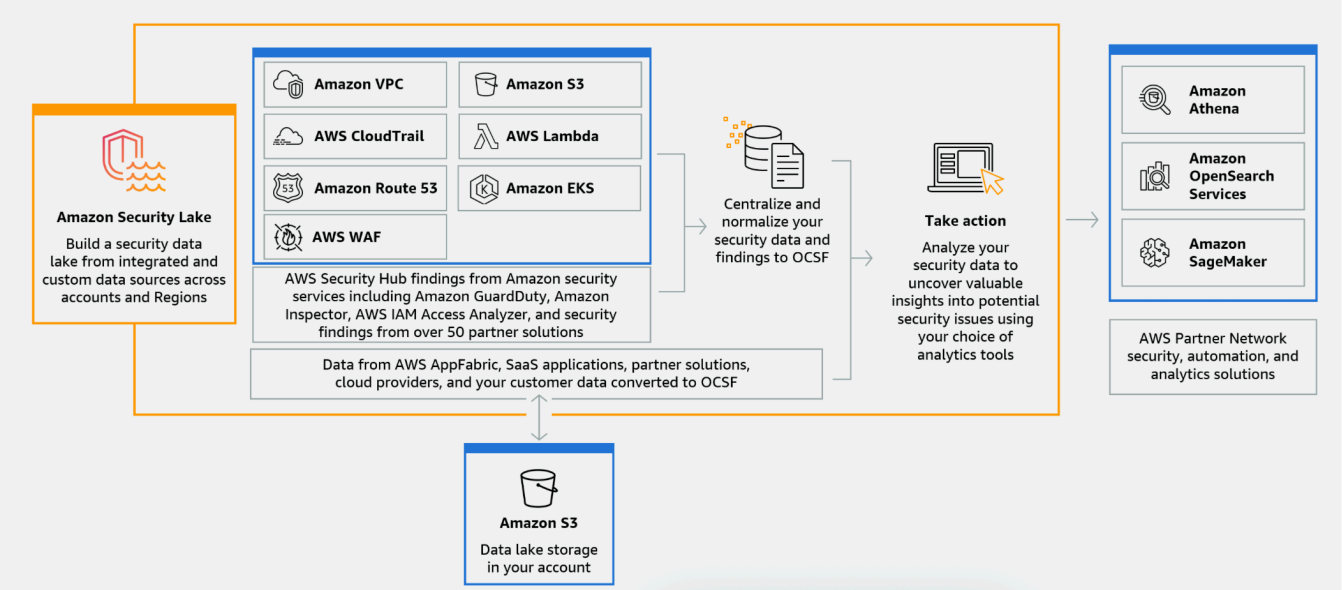
Amazon Security Lake is a fully managed security data lake service. You can use Security Lake to automatically centralize security data from AWS environments, SaaS providers, on premises, cloud sources, and third-party sources into a purpose-built data lake that's stored in your AWS account. Security Lake helps you analyze security data, so you can get a more complete understanding of your security posture across the entire organization. With Security Lake, you can also improve the protection of your workloads, applications, and data.

The data lake is backed by Amazon Simple Storage Service (Amazon S3) buckets, and you retain ownership over your data.

Security Lake automates the collection of security-related log and event data from integrated AWS services and third-party services. It also helps you manage the lifecycle of data with customizable retention and replication settings. Security Lake converts ingested data into Apache Parquet format and a standard open-source schema called the Open Cybersecurity Schema Framework (OCSF). With OCSF support, Security Lake normalizes and combines security data from AWS and a broad range of enterprise security data sources.

Other AWS services and third-party services can subscribe to the data that's stored in Security Lake for incident response and security data analytics.

Overview of Security Lake



via - https://docs.aws.amazon.com/security-lake/latest/userguide/what-is-security-lake.html

Incorrect options:

Deploy a custom Lambda function to aggregate security logs from multiple AWS accounts. Format the data into CSV files and upload them to a central S3 bucket for analysis - While this approach centralizes the logs, it requires considerable custom development and maintenance, including cross-account access setup, log parsing logic, error handling, and consistent formatting. Additionally, storing data in CSV format makes automated analytics and threat detection more difficult, and there's no inherent support for schema standardization or normalization. This option does not meet the requirement for minimal development effort.

Use Amazon Athena with predefined SQL queries to scan security logs stored in multiple S3 buckets. Visualize the findings by exporting results to an Amazon QuickSight dashboard - Although Athena and QuickSight can be useful for analysis, this solution does not address centralized log collection or normalization. The organization would still need to manage the log ingestion, parsing, and schema management manually, often across multiple S3 buckets. This setup increases operational burden and lacks out-of-the-box security-specific integrations and formatting. It also fails to provide proactive threat detection or standardized data aggregation.

Set up a data lake using AWS Lake Formation to collect and organize security event logs. Use AWS Glue to perform ETL operations and standardize the log formats for centralized analysis - While AWS Lake Formation is a powerful service for building and securing data lakes, it

is primarily designed for structured and semi-structured business data, not for native ingestion and normalization of security event logs. Using Lake Formation in this context would require custom pipelines, ETL jobs, and log parsers, all of which significantly increase the development and operational overhead. Unlike Amazon Security Lake, which provides native support for ingesting logs from AWS security services (e.g., CloudTrail, GuardDuty, VPC Flow Logs) and automatically standardizes them into OCSF format, Lake Formation does not offer purpose-built features for centralized security data management.

References:

https://docs.aws.amazon.com/security-lake/latest/userguide/what-is-security-lake.html

https://docs.aws.amazon.com/lake-formation/latest/dg/what-is-lake-formation.html

Domain

Design Secure Architectures

Question 49 **Correct**

A financial institution is transitioning its critical back-office systems to AWS. These systems currently rely on Microsoft SQL Server databases hosted on on-premises infrastructure. The data is highly sensitive and subject to regulatory compliance. The organization wants to enhance security and minimize database management tasks as part of the migration.

Which solution will best meet these goals with the least operational burden?

Your answer is correct

Migrate the SQL Server databases to a Multi-AZ Amazon RDS for SQL Server deployment. Enable encryption at rest by using an AWS Key Management Service (AWS KMS) managed key

Export the SQL Server databases to CSV format and store them in Amazon S3 with S3 bucket policies for access control. Use AWS Backup for data protection

Migrate the SQL Server databases to Amazon EC2 instances with encrypted EBS volumes. Use an AWS KMS customer managed key to enable encryption

Move the SQL Server data into Amazon Timestream to gain time series insights. Use AWS CloudTrail to monitor access to the data

Overall explanation

Correct option:

Migrate the SQL Server databases to a Multi-AZ Amazon RDS for SQL Server deployment. Enable encryption at rest by using an AWS Key Management Service (AWS KMS) managed key

Amazon RDS for SQL Server is a fully managed relational database service, which significantly reduces operational overhead by automating backups, patching, high availability, and monitoring. A Multi-AZ deployment ensures high availability and failover support, which is crucial for critical financial workloads. With RDS, encryption at rest using AWS KMS managed keys can be enabled easily for compliance and security. This architecture provides a secure, resilient, and low-maintenance solution for running SQL Server in the cloud.

Incorrect options:

Migrate the SQL Server databases to Amazon EC2 instances with encrypted EBS volumes. Use an AWS KMS customer managed key to enable encryption - While hosting SQL Server on EC2 with encrypted Amazon EBS volumes provides control over database configuration, it also significantly increases operational overhead. The organization would need to manage OS-level patching, SQL Server updates, backups, scaling, and high availability. Even though encryption with a KMS CMK enhances security, this approach lacks the automation and ease of management that a managed service like Amazon RDS offers, making it less suitable for reducing operational tasks.

Export the SQL Server databases to CSV format and store them in Amazon S3 with S3 bucket policies for access control. Use AWS Backup for data protection - While storing data in Amazon S3 can be secure and cost-effective, this solution does not replicate the functionality of a relational database and introduces limitations around querying, indexing, and transactional

consistency. Additionally, converting the data to CSV and storing it in S3 loses the relational structure, making this unsuitable for migrating a live, critical system that relies on SQL Server's features. This setup also does not offer real-time access to an operational database.

Move the SQL Server data into Amazon Timestream to gain time series insights. Use AWS CloudTrail to monitor access to the data - Amazon Timestream is designed for time series data such as telemetry and metrics, not general-purpose relational database workloads like those used in financial systems. It does not support SQL Server features like stored procedures, triggers, or complex joins. Also, using CloudTrail to monitor access logs does not provide the granular security mechanisms required for database-level access control and compliance in this use case.

References:

https://docs.aws.amazon.com/AmazonRDS/latest/UserGuide/CHAP_SQLServer.html

https://docs.aws.amazon.com/AmazonRDS/latest/UserGuide/Overview.Encryption.html

Domain

Design Secure Architectures

Question 50 **Correct**

The engineering manager for a content management application wants to set up Amazon RDS read replicas to provide enhanced performance and read scalability. The manager wants to understand the data transfer charges while setting up Amazon RDS read replicas.

Which of the following would you identify as correct regarding the data transfer charges for Amazon RDS read replicas?

There are data transfer charges for replicating data within the same Availability Zone (AZ)

There are no data transfer charges for replicating data across AWS Regions

Your answer is correct

There are data transfer charges for replicating data across AWS Regions

There are data transfer charges for replicating data within the same AWS Region

Overall explanation

Correct option:

There are data transfer charges for replicating data across AWS Regions

Amazon RDS Read Replicas provide enhanced performance and durability for Amazon RDS database (DB) instances. They make it easy to elastically scale out beyond the capacity constraints of a single DB instance for read-heavy database workloads.

A read replica is billed as a standard DB Instance and at the same rates. You are not charged for the data transfer incurred in replicating data between your source DB instance and read replica within the same AWS Region.

Q: How much do read replicas cost? When does billing begin and end?

A read replica is billed as a standard DB Instance and at the same rates. Just like a standard DB instance, the rate per "DB Instance hour" for a read replica is determined by the DB instance class of the read replica – please see pricing page for up-to-date pricing. You are not charged for the data transfer incurred in replicating data between your source DB instance and read replica within the same AWS Region.

Billing for a read replica begins as soon as the replica has been successfully created (i.e. when status is listed as "active"). The read replica will continue being billed at standard Amazon RDS DB instance hour rates until you issue a command to delete it.

via - https://aws.amazon.com/rds/faqs/

Incorrect options:

There are data transfer charges for replicating data within the same Availability Zone (AZ)

There are data transfer charges for replicating data within the same AWS Region

There are no data transfer charges for replicating data across AWS Regions

These three options contradict the explanation provided above, so these options are incorrect.

Reference:

https://aws.amazon.com/rds/faqs/

Domain

Design Cost-Optimized Architectures

Question 51 Incorrect

A financial services company wants to store confidential data in Amazon S3 and it needs to meet the following data security and compliance norms:

1. Encryption key usage must be logged for auditing purposes
2. Encryption Keys must be rotated every year
3. The data must be encrypted at rest

Which is the MOST operationally efficient solution?

Server-side encryption (SSE-S3) with automatic key rotation

Your answer is incorrect

Server-side encryption with customer-provided keys (SSE-C) with automatic key rotation

Correct answer

Server-side encryption with AWS Key Management Service (AWS KMS) keys (SSE-KMS) with automatic key rotation

Server-side encryption with AWS Key Management Service (AWS KMS) keys (SSE-KMS) with manual key rotation

Overall explanation

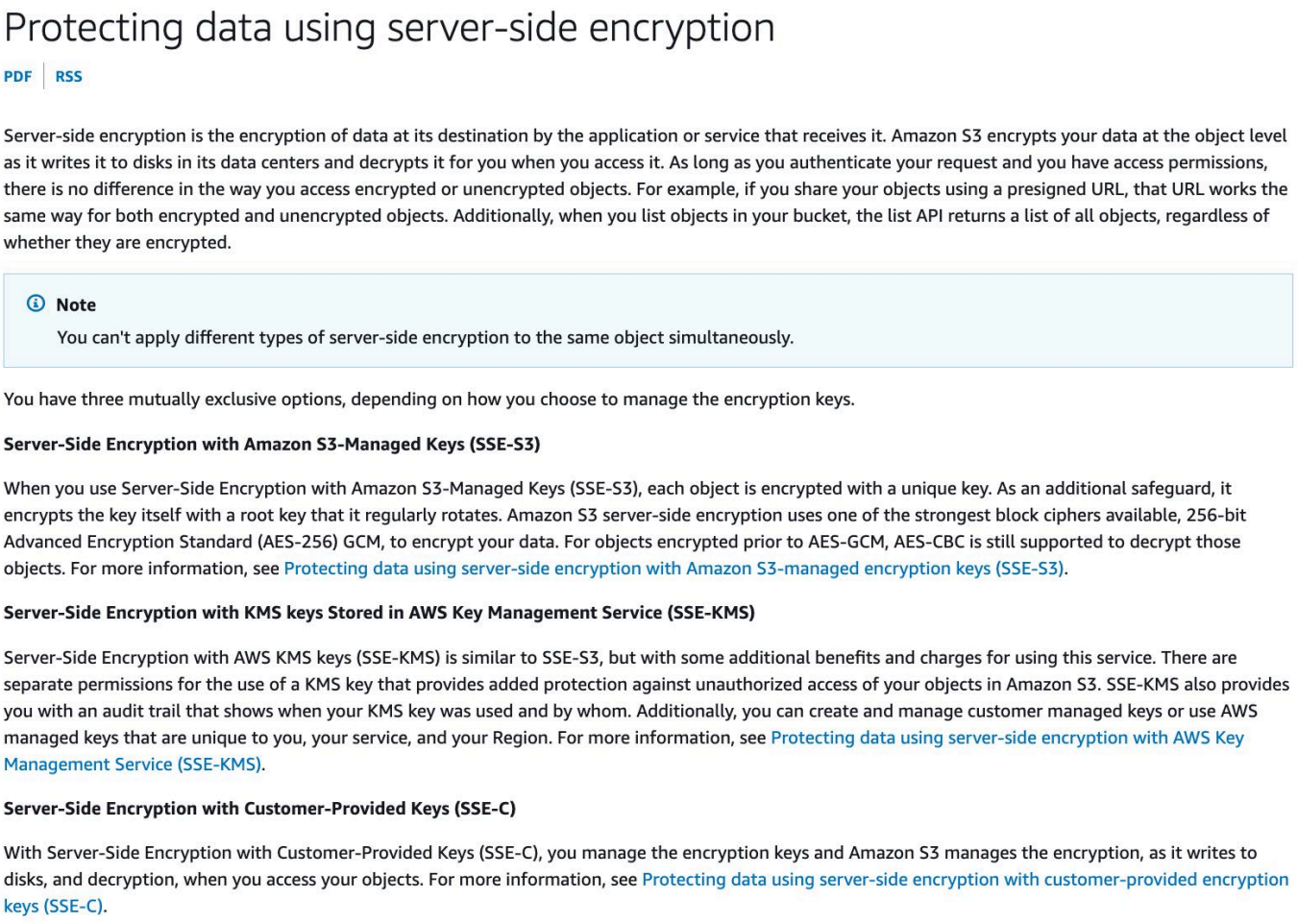
Correct option:

Server-side encryption with AWS Key Management Service (AWS KMS) keys (SSE-KMS) with automatic key rotation

Server-side encryption is the encryption of data at its destination by the application or service that receives it. Amazon S3 encrypts your data at the object level as it writes it to disks in its data centers and decrypts it for you when you access it.

Amazon S3 now applies server-side encryption with Amazon S3 managed keys (SSE-S3) as the base level of encryption for every bucket in Amazon S3. Starting January 5, 2023, all new object uploads to Amazon S3 are automatically encrypted at no additional cost and with no impact on performance.

Amazon S3 server-side encryption



Protecting data using server-side encryption

PDF | RSS

Server-side encryption is the encryption of data at its destination by the application or service that receives it. Amazon S3 encrypts your data at the object level as it writes it to disks in its data centers and decrypts it for you when you access it. As long as you authenticate your request and you have access permissions, there is no difference in the way you access encrypted or unencrypted objects. For example, if you share your objects using a presigned URL, that URL works the same way for both encrypted and unencrypted objects. Additionally, when you list objects in your bucket, the list API returns a list of all objects, regardless of whether they are encrypted.

Note

You can't apply different types of server-side encryption to the same object simultaneously.

You have three mutually exclusive options, depending on how you choose to manage the encryption keys.

Server-Side Encryption with Amazon S3-Managed Keys (SSE-S3)

When you use Server-Side Encryption with Amazon S3-Managed Keys (SSE-S3), each object is encrypted with a unique key. As an additional safeguard, it encrypts the key itself with a root key that it regularly rotates. Amazon S3 server-side encryption uses one of the strongest block ciphers available, 256-bit Advanced Encryption Standard (AES-256) GCM, to encrypt your data. For objects encrypted prior to AES-GCM, AES-CBC is still supported to decrypt those objects. For more information, see Protecting data using server-side encryption with Amazon S3-managed encryption keys (SSE-S3).

Server-Side Encryption with KMS keys Stored in AWS Key Management Service (SSE-KMS)

Server-Side Encryption with AWS KMS keys (SSE-KMS) is similar to SSE-S3, but with some additional benefits and charges for using this service. There are separate permissions for the use of a KMS key that provides added protection against unauthorized access of your objects in Amazon S3. SSE-KMS also provides you with an audit trail that shows when your KMS key was used and by whom. Additionally, you can create and manage customer managed keys or use AWS managed keys that are unique to you, your service, and your Region. For more information, see Protecting data using server-side encryption with AWS Key Management Service (SSE-KMS).

Server-Side Encryption with Customer-Provided Keys (SSE-C)

With Server-Side Encryption with Customer-Provided Keys (SSE-C), you manage the encryption keys and Amazon S3 manages the encryption, as it writes to disks, and decryption, when you access your objects. For more information, see Protecting data using server-side encryption with customer-provided encryption keys (SSE-C).

via - https://docs.aws.amazon.com/AmazonS3/latest/userguide/serv-side-encryption.html

AWS KMS is a service that combines secure, highly available hardware and software to provide a key management system scaled for the cloud. Amazon S3 uses server-side encryption with AWS KMS (SSE-KMS) to encrypt your S3 object data. Also, when SSE-KMS is requested for the object, the S3 checksum as part of the object's metadata, is stored in encrypted form.

If you use KMS keys, you can use AWS KMS through the AWS Management Console or the AWS KMS API to do the following:

1. Centrally create, view, edit, monitor, enable or disable, rotate, and schedule deletion of KMS keys.
2. Define the policies that control how and by whom KMS keys can be used.
3. Audit their usage to prove that they are being used correctly. Auditing is supported by the AWS KMS API, but not by the AWS KMSAWS Management Console.

When you enable automatic key rotation for a KMS key, AWS KMS generates new cryptographic material for the KMS key every year.

AWS KMS keys:

Rotating AWS KMS keys

PDF | RSS

Cryptographic best practices discourage extensive reuse of encryption keys. To create new cryptographic material for your customer managed keys, you can create new KMS keys, and then change your applications or aliases to use the new KMS keys. Or, you can enable automatic key rotation for an existing KMS key.

When you enable *automatic key rotation* for a KMS key, AWS KMS generates new cryptographic material for the KMS key every year. AWS KMS saves all previous versions of the cryptographic material in perpetuity so you can decrypt any data encrypted with that KMS key. AWS KMS does not delete any rotated key material until you delete the KMS key. You can track the rotation of key material for your KMS keys in Amazon CloudWatch and AWS CloudTrail.

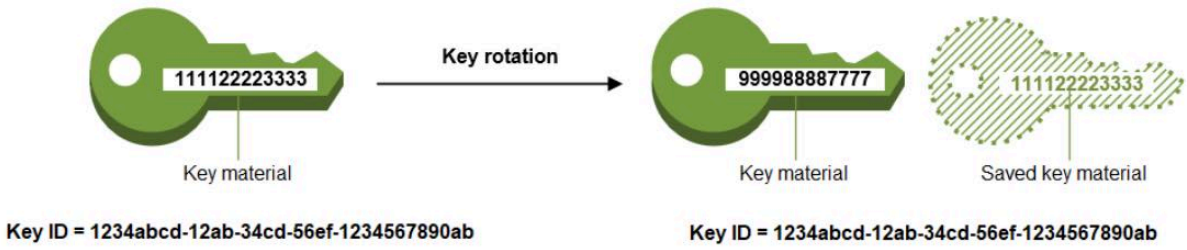
When you use a rotated KMS key to encrypt data, AWS KMS uses the current key material. When you use the rotated KMS key to decrypt ciphertext, AWS KMS uses the version of the key material that was used to encrypt it. You cannot request a particular version of the key material. Because AWS KMS transparently decrypts with the appropriate key material, you can safely use a rotated KMS key in applications and AWS services without code changes.

However, automatic key rotation has no effect on the data that the KMS key protects. It does not rotate the data keys that the KMS key generated or re-encrypt any data protected by the KMS key, and it will not mitigate the effect of a compromised data key.

AWS KMS supports automatic key rotation only for symmetric encryption KMS keys with key material that AWS KMS creates. Automatic rotation is optional for customer managed KMS keys. AWS KMS always rotates the key material for AWS managed KMS keys every year. Rotation of AWS owned KMS keys varies.

> **Note**
>
> The rotation interval for AWS managed keys changed in May 2022. For details, see AWS managed keys.

Key rotation changes only the *key material*, which is the cryptographic secret that is used in encryption operations. The KMS key is the same logical resource, regardless of whether or how many times its key material changes. The properties of the KMS key do not change, as shown in the following image.



via - https://docs.aws.amazon.com/kms/latest/developerguide/rotate-keys.html

For the given use case, you can set up server-side encryption with AWS KMS Keys (SSE-KMS) with automatic key rotation.

Incorrect options:

Server-side encryption with AWS Key Management Service (AWS KMS) keys (SSE-KMS) with manual key rotation - Although it is possible to manually rotate the AWS KMS key, it is not the best fit solution as it is not operationally efficient.

Server-side encryption (SSE-S3) with automatic key rotation - When you use Server-Side Encryption with Amazon S3-Managed Keys (SSE-S3), each object is encrypted with a unique key. As an additional safeguard, it encrypts the key itself with a root key that it regularly rotates.

However, with SSE-S3, you cannot log the usage of the encryption key for auditing purposes. So this option is incorrect.

Server-side encryption with customer-provided keys (SSE-C) with automatic key rotation - It is possible to automatically rotate the customer-provided keys but you will need to develop the underlying solution to automate the key rotation. Therefore, this option is not operationally efficient.

References:

https://docs.aws.amazon.com/kms/latest/developerguide/concepts.html#master_keys

https://docs.aws.amazon.com/kms/latest/developerguide/rotate-keys.html

https://docs.aws.amazon.com/AmazonS3/latest/userguide/serv-side-encryption.html

Domain

Design Secure Architectures

Question 52 **Correct**

A manufacturing analytics company has a large collection of automated scripts that perform data cleanup, validation, and system integration tasks. These scripts are currently run by a local Linux cron scheduler and have an execution time of up to 30 minutes. The company wants to migrate these scripts to AWS without significant changes, and would prefer a containerized, serverless architecture that automatically scales and can respond to event-based triggers in the future. The solution must minimize infrastructure management.

Which solution will best meet these requirements with minimal refactoring and operational overhead?

Convert each script into a Lambda function and package it in a zip archive. Use Amazon EventBridge Scheduler to run the functions on a fixed schedule. Use Amazon S3 to store function outputs and logs

Create a container image for each script. Use AWS Step Functions to define a workflow for all scheduled tasks. Use a Wait state to delay execution and run tasks using Step Functions' RunTask integration with ECS Fargate

Your answer is correct

Package the scripts into a container image. Use Amazon EventBridge Scheduler to define cron-based recurring schedules. Configure EventBridge Scheduler to invoke AWS Fargate tasks using Amazon ECS

Package the scripts into a container image. Deploy the image to AWS Batch with a managed compute environment on Amazon EC2. Define scheduling policies in AWS Batch to trigger jobs according to cron expressions

Overall explanation

Correct option:

Package the scripts into a container image. Use Amazon EventBridge Scheduler to define cron-based recurring schedules. Configure EventBridge Scheduler to invoke AWS Fargate tasks using Amazon ECS

This solution best meets all the requirements. Amazon EventBridge Scheduler is a fully managed, serverless scheduler that allows you to define high-scale cron and rate-based jobs without maintaining any scheduling infrastructure. By packaging the legacy cron scripts as containers and running them as ECS Fargate tasks, the company avoids the need to provision EC2 instances or manage runtime environments. The solution is serverless, scalable, and compatible with future event-based triggers using EventBridge's native integration.

Incorrect options:

Convert each script into a Lambda function and package it in a zip archive. Use Amazon EventBridge Scheduler to run the functions on a fixed schedule. Use Amazon S3 to store function outputs and logs - While AWS Lambda integrates well with EventBridge Scheduler, Lambda functions have a hard timeout of 15 minutes. This makes them unsuitable for cron jobs that run up to 30 minutes, as stated in the scenario. Additionally, refactoring each script into a separate Lambda function would require significant code and dependency changes, violating the requirement for minimal refactoring.

Package the scripts into a container image. Deploy the image to AWS Batch with a managed compute environment on Amazon EC2. Define scheduling policies in AWS Batch to trigger jobs according to cron expressions - AWS Batch supports long-running jobs and scheduling, but it's designed for batch compute jobs with queues and dependency management, not for

lightweight, cron-like container jobs. Batch also has longer setup and initialization times and typically requires more configuration and compute environment management. Using EC2-based compute environments increases operational overhead.

Create a container image for each script. Use AWS Step Functions to define a workflow for all scheduled tasks. Use a Wait state to delay execution and run tasks using Step Functions' RunTask integration with ECS Fargate - AWS Step Functions can schedule tasks using Wait states, but this approach introduces unnecessary orchestration complexity for simple cron workloads. Step Functions are best suited for multi-step workflows, not individual recurring jobs.

References:

https://docs.aws.amazon.com/scheduler/latest/UserGuide/what-is-scheduler.html

https://docs.aws.amazon.com/lambda/latest/dg/gettingstarted-limits.html

https://docs.aws.amazon.com/step-functions/latest/dg/connect-ecs.html

https://docs.aws.amazon.com/batch/latest/userguide/what-is-batch.html

Domain

Design High-Performing Architectures

Question 53 Incorrect

A health-care solutions company wants to run their applications on single-tenant hardware to meet regulatory guidelines.

Which of the following is the MOST cost-effective way of isolating their Amazon Elastic Compute Cloud (Amazon EC2)instances to a single tenant?

Correct answer

Dedicated Instances

On-Demand Instances

Your answer is incorrect

Dedicated Hosts

Spot Instances

Overall explanation

Correct option:

Dedicated Instances

Dedicated Instances are Amazon EC2 instances that run in a virtual private cloud (VPC) on hardware that's dedicated to a single customer. Dedicated Instances that belong to different AWS accounts are physically isolated at a hardware level, even if those accounts are linked to a single-payer account. However, Dedicated Instances may share hardware with other instances from the same AWS account that are not Dedicated Instances.

A Dedicated Host is also a physical server that's dedicated for your use. With a Dedicated Host, you have visibility and control over how instances are placed on the server.

Differences between Dedicated Hosts and Dedicated Instances:

Differences between Dedicated Hosts and Dedicated Instances

Dedicated Hosts and Dedicated Instances can both be used to launch Amazon EC2 instances onto physical servers that are dedicated for your use.

There are no performance, security, or physical differences between Dedicated Instances and instances on Dedicated Hosts. However, there are some differences between the two. The following table highlights some of the key differences between Dedicated Hosts and Dedicated Instances:

	Dedicated Host	Dedicated Instance
Billing	Per-host billing	Per-instance billing
Visibility of sockets, cores, and host ID	Provides visibility of the number of sockets and physical cores	No visibility
Host and instance affinity	Allows you to consistently deploy your instances to the same physical server over time	Not supported
Targeted instance placement	Provides additional visibility and control over how instances are placed on a physical server	Not supported
Automatic instance recovery	Supported. For more information, see Host recovery.	Supported
Bring Your Own License (BYOL)	Supported	Not supported

via - https://docs.aws.amazon.com/AWSEC2/latest/UserGuide/dedicated-hosts-overview.html#dedicated-hosts-dedicated-instances

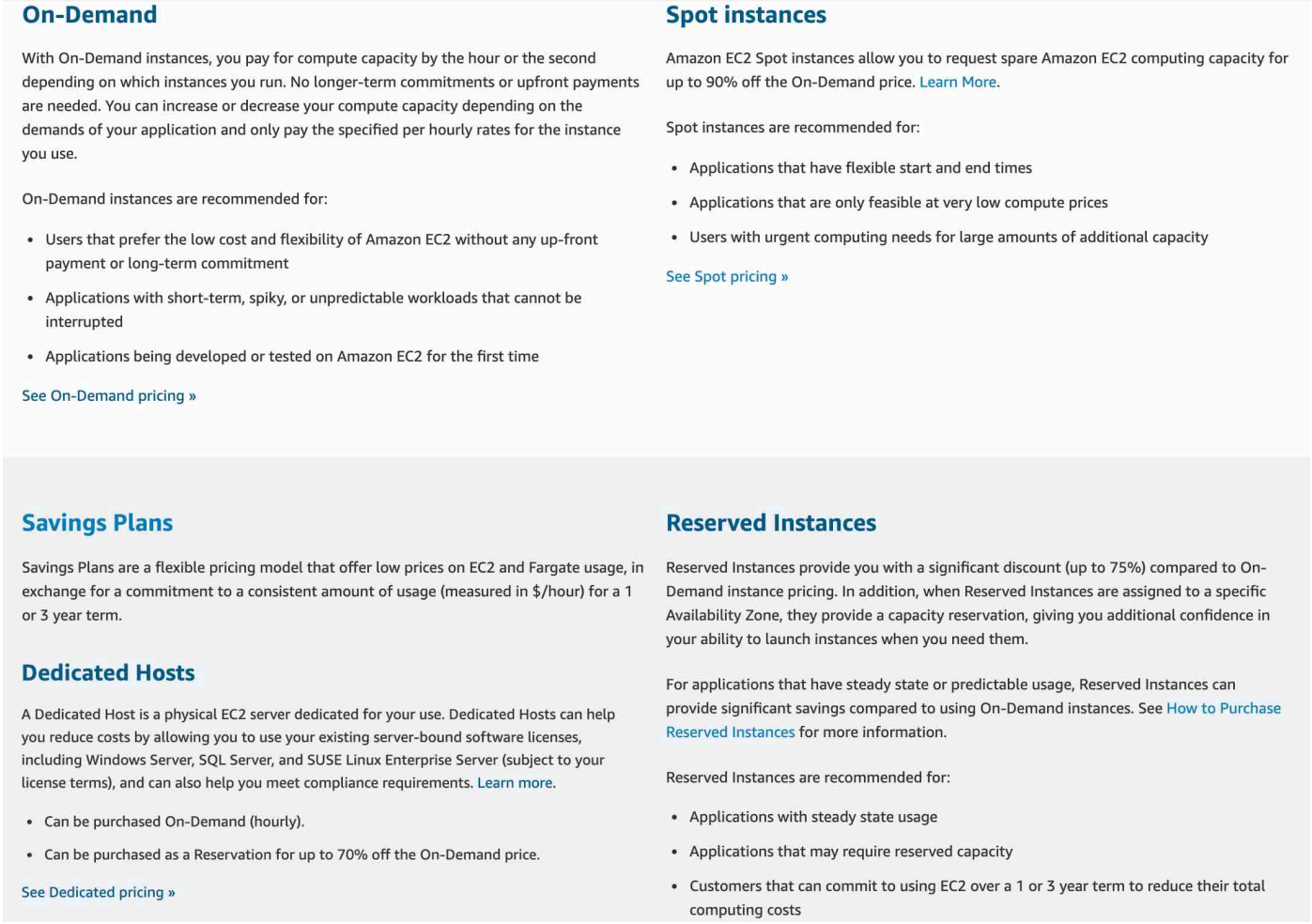
Incorrect options:

Spot Instances - A Spot Instance is an unused Amazon EC2 instance that is available for less than the On-Demand price. Your Spot Instance runs whenever capacity is available and the maximum price per hour for your request exceeds the Spot price. Any instance present with unused capacity will be allocated. Even though this is cost-effective, it does not fulfill the single-tenant hardware requirement of the client and hence is not the correct option.

Dedicated Hosts - An Amazon EC2 Dedicated Host is a physical server with EC2 instance capacity fully dedicated to your use. Dedicated Hosts allow you to use your existing software licenses on EC2 instances. With a Dedicated Host, you have visibility and control over how instances are placed on the server. This option is costlier than the Dedicated Instance and hence is not the right choice for the current requirement.

On-Demand Instances - With On-Demand Instances, you pay for compute capacity by the second with no long-term commitments. You have full control over its lifecycle—you decide when to launch, stop, hibernate, start, reboot, or terminate it. Hardware isolation is not possible and on-demand has one of the costliest instance charges and hence is not the correct answer for current requirements.

High Level Overview of Amazon EC2 Instance Purchase Options:



via - https://aws.amazon.com/ec2/pricing/

References:

https://docs.aws.amazon.com/AWSEC2/latest/UserGuide/dedicated-instance.html

https://docs.aws.amazon.com/AWSEC2/latest/UserGuide/instance-purchasing-options.html

Domain

Design Secure Architectures

Question 54 Incorrect

Consider the following policy associated with an IAM group containing several users:

```
{
    "Version":"2012-10-17",
    "Id":"EC2TerminationPolicy",
    "Statement":[
        {
            "Effect":"Deny",
            "Action":"ec2:*",
            "Resource":"*",
            "Condition":{
                "StringNotEquals":{
```

```
                    "ec2:Region":"us-west-1"
                }
            }
        },
        {
            "Effect":"Allow",
            "Action":"ec2:TerminateInstances",
            "Resource":"*",
            "Condition":{
                "IpAddress":{
                    "aws:SourceIp":"10.200.200.0/24"
                }
            }
        }
    ]
}
```

Which of the following options is correct?

Correct answer

Users belonging to the IAM user group can terminate an Amazon EC2 instance in the `us-west-1` region when the user's source IP is 10.200.200.200

Users belonging to the IAM user group cannot terminate an Amazon EC2 instance in the `us-west-1` region when the user's source IP is 10.200.200.200

Users belonging to the IAM user group can terminate an Amazon EC2 instance belonging to any region except the `us-west-1` region when the user's source IP is 10.200.200.200

Your answer is incorrect

Users belonging to the IAM user group can terminate an Amazon EC2 instance in the `us-west-1` region when the EC2 instance's IP address is 10.200.200.200

Overall explanation

Correct option:

Users belonging to the IAM user group can terminate an Amazon EC2 instance in the `us-west-1` region when the user's source IP is 10.200.200.200

The given policy denies all EC2 specification actions on all resources when the region of the underlying resource is not `us-west-1`. The policy allows the terminate EC2 action on all resources when the source IP address is in the CIDR range 10.200.200.0/24, therefore it would allow the user with the source IP 10.200.200.200 to terminate the Amazon EC2 instance.

Incorrect options:

Users belonging to the IAM user group cannot terminate an Amazon EC2 instance in the `us-west-1` region when the user's source IP is 10.200.200.200

Users belonging to the IAM user group can terminate an Amazon EC2 instance in the `us-west-1` region when the EC2 instance's IP address is 10.200.200.200

Users belonging to the IAM user group can terminate an Amazon EC2 instance belonging to any region except the `us-west-1` region when the user's source IP is 10.200.200.200

These three options contradict the explanation provided above, so these options are incorrect.

Reference:

https://docs.aws.amazon.com/IAM/latest/UserGuide/reference_policies_evaluation-logic.html

Domain

Design Secure Architectures

Question 55 Incorrect

A systems administrator has created a private hosted zone and associated it with a Virtual Private Cloud (VPC). However, the Domain Name System (DNS) queries for the private hosted zone remain unresolved.

As a Solutions Architect, can you identify the Amazon Virtual Private Cloud (Amazon VPC) options to be configured in order to get the private hosted zone to work?

Remove any overlapping namespaces for the private and public hosted zones

Correct answer

Enable DNS hostnames and DNS resolution for private hosted zones

Your answer is incorrect

Fix conflicts between your private hosted zone and any Resolver rule that routes traffic to your network for the same domain name, as it results in ambiguity over the route to be taken

Fix the Name server (NS) record and Start Of Authority (SOA) records that may have been created with wrong configurations

Overall explanation

Correct option:

Enable DNS hostnames and DNS resolution for private hosted zones

DNS hostnames and DNS resolution are required settings for private hosted zones. DNS queries for private hosted zones can be resolved by the Amazon-provided VPC DNS server only. As a result, these options must be enabled for your private hosted zone to work.

DNS hostnames: For non-default virtual private clouds that aren't created using the Amazon VPC wizard, this option is disabled by default. If you create a private hosted zone for a domain and create records in the zone without enabling DNS hostnames, private hosted zones aren't enabled. To use a private hosted zone, this option must be enabled.

DNS resolution: Private hosted zones accept DNS queries only from a VPC DNS server. The IP address of the VPC DNS server is the reserved IP address at the base of the VPC IPv4 network range plus two. Enabling DNS resolution allows you to use the VPC DNS server as a Resolver for performing DNS resolution. Keep this option disabled if you're using a custom DNS server in the DHCP Options set, and you're not using a private hosted zone.

Incorrect options:

Remove any overlapping namespaces for the private and public hosted zones - If you have private and public hosted zones that have overlapping namespaces, such as example.com and accounting.example.com, then the Resolver routes traffic based on the most specific match. It won't result in unresolved queries, hence this option is wrong.

Fix the Name server (NS) record and Start Of Authority (SOA) records that may have been created with wrong configurations - When you create a hosted zone, Amazon Route 53 automatically creates a name server (NS) record and a start of authority (SOA) record for the zone for public hosted zone. However, this issue is about the private hosted zone, hence this is an incorrect option.

Fix conflicts between your private hosted zone and any Resolver rule that routes traffic to your network for the same domain name, as it results in ambiguity over the route to be taken - If you have a private hosted zone (example.com) and a Resolver rule that routes traffic to your network for the same domain name, the Resolver rule takes precedence. It won't result in unresolved queries.

References:

https://aws.amazon.com/premiumsupport/knowledge-center/vpc-enable-private-hosted-zone/

https://docs.aws.amazon.com/Route53/latest/DeveloperGuide/hosted-zone-private-considerations.html

https://docs.aws.amazon.com/Route53/latest/DeveloperGuide/hosted-zone-public-considerations.html

Domain

Design Secure Architectures

Question 56 Incorrect

To improve the performance and security of the application, the engineering team at a company has created an Amazon CloudFront distribution with an Application Load Balancer as the custom origin. The team has also set up an AWS Web Application Firewall (AWS WAF) with Amazon CloudFront distribution. The security team at the company has noticed a surge in malicious attacks from a specific IP address to steal sensitive data stored on the Amazon EC2 instances.

As a solutions architect, which of the following actions would you recommend to stop the attacks?

Create a ticket with AWS support to take action against the malicious IP

Your answer is incorrect

Create a deny rule for the malicious IP in the Security Groups associated with each of the instances

Correct answer

Create an IP match condition in the AWS WAF to block the malicious IP address

Create a deny rule for the malicious IP in the network access control list (network ACL) associated with each of the instances

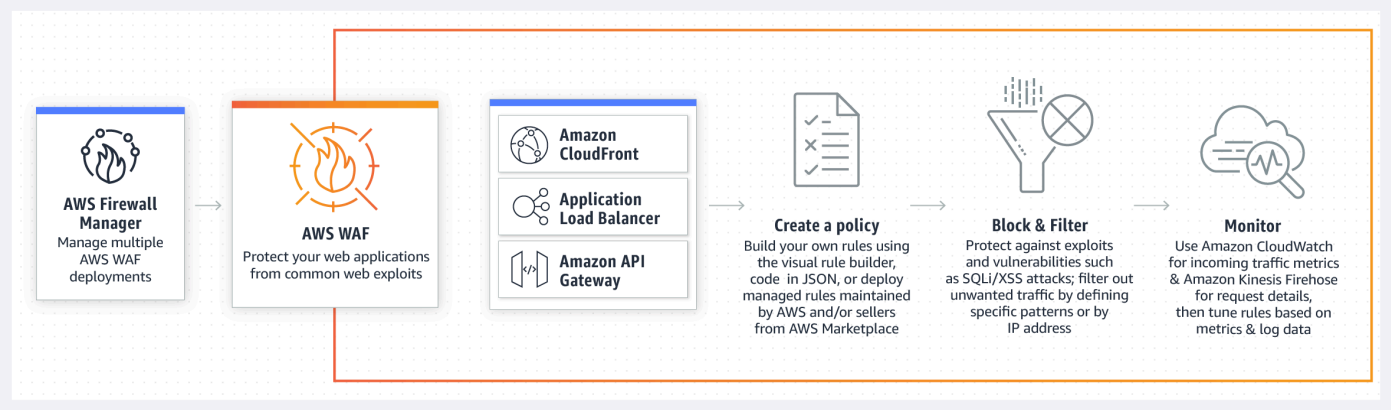
Overall explanation

Correct option:

Create an IP match condition in the AWS WAF to block the malicious IP address

AWS WAF is a web application firewall that helps protect your web applications or APIs against common web exploits that may affect availability, compromise security, or consume excessive resources. AWS WAF gives you control over how traffic reaches your applications by enabling you to create security rules that block common attack patterns, such as SQL injection or cross-site scripting, and rules that filter out specific traffic patterns you define.

How AWS WAF Works:



via - https://aws.amazon.com/waf/

If you want to allow or block web requests based on the IP addresses that the requests originate from, create one or more IP match conditions. An IP match condition lists up to 10,000 IP addresses or IP address ranges that your requests originate from. So, this option is correct.

Incorrect options:

Create a deny rule for the malicious IP in the network access control list (network ACL) associated with each of the instances - Network access control list (network ACL) are not associated with instances. So this option is also ruled out.

Create a deny rule for the malicious IP in the Security Groups associated with each of the instances - You cannot deny rules in Security Groups. So this option is ruled out.

Create a ticket with AWS support to take action against the malicious IP - Managing the security of your application is your responsibility, not that of AWS, so you cannot raise a ticket for this issue.

Reference:

https://docs.aws.amazon.com/waf/latest/developerguide/classic-web-acl-ip-conditions.html

Domain

Design Secure Architectures

Question 57 Correct

A financial services company has deployed its flagship application on Amazon EC2 instances. Since the application handles sensitive customer data, the security team at the company wants to ensure that any third-party Secure Sockets Layer certificate (SSL certificate) SSL/Transport Layer Security (TLS) certificates configured on Amazon EC2 instances via the AWS Certificate Manager (ACM) are renewed before their expiry date. The company has hired you as an AWS Certified Solutions Architect Associate to build a solution that notifies the security team 30 days before the certificate expiration. The solution should require the least amount of scripting and maintenance effort.

What will you recommend?

Monitor the `days to expiry` Amazon CloudWatch metric for certificates created via ACM. Create a CloudWatch alarm to monitor such certificates based on the `days to expiry` metric and then trigger a custom action of notifying the security team

Your answer is correct

Leverage AWS Config managed rule to check if any third-party SSL/TLS certificates imported into ACM are marked for expiration within 30 days. Configure the rule to trigger an Amazon SNS notification to the security team if any certificate expires within 30 days

Leverage AWS Config managed rule to check if any SSL/TLS certificates created via ACM are marked for expiration within 30 days. Configure the rule to trigger an Amazon SNS notification to the security team if any certificate expires within 30 days

Monitor the `days to expiry` Amazon CloudWatch metric for certificates imported into ACM. Create a CloudWatch alarm to monitor such certificates based on the `days to expiry` metric and then trigger a custom action of notifying the security team

Overall explanation

Correct option:

Leverage AWS Config managed rule to check if any third-party SSL/TLS certificates imported into ACM are marked for expiration within 30 days. Configure the rule to trigger an Amazon SNS notification to the security team if any certificate expires within 30 days

AWS Certificate Manager is a service that lets you easily provision, manage, and deploy public and private Secure Sockets Layer/Transport Layer Security (SSL/TLS) certificates for use with AWS services and your internal connected resources. SSL/TLS certificates are used to secure network communications and establish the identity of websites over the Internet as well as resources on private networks.

AWS Config provides a detailed view of the configuration of AWS resources in your AWS account. This includes how the resources are related to one another and how they were configured in the past so that you can see how the configurations and relationships change over time.

How AWS Config Works

PDF | RSS

When you turn on AWS Config, it first discovers the supported AWS resources that exist in your account and generates a configuration item for each resource.

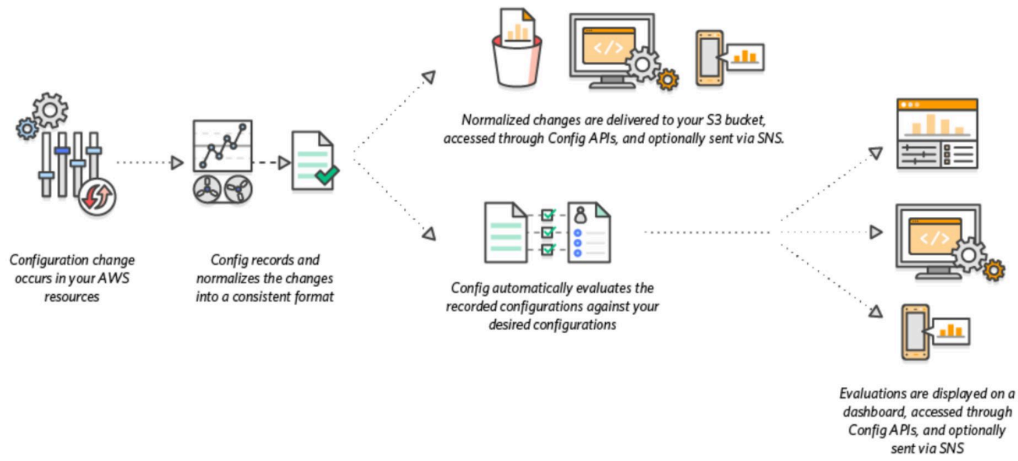
AWS Config also generates configuration items when the configuration of a resource changes, and it maintains historical records of the configuration items of your resources from the time you start the configuration recorder. By default, AWS Config creates configuration items for every supported resource in the region. If you don't want AWS Config to create configuration items for all supported resources, you can specify the resource types that you want it to track.

AWS Config keeps track of all changes to your resources by invoking the Describe or the List API call for each resource in your account. The service uses those same API calls to capture configuration details for all related resources.

For example, removing an egress rule from a VPC security group causes AWS Config to invoke a Describe API call on the security group. AWS Config then invokes a Describe API call on all of the instances associated with the security group. The updated configurations of the security group (the resource) and of each instance (the related resources) are recorded as configuration items and delivered in a configuration stream to an Amazon Simple Storage Service (Amazon S3) bucket.

AWS Config also tracks the configuration changes that were not initiated by the API. AWS Config examines the resource configurations periodically and generates configuration items for the configurations that have changed.

If you are using AWS Config rules, AWS Config continuously evaluates your AWS resource configurations for desired settings. Depending on the rule, AWS Config will evaluate your resources either in response to configuration changes or periodically. Each rule is associated with an AWS Lambda function, which contains the evaluation logic for the rule. When AWS Config evaluates your resources, it invokes the rule's AWS Lambda function. The function returns the compliance status of the evaluated resources. If a resource violates the conditions of a rule, AWS Config flags the resource and the rule as noncompliant. When the compliance status of a resource changes, AWS Config sends a notification to your Amazon SNS topic.



via - https://docs.aws.amazon.com/config/latest/developerguide/how-does-config-work.html

AWS Config provides AWS-managed rules, which are predefined, customizable rules that AWS Config uses to evaluate whether your AWS resources comply with common best practices. You can leverage an AWS Config managed rule to check if any ACM certificates in your account are marked for expiration within the specified number of days. Certificates provided by ACM are automatically renewed. ACM does not automatically renew the certificates that you import. The rule is NON_COMPLIANT if your certificates are about to expire.

acm-certificate-expiration-check

PDF | RSS

Checks if AWS Certificate Manager Certificates in your account are marked for expiration within the specified number of days. Certificates provided by ACM are automatically renewed. ACM does not automatically renew certificates that you import. The rule is NON_COMPLIANT if your certificates are about to expire.

Identifier: ACM_CERTIFICATE_EXPIRATION_CHECK

Trigger type: Configuration changes

AWS Region: All supported AWS regions except China (Beijing), China (Ningxia), Asia Pacific (Osaka), Europe (Milan) Region

Parameters:

daysToExpiration (Optional)
Type: int
Default: 14
Specify the number of days before the rule flags the ACM Certificate as noncompliant.

via - https://docs.aws.amazon.com/config/latest/developerguide/how-does-config-work.html

You can configure AWS Config to stream configuration changes and notifications to an Amazon SNS topic. For example, when a resource is updated, you can get a notification sent to your email, so that you can view the changes. You can also be notified when AWS Config evaluates your custom or managed rules against your resources.

Incorrect options:

Monitor the `days to expiry` Amazon CloudWatch metric for certificates imported into ACM. Create a CloudWatch alarm to monitor such certificates based on the `days to expiry` metric and then trigger a custom action of notifying the security team - AWS Certificate Manager (ACM) does not attempt to renew third-party certificates that are imported. Also, an administrator needs to reconfigure missing DNS records for certificates that use DNS validation if the record was removed for any reason after the certificate was issued. Metrics and events provide you visibility into such certificates that require intervention to continue the renewal process. Amazon CloudWatch metrics and Amazon EventBridge events are enabled for all certificates that are managed by ACM. Users can monitor `days to expiry` as a metric for ACM certificates through Amazon CloudWatch. An Amazon EventBridge expiry event is published for any certificate that is at least 45 days away from expiry by default. Users can build alarms to monitor certificates based on days to expiry and also trigger custom actions such as calling a Lambda function or paging an administrator.

It is certainly possible to use the `days to expiry` CloudWatch metric to build a CloudWatch alarm to monitor the imported ACM certificates. The alarm will, in turn, trigger a notification to the security team. But this option needs more configuration effort than directly using the AWS Config managed rule that is available off-the-shelf.

Leverage AWS Config managed rule to check if any SSL/TLS certificates created via ACM are marked for expiration within 30 days. Configure the rule to trigger an Amazon SNS notification to the security team if any certificate expires within 30 days

Monitor the `days to expiry` Amazon CloudWatch metric for certificates created via ACM. Create a CloudWatch alarm to monitor such certificates based on the `days to expiry` metric and then trigger a custom action of notifying the security team

Any SSL/TLS certificates created via ACM do not need any monitoring/intervention for expiration. ACM automatically renews such certificates. Hence both these options are incorrect.

References:

https://docs.aws.amazon.com/config/latest/developerguide/WhatIsConfig.html

https://docs.aws.amazon.com/config/latest/developerguide/how-does-config-work.html

https://docs.aws.amazon.com/config/latest/developerguide/evaluate-config.html

https://docs.aws.amazon.com/config/latest/developerguide/acm-certificate-expiration-check.html

https://aws.amazon.com/blogs/security/how-to-monitor-expirations-of-imported-certificates-in-aws-certificate-manager-acm/

Domain

Design Secure Architectures

Question 58 Incorrect

You have multiple AWS accounts within a single AWS Region managed by AWS Organizations and you would like to ensure all Amazon EC2 instances in all these accounts can communicate privately. Which of the following solutions provides the capability at the CHEAPEST cost?

Create a Private Link between all the Amazon EC2 instances

Your answer is incorrect

Create a VPC peering connection between all virtual private cloud (VPCs)

Create an AWS Transit Gateway and link all the virtual private cloud (VPCs) in all the accounts together

Correct answer

Create a virtual private cloud (VPC) in an account and share one or more of its subnets with the other accounts using Resource Access Manager

Overall explanation

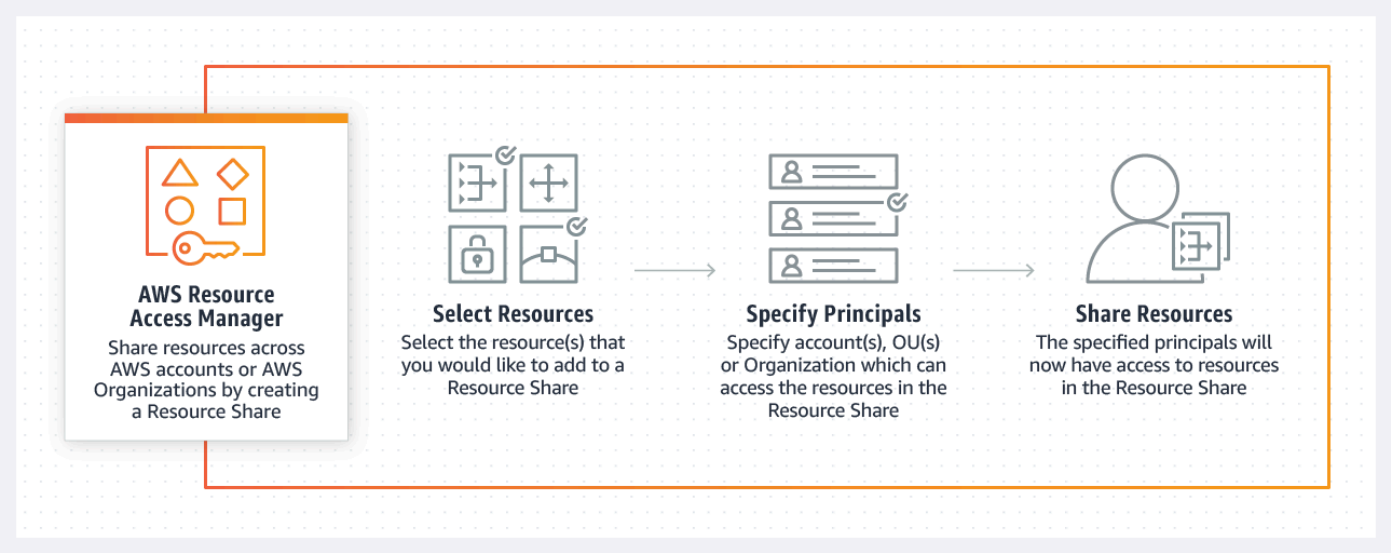
Correct option:

Create a virtual private cloud (VPC) in an account and share one or more of its subnets with the other accounts using Resource Access Manager

AWS Resource Access Manager (RAM) is a service that enables you to easily and securely share AWS resources with any AWS account or within your AWS Organization. You can share AWS Transit Gateways, Subnets, AWS License Manager configurations, and Amazon Route 53 Resolver rules resources with RAM. RAM eliminates the need to create duplicate resources in multiple accounts, reducing the operational overhead of managing those resources in every single account you own. You can create resources centrally in a multi-account environment, and use RAM to share those resources across accounts in three simple steps: create a Resource Share, specify resources, and specify accounts. RAM is available to you at no additional charge.

The correct solution is to share the subnet(s) within a VPC using RAM. This will allow all Amazon EC2 instances to be deployed in the same VPC (although from different accounts) and easily communicate with one another.

How AWS Resource Access Manager (AWS RAM) Works:



via - https://aws.amazon.com/ram/

Incorrect options:

Create a Private Link between all the Amazon EC2 instances - AWS PrivateLink simplifies the security of data shared with cloud-based applications by eliminating the exposure of data to the public Internet. AWS PrivateLink provides private connectivity between VPCs, AWS services, and on-premises applications, securely on the Amazon network. Private Link is a distractor in this question. Private Link is leveraged to create a private connection between an application that is fronted by an NLB in an account, and an Elastic Network Interface (ENI) in another account, without the need of VPC peering and allowing the connections between the two to remain within the AWS network.

Create a VPC peering connection between all virtual private cloud (VPCs) - A VPC peering connection is a networking connection between two VPCs that enables you to route traffic between them using private IPv4 addresses or IPv6 addresses. Instances in either VPC can communicate with each other as if they are within the same network. You can create a VPC peering connection between your VPCs, or with a VPC in another AWS account. The VPCs can be in different regions (also known as an inter-region VPC peering connection). VPC peering connections will work, but won't efficiently scale if you add more accounts (you'll have to create many connections).

Create an AWS Transit Gateway and link all the virtual private cloud (VPCs) in all the accounts together - AWS Transit Gateway is a service that enables customers to connect their Amazon Virtual Private Clouds (VPCs) and their on-premises networks to a single gateway. A Transit Gateway will work but will be an expensive solution. Here we want to minimize cost.

References:

https://aws.amazon.com/ram/

https://aws.amazon.com/privatelink/

https://docs.aws.amazon.com/vpc/latest/peering/what-is-vpc-peering.html

https://aws.amazon.com/transit-gateway/

Domain

Design Cost-Optimized Architectures

Question 59 Incorrect

A media company is migrating its flagship application from its on-premises data center to AWS for improving the application's read-scaling capability as well as its availability. The existing architecture leverages a Microsoft SQL Server database that sees a heavy read load. The engineering team does a full copy of the production database at the start of the business day to populate a dev database. During this period, application users face high latency leading to a bad user experience.

The company is looking at alternate database options and migrating database engines if required. What would you suggest?

Your answer is incorrect

Leverage Amazon RDS for SQL server with a Multi-AZ deployment and read replicas. Use the read replica as the dev database

Leverage Amazon RDS for MySQL with a Multi-AZ deployment and use the standby instance as the dev database

Correct answer

Leverage Amazon Aurora MySQL with Multi-AZ Aurora Replicas and create the dev database by restoring from the automated backups of Amazon Aurora

Leverage Amazon Aurora MySQL with Multi-AZ Aurora Replicas and restore the dev database via mysqldump

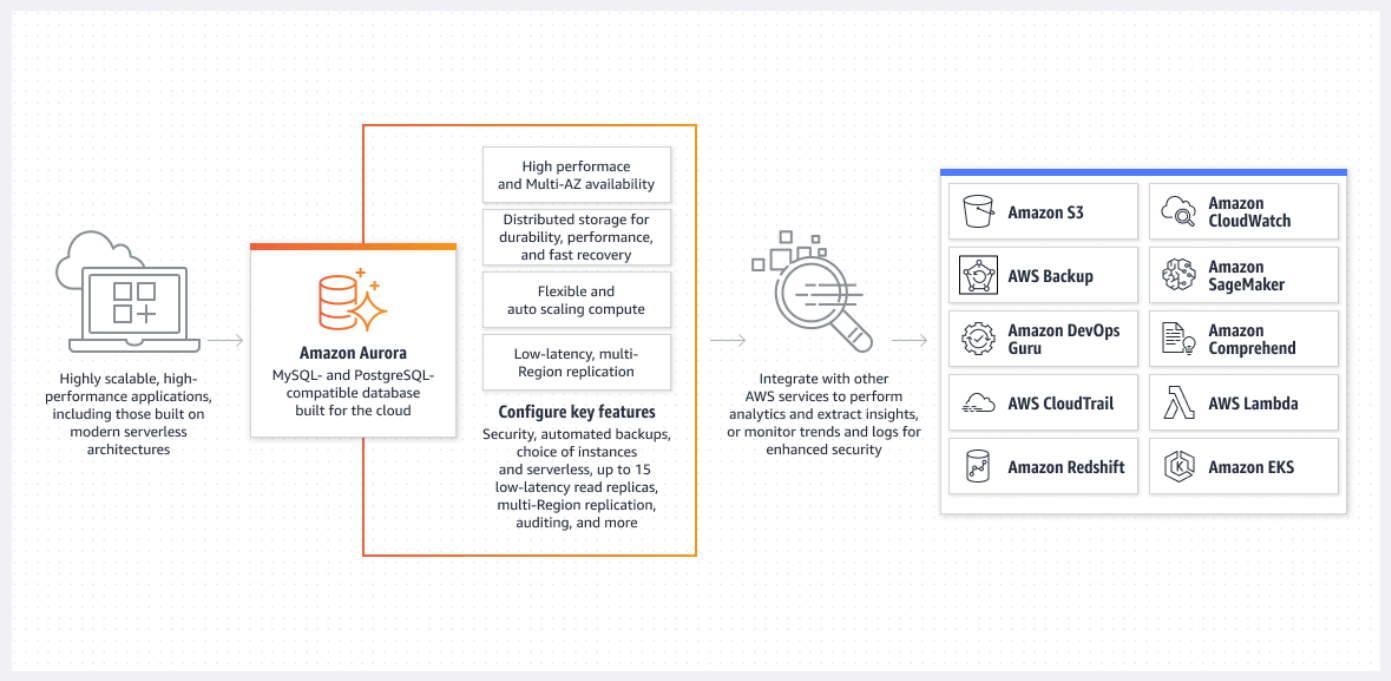
Overall explanation

Correct option:

Leverage Amazon Aurora MySQL with Multi-AZ Aurora Replicas and create the dev database by restoring from the automated backups of Amazon Aurora

Amazon Aurora (Aurora) is a fully managed relational database engine that's compatible with MySQL and PostgreSQL. An Amazon Aurora DB cluster consists of one or more DB instances and a cluster volume that manages the data for those DB instances. An Aurora cluster volume is a virtual database storage volume that spans multiple Availability Zones (AZs), with each Availability Zone (AZ) having a copy of the Amazon Aurora DB cluster data. Aurora supports Multi-AZ Aurora Replicas that improve the application's read-scaling and availability.

Amazon Aurora Overview:



via - https://aws.amazon.com/rds/aurora/

Aurora backs up your cluster volume automatically and retains restore data for the length of the backup retention period. Aurora backups are continuous and incremental so you can quickly restore to any point within the backup retention period. No performance impact or interruption of database service occurs as backup data is being written.

Backups

Aurora backs up your cluster volume automatically and retains restore data for the length of the *backup retention period*. Aurora backups are continuous and incremental so you can quickly restore to any point within the backup retention period. No performance impact or interruption of database service occurs as backup data is being written. You can specify a backup retention period, from 1 to 35 days, when you create or modify a DB cluster. Aurora backups are stored in Amazon S3.

If you want to retain a backup beyond the backup retention period, you can also take a snapshot of the data in your cluster volume. Because Aurora retains incremental restore data for the entire backup retention period, you only need to create a snapshot for data that you want to retain beyond the backup retention period. You can create a new DB cluster from the snapshot.

Note

- For Amazon Aurora DB clusters, the default backup retention period is one day regardless of how the DB cluster is created.
- You can't disable automated backups on Aurora. The backup retention period for Aurora is managed by the DB cluster.

via - https://docs.aws.amazon.com/AmazonRDS/latest/AuroraUserGuide/Aurora.Managing.Backups.html

Automated backups occur daily during the preferred backup window. If the backup requires more time than allotted to the backup window, the backup continues after the window ends, until it finishes. The backup window can't overlap with the weekly maintenance window for the DB cluster. Aurora backups are continuous and incremental, but the backup window is used to create a daily system backup that is preserved within the backup retention period. The latest restorable time for a DB cluster is the most recent point at which you can restore your DB cluster, typically within 5 minutes of the current time.

For the given use case, you can create the dev database by restoring from the automated backups of Amazon Aurora.

Incorrect options:

Leverage Amazon Aurora MySQL with Multi-AZ Aurora Replicas and restore the dev database via mysqldump - Restoring the dev database via mysqldump would still result in a

significant load on the primary DB, so this option fails to address the given requirement.

Leverage Amazon RDS for MySQL with a Multi-AZ deployment and use the standby instance as the dev database - The standby is there just for handling failover in a Multi-AZ deployment. You cannot access the standby instance and use it as a dev database. Hence this option is incorrect.

Leverage Amazon RDS for SQL server with a Multi-AZ deployment and read replicas. Use the read replica as the dev database - Amazon RDS supports Multi-AZ deployments for Microsoft SQL Server by using either SQL Server Database Mirroring (DBM) or Always On Availability Groups (AGs). Amazon RDS monitors and maintains the health of your Multi-AZ deployment.

Multi-AZ deployments provide increased availability, data durability, and fault tolerance for DB instances. In the event of planned database maintenance or unplanned service disruption, Amazon RDS automatically fails over to the up-to-date secondary DB instance. For SQL Server, I/O activity is suspended briefly during backup for Multi-AZ deployments.

A read replica is only meant to serve read traffic. The primary purpose of the read replica is to replicate the data in the primary DB instance. A read replica cannot be used as a dev database because it does not allow any database write operations.

References:

https://docs.aws.amazon.com/AmazonRDS/latest/AuroraUserGuide/Aurora.Managing.Backups.html

https://docs.aws.amazon.com/AmazonRDS/latest/UserGuide/SQLServer.ReadReplicas.html

Domain

Design Resilient Architectures

Question 60 **Incorrect**

The DevOps team at a major financial services company uses Multi-Availability Zone (Multi-AZ) deployment for its MySQL Amazon RDS database in order to automate its database replication and augment data durability. The DevOps team has scheduled a maintenance window for a database engine level upgrade for the coming weekend.

Which of the following is the correct outcome during the maintenance window?

Any database engine level upgrade for an Amazon RDS database instance with Multi-AZ deployment triggers both the primary and standby database instances to be upgraded at the same time. However, this does not cause any downtime until the upgrade is complete

Correct answer

Any database engine level upgrade for an Amazon RDS database instance with Multi-AZ deployment triggers both the primary and standby database instances to be upgraded at the same time. This causes downtime until the upgrade is complete

Your answer is incorrect

Any database engine level upgrade for an Amazon RDS database instance with Multi-AZ deployment triggers the primary database instance to be upgraded which is then followed by the upgrade of the standby database instance. This does not cause any downtime for the duration of the upgrade

Any database engine level upgrade for an Amazon RDS database instance with Multi-AZ deployment triggers the standby database instance to be upgraded which is then followed by the upgrade of the primary database instance. This does not cause any downtime for the duration of the upgrade

Overall explanation

Correct option:

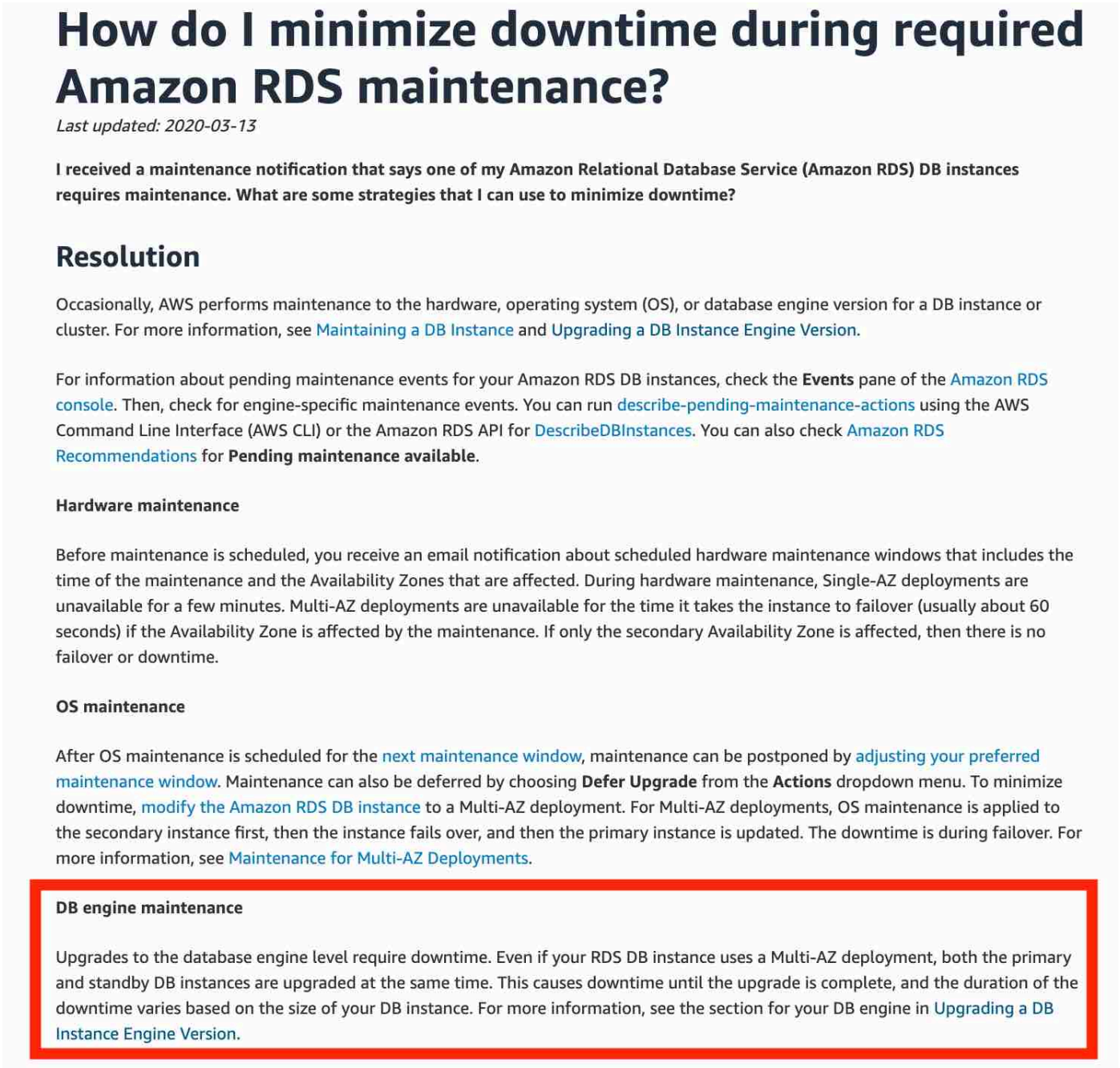
Any database engine level upgrade for an Amazon RDS database instance with Multi-AZ deployment triggers both the primary and standby database instances to be upgraded at the same time. This causes downtime until the upgrade is complete

Amazon Relational Database Service (Amazon RDS) makes it easy to set up, operate, and scale a relational database in the cloud. It provides cost-efficient and resizable capacity while automating

time-consuming administration tasks such as hardware provisioning, database setup, patching, and backups.

Upgrades to the database engine level require downtime. Even if your Amazon RDS DB instance uses a Multi-AZ deployment, both the primary and standby DB instances are upgraded at the same time. This causes downtime until the upgrade is complete, and the duration of the downtime varies based on the size of your database instance.

Amazon RDS DB Engine Maintenance:



How do I minimize downtime during required Amazon RDS maintenance?

Last updated: 2020-03-13

I received a maintenance notification that says one of my Amazon Relational Database Service (Amazon RDS) DB instances requires maintenance. What are some strategies that I can use to minimize downtime?

Resolution

Occasionally, AWS performs maintenance to the hardware, operating system (OS), or database engine version for a DB instance or cluster. For more information, see Maintaining a DB Instance and Upgrading a DB Instance Engine Version.

For information about pending maintenance events for your Amazon RDS DB instances, check the **Events** pane of the Amazon RDS console. Then, check for engine-specific maintenance events. You can run describe-pending-maintenance-actions using the AWS Command Line Interface (AWS CLI) or the Amazon RDS API for DescribeDBInstances. You can also check Amazon RDS Recommendations for **Pending maintenance available**.

Hardware maintenance

Before maintenance is scheduled, you receive an email notification about scheduled hardware maintenance windows that includes the time of the maintenance and the Availability Zones that are affected. During hardware maintenance, Single-AZ deployments are unavailable for a few minutes. Multi-AZ deployments are unavailable for the time it takes the instance to failover (usually about 60 seconds) if the Availability Zone is affected by the maintenance. If only the secondary Availability Zone is affected, then there is no failover or downtime.

OS maintenance

After OS maintenance is scheduled for the next maintenance window, maintenance can be postponed by adjusting your preferred maintenance window. Maintenance can also be deferred by choosing **Defer Upgrade** from the **Actions** dropdown menu. To minimize downtime, modify the Amazon RDS DB instance to a Multi-AZ deployment. For Multi-AZ deployments, OS maintenance is applied to the secondary instance first, then the instance fails over, and then the primary instance is updated. The downtime is during failover. For more information, see Maintenance for Multi-AZ Deployments.

DB engine maintenance

Upgrades to the database engine level require downtime. Even if your RDS DB instance uses a Multi-AZ deployment, both the primary and standby DB instances are upgraded at the same time. This causes downtime until the upgrade is complete, and the duration of the downtime varies based on the size of your DB instance. For more information, see the section for your DB engine in Upgrading a DB Instance Engine Version.

via - https://aws.amazon.com/premiumsupport/knowledge-center/rds-required-maintenance/

Incorrect options:

Any database engine level upgrade for an Amazon RDS database instance with Multi-AZ deployment triggers both the primary and standby database instances to be upgraded at the same time. However, this does not cause any downtime until the upgrade is complete - For Amazon RDS database engine level upgrade, primary and standby database instances are upgraded at the same time and it causes downtime until the upgrade is complete, hence this option is incorrect.

Any database engine level upgrade for an Amazon RDS database instance with Multi-AZ deployment triggers the standby database instance to be upgraded which is then followed by the upgrade of the primary database instance. This does not cause any downtime for the duration of the upgrade - For Amazon RDS database engine level upgrade, primary and standby database instances are upgraded at the same time and it causes downtime until the upgrade is complete, hence this option is incorrect.

Any database engine level upgrade for an Amazon RDS database instance with Multi-AZ deployment triggers the primary database instance to be upgraded which is then followed by the upgrade of the standby database instance. This does not cause any downtime for the duration of the upgrade - For Amazon RDS database engine level upgrade, primary and standby database instances are upgraded at the same time and it causes downtime until the upgrade is complete, hence this option is incorrect.

Reference:

https://aws.amazon.com/premiumsupport/knowledge-center/rds-required-maintenance/

Domain

Design Resilient Architectures

Question 61 Incorrect

An IT company has built a solution wherein an Amazon Redshift cluster writes data to an Amazon S3 bucket belonging to a different AWS account. However, it is found that the files created in the Amazon S3 bucket using the UNLOAD command from the Amazon Redshift cluster are not even accessible to the Amazon S3 bucket owner.

What could be the reason for this denial of permission for the bucket owner?

Correct answer

By default, an Amazon S3 object is owned by the AWS account that uploaded it. So the Amazon S3 bucket owner will not implicitly have access to the objects written by the Amazon Redshift cluster

The owner of an Amazon S3 bucket has implicit access to all objects in his bucket. Permissions are set on objects after they are completely copied to the target location. Since the

owner is unable to access the uploaded files, the write operation may be still in progress

When objects are uploaded to Amazon S3 bucket from a different AWS account, the S3 bucket owner will get implicit permissions to access these objects. This issue seems to be due to an upload error that can be fixed by providing manual access from AWS console

Your answer is incorrect

When two different AWS accounts are accessing an Amazon S3 bucket, both the accounts must share the bucket policies. An erroneous policy can lead to such permission failures

Overall explanation

Correct option:

By default, an Amazon S3 object is owned by the AWS account that uploaded it. So the Amazon S3 bucket owner will not implicitly have access to the objects written by the Amazon Redshift cluster - By default, an Amazon S3 object is owned by the AWS account that uploaded it. This is true even when the bucket is owned by another account. Because the Amazon Redshift data files from the UNLOAD command were put into your bucket by another account, you (the bucket owner) don't have default permission to access those files.

To get access to the data files, an AWS Identity and Access Management (IAM) role with cross-account permissions must run the UNLOAD command again. Follow these steps to set up the Amazon Redshift cluster with cross-account permissions to the bucket:

1. From the account of the Amazon S3 bucket, create an IAM role (Bucket Role) with permissions to the bucket.
2. From the account of the Amazon Redshift cluster, create another IAM role (Cluster Role) with permissions to assume the Bucket Role.
3. Update the Bucket Role to grant bucket access and create a trust relationship with the Cluster Role.

4. From the Amazon Redshift cluster, run the UNLOAD command using the Cluster Role and Bucket Role.

This solution doesn't apply to Amazon Redshift clusters or Amazon S3 buckets that use server-side encryption with AWS Key Management Service (AWS KMS).

Incorrect options:

When objects are uploaded to Amazon S3 bucket from a different AWS account, the S3 bucket owner will get implicit permissions to access these objects. This issue seems to be due to an upload error that can be fixed by providing manual access from AWS console - By default, an Amazon S3 object is owned by the AWS account that uploaded it. So, the bucket owner will not have any default permissions on the objects. Therefore, this option is incorrect.

The owner of an Amazon S3 bucket has implicit access to all objects in his bucket. Permissions are set on objects after they are completely copied to the target location. Since the owner is unable to access the uploaded files, the write operation may be still in progress - This is an incorrect statement, given only as a distractor.

When two different AWS accounts are accessing an Amazon S3 bucket, both the accounts must share the bucket policies. An erroneous policy can lead to such permission failures - This is an incorrect statement, given only as a distractor.

Reference:

https://aws.amazon.com/premiumsupport/knowledge-center/s3-access-denied-redshift-unload/

Domain

Design Secure Architectures

Question 62 **Correct**

A startup has just developed a video backup service hosted on a fleet of Amazon EC2 instances. The Amazon EC2 instances are behind an Application Load Balancer and the instances are using Amazon Elastic Block Store (Amazon EBS) Volumes for storage. The service provides authenticated users the ability to upload videos that are then saved on the EBS volume attached to a given instance. On the first day of the beta launch, users start complaining that they can see only some of the videos in their uploaded videos backup. Every time the users log into the website, they claim to see a different subset of their uploaded videos.

Which of the following is the MOST optimal solution to make sure that users can view all the uploaded videos? (Select two)

Your selection is correct

Mount Amazon Elastic File System (Amazon EFS) on all Amazon EC2 instances. Write a one time job to copy the videos from all Amazon EBS volumes to Amazon EFS. Modify the application to use Amazon EFS for storing the videos

Write a one time job to copy the videos from all Amazon EBS volumes to Amazon RDS and then modify the application to use Amazon RDS for storing the videos

Write a one time job to copy the videos from all Amazon EBS volumes to Amazon DynamoDB and then modify the application to use Amazon DynamoDB for storing the videos

Your selection is correct

Write a one time job to copy the videos from all Amazon EBS volumes to Amazon S3 and then modify the application to use Amazon S3 standard for storing the videos

Write a one time job to copy the videos from all Amazon EBS volumes to Amazon S3 Glacier Deep Archive and then modify the application to use Amazon S3 Glacier Deep Archive for storing the videos

Overall explanation

Correct options:

Write a one time job to copy the videos from all Amazon EBS volumes to Amazon S3 and then modify the application to use Amazon S3 standard for storing the videos

Mount Amazon Elastic File System (Amazon EFS) on all Amazon EC2 instances. Write a one time job to copy the videos from all Amazon EBS volumes to Amazon EFS. Modify the application to use Amazon EFS for storing the videos

Amazon Elastic Block Store (EBS) is an easy to use, high-performance block storage service designed for use with Amazon Elastic Compute Cloud (EC2) for both throughput and transaction-intensive workloads at any scale.

Amazon Elastic File System (Amazon EFS) provides a simple, scalable, fully managed elastic NFS file system for use with AWS Cloud services and on-premises resources. It is built to scale on-demand to petabytes without disrupting applications, growing and shrinking automatically as you add and remove files, eliminating the need to provision and manage capacity to accommodate growth.

Amazon Simple Storage Service (Amazon S3) is an object storage service that offers industry-leading scalability, data availability, security, and performance.

As Amazon EBS volumes are attached locally to the Amazon EC2 instances, therefore the uploaded videos are tied to specific Amazon EC2 instances. Every time the user logs in, they are directed to a different instance and therefore their videos get dispersed across multiple EBS volumes. The correct solution is to use either Amazon S3 or Amazon EFS to store the user videos.

Incorrect options:

Write a one time job to copy the videos from all Amazon EBS volumes to Amazon S3 Glacier Deep Archive and then modify the application to use Amazon S3 Glacier Deep Archive for storing the videos - Amazon S3 Glacier Deep Archive is meant to be used for long term data archival. It cannot be used to serve static content such as videos or images via a web application. So this option is incorrect.

Write a one time job to copy the videos from all Amazon EBS volumes to Amazon RDS and then modify the application to use Amazon RDS for storing the videos - Amazon RDS is a relational database and not the right candidate for storing videos.

Write a one time job to copy the videos from all Amazon EBS volumes to Amazon DynamoDB and then modify the application to use Amazon DynamoDB for storing the videos - Amazon DynamoDB is a NoSQL database and not the right candidate for storing videos.

Reference:

https://aws.amazon.com/ebs/

Domain

Design Resilient Architectures

Question 63 Correct

A company has historically operated only in the `us-east-1` region and stores encrypted data in Amazon S3 using SSE-KMS. As part of enhancing its security posture as well as improving the backup and recovery architecture, the company wants to store the encrypted data in Amazon S3 that is replicated into the `us-west-1` AWS region. The security policies mandate that the data must be encrypted and decrypted using the same key in both AWS regions.

Which of the following represents the best solution to address these requirements?

Your answer is correct

Create a new Amazon S3 bucket in the `us-east-1` region with replication enabled from this new bucket into another bucket in `us-west-1` region. Enable SSE-KMS encryption on the new bucket in `us-east-1` region by using an AWS KMS multi-region key. Copy the existing data from the current Amazon S3 bucket in `us-east-1` region into this new Amazon S3 bucket in `us-east-1` region

Create an Amazon CloudWatch scheduled rule to invoke an AWS Lambda function to copy the daily data from the source bucket in `us-east-1` region to the destination bucket in `us-west-1` region. Provide AWS KMS key access to the AWS Lambda function for encryption and decryption operations on the data in the source and destination Amazon S3 buckets

Change the AWS KMS single region key used for the current Amazon S3 bucket into an AWS KMS multi-region key. Enable Amazon S3 batch replication for the existing data in the current bucket in `us-east-1` region into another bucket in `us-west-1` region

Enable replication for the current bucket in `us-east-1` region into another bucket in `us-west-1` region. Share the existing AWS KMS key from `us-east-1` region to `us-west-1` region

Overall explanation

Correct option:

Create a new Amazon S3 bucket in the `us-east-1` region with replication enabled from this new bucket into another bucket in `us-west-1` region. Enable SSE-KMS encryption on the new bucket in `us-east-1` region by using an AWS KMS multi-region key. Copy the existing data from the current Amazon S3 bucket in `us-east-1` region into this new Amazon S3 bucket in `us-east-1` region

AWS KMS supports multi-region keys, which are AWS KMS keys in different AWS regions that can be used interchangeably – as though you had the same key in multiple regions. Each set of related multi-region keys has the same key material and key ID, so you can encrypt data in one AWS region and decrypt it in a different AWS region without re-encrypting or making a cross-region call to AWS KMS.

You can use multi-region AWS KMS keys in Amazon S3. However, Amazon S3 currently treats multi-region keys as though they were single-region keys, and does not use the multi-region features of the key.

Multi-region AWS KMS keys:

Multi-Region keys in AWS KMS

PDF | RSS

AWS KMS supports *multi-Region keys*, which are AWS KMS keys in different AWS Regions that can be used interchangeably – as though you had the same key in multiple Regions. Each set of *related* multi-Region keys has the same key material and key ID, so you can encrypt data in one AWS Region and decrypt it in a different AWS Region without re-encrypting or making a cross-Region call to AWS KMS.

Like all KMS keys, multi-Region keys never leave AWS KMS unencrypted. You can create symmetric or asymmetric multi-Region keys for encryption or signing, create HMAC multi-Region keys for generating and verifying HMAC tags, and create multi-Region keys with imported key material or key material that AWS KMS generates. You must manage each multi-Region key independently, including creating aliases and tags, setting their key policies and grants, and enabling and disabling them selectively. You can use multi-Region keys in all cryptographic operations that you can do with single-Region keys.

Multi-Region keys are a flexible and powerful solution for many common data security scenarios.

Disaster recovery

In a backup and recovery architecture, multi-Region keys let you process encrypted data without interruption even in the event of an AWS Region outage. Data maintained in backup Regions can be decrypted in the backup Region, and data newly encrypted in the backup Region can be decrypted in the primary Region when that Region is restored.

Global data management

Businesses that operate globally need globally distributed data that is available consistently across AWS Regions. You can create multi-Region keys in all Regions where your data resides, then use the keys as though they were a single-Region key without the latency of a cross-Region call or the cost of re-encrypting data under a different key in each Region.

Distributed signing applications

Applications that require cross-Region signature capabilities can use multi-Region asymmetric signing keys to generate identical digital signatures consistently and repeatedly in different AWS Regions.

If you use certificate chaining with a single global trust store (for a single root certification authority (CA), and Regional intermediate CAs signed by the root CA, you don't need multi-Region keys. However, if your system doesn't support intermediate CAs, such as application signing, you can use multi-Region keys to bring consistency to Regional certifications.

Active-active applications that span multiple Regions

Some workloads and applications can span multiple Regions in active-active architectures. For these applications, multi-Region keys can reduce complexity by providing the same key material for concurrent encrypt and decrypt operations on data that might be moving across Region boundaries.

You can use multi-Region keys with client-side encryption libraries, such as the AWS Encryption SDK, the DynamoDB Encryption Client, and Amazon S3 client-side encryption. For an example of using multi-Region keys with Amazon DynamoDB global tables and the DynamoDB Encryption Client, see Encrypt global data client-side with AWS KMS multi-Region keys in the AWS Security Blog.

AWS services that integrate with AWS KMS for encryption at rest or digital signatures currently treat multi-Region keys as though they were single-Region keys. They might re-wrap or re-encrypt data moved between Regions. For example, Amazon S3 cross-region replication decrypts and re-encrypts data under a KMS key in the destination Region, even when replicating objects protected by a multi-Region key.

Multi-Region keys are not global. You create a multi-Region primary key and then replicate it into Regions that you select within an AWS partition. Then you manage the multi-Region key in each Region independently. Neither AWS nor AWS KMS ever automatically creates or replicates multi-Region keys into any Region on your behalf. AWS managed keys, the KMS keys that AWS services create in your account for you, are always single-Region keys.

You cannot convert an existing single-Region key to a multi-Region key. This design ensures that all data protected with existing single-Region keys maintain the same data residency and data sovereignty properties.

For most data security needs, the Regional isolation and fault tolerance of Regional resources make standard AWS KMS single-Region keys a best-fit solution. However, when you need to encrypt or sign data in client-side applications across multiple Regions, multi-Region keys might be the solution.

via - https://docs.aws.amazon.com/kms/latest/developerguide/multi-region-keys-overview.html

For the given use case, you must create a new bucket in the `us-east-1` region with replication enabled from this new bucket into another bucket in `us-west-1` region. This would ensure that the data is available in another region for backup and recovery purposes. You should also enable SSE-KMS encryption on the new bucket in `us-east-1` region by using an AWS KMS multi-region key so that the data can be encrypted and decrypted using the same key in both AWS regions. Since the existing data in the current bucket was encrypted using the AWS KMS key restricted to the `us-east-1` region, so data must be copied to the new bucket in `us-east-1` region for replication as well as multi-region KMS key based encryption to kick-in.

To require server-side encryption of all objects in a particular Amazon S3 bucket, you can use a policy. For example, the following bucket policy denies the upload object (s3:PutObject) permission to everyone if the request does not include the x-amz-server-side-encryption header requesting server-side encryption with SSE-KMS.

```
{
   "Version":"2012-10-17",
   "Id":"PutObjectPolicy",
   "Statement":[{
         "Sid":"DenyUnEncryptedObjectUploads",
         "Effect":"Deny",
         "Principal":"*",
         "Action":"s3:PutObject",
         "Resource":"arn:aws:s3:::DOC-EXAMPLE-BUCKET1/*",
         "Condition":{
            "StringNotEquals":{
               "s3:x-amz-server-side-encryption":"aws:kms"
            }
         }
      }
   ]
}
```

The following example IAM policies show statements for using AWS KMS server-side encryption with replication.

In this example, the encryption context is the object ARN. If you use SSE-KMS with an Amazon S3 Bucket Key enabled, you must use the bucket ARN as the encryption context.

```
{
    "Version": "2012-10-17",
    "Statement": [{
            "Action": ["kms:Decrypt"],
            "Effect": "Allow",
            "Resource": "List of AWS KMS key ARNs used to encrypt source
objects.",
            "Condition": {
                "StringLike": {
```

```
                    "kms:ViaService": "s3.source-bucket-region.amazonaws.com",
                    "kms:EncryptionContext:aws:s3:arn": "arn:aws:s3:::source-
bucket-name/key-prefix1/*"
                }
            }
        },

        {
            "Action": ["kms:Encrypt"],
            "Effect": "Allow",
            "Resource": "AWS KMS key ARNs (for the AWS Region of the destination
bucket 1). Used to encrypt object replicas created in destination bucket 1.",
            "Condition": {
                "StringLike": {
                    "kms:ViaService": "s3.destination-bucket-1-
region.amazonaws.com",
                    "kms:EncryptionContext:aws:s3:arn": "arn:aws:s3:::destination-
bucket-name-1/key-prefix1/*"
                }
            }
        },
        {
            "Action": ["kms:Encrypt"],
            "Effect": "Allow",
            "Resource": "AWS KMS key ARNs (for the AWS Region of destination
bucket 2). Used to encrypt object replicas created in destination bucket 2.",
            "Condition": {
                "StringLike": {
                    "kms:ViaService": "s3.destination-bucket-2-
region.amazonaws.com",
                    "kms:EncryptionContext:aws:s3:arn": "arn:aws:s3:::destination-
bucket-2-name/key-prefix1*"
                }
            }
        }
    ]
}
```

Incorrect options:

Change the AWS KMS single region key used for the current Amazon S3 bucket into an AWS KMS multi-region key. Enable Amazon S3 batch replication for the existing data in the current bucket in `us-east-1` region into another bucket in `us-west-1` region - Amazon S3 batch replication can certainly be used to replicate the existing data in the current bucket in `us-east-1` region into another bucket in `us-west-1` region.

However, you cannot convert an existing single-Region key to a multi-Region key. This design ensures that all data protected with existing single-Region keys maintain the same data residency and data sovereignty properties. So this option is incorrect.

Enable replication for the current bucket in `us-east-1` region into another bucket in `us-west-1` region. Share the existing AWS KMS key from `us-east-1` region to `us-west-1` region - You cannot share an AWS KMS key to another region, so this option is incorrect.

Create an Amazon CloudWatch scheduled rule to invoke an AWS Lambda function to copy the daily data from the source bucket in `us-east-1` region to the destination bucket in `us-west-1` region. Provide AWS KMS key access to the AWS Lambda function for encryption and decryption operations on the data in the source and destination Amazon S3 buckets - This option is a distractor as the daily frequency of data replication would result in significant data loss in case of a disaster. In addition, this option involves significant development effort to create the functionality to reliably replicate the data from source to destination buckets. So this option is not the best fit for the given use case.

References:

https://docs.aws.amazon.com/kms/latest/developerguide/multi-region-keys-overview.html

https://docs.aws.amazon.com/AmazonS3/latest/userguide/replication-config-for-kms-objects.html

Domain

Design Secure Architectures

Question 64 **Correct**

A developer needs to implement an AWS Lambda function in AWS account A that accesses an Amazon Simple Storage Service (Amazon S3) bucket in AWS account B.

As a Solutions Architect, which of the following will you recommend to meet this requirement?

Create an IAM role for the AWS Lambda function that grants access to the Amazon S3 bucket. Set the IAM role as the Lambda function's execution role and that would give the AWS Lambda function cross-account access to the Amazon S3 bucket

AWS Lambda cannot access resources across AWS accounts. Use Identity federation to work around this limitation of Lambda

The Amazon S3 bucket owner should make the bucket public so that it can be accessed by the AWS Lambda function in the other AWS account

Your answer is correct

Create an IAM role for the AWS Lambda function that grants access to the Amazon S3 bucket. Set the IAM role as the AWS Lambda function's execution role. Make sure that the bucket policy also grants access to the AWS Lambda function's execution role

Overall explanation

Correct option:

Create an IAM role for the AWS Lambda function that grants access to the Amazon S3 bucket. Set the IAM role as the AWS Lambda function's execution role. Make sure that the bucket policy also grants access to the AWS Lambda function's execution role

If the IAM role that you create for the Lambda function is in the same AWS account as the bucket, then you don't need to grant Amazon S3 permissions on both the IAM role and the bucket policy. Instead, you can grant the permissions on the IAM role and then verify that the bucket policy doesn't explicitly deny access to the Lambda function role. If the IAM role and the bucket are in different accounts, then you need to grant Amazon S3 permissions on both the IAM role and the bucket policy. Therefore, this is the right way of giving access to AWS Lambda for the given use-case.

Complete list of steps to be followed:

I want my AWS Lambda function to be able to access my Amazon Simple Storage Service (Amazon S3) bucket. How can I do that?

Short Description

Follow these steps:

1. Create an AWS Identity and Access Management (IAM) role for the Lambda function that also grants access to the S3 bucket.
2. Set the IAM role as the Lambda function's execution role.
3. Verify that the bucket policy grants access to the Lambda function's execution role.

Important: If the IAM role that you create for the Lambda function is in the same AWS account as the bucket, then you don't need to grant Amazon S3 permissions on both the IAM role and the bucket policy. Instead, you can grant the permissions on the IAM role and then verify that the bucket policy doesn't explicitly deny access to the Lambda function role. As an example, the following procedure grants Amazon S3 permissions on the IAM role. If the IAM role and the bucket are in different accounts, then you need to grant Amazon S3 permissions on both the IAM role and the bucket policy.

via - https://aws.amazon.com/premiumsupport/knowledge-center/lambda-execution-role-s3-bucket/

Incorrect options:

AWS Lambda cannot access resources across AWS accounts. Use Identity federation to work around this limitation of Lambda - This is an incorrect statement, used only as a distractor.

Create an IAM role for the AWS Lambda function that grants access to the Amazon S3 bucket. Set the IAM role as the Lambda function's execution role and that would give the AWS Lambda function cross-account access to the Amazon S3 bucket - When the execution role of AWS Lambda and Amazon S3 bucket to be accessed are from different accounts, then you need to grant Amazon S3 bucket access permissions to the IAM role and also ensure that the bucket policy grants access to the AWS Lambda function's execution role.

The Amazon S3 bucket owner should make the bucket public so that it can be accessed by the AWS Lambda function in the other AWS account - Making the Amazon S3 bucket public for the given use-case will be considered as a security bad practice. It's usually done for very few use-cases such as hosting a website on Amazon S3. Therefore this option is incorrect.

Reference:

https://aws.amazon.com/premiumsupport/knowledge-center/lambda-execution-role-s3-bucket/

Domain

Design Secure Architectures

Question 65 Correct

Upon a security review of your AWS account, an AWS consultant has found that a few Amazon RDS databases are unencrypted. As a Solutions Architect, what steps must be taken to encrypt the Amazon RDS databases?

Enable Multi-AZ for the database, and make sure the standby instance is encrypted. Stop the main database to that the standby database kicks in, then disable Multi-AZ

Enable encryption on the Amazon RDS database using the AWS Console

Your answer is correct

Take a snapshot of the database, copy it as an encrypted snapshot, and restore a database from the encrypted snapshot. Terminate the previous database

Create a Read Replica of the database, and encrypt the read replica. Promote the read replica as a standalone database, and terminate the previous database

Overall explanation

Correct option:

Take a snapshot of the database, copy it as an encrypted snapshot, and restore a database from the encrypted snapshot. Terminate the previous database

Amazon Relational Database Service (Amazon RDS) makes it easy to set up, operate, and scale a relational database in the cloud. It provides cost-efficient and resizable capacity while automating time-consuming administration tasks such as hardware provisioning, database setup, patching and backups.

You can encrypt your Amazon RDS DB instances and snapshots at rest by enabling the encryption option for your Amazon RDS DB instances. Data that is encrypted at rest includes the underlying storage for DB instances, its automated backups, read replicas, and snapshots.

You can only enable encryption for an Amazon RDS DB instance when you create it, not after the DB instance is created. However, because you can encrypt a copy of an unencrypted DB snapshot, you can effectively add encryption to an unencrypted DB instance. That is, you can create a snapshot of your DB instance, and then create an encrypted copy of that snapshot. So this is the correct option.

Incorrect options:

Create a Read Replica of the database, and encrypt the read replica. Promote the read replica as a standalone database, and terminate the previous database - If the master is not encrypted, the read replicas cannot be encrypted. So this option is incorrect.

Enable Multi-AZ for the database, and make sure the standby instance is encrypted. Stop the main database to that the standby database kicks in, then disable Multi-AZ - Multi-AZ is to help with High Availability, not encryption. So this option is incorrect.

Enable encryption on the Amazon RDS database using the AWS Console - There is no direct option to encrypt an Amazon RDS database using the AWS Console.

Steps to encrypt an un-encrypted RDS database: 1. Create a snapshot of the un-encrypted database 2. Copy the snapshot and enable encryption for the snapshot 3. Restore the database from the encrypted snapshot 4. Migrate applications to the new database, and delete the old database

Reference:

https://docs.aws.amazon.com/AmazonRDS/latest/UserGuide/Overview.Encryption.html

Domain

Design Secure Architectures