AWS SAP-C02 Exam Questions

Total Questions: 530+
Demo Questions: 35
Version: Updated for 2025

Prepared and Verified by Cert Empire – Your Trusted IT Certification Partner

For Access to the full set of Updated Questions – Visit:
AWS SAP-C02 Exam Questions by Cert Empire

Question: 1

A solutions architect at a large company needs to set up network security tor outbound traffic to the
internet from all AWS accounts within an organization in AWS Organizations. The organization has
more than 100 AWS accounts, and the accounts route to each other by using a centralized AWS Transit Gateway. Each account has both an internet gateway and a NAT gateway tor outbound traffic
to the internet The company deploys resources only into a single AWS Region.
The company needs the ability to add centrally managed rule-based filtering on all outbound traffic
to the internet for all AWS accounts in the organization. The peak load of outbound traffic will not exceed 25 Gbps in each Availability Zone.
Which solution meets these requirements?

A. Create a new VPC for outbound traffic to the internet. Connect the existing transit gateway to the
new VPC. Configure a new NAT gateway. Create an Auto Scaling group of Amazon EC2 instances that
run an open-source internet proxy for rule-based filtering across all Availability Zones in the Region.

Modify all default routes to point to the proxy's Auto Scaling group.

B. Create a new VPC for outbound traffic to the internet. Connect the existing transit gateway to the
new VPC. Configure a new NAT gateway. Use an AWSNetwork Firewall firewall for rule-based filtering. Create Network Firewall endpoints in each Availability Zone. Modify all default routes to point to the Network Firewall endpoints.

C. Create an AWS Network Firewall firewall for rule-based filtering in each AWS account. Modify all
default routes to point to the Network Firewall firewalls in each account.

D. In each AWS account, create an Auto Scaling group of network-optimized Amazon EC2 instances
that run an open-source internet proxy for rule-based filtering. Modify all default routes to point to the proxy's Auto Scaling group.

Answer:

B

Explanation:

This solution describes the standard AWS architecture for centralized egress traffic filtering. By creating a dedicated inspection VPC attached to the Transit Gateway, all outbound traffic from the 100+ spoke VPCs can be routed through a single, centrally managed point. AWS Network Firewall is a managed, highly available, and scalable service designed for this exact use case, providing stateful, rule-based filtering. Creating Network Firewall endpoints in each Availability Zone ensures high availability and allows traffic to be inspected without crossing AZ boundaries, which is efficient. This architecture simplifies management, policy enforcement, and monitoring for the entire organization.

Why Incorrect Options are Wrong:

A. Using self-managed EC2 instances for proxying adds significant operational overhead for patching, scaling, and maintenance compared to using a managed service like AWS Network Firewall.

C. This approach is decentralized, contradicting the core requirement for a centrally managed solution. Managing firewall rules across more than 100 individual accounts would be complex and inefficient.

D. This is the least optimal solution, combining the high operational overhead of self-managed EC2 proxies with the complexity and cost of a decentralized, per-account deployment.

References:

1. AWS Documentation, AWS Network Firewall Developer Guide: The "Deployment models for AWS Network Firewall" section explicitly details the "Centralized inspection architecture with AWS Transit Gateway." It describes routing traffic from spoke VPCs to an inspection VPC that contains the Network Firewall endpoints for centralized filtering before egressing to the internet. (See: AWS Network Firewall Developer Guide - Firewall deployment models - Centralized inspection architecture).
2. AWS Whitepaper, "Building a Scalable and Secure Multi-VPC AWS Network Infrastructure": This whitepaper discusses using a central "services" or "inspection" VPC connected to a Transit Gateway to centralize security functions. Section "Centralized network security" describes how services like AWS Network Firewall can be deployed in this VPC to inspect traffic to and from the internet. (See page 21, "Centralized network security").
3. AWS Architecture Blog, "Deployment models for AWS Network Firewall": This blog post provides detailed diagrams and explanations for various deployment models. The "Centralized egress" model shows a Transit Gateway routing traffic from multiple spoke VPCs to an inspection VPC, where Network Firewall filters traffic before it goes to a NAT Gateway and out to the

internet. (See section: Deployment models - Centralized egress).

Question: 2

A company has an on-premises Microsoft SOL Server database that writes a nightly 200 GB export to
a local drive. The company wants to move the backups to more robust cloud storage on Amazon S3.
The company has set up a 10 Gbps AWS Direct Connect connection between the on-premises data
center and AWS.
Which solution meets these requirements MOST cost-effectively?

A. Create a new S3 bucket. Deploy an AWS Storage Gateway file gateway within the VPC that Is connected to the Direct Connect connection. Create a new SMB file share. Write nightly database exports to the new SMB file share.

B. Create an Amazon FSx for Windows File Server Single-AZ file system within the VPC that is connected to the Direct Connect connection. Create a new SMB file share. Write nightly database exports to an SMB file share on the Amazon FSx file system. Enable nightly backups.

C. Create an Amazon FSx for Windows File Server Multi-AZ file system within the VPC that is connected to the Direct Connect connection. Create a new SMB file share. Write nightly database exports to an SMB file share on the Amazon FSx file system. Enable nightly backups.

D. Create a new S3 bucket. Deploy an AWS Storage Gateway volume gateway within the VPC that Is
connected to the Direct Connect connection. Create a new SMB file share. Write nightly database exports to the new SMB file share on the volume gateway, and automate copies of this data to an S3
bucket.

Answer:

A

Explanation:

The most cost-effective solution is to use an AWS Storage Gateway file gateway. The file gateway is deployed on-premises and presents a Server Message Block (SMB) file share. The existing on-premises SQL Server can write its nightly export directly to this share with no changes to the backup process. The gateway then efficiently and asynchronously transfers the backup file as an object to the designated Amazon S3 bucket over the Direct Connect connection. This approach leverages the low cost and high durability of Amazon S3 for backup storage while providing a seamless local integration point.

Why Incorrect Options are Wrong:

B. Amazon FSx for Windows File Server is a fully managed file system, which is more complex and significantly more expensive than using S3 for simple backup storage.

C. A Multi-AZ deployment of Amazon FSx for Windows File Server is even more expensive than a Single-AZ deployment and provides high availability that is unnecessary for a nightly backup target.

D. A volume gateway provides block-level storage via iSCSI, not file-level access via SMB. The description is technically incorrect; a file gateway is the service that provides an SMB interface.

References:

1. AWS Storage Gateway User Guide, "What is Amazon S3 File Gateway?": "Amazon S3 File Gateway presents a file interface that enables you to store files as objects in Amazon S3 using the industry-standard NFS and SMB file protocols... The gateway provides a virtual on-premises file share..." This confirms that a file gateway provides the required SMB share for the on-premises server.
2. AWS Storage Gateway User Guide, "Choosing a gateway solution": "For on-premises file-based applications that need a file protocol (NFS or SMB) interface to Amazon S3, we recommend S3 File Gateway." This directly validates the choice of a file gateway for this use case.
3. Amazon FSx for Windows File Server, "Pricing": The pricing for FSx includes costs for storage capacity, throughput capacity, and backups. This multi-faceted pricing model is more expensive for storing large, infrequently accessed backup files compared to the simple per-GB storage pricing of Amazon S3.
4. AWS Storage Gateway User Guide, "What is a volume gateway?": "A volume gateway provides cloud-backed storage volumes that you can mount as Internet Small Computer System Interface (iSCSI) devices from your on-premises application servers." This confirms that volume gateways provide block storage, not the required SMB file share.

Question: 3

A company is running an application on Amazon EC2 instances in the AWS Cloud. The application is
using a MongoDB database with a replica set as its data tier. The MongoDB database is installed on
systems in the company's on-premises data center and is accessible through an AWS Direct Connect
connection to the data center environment.
A solutions architect must migrate the on-premises MongoDB database to Amazon DocumentDB (with MongoDB compatibility).
Which strategy should the solutions architect choose to perform this migration?

A. Create a fleet of EC2 instances. Install MongoDB Community Edition on the EC2 instances, and
create a database. Configure continuous synchronous replication with the database that is running in
the on-premises data center.

B. Create an AWS Database Migration Service (AWS DMS) replication instance. Create a source endpoint for the on-premises MongoDB database by using change data capture (CDC). Create a target endpoint for the Amazon DocumentDB database. Create and run a DMS migration task.

C. Create a data migration pipeline by using AWS Data Pipeline. Define data nodes for the on-premises MongoDB database and the Amazon DocumentDB database. Create a scheduled task to run
the data pipeline.

D. Create a source endpoint for the on-premises MongoDB database by using AWS Glue crawlers.
Configure continuous asynchronous replication between the MongoDB database and the Amazon DocumentDB database.

Answer:

B

Explanation:

AWS Database Migration Service (DMS) is the designated AWS service for migrating databases to the AWS Cloud. It supports heterogeneous migrations, such as from an on-premises MongoDB database to Amazon DocumentDB. The process involves creating a DMS replication instance to perform the work, defining a source endpoint for the on-premises MongoDB, and a target endpoint for the Amazon DocumentDB cluster. By enabling Change Data Capture (CDC) on the source, DMS can perform a full load of existing data and then continuously replicate changes

from the source to the target, facilitating a migration with minimal downtime. This approach is the standard and most efficient method recommended by AWS for this scenario.

Why Incorrect Options are Wrong:

A. This option migrates the database to MongoDB running on EC2 instances, not to the required target of Amazon DocumentDB, which is a managed service.
C. AWS Data Pipeline is a data workflow orchestration service for ETL tasks, not a dedicated database migration tool. DMS is purpose-built for live database replication and migration.
D. AWS Glue crawlers are used to populate the AWS Glue Data Catalog for ETL purposes. They do not create migration endpoints or handle database replication.

References:

1. AWS Database Migration Service User Guide. (2023). Migrating to Amazon DocumentDB (with MongoDB compatibility). AWS Documentation. Retrieved from https://docs.aws.amazon.com/dms/latest/userguide/CHAPTarget.DocumentDB.html. This document outlines the use of AWS DMS as the primary tool for migrating to Amazon DocumentDB.
2. AWS Database Migration Service User Guide. (2023). Sources for data migration in AWS DMS. AWS Documentation. Retrieved from https://docs.aws.amazon.com/dms/latest/userguide/CHAPSource.html. Under the "MongoDB" section, it confirms that MongoDB is a supported source for DMS.
3. AWS Database Migration Service User Guide. (2023). Using a MongoDB database as a source for AWS DMS. AWS Documentation. Retrieved from https://docs.aws.amazon.com/dms/latest/userguide/CHAPSource.MongoDB.html. This page details the prerequisites and configuration for using MongoDB as a source, including the use of Change Data Capture (CDC) for ongoing replication.
4. AWS Prescriptive Guidance. (2023). Migrate from MongoDB to Amazon DocumentDB using the online method. AWS Documentation. Retrieved from https://docs.aws.amazon.com/prescriptive-guidance/latest/patterns/migrate-from-mongodb-to-amazon-documentdb-using-the-online-method.html. This pattern explicitly details the architecture and steps using AWS DMS for a minimal-downtime migration from MongoDB to DocumentDB.

Question: 4

A large company runs workloads in VPCs that are deployed across hundreds of AWS accounts. Each
VPC consists to public subnets and private subnets that span across multiple Availability Zones. NAT
gateways are deployed in the public subnets and allow outbound connectivity to the internet from the private subnets.
A solutions architect is working on a hub-and-spoke design. All private subnets in the spoke VPCs
must route traffic to the internet through an egress VPC. The solutions architect already has deployed
a NAT gateway in an egress VPC in a central AWS account.
Which set of additional steps should the solutions architect take to meet these requirements?

A. Create peering connections between the egress VPC and the spoke VPCs. Configure the required
routing to allow access to the internet.

B. Create a transit gateway, and share it with the existing AWS accounts. Attach existing VPCs to the
transit gateway Configure the required routing to allow access to the internet.

C. Create a transit gateway in every account. Attach the NAT gateway to the transit gateways. Configure the required routing to allow access to the internet.

D. Create an AWS PrivateLink connection between the egress VPC and the spoke VPCs. Configure the
required routing to allow access to the internet

Answer:

B

Explanation:

AWS Transit Gateway is designed to function as a cloud router, simplifying network connectivity at scale. For a hub-and-spoke architecture spanning hundreds of accounts, a single Transit Gateway in a central account can be shared with all other accounts using AWS Resource Access Manager (RAM). Spoke VPCs are then attached to the shared Transit Gateway. By configuring route tables in the spoke VPCs to point the default route (0.0.0.0/0) to the Transit Gateway, and the Transit Gateway's route table to forward that traffic to the egress VPC attachment, all outbound internet traffic is centralized. This is the standard, most scalable, and manageable AWS pattern for this requirement.

Why Incorrect Options are Wrong:

A. VPC peering is not transitive and does not scale effectively for hundreds of VPCs. Managing individual peering connections and routes would be operationally complex and inefficient.
C. Creating a Transit Gateway in every account is the opposite of a centralized model. It would increase complexity and cost without achieving the goal of a single egress point.
D. AWS PrivateLink is used to privately expose specific services to other VPCs. It is not a solution for routing general internet-bound traffic from an entire subnet or VPC.

References:

1. AWS Documentation, AWS Transit Gateway, "Centralized outbound routing to the internet": This document explicitly describes the required architecture. It states, "You can use an AWS Transit Gateway to route outbound traffic from multiple VPCs through a single egress VPC... The transit gateway is in its own account (the network services account) and is shared with the other accounts in the organization." This directly supports option B.
2. AWS Documentation, AWS Transit Gateway, "Example: Centralized router": This section details how a Transit Gateway acts as a central hub. "You can attach all your hybrid and VPC connections to a single Transit Gateway, which acts as a central router... This simplifies management and reduces operational costs."
3. AWS Documentation, Amazon VPC, "VPC peering basics and limitations": This document clarifies a key limitation of VPC peering: "A VPC peering connection is a one to one relationship between two VPCs... Transitive peering relationships are not supported." This confirms why option A is incorrect for routing traffic through a hub VPC to the internet.
4. AWS Documentation, AWS PrivateLink, "AWS PrivateLink concepts": This document explains that PrivateLink provides private connectivity between VPCs, AWS services, and on-premises networks. Its function is to create service endpoints, not to act as a general-purpose router for internet traffic, which makes option D unsuitable.

Question: 5

A company needs to migrate its customer transactions database from on premises to AWS. The database resides on an Oracle DB instance that runs on a Linux server. According to a new security
requirement, the company must rotate the database password each year.
Which solution will meet these requirements with the LEAST operational overhead?

A. Convert the database to Amazon DynamoDB by using the AWS Schema Conversion Tool (AWS
SCT). Store the password in AWS Systems Manager Parameter Store. Create an Amazon CloudWatch
alarm to invoke an AWS Lambda function for yearly password rotation.

B. Migrate the database to Amazon RDS for Oracle. Store the password in AWS Secrets Manager. Turn
on automatic rotation. Configure a yearly rotation schedule.

C. Migrate the database to an Amazon EC2 instance. Use AWS Systems Manager Parameter Store to
keep and rotate the connection string by using an AWS Lambda function on a yearly schedule

D. Migrate the database to Amazon Neptune by using the AWS Schema Conversion Tool AWS SCT).
Create an Amazon CloudWatch alarm to invoke an AWS Lambda function for yearly password rotation.

Answer:

B

Explanation:

This solution provides the path of least operational overhead for both the migration and the security requirement. Migrating an on-premises Oracle database to Amazon RDS for Oracle is a standard re-platforming strategy that offloads database administration tasks (patching, backups, scaling) to AWS. AWS Secrets Manager is designed specifically for this use case; it can securely store the database credentials and has a built-in, managed feature for automatic password rotation for supported services like Amazon RDS. Configuring a yearly rotation schedule is a simple, declarative step within the Secrets Manager console or API, requiring no custom code or infrastructure management.

Why Incorrect Options are Wrong:

A. Converting a relational Oracle database to NoSQL (DynamoDB) is a complex re-architecture, not a low-overhead migration, and is unsuitable for a typical transactional database without significant application changes.

C. Running the database on an EC2 instance retains the high operational overhead of managing the operating system, patching, and database software, which contradicts the goal of minimizing overhead.

D. Converting a relational database to a graph database (Neptune) is an even more drastic and inappropriate re-architecture for a standard customer transactions database, representing maximum operational overhead.

References:

1. AWS Secrets Manager User Guide: "Secrets Manager can rotate secrets for supported AWS services... For example, for a database, you can require your applications to retrieve the credentials from Secrets Manager each time they connect to the database. When Secrets Manager rotates the credentials, it updates the database with the new credentials and also updates the secret in Secrets Manager." (See section: Rotate AWS Secrets Manager secrets).
2. Amazon RDS User Guide: "Amazon RDS is a managed service that makes it easier to set up, operate, and scale a relational database in the AWS Cloud. It automates time-consuming administration tasks such as hardware provisioning, database setup, patching, and backups." (See section: What is Amazon Relational Database Service (Amazon RDS)?). This highlights the reduced operational overhead compared to managing a database on EC2.
3. AWS Secrets Manager User Guide: "For an Amazon RDS DB instance... you can turn on automatic rotation for your secret. When you turn on automatic rotation, you choose a rotation schedule... Secrets Manager uses a Lambda function to rotate the secret." (See section: Rotation for Amazon RDS databases). This confirms the native, low-overhead capability.

Question: 6

A financial services company loaded millions of historical stock trades into an Amazon DynamoDB
table. The table uses on-demand capacity mode. Once each day at midnight, a few million new records are loaded into the table. Application read activity against the table happens in bursts throughout the day. and a limited set of keys are repeatedly looked up. The company needs to reduce costs associated with DynamoDB.
Which strategy should a solutions architect recommend to meet this requirement?

A. Deploy an Amazon ElastiCache cluster in front of the DynamoDB table.

B. Deploy DynamoDB Accelerator (DAX). Configure DynamoDB auto scaling. Purchase Savings Plans
in Cost Explorer

C. Use provisioned capacity mode. Purchase Savings Plans in Cost Explorer.

D. Deploy DynamoDB Accelerator (DAX). Use provisioned capacity mode. Configure DynamoDB auto
scaling.

Answer:

D

Explanation:

The described workload has two distinct patterns ideal for cost optimization. First, the "read activity... in bursts" and "limited set of keys are repeatedly looked up" is a classic use case for an in-memory cache. DynamoDB Accelerator (DAX) is a fully managed, highly available, in-memory cache for DynamoDB that can significantly reduce read costs and improve latency by serving frequent reads from the cache. Second, the daily bulk load is a predictable, high-volume write event. Switching from on-demand to provisioned capacity mode with auto scaling allows the table to handle this predictable spike cost-effectively, scaling up for the load and down afterward, which is typically cheaper than on-demand for such patterns.

Why Incorrect Options are Wrong:

A. While ElastiCache is a caching service, DAX is purpose-built for DynamoDB, offering a transparent implementation that requires minimal application code changes compared to a separate ElastiCache cluster.
B. This option is contradictory. DynamoDB auto scaling and Savings Plans are features applicable to provisioned capacity mode, not on-demand mode, which the table is currently using.
C. This approach only addresses the capacity mode but fails to optimize the read-intensive, hot-key access pattern. Without DAX, provisioned read capacity would need to be high to handle

bursts, diminishing cost savings.

References:

1. AWS Documentation, DynamoDB Developer Guide, "In-Memory Caching with DynamoDB Accelerator (DAX)": "DAX is intended for use with read-intensive applications... DAX can reduce response times for eventually consistent reads from milliseconds to microseconds... This reduces the need to overprovision read capacity units, which in turn reduces your DynamoDB read costs."
2. AWS Documentation, DynamoDB Developer Guide, "Read/write capacity mode": "Provisioned mode is a good option if... you can forecast your capacity requirements to control costs." and "DynamoDB auto scaling... actively manages throughput capacity for tables and global secondary indexes. With auto scaling, you define a range... DynamoDB auto scaling seeks to maintain your target utilization within that range." This combination is ideal for the predictable daily write spike.
3. AWS Documentation, DynamoDB Developer Guide, "When should I use DAX?": This section explicitly lists use cases such as applications requiring the fastest possible response time for reads and those with read-heavy or bursty workloads, which directly matches the scenario.
4. AWS Well-Architected Framework, Cost Optimization Pillar Whitepaper, "Expenditure and usage awareness - COST 2": The whitepaper emphasizes selecting the right pricing model. For DynamoDB, this involves choosing between on-demand and provisioned capacity. For workloads with some predictability, like the daily load described, provisioned capacity with auto scaling is the recommended cost-effective choice over on-demand.

Question: 7

A company wants to manage the costs associated with a group of 20 applications that are infrequently used, but are still business-critical, by migrating to AWS. The applications are a mix of
Java and Node.js spread across different instance clusters. The company wants to minimize costs
while standardizing by using a single deployment methodology.
Most of the applications are part of month-end processing routines with a small number of concurrent users, but they are occasionally run at other times Average application memory consumption is less than 1 GB. though some applications use as much as 2.5 GB of memory during
peak processing. The most important application in the group is a billing report written in Java that
accesses multiple data sources and often runs for several hours.
Which is the MOST cost-effective solution?

A. Deploy a separate AWS Lambda function tor each application. Use AWS CloudTrail logs and Amazon CloudWatch alarms to verify completion of critical jobs.

B. Deploy Amazon ECS containers on Amazon EC2 with Auto Scaling configured for memory utilization of 75%. Deploy an ECS task for each application being migrated with ECS task scaling. Monitor services and hosts by using Amazon CloudWatch.

C. Deploy AWS Elastic Beanstalk for each application with Auto Scaling to ensure that all requests
have sufficient resources. Monitor each AWS Elastic Beanstalk deployment by using CloudWatch alarms.

D. Deploy a new Amazon EC2 instance cluster that co-hosts all applications by using EC2 Auto Scaling
and Application Load Balancers. Scale cluster size based on a custom metric set on instance memory
utilization. Purchase 3-year Reserved Instance reservations equal to the GroupMaxSize parameter of
the Auto Scaling group.

Answer:

B

Explanation:

This solution is the most cost-effective because it leverages containerization with Amazon ECS on a shared cluster of Amazon EC2 instances. This "bin packing" approach allows multiple, infrequently used applications to share the resources of a few EC2 instances, drastically reducing costs compared to dedicating infrastructure to each application. The EC2 Auto Scaling group for the cluster can scale down to a minimal size (or even zero) when the applications are idle. Crucially, unlike AWS Lambda, ECS tasks can run for the several hours required by the critical billing report, while still providing a standardized deployment methodology for all Java and Node.js applications.

Why Incorrect Options are Wrong:

A. AWS Lambda has a maximum execution timeout of 15 minutes, which makes it unsuitable for the billing report application that runs for several hours.
C. Deploying a separate AWS Elastic Beanstalk environment for each of the 20 applications would create significant resource overhead (e.g., load balancers, EC2 instances) and would not be cost-effective.
D. Purchasing 3-year Reserved Instances for the maximum size of an Auto Scaling group is financially unsound for an infrequently used workload, as it means paying for peak capacity at all times.

References:

1. Amazon ECS for Long-Running Tasks and Cost-Effectiveness: The Amazon ECS documentation highlights its suitability for long-running processes and the cost benefits of the EC2 launch type. "With the EC2 launch type, you can run your containerized applications on a cluster of Amazon EC2 instances that you manage... Amazon ECS leverages EC2 Auto Scaling to manage the scaling of the EC2 instances." This supports bin-packing on a scalable, shared cluster.
Source: AWS Documentation, Amazon ECS Developer Guide, "What is Amazon ECS?", Section: "Amazon ECS on Amazon EC2 instances".
2. AWS Lambda Execution Time Limit: The official AWS Lambda documentation explicitly states the service's execution time limit, which disqualifies it for the scenario's long-running job. "Timeout - The amount of time that Lambda allows a function to run before stopping it. The default is 3 seconds. The maximum value is 900 seconds (15 minutes)."
Source: AWS Documentation, AWS Lambda Developer Guide, "Configuring Lambda function options", Section: "Basic settings".
3. AWS Elastic Beanstalk Environment Structure: The documentation shows that each environment is a self-contained unit with its own resources, making a per-application deployment costly. "An environment is a collection of AWS resources running an application version."
Source: AWS Documentation, AWS Elastic Beanstalk Developer Guide, "AWS Elastic Beanstalk

concepts", Section: "Environments".
4. Reserved Instance Use Case: The EC2 documentation clarifies that Reserved Instances are for predictable, steady-state workloads, not infrequent ones. "Amazon EC2 Reserved Instances (RI) provide a significant discount (up to 72%) compared to On-Demand pricing and provide a capacity reservation when used in a specific Availability Zone. RIs are recommended for applications with steady state or predictable usage."
Source: AWS Documentation, Amazon EC2 User Guide for Linux Instances, "Reserved Instances".

Question: 8

During an audit, a security team discovered that a development team was putting IAM user secret access keys in their code and then committing it to an AWS CodeCommit repository. The security team wants to automatically find and remediate instances of this security vulnerability.
Which solution will ensure that the credentials are appropriately secured automatically7

A. Run a script nightly using AWS Systems Manager Run Command to search tor credentials on the
development instances. If found. use AWS Secrets Manager to rotate the credentials.

B. Use a scheduled AWS Lambda function to download and scan the application code from CodeCommit. If credentials are found, generate new credentials and store them in AWS KMS.

C. Configure Amazon Made to scan for credentials in CodeCommit repositories. If credentials are found, trigger an AWS Lambda function to disable the credentials and notify the user.

D. Configure a CodeCommit trigger to invoke an AWS Lambda function to scan new code submissions
for credentials. It credentials are found, disable them in AWS IAM and notify the user

Answer:

D

Explanation:

This solution provides an automated, event-driven response that directly addresses the vulnerability at its source. By configuring an AWS CodeCommit trigger for push events, a scan is initiated immediately upon code submission. An AWS Lambda function is the appropriate serverless compute service to execute the scan. If credentials are found, the function can use IAM API calls to immediately disable the compromised access key by setting its status to Inactive. This is the most effective and immediate remediation, preventing the key's use while the user is notified to rotate it properly.

Why Incorrect Options are Wrong:

A. This approach is incorrect because it scans development instances, not the CodeCommit repository where the credentials were committed. It is also not a real-time solution, as it runs nightly.
B. This uses a scheduled function, which introduces a delay between the commit and the scan. The remediation of storing new credentials in AWS KMS does not address the compromised, active key.
C. This is incorrect because Amazon Macie is a data security service that discovers and protects sensitive data in Amazon S3, not in AWS CodeCommit repositories.

References:

1. AWS CodeCommit Developer Guide, "Create a trigger for a CodeCommit repository": This document explains how to create triggers that respond to repository events, such as a push to a branch. It explicitly states that a trigger can invoke an AWS Lambda function. (See section: "Tutorial: Create a trigger for a CodeCommit repository").
2. AWS IAM API Reference, "UpdateAccessKey": This API action is used to change the status of a specified access key. The documentation confirms that the Status parameter can be set to Inactive, which disables the key without deleting it. This is the correct remediation step.
3. AWS Lambda Developer Guide, "Using AWS Lambda with AWS CodeCommit": This guide details the event-driven pattern where CodeCommit triggers invoke a Lambda function. The Lambda function receives event data about the commit, enabling it to retrieve and scan the code.
4. Amazon Macie User Guide, "How Amazon Macie works": The official documentation states, "Amazon Macie is a data security service that discovers sensitive data by using machine learning and pattern matching, provides visibility into data security risks, and enables automated protection against those risks." It specifies its integration with Amazon S3 for data discovery. There is no mention of native support for AWS CodeCommit.

Question: 9

A company is rearchitecting its applications to run on AWS. The company's infrastructure includes
multiple Amazon EC2 instances. The company's development team needs different levels of access.
The company wants to implement a policy that requires all Windows EC2 instances to be joined to an
Active Directory domain on AWS. The company also wants to Implement enhanced security processes such as multi-factor authentication (MFA). The company wants to use managed AWS services wherever possible.
Which solution will meet these requirements?

A. Create an AWS Directory Service for Microsoft Active Directory implementation. Launch an Amazon Workspace. Connect to and use the Workspace for domain security configuration tasks.

B. Create an AWS Directory Service for Microsoft Active Directory implementation. Launch an EC2
instance. Connect to and use the EC2 instance for domain security configuration tasks.

C. Create an AWS Directory Service Simple AD implementation. Launch an EC2 instance. Connect to
and use the EC2 instance for domain security configuration tasks.

D. Create an AWS Directory Service Simple AD implementation. Launch an Amazon Workspace. Connect to and use the Workspace for domain security configuration tasks.

Answer:

A

Explanation:

AWS Directory Service for Microsoft Active Directory (AWS Managed Microsoft AD) provides a fully-managed, Windows-compatible directory that EC2 Windows instances can join automatically. It natively supports Group Policy and multi-factor authentication through RADIUS, satisfying the company's security and MFA requirements. Pairing this with Amazon WorkSpaces keeps the administrative jump-host fully managed as well, eliminating the need to operate an EC2 bastion while still allowing domain, GPO, and security-policy configuration work. Option A therefore uses only managed AWS services and meets every stated requirement.

Why Incorrect Options are Wrong:

B. Uses EC2 for admin tasks; EC2 is customer-managed, contradicting "use managed services wherever possible."
C. Simple AD lacks MFA support and advanced AD features; cannot enforce required security policies.
D. Same Simple AD limitations as C; MFA and advanced policies unsupported despite WorkSpaces.

References:

1. AWS Directory Service Admin Guide - "Enable multi-factor authentication for AWS Managed Microsoft AD" (Section: 'Enable MFA'; pp. 313-318, 2023-08-15).
2. AWS Directory Service Admin Guide - "Seamless domain join for Windows EC2 instances" (Section: 'Seamless Domain Join'; pp. 147-150).
3. AWS Directory Service Admin Guide - "Simple AD limitations" (Table: Feature support; pp. 34).
4. Amazon WorkSpaces Admin Guide - "Integrating with AWS Managed Microsoft AD and MFA via RADIUS" (Section: 'MFA Setup'; pp. 229-233).
5. AWS Security Best Practices Whitepaper - "Prefer managed services to reduce operational burden" (Section 2.1, 2023-04).

Question: 10

A company is running an application in the AWS Cloud. The application consists of microservices that
run on a fleet of Amazon EC2 instances in multiple Availability Zones behind an Application Load Balancer. The company recently added a new REST API that was implemented in Amazon API Gateway. Some of the older microservices that run on EC2 instances need to call this new API. The company does not want the API to be accessible from the public internet and does not want proprietary data to traverse the public internet
What should a solutions architect do to meet these requirements?

A. Create an AWS Site-to-Site VPN connection between the VPC and the API Gateway. Use API Gateway to generate a unique API key for each microservice. Configure the API methods to require
the key.

B. Create an interface VPC endpoint for API Gateway, and set an endpoint policy to only allow access
to the specific API Add a resource policy to API Gateway to only allow access from the VPC endpoint.
Change the API Gateway endpoint type to private.

C. Modify the API Gateway to use 1AM authentication. Update the 1AM policy for the 1AM role that
is assigned to the EC2 Instances to allow access to the API Gateway. Move the API Gateway into a
new VPC Deploy a transit gateway and connect the VPCs.

D. Create an accelerator in AWS Global Accelerator, and connect the accelerator to the API Gateway.
Update the route table for all VPC subnets with a route to the created Global Accelerator endpoint IP
address. Add an API key for each service to use for authentication.

Answer:

B

Explanation:

This solution correctly establishes secure, private connectivity. Creating a private API Gateway endpoint makes the API accessible only from within a VPC, not from the public internet. An interface VPC endpoint for API Gateway (execute-api) creates a private network path from the EC2 instances' VPC to the API Gateway service, ensuring traffic does not traverse the public internet. Attaching a resource policy to the API that explicitly allows access only from the

specified VPC endpoint provides a robust security layer, fulfilling all the company's requirements for private and secure communication.

Why Incorrect Options are Wrong:

A. An AWS Site-to-Site VPN connects a VPC to an on-premises network, not to a managed AWS service like API Gateway. This does not solve the private connectivity requirement.
C. IAM authentication provides authorization but does not make the API endpoint private. API Gateway cannot be "moved" into a VPC, and a transit gateway is for inter-VPC networking, not service access.
D. AWS Global Accelerator is designed to improve performance for public-facing applications by routing traffic over the AWS global network. It is for public, not private, endpoints.

References:

1. AWS API Gateway Developer Guide, "Creating a private API in Amazon API Gateway": This guide states, "Private APIs are API endpoints that can only be accessed from your Amazon Virtual Private Cloud (VPC) through an interface VPC endpoint. This is an endpoint network interface that you create in your VPC." This directly supports using a private API endpoint type and a VPC endpoint.
2. AWS API Gateway Developer Guide, "Control access to a private API": This section details the use of resource policies. It explains, "To grant the access, you create a resource policy... and attach it to the API." It provides examples of policies that grant access based on the source VPC or VPC endpoint (aws:sourceVpce). This supports the resource policy and endpoint policy components of the answer.
3. AWS PrivateLink Guide, "Interface VPC endpoints (AWS PrivateLink)": This document explains the core technology. It states, "When you use an interface VPC endpoint, communication between your VPC and the AWS service is conducted entirely within the AWS network." This confirms that data does not traverse the public internet.

Question: 11

A company has more than 10.000 sensors that send data to an on-premises Apache Kafka server by
using the Message Queuing Telemetry Transport (MQTT) protocol. The on-premises Kafka server transforms the data and then stores the results as objects in an Amazon S3 bucket.
Recently, the Kafka server crashed. The company lost sensor data while the server was being restored. A solutions architect must create a new design on AWS that is highly available and scalable
to prevent a similar occurrence.
Which solution will meet these requirements?

A. Launch two Amazon EC2 instances to host the Kafka server in an active/standby configuration across two Availability Zones. Create a domain name in Amazon Route 53. Create a Route 53 failover
policy. Route the sensors to send the data to the domain name.

B. Migrate the on-premises Kafka server to Amazon Managed Streaming for Apache Kafka (Amazon
MSK). Create a Network Load Balancer (NLB) that points to the Amazon MSK broker Enable NL8 health checks. Route the sensors to send the data to the NLB.

C. Deploy AWS IoT Core, and connect it to an Amazon Kinesis Data Firehose delivery stream. Use an
AWS Lambda function to handle data transformation. Route the sensors to send the data to AWS IoT
Core.

D. Deploy AWS IoT Core, and launch an Amazon EC2 instance to host the Kafka server. Configure AWS
IoT Core to send the data to the EC2 instance. Route the sensors to send the data to AWS IoT Core.

Answer:

C

Explanation:

This solution provides a fully managed, serverless, and highly available architecture. AWS IoT Core is designed for scalable IoT device connectivity and natively supports the required MQTT protocol. The IoT Rules Engine can route incoming data to an Amazon Kinesis Data Firehose delivery stream. Firehose reliably streams the data and can invoke an AWS Lambda function for the required data transformation before durably storing the results in Amazon S3. This design eliminates single points of failure and the operational overhead of managing servers, directly

addressing the requirements for high availability and scalability.

Why Incorrect Options are Wrong:

A. This is a self-managed solution on EC2. It introduces operational overhead and is less scalable and resilient than a fully managed, serverless architecture.
B. Amazon MSK does not natively support the MQTT protocol used by the sensors. An additional MQTT broker would be required, making this solution incomplete.
D. This re-introduces a self-managed Kafka server on EC2, which was the original point of failure, failing to meet the high availability and scalability requirements.

References:

1. AWS IoT Core Documentation: "AWS IoT Core supports devices and clients that use the MQTT and the MQTT over WSS protocols to publish and subscribe to messages... AWS IoT Core provides a secure, highly available, and scalable message broker".
Source: AWS IoT Developer Guide, "Protocols, port mappings, and endpoints".
2. AWS IoT Core Rules Engine: "The Rules Engine evaluates inbound messages published into AWS IoT Core and transforms and delivers them to another device or a cloud service, based on business rules you define... A rule can apply to data from one or many devices, can take one or many actions in parallel." Actions include sending data to Kinesis Data Firehose.

Source: AWS IoT Developer Guide, "Rules for AWS IoT".
3. Amazon Kinesis Data Firehose Documentation: "Amazon Kinesis Data Firehose is a fully managed service for delivering real-time streaming data to destinations such as Amazon Simple Storage Service (Amazon S3)... With Kinesis Data Firehose, you can also transform your data before delivering it. You can configure Kinesis Data Firehose to invoke your AWS Lambda function to transform incoming source data and deliver the transformed data to destinations."
Source: Amazon Kinesis Data Firehose Developer Guide, "What Is Amazon Kinesis Data Firehose?".
4. AWS Whitepaper - AWS IoT Lens, AWS Well-Architected Framework: This paper emphasizes using managed services to improve reliability and reduce operational overhead. "Use managed services for undifferentiated parts of the architecture. For example, AWS IoT Core provides a managed message broker... This reduces the operational burden of managing infrastructure for the workload."
Source: AWS Well-Architected Framework, AWS IoT Lens, REL 1: "How do you manage your workload's connectivity?", Page 10.

Question: 12

A company is developing a gene reporting device that will collect genomic information to assist researchers with collecting large samples of data from a diverse population. The device will push 8
KB of genomic data every second to a data platform that will need to process and analyze the data
and provide information back to researchers. The data platform must meet the following requirements:
•Provide near-real-time analytics of the inbound genomic data
•Ensure the data is flexible, parallel, and durable
•Deliver results of processing to a data warehouse
Which strategy should a solutions architect use to meet these requirements?

A. Use Amazon Kinesis Data Firehose to collect the inbound sensor data, analyze the data with Kinesis clients, and save the results to an Amazon RDS instance.

B. Use Amazon Kinesis Data Streams to collect the inbound sensor data, analyze the data with Kinesis
clients, and save the results to an Amazon Redshift cluster using Amazon EMR.

C. Use Amazon S3 to collect the inbound device data, analyze the data from Amazon SOS with Kinesis, and save the results to an Amazon Redshift cluster.

D. Use an Amazon API Gateway to put requests into an Amazon SQS queue, analyze the data with an
AWS Lambda function, and save the results to an Amazon Redshift cluster using Amazon EMR.

Answer:

B

Explanation:

This scenario requires a solution for ingesting and processing a continuous stream of data in near-real-time, with the results being loaded into a data warehouse.
Amazon Kinesis Data Streams is the ideal service for capturing and processing large streams of data records in real time. It provides durability and elasticity, and its sharded architecture allows for parallel processing. Consumer applications, often referred to as Kinesis clients (e.g., applications using the Kinesis Client Library, AWS Lambda, or Amazon Kinesis Data Analytics), can read from the stream to perform near-real-time analytics. The processed data can then be loaded into Amazon Redshift, a petabyte-scale data warehouse service. Using Amazon EMR is a valid pattern for performing complex transformations on the streaming data before loading it into Redshift.

Why Incorrect Options are Wrong:

A. Amazon RDS is an Online Transaction Processing (OLTP) database, not a data warehouse (OLAP) as required by the question. Kinesis Data Firehose is less flexible for custom real-time analytics than Kinesis Data Streams.

C. Using Amazon S3 for direct, high-velocity data ingestion does not meet the near-real-time analytics requirement. The description "analyze the data from Amazon SQS with Kinesis" is technically incoherent.

D. While API Gateway and SQS can handle data ingestion, Amazon SQS is a message queue, not a persistent, replayable stream. Kinesis Data Streams is better suited for continuous real-time data stream processing.

References:

1. Amazon Kinesis Data Streams Developer Guide: "You can use Kinesis Data Streams to collect and process large streams of data records in real time... The processed records can then be sent to dashboards, used to generate alerts, dynamically change pricing and advertising strategies, or send data to a variety of other AWS services." This aligns with the requirement for real-time collection and analysis.
Source: AWS Documentation, What Is Amazon Kinesis Data Streams?, Introduction.
2. Amazon Kinesis Data Streams Developer Guide: "The consumers of a stream, known as Amazon Kinesis Data Streams applications, can be built using the Kinesis Client Library (KCL)..." This supports the "analyze the data with Kinesis clients" part of the correct answer.
Source: AWS Documentation, Reading Data from Amazon Kinesis Data Streams, Reading from a Stream.
3. AWS Whitepaper - Streaming Data Solutions on AWS with Amazon Kinesis: "Amazon Kinesis Streams enables you to build custom applications that process or analyze streaming data for specialized needs... For example, you can send the output of your Amazon Kinesis application to Amazon S3, Amazon DynamoDB, Amazon EMR, and Amazon Redshift." This confirms the architectural pattern of Kinesis Streams feeding a data warehouse like Redshift, potentially via EMR.
Source: AWS Whitepaper, Streaming Data Solutions on AWS with Amazon Kinesis, Page 5, "Amazon Kinesis Streams".
4. Amazon Redshift Documentation: "Amazon Redshift is a fully managed, petabyte-scale data warehouse service in the cloud." This confirms Redshift as the appropriate data warehouse destination.
Source: AWS Documentation, Amazon Redshift Cluster Management Guide, "What is Amazon Redshift?".

Question: 13

A media storage application uploads user photos to Amazon S3 for processing by AWS Lambda functions. Application state is stored in Amazon DynamoOB tables. Users are reporting that some uploaded photos are not being processed properly. The application developers trace the logs and find that Lambda is experiencing photo processing issues when thousands of users upload photos
simultaneously. The issues are the result of Lambda concurrency limits and the performance of DynamoDB when data is saved.
Which combination of actions should a solutions architect take to increase the performance and reliability of the application? (Select TWO.)

A. Evaluate and adjust the RCUs for the DynamoDB tables.

B. Evaluate and adjust the WCUs for the DynamoDB tables.

C. Add an Amazon ElastiCache layer to increase the performance of Lambda functions.

D. Add an Amazon Simple Queue Service (Amazon SQS) queue and reprocessing logic between Amazon S3 and the Lambda functions.

E. Use S3 Transfer Acceleration to provide lower latency to users.

Answer:

B, D

Explanation:

The scenario describes two specific problems during peak load: hitting AWS Lambda concurrency limits and poor Amazon DynamoDB performance when saving data.
1. Adding an Amazon SQS queue (D) between Amazon S3 and the Lambda functions decouples the components. Instead of S3 invoking thousands of Lambda functions directly and simultaneously, it sends messages to an SQS queue. Lambda can then poll this queue at a controlled rate, smoothing out the processing spikes and preventing the account from hitting its concurrency limits. This also enhances reliability, as SQS can retain messages for reprocessing if a function fails.
2. Adjusting the WCUs for the DynamoDB tables (B) directly addresses the performance issue "when data is saved." Saving data is a write operation, managed by Write Capacity Units (WCUs). Insufficient WCUs during a spike in traffic will lead to write throttling and performance degradation. Evaluating and increasing the provisioned WCUs (or switching to on-demand capacity mode) will ensure the database can handle the write load.

Why Incorrect Options are Wrong:

A. Evaluate and adjust the RCUs for the DynamoDB tables.
RCUs (Read Capacity Units) are for read performance. The problem explicitly states the issue occurs when data is saved (a write operation).
C. Add an Amazon ElastiCache layer to increase the performance of Lambda functions.
ElastiCache is a caching service used to speed up data retrieval (reads). It does not solve issues with Lambda invocation concurrency or database write performance.
E. Use S3 Transfer Acceleration to provide lower latency to users.
S3 Transfer Acceleration speeds up the initial file upload to S3 but has no impact on the backend processing pipeline (Lambda and DynamoDB) where the bottlenecks occur.

References:

1. Using Lambda with Amazon SQS: The AWS Lambda Developer Guide explains how SQS can be used to buffer events and control concurrency. "For standard queues, Lambda uses long polling to poll a queue until it becomes active..... Lambda reads messages in batches and invokes your function once for each batch. ....your function can scale up to process more batches in parallel." This directly addresses the concurrency issue.
Source: AWS Lambda Developer Guide, "Using Lambda with Amazon SQS", Section: "Event source mapping".
2. DynamoDB Write Capacity Units (WCUs): The Amazon DynamoDB Developer Guide details how write throughput is managed. "For writes, a write capacity unit (WCU) represents one write per second for an item up to 1 KB in size. If you need to write an item that is larger than 1 KB, DynamoDB must consume additional write capacity units." Insufficient WCUs cause ProvisionedThroughputExceededException errors.
Source: Amazon DynamoDB Developer Guide, "Read/write capacity mode", Section: "Provisioned mode".
3. Decoupling with Queues Pattern: The AWS Well-Architected Framework discusses using queues to create resilient and scalable systems. "Decoupling components with a message queue can help you build more resilient systems.....A queue can also act as a buffer, smoothing out temporary spikes in traffic without losing requests." This pattern is precisely what option D implements.
Source: AWS Well-Architected Framework, Reliability Pillar, "REL 12: How do you design your workload to withstand component failures?", Section: "Decouple components".

Question: 14

A company runs many workloads on AWS and uses AWS Organizations to manage its accounts. The
workloads are hosted on Amazon EC2. AWS Fargate. and AWS Lambd
a. Some of the workloads have unpredictable demand. Accounts record high usage in some months
and low usage in other months.
The company wants to optimize its compute costs over the next 3 years A solutions architect obtains
a 6-month average for each of the accounts across the organization to calculate usage.
Which solution will provide the MOST cost savings for all the organization's compute usage?

A. Purchase Reserved Instances for the organization to match the size and number of the most common EC2 instances from the member accounts.

B. Purchase a Compute Savings Plan for the organization from the management account by using the
recommendation at the management account level

C. Purchase Reserved Instances for each member account that had high EC2 usage according to the
data from the last 6 months.

D. Purchase an EC2 Instance Savings Plan for each member account from the management account
based on EC2 usage data from the last 6 months.

Answer:

B

Explanation:

Compute Savings Plans provide the most flexibility and the highest potential savings in this scenario. They automatically apply to usage across Amazon EC2, AWS Fargate, and AWS Lambda, regardless of instance family, size, OS, tenancy, or AWS Region. This directly addresses the requirement to optimize costs for all the company's specified compute services. Purchasing the plan from the management account allows the commitment to be shared across all member accounts in the AWS Organization. This aggregates the unpredictable usage from various accounts, ensuring the commitment is fully utilized and maximizing overall savings, which is superior to instance-specific or service-specific commitments.

Why Incorrect Options are Wrong:

A. Reserved Instances (RIs) only apply to Amazon EC2 usage. They do not provide any savings for AWS Fargate or AWS Lambda, failing to cover all the organization's compute usage.

C. This is inefficient. Purchasing RIs at the member account level and only for EC2 ignores Fargate/Lambda costs and fails to aggregate usage across the entire organization for maximum benefit.

D. An EC2 Instance Savings Plan, as the name implies, is limited to EC2 instances. It does not apply to AWS Fargate or AWS Lambda, thus not providing a comprehensive cost-saving solution.

References:

1. AWS Savings Plans User Guide: "Compute Savings Plans provide the most flexibility and help to reduce your costs by up to 66%. These plans automatically apply to EC2 instance usage regardless of instance family, size, OS, tenancy or AWS Region. These also apply to Fargate and Lambda usage." This confirms that Compute Savings Plans cover all three services mentioned in the question.

Source: AWS Savings Plans User Guide, "Savings Plans overview" section.

2. AWS Savings Plans User Guide: "When you purchase Savings Plans, the prices for On-Demand Instances are discounted... If you have multiple accounts in AWS Organizations, you can use the cost and usage sharing feature. Savings Plans can be applied to usage across all accounts." This supports purchasing from the management account to benefit the entire organization.

Source: AWS Savings Plans User Guide, "How Savings Plans work" section.

3. AWS Cost Management Documentation: "We recommend that you always purchase Savings Plans in the management account. This provides sharing and flexibility benefits across all accounts in your consolidated billing family." This establishes the best practice of centralized purchasing for maximum benefit.

Source: AWS Cost Management User Guide, "Managing your Savings Plans", "Best practices" subsection.

4. AWS Fargate Pricing Documentation: "AWS Fargate also has an optional compute savings pricing model, Compute Savings Plans, that can save you money in exchange for a commitment to a consistent amount of usage..." This explicitly confirms Savings Plans apply to Fargate.

Source: AWS Fargate Pricing page, "Savings Plans" section.

Question: 15

A company wants to migrate its on-premises application to AWS. The database for the application
stores structured product data and temporary user session dat
a. The company needs to decouple the product data from the user session data. The company also
needs to implement replication in another AWS Region for disaster recovery.
Which solution will meet these requirements with the HIGHEST performance?

A. Create an Amazon RDS DB instance with separate schemas to host the product data and the user
session data. Configure a read replica for the DB instance in another Region.

B. Create an Amazon RDS DB instance to host the product data. Configure a read replica for the DB
instance in another Region. Create a global datastore in Amazon ElastiCache for Memcached to host
the user session data.

C. Create two Amazon DynamoDB global tables. Use one global table to host the product data Use
the other global table to host the user session data. Use DynamoDB Accelerator (DAX) for caching.

D. Create an Amazon RDS DB instance to host the product data. Configure a read replica for the DB
instance in another Region. Create an Amazon DynamoDB global table to host the user session data

Answer:

D

Explanation:

This solution provides the highest performance by using the most appropriate, purpose-built AWS service for each distinct data type. Amazon RDS is the optimal choice for structured, relational product data, and configuring a cross-Region read replica directly satisfies the disaster recovery requirement for this critical data.
For temporary user session data, which is typically a high-throughput key-value workload, Amazon DynamoDB is an ideal fit, offering single-digit millisecond latency. Using a DynamoDB global table provides a fully managed, multi-Region, active-active database, which not only delivers low-latency access but also fulfills the disaster recovery requirement for the session data seamlessly. This architecture correctly decouples the workloads, preventing resource contention

and maximizing performance for the entire application.

Why Incorrect Options are Wrong:

A. Storing both data types in a single RDS instance fails to properly decouple the workloads, leading to resource contention that would degrade performance.
B. This option is technically incorrect. Amazon ElastiCache for Memcached does not support native replication or a feature called "global datastore," thereby failing to meet the disaster recovery requirement.
C. While DynamoDB is a high-performance database, Amazon RDS is generally a more suitable and performant choice for structured product data that often involves complex relational queries and joins.

References:

1. Amazon RDS Read Replicas for DR: AWS Documentation, "Working with read replicas," states, "You can use a read replica as a standby instance for your source DB instance. If the source DB instance fails, you can promote the read replica... For disaster recovery, you can promote a cross-Region read replica to be the new primary DB instance." (Source: AWS RDS User Guide, "Working with read replicas" section).
2. Amazon DynamoDB for Session State & DR: AWS Documentation, "Amazon DynamoDB for web session management," highlights its suitability: "Amazon DynamoDB offers single-digit millisecond latency, making it a great fit for storing session data... With global tables, you can give users in different regions fast, local access to their session data." (Source: AWS Solutions Library, "Web session management with Amazon DynamoDB").

3. ElastiCache for Memcached Replication: AWS Documentation, "Replication: Redis (cluster mode disabled) vs. Memcached," explicitly states, "Memcached doesn't support replication." This confirms that option B cannot fulfill the disaster recovery requirement as written. (Source: AWS ElastiCache User Guide, "Comparing Memcached and Redis" section).
4. Choosing the Right Database: AWS Documentation, "Databases on AWS," guides users to select services based on data models. It positions Amazon RDS for relational workloads ("structured product data") and Amazon DynamoDB for key-value workloads ("user session data"), supporting the architectural choice in option D. (Source: aws.amazon.com/products/databases/).

Question: 16

A company uses AWS Organizations to manage its AWS accounts. The company needs a list of all its
Amazon EC2 instances that have underutilized CPU or memory usage. The company also needs recommendations for how to downsize these underutilized instances.
Which solution will meet these requirements with the LEAST effort?

A. Install a CPU and memory monitoring tool from AWS Marketplace on all the EC2 Instances. Store
the findings in Amazon S3. Implement a Python script to identify underutilized instances. Reference
EC2 instance pricing information for recommendations about downsizing options.

B. Install the Amazon CloudWatch agent on all the EC2 instances by using AWS Systems Manager.
Retrieve the resource op! nization recommendations from AWS Cost Explorer in the organization's
management account. Use the recommendations to downsize underutilized instances in all accounts
of the organization.

C. Install the Amazon CloudWatch agent on all the EC2 instances by using AWS Systems Manager.
Retrieve the resource optimization recommendations from AWS Cost Explorer in each account of the
organization. Use the recommendations to downsize underutilized instances in all accounts of the organization.

D. Install the Amazon CloudWatch agent on all the EC2 instances by using AWS Systems Manager
Create an AWS Lambda function to extract CPU and memory usage from all the EC2 instances. Store
the findings as files in Amazon S3. Use Amazon Athena to find underutilized instances. Reference EC2
instance pricing information for recommendations about downsizing options.

Answer:

B

Explanation:

The most efficient solution with the least effort is to leverage AWS-native, managed services designed for this purpose. AWS Cost Explorer provides resource optimization recommendations, including rightsizing for underutilized EC2 instances. To include memory utilization in its analysis, the Amazon CloudWatch agent must be installed on the instances. AWS Systems Manager is the most efficient way to deploy the agent at scale across all accounts in an AWS Organization. By accessing Cost Explorer from the management account, the company can get a consolidated view and recommendations for all member accounts, fulfilling all requirements with minimal custom development.

Why Incorrect Options are Wrong:

A. This is a high-effort, custom solution requiring the installation of third-party tools, custom scripting, and manual correlation with pricing data, which contradicts the "least effort" requirement.

C. This approach is inefficient. While it uses the correct tools, retrieving recommendations from each account individually requires more effort than using the centralized view available in the management account.

D. This is another high-effort, custom solution. It requires developing a Lambda function, setting up a data pipeline with S3 and Athena, and building custom queries, which is significantly more work than using the built-in AWS Cost Explorer feature.

References:

1. AWS Cost Explorer Rightsizing Recommendations: "AWS Cost Explorer provides you with EC2 instance rightsizing recommendations to help you identify idle and underutilized instances... To receive rightsizing recommendations that are based on memory utilization data, you must first enable memory data collection by installing the Amazon CloudWatch agent..."
Source: AWS Documentation, AWS Cost Management User Guide, "Rightsizing recommendations".

2. Using Cost Explorer with AWS Organizations: "If you have a management account in an organization in AWS Organizations, you can view the costs and usage for the member accounts. The rightsizing recommendations page in the management account provides recommendations for all member accounts in your organization."
Source: AWS Documentation, AWS Cost Management User Guide, "Viewing rightsizing recommendations for member accounts in your organization".

3. Deploying the CloudWatch Agent with Systems Manager: "To install the CloudWatch agent on a new instance or an existing instance, you can use AWS Systems Manager... Systems Manager Distributor simplifies and scales the process of distributing and maintaining the agent packages."
Source: AWS Documentation, Amazon CloudWatch User Guide, "Installing the CloudWatch agent by using AWS Systems Manager".

Question: 17

A company needs to create and manage multiple AWS accounts for a number of departments from a
central location. The security team requires read-only access to all accounts from its own AWS account. The company is using AWS Organizations and created an account for the security team. How should a solutions architect meet these requirements?

A. Use the OrganizationAccountAccessRole IAM role to create a new IAM policy with read-only access in each member account. Establish a trust relationship between the IAM policy in each member account and the security account. Ask the security team to use the IAM policy to gain access.

B. Use the Organization AccountAccessRole IAM role to create a new IAM role with read-only access
in each member account. Establish a trust relationship between the IAM role in each member account and the security account. Ask the security team to use the IAM role to gain access.

C. Ask the security team to use AWS Security Token Service (AWS STS) lo call the AssumeRole API tor
the Organization AccountAccessRole IAM role in the management account from the security account.
Use the generated temporary credentials to gain access.

D. Ask the security team to use AWS Security Token Service (AWS STS) to call the AssumeRole API for
the Organization AccountAccessRole IAM role in the member account from the security account. Use
the generated temporary credentials to gain access.

Answer:

B

Explanation:

This solution follows the principle of least privilege and standard AWS best practices for cross-account access within an organization. The OrganizationAccountAccessRole provides administrative access from the management account into member accounts. This administrative role should be used to provision a new, separate IAM role in each member account with only the required read-only permissions. This new role's trust policy is configured to explicitly allow the security account to assume it. This separates administrative setup from routine auditing, ensuring the security team has only the access they need, and the process can be automated and managed centrally from the management account.

Why Incorrect Options are Wrong:

A. A trust relationship is established with an IAM principal (like a role), not an IAM policy. Policies define permissions and are attached to principals.

C. The OrganizationAccountAccessRole exists in member accounts to be assumed by the management account, not the other way around. The role is not present in the management account.

D. The default trust policy for the OrganizationAccountAccessRole only allows the management account to assume it. The security account would be denied access without modifying this trust policy first.

References:

1. AWS IAM User Guide, "Tutorial: Delegate access across AWS accounts using IAM roles": This tutorial details the standard procedure for creating a role in a target account (the member account) that trusts a principal in another account (the security account) to assume it. This directly supports the methodology described in the correct answer (B).
2. AWS Organizations User Guide, "Accessing and administering the member accounts in your organization": This section explains that when you create a member account, AWS Organizations creates the OrganizationAccountAccessRole within that member account. The documentation states, "By default, only the management account can assume this role." This directly refutes the viability of option D.

3. AWS IAM User Guide, "Role terms and concepts", Section: "Trust policy": This section clarifies that a role trust policy is a required part of an IAM role that "defines which principals (accounts, users, roles, and federated users) can assume the role." This confirms that trust is established with a role, not a policy, invalidating the premise of option A.

Question: 18

A company runs an unauthenticated static website (www.example.com) that includes a registration
form for users. The website uses Amazon S3 for hosting and uses Amazon CloudFront as the content
delivery network with AWS WAF configured. When the registration form is submitted, the website calls an Amazon API Gateway API endpoint that invokes an AWS Lambda function to process the
payload and forward the payload to an external API call.
During testing, a solutions architect encounters a cross-origin resource sharing (CORS) error. The
solutions architect confirms that the CloudFront distribution origin has the Access-Control-Allow-Origin header set towww.example.com.
What should the solutions architect do to resolve the error?

A. Change the CORS configuration on the S3 bucket. Add rules for CORS to the Allowed Origin element forwww.example.com.

B. Enable the CORS setting in AWS WAF. Create a web ACL rule in which the Access-Control-Allow-
Origin header is set towww.example.com.

C. Enable the CORS setting on the API Gateway API endpoint. Ensure that the API endpoint is configured to return all responses that have the Access-Control -Allow-Origin header set towww.example.com.

D. Enable the CORS setting on the Lambda function. Ensure that the return code of the function has
the Access-Control-Allow-Origin header set towww.example.com.

Answer:

C

Explanation:

The cross-origin request originates from the browser-hosted website (www.example.com) and targets the Amazon API Gateway endpoint. For the browser to permit this request, the destination server (API Gateway) must respond with the appropriate Cross-Origin Resource Sharing (CORS) headers, specifically Access-Control-Allow-Origin. Amazon API Gateway provides a built-in feature to enable and configure CORS. This automatically creates the necessary OPTIONS preflight method and adds the required CORS headers to the method's integration responses. Configuring CORS directly on the API Gateway resource is the standard and most efficient method to resolve this issue.

Why Incorrect Options are Wrong:

A. The S3 bucket hosts the static website; it is the source of the cross-origin request, not the destination API endpoint that needs to return the CORS headers.
B. AWS WAF is a web application firewall used to filter and monitor HTTP/S requests. It does not manage or add application-level response headers like Access-Control-Allow-Origin.
D. While the Lambda function can be coded to return CORS headers, the best practice is to manage this at the API Gateway level. API Gateway can handle preflight OPTIONS requests without invoking the backend, which is more efficient.

References:

1. AWS API Gateway Developer Guide: "Enabling CORS for a REST API resource". This document states, "After you enable CORS for a resource, API Gateway automatically adds an OPTIONS preflight method to the resource... API Gateway then sends a response to the OPTIONS preflight request with the required Access-Control-Allow- headers".
2. AWS API Gateway Developer Guide: "CORS support for HTTP APIs". This guide explains, "Cross-origin resource sharing (CORS) is a browser security feature that restricts cross-origin HTTP requests... To support CORS, you must configure your API to send CORS headers in its responses." This confirms the API endpoint is the correct place for configuration.
3. AWS S3 User Guide: "Using cross-origin resource sharing (CORS)". This documentation clarifies that S3 CORS configuration is for allowing cross-origin requests to the S3 bucket itself, not for API calls made from a website hosted on S3.
4. AWS WAF Developer Guide: "How AWS WAF works". This guide describes WAF's function as inspecting incoming web requests and blocking or allowing them based on rules, confirming it does not modify response headers for CORS.

Question: 19

A large education company recently introduced Amazon Workspaces to provide access to internal
applications across multiple universities. The company is storing user profiles on an Amazon FSx (or
Windows File Server file system. The tile system is configured with a DNS alias and is connected to a
self-managed Active Directory. As more users begin to use the Workspaces, login time increases to
unacceptable levels.
An investigation reveals a degradation in performance of the file system. The company created the
file system on HDD storage with a throughput of 16 MBps. A solutions architect must improve the performance of the file system during a defined maintenance window.
What should the solutions architect do to meet these requirements with the LEAST administrative effort?

A. Use AWS Backup to create a point-In-lime backup of the file system. Restore the backup to a new
FSx for Windows File Server file system. Select SSD as the storage type Select 32 MBps as the

throughput capacity. When the backup and restore process Is completed, adjust the DNS alias accordingly. Delete the original file system.

B. Disconnect users from the file system. In the Amazon FSx console, update the throughput capacity
to 32 MBps. Update the storage type to SSD. Reconnect users to the file system.

C. Deploy an AWS DataSync agent onto a new Amazon EC2 Instance. Create a task. Configure the
existing file system as the source location. Configure a new FSx for Windows File Server file system
with SSD storage and 32 MBps of throughput as the target location. Schedule the task. When the task
is completed, adjust the DNS alias accordingly. Delete the original file system.

D. Enable shadow copies on the existing file system by using a Windows PowerShell command. Schedule the shadow copy job to create a point-in-time backup of the file system. Choose to restore
previous versions. Create a new FSx for Windows File Server file system with SSD storage and 32
MBps of throughput. When the copy job is completed, adjust the DNS alias. Delete the original file

system.

Answer:

A

Explanation:

The most efficient method with the least administrative effort is to use the native AWS Backup and restore functionality. This fully managed service allows creating a point-in-time backup of the existing Amazon FSx file system. During the restore process, a new file system can be created with updated specifications, including changing the storage type from HDD to SSD and increasing the throughput capacity. This approach is highly integrated within the AWS ecosystem, requires minimal configuration, and avoids the overhead of setting up additional infrastructure like a DataSync agent. After the new file system is ready, a simple update to the DNS alias redirects users, completing the migration seamlessly.

Why Incorrect Options are Wrong:

B: This is incorrect because Amazon FSx for Windows File Server does not support changing the storage type (from HDD to SSD) of an existing file system in-place. A new file system must be created.

C: While AWS DataSync is a valid migration tool, it requires more administrative effort than AWS Backup. It involves deploying an agent, configuring network paths, and creating and managing a migration task, which is more complex.

D: Windows Shadow Copies are designed for point-in-time, file-level recovery on an existing volume, not for migrating an entire file system to a new, separate instance. This is not the correct tool for this task.

References:

1. AWS Documentation - Restoring an Amazon FSx backup: "When you restore a file system, you create a new Amazon FSx file system... You can modify the following file system configuration settings when you restore a backup: ... VPC, Subnets, Windows authentication, Daily automatic backup window, Storage type, SSD IOPS, Throughput capacity..." This confirms that restoring a backup allows for changing the storage type and throughput, supporting option A.
Source: AWS Documentation, Amazon FSx for Windows File Server User Guide, "Working with backups", "Restoring backups".
2. AWS Documentation - Updating file system properties: "You can modify a file system's storage capacity and throughput capacity... You can't modify a file system's storage type or DNS name." This confirms that option B is not possible as the storage type cannot be changed in-place.
Source: AWS Documentation, Amazon FSx for Windows File Server User Guide, "Managing file systems", "Updating file system properties".
3. AWS Documentation - Migrating to FSx for Windows File Server: This page outlines migration

options. While it lists AWS DataSync as a primary option, the setup involves more steps (deploying an agent, creating locations and tasks) than the integrated AWS Backup/Restore process, making option C less optimal for the "least administrative effort" requirement.
Source: AWS Documentation, Amazon FSx for Windows File Server User Guide, "Migrating to FSx for Windows File Server".

Question: 20

A company has an application that is deployed on Amazon EC2 instances behind an Application Load
Balancer (ALB). The instances are part of an Auto Scaling group.Theapplication has unpredictable
workloads and frequently scales out and in. The company's development team wants to analyze application logs to find ways to improve the application's performance. However, the logs are no longer available after instances scale in.
Which solution will give the development team the ability to view the application logs after a scale-
in event?

A. Enable access logs for the ALB. Store the logs in an Amazon S3 bucket.

B. Configure the EC2 instances lo publish logs to Amazon CloudWatch Logs by using the unified CloudWatch agent.

C. Modify the Auto Scaling group to use a step scaling policy.

D. Instrument the application with AWS X-Ray tracing.

Answer:

B

Explanation:

The primary issue is the loss of application logs when EC2 instances are terminated during scale-in events. The unified CloudWatch agent is designed to solve this problem. By installing and configuring the agent on the EC2 instances, application logs can be streamed in near real-time to Amazon CloudWatch Logs. CloudWatch Logs provides a centralized and durable storage solution, ensuring that logs from all instances, including terminated ones, are preserved and available for analysis by the development team. This directly addresses the requirement to view logs after a scale-in event.

Why Incorrect Options are Wrong:

A. ALB access logs contain data about requests to the load balancer, not the application-specific logs generated by the code running on the EC2 instances.
C. A step scaling policy adjusts how an Auto Scaling group scales. It does not address the persistence of logs on the instances that are terminated.
D. AWS X-Ray is a service for tracing and analyzing request flows and performance bottlenecks, not for aggregating and storing detailed application log files.

References:

1. AWS CloudWatch User Guide: "You can use the unified CloudWatch agent to collect both system-level metrics and log files from Amazon EC2 instances... The logs collected by the CloudWatch agent are processed and stored in Amazon CloudWatch Logs."
Source: AWS Documentation, CloudWatch User Guide, "Collecting metrics and logs from Amazon EC2 instances and on-premises servers with the CloudWatch agent".
2. AWS Elastic Load Balancing User Guide: "The access log contains information about the requests sent to your load balancer, such as the time the request was received, the client's IP address, latencies, request paths, and server responses."
Source: AWS Documentation, User Guide for Application Load Balancers, "Access logs for your Application Load Balancer".
3. AWS X-Ray Developer Guide: "AWS X-Ray helps developers analyze and debug production, distributed applications... With X-Ray, you can understand how your application and its underlying services are performing to identify and troubleshoot the root cause of performance issues and errors."
Source: AWS Documentation, AWS X-Ray Developer Guide, "What is AWS X-Ray?".
4. Amazon EC2 Auto Scaling User Guide: "With step scaling policies, you choose scaling metrics and threshold values for the CloudWatch alarms that trigger the scaling process... A step scaling policy increases or decreases the current capacity of your Auto Scaling group."
Source: AWS Documentation, Amazon EC2 Auto Scaling User Guide, "Step and simple scaling policies for Amazon EC2 Auto Scaling".

Question: 21

A company has an application that generates reports and stores them in an Amazon S3 bucket When
a user accesses their report, the application generates a signed URL to allow the user to download
the report. The company's security team has discovered that the files are public and that anyone can
download them without authentication The company has suspended the generation of new reports
until the problem is resolved.
Which set of actions will immediately remediate the security issue without impacting the application's normal workflow?

A. Create an AWS Lambda function that applies a deny all policy for users who are not authenticated.
Create a scheduled event to invoke the Lambda function

B. Review the AWS Trusted Advisor bucket permissions check and implement the recommended actions.

C. Run a script that puts a private ACL on all of the objects in the bucket.

D. Use the Block Public Access feature in Amazon S3 to set the IgnorePublicAcls option to TRUE on
the bucket.

Answer:

D

Explanation:

The most immediate and effective solution is to use the Amazon S3 Block Public Access feature. Setting the IgnorePublicAcls option to TRUE at the bucket level instructs Amazon S3 to ignore any public Access Control Lists (ACLs) on all existing and future objects within that bucket. This instantly revokes public access without needing to modify individual objects. This remediation does not impact the application's workflow because presigned URLs grant temporary access based on the permissions of the IAM principal that created them, not on the object's public accessibility status.

Why Incorrect Options are Wrong:

A: A scheduled Lambda function is not an immediate solution, as it runs on a delay. It is also an overly complex and indirect method for managing bucket permissions.
B: AWS Trusted Advisor is a diagnostic and advisory tool. It identifies the problem and recommends solutions but does not perform the remediation action itself.
C: Running a script to modify individual object ACLs can be time-consuming for a large number of objects and does not prevent newly created objects from being made public.

References:

1. Amazon S3 User Guide, "Blocking public access to your Amazon S3 storage": This document details the S3 Block Public Access feature. It states, "When you set IgnorePublicAcls to TRUE for a bucket, Amazon S3 ignores all public ACLs on that bucket and the objects that it contains." This confirms the immediate effect of the setting. (Source: AWS Official Documentation)
Reference: https://docs.aws.amazon.com/AmazonS3/latest/userguide/access-control-block-public-access.html, Section: "Block Public Access settings".
2. Amazon S3 User Guide, "Sharing objects using presigned URLs": This guide explains that presigned URLs grant access based on the credentials of the creator. "Anyone who receives the presigned URL then has access to the object as if they were the original user who created the presigned URL." This clarifies why blocking public access does not affect the application's workflow. (Source: AWS Official Documentation)

Reference: https://docs.aws.amazon.com/AmazonS3/latest/userguide/ShareObjectPreSignedURL.html, Section: "How presigned URLs work".
3. AWS Security Blog, "Amazon S3 Block Public Access - Another Layer of Protection for Your Accounts and Buckets": This article emphasizes that Block Public Access is a centralized control to prevent accidental public exposure. "You can apply these settings to individual buckets or to all buckets in your account... This setting overrides object ACLs that grant public access." (Source: AWS Official Documentation)
Reference: https://aws.amazon.com/blogs/aws/amazon-s3-block-public-access-another-layer-of-protection-for-your-accounts-and-buckets/, Paragraphs 3 and 4.

Question: 22

A large company recently experienced an unexpected increase in Amazon RDS and Amazon DynamoDB costs. The company needs to increase visibility into details of AWS Billing and Cost Management There are various accounts associated with AWS Organizations, including many development and production accounts There is no consistent tagging strategy across the organization, but there are guidelines in place that require all infrastructure to be deployed using AWS CloudFormation with consistent tagging. Management requires cost center numbers and project ID numbers for all existing and future DynamoDB tables and RDS instances.
Which strategy should the solutions architect provide to meet these requirements?

A. Use Tag Editor to tag existing resources Create cost allocation tags to define the cost center and
project ID and allow 24 hours for tags to propagate to existing resources.

B. Use an AWS Config rule to alert the finance team of untagged resources Create a centralized AWS
Lambda based solution to tag untagged RDS databases and DynamoDB resources every hour using a
cross-account role.

C. Use Tag Editor to tag existing resources Create cost allocation tags to define the cost center and

project ID Use SCPs to restrict resource creation that do not have the cost center and project ID on
the resource.

D. Create cost allocation tags to define the cost center and project ID and allow 24 hours for tags to
propagate to existing resources Update existing federated roles to restrict privileges to provision resources that do not include the cost center and project ID on the resource.

Answer:

C

Explanation:

This strategy provides a complete solution by addressing both existing and future resources. First, using Tag Editor allows for the bulk tagging of existing RDS and DynamoDB resources across all accounts in the organization, which is necessary for immediate remediation. Second, creating cost allocation tags and activating them in the Billing and Cost Management console makes these tags visible in cost reports, fulfilling the primary requirement for cost visibility. Finally, implementing Service Control Policies (SCPs) at the AWS Organizations level provides a robust, preventive control. An SCP can deny resource creation APIs (e.g., rds:CreateDBInstance,

dynamodb:CreateTable) if the required cost-center and project-id tags are not present in the request, thus enforcing the tagging policy for all future resources.

Why Incorrect Options are Wrong:

A: This option only addresses existing resources with Tag Editor and enables cost allocation. It lacks a mechanism to enforce the tagging policy on future resources, failing to meet a key requirement.
B: This is a detective and reactive approach. AWS Config rules and Lambda functions would identify and tag non-compliant resources after they have been created, rather than preventing their creation in the first place.
D: This option fails to address the existing untagged resources. While using IAM policies on federated roles can enforce tagging, it is less scalable and harder to manage across many accounts than using a centralized SCP.

References:

1. Service Control Policies (SCPs): AWS Organizations User Guide. "SCPs are 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." The guide provides examples of using condition keys like aws:RequestTag/tag-key to require tags on resource creation. (See: AWS Organizations User Guide, "Service control policies (SCPs)", "Examples of SCPs", section on "Require a tag on created resources").

2. Tag Editor: AWS Resource Groups and Tag Editor User Guide. "With Tag Editor, you can add, edit, or delete tags for multiple resources at once... You can find resources to tag by searching for resources in the regions you specify." (See: AWS Resource Groups User Guide, "Working with Tag Editor").
3. Cost Allocation Tags: AWS Billing and Cost Management User Guide. "You can use tags to organize your resources, and cost allocation tags to track your AWS costs on a detailed level... After you activate cost allocation tags, AWS uses the cost allocation tags to organize your resource costs on your cost allocation report." (See: AWS Billing User Guide, "Using cost allocation tags", section on "Activating user-defined cost allocation tags").
4. AWS Config Rules: AWS Config Developer Guide. "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." This highlights its role as an evaluation/detective tool, not a preventive one. (See: AWS Config Developer Guide, "Working with AWS Config Rules").

Question: 23

A company is serving files to its customers through an SFTP server that is accessible over the internet
The SFTP server is running on a single Amazon EC2 instance with an Elastic IP address attached
Customers connect to the SFTP server through its Elastic IP address and use SSH for authentication
The EC2 instance also has an attached security group that allows access from all customer IP addresses.
A solutions architect must implement a solution to improve availability minimize the complexity of infrastructure management and minimize the disruption to customers who access files. The solution
must not change the way customers connect
Which solution will meet these requirements?

A. Disassociate the Elastic IP address from the EC2 instance Create an Amazon S3 bucket to be used
for SFTP file hosting Create an AWS Transfer Family server. Configure the Transfer Family server with
a publicly accessible endpoint Associate the SFTP Elastic IP address with the new endpoint.

Point the
Transfer Family server to the S3 bucket Sync all files from the SFTP server to the S3 bucket.

B. Disassociate the Elastic IP address from the EC2 instance Create an Amazon S3 bucket to be used
for SFTP file hosting Create an AWS Transfer Family Server Configure the Transfer Family server with
a VPC-hosted, internet-facing endpoint Associate the SFTP Elastic IP address with the new endpoint
Attach the security group with customer IP addresses to the new endpoint Point the Transfer Family
server to the S3 bucket. Sync all files from the SFTP server to the S3 bucket.

C. Disassociate the Elastic IP address from the EC2 instance. Create a new Amazon Elastic File System
(Amazon EFS) file system to be used for SFTP file hosting. Create an AWS Fargate task definition to
run an SFTP server Specify the EFS file system as a mount in the task definition Create a Fargate
service by using the task definition, and place a Network Load Balancer (NLB) in front of the service.

When configuring the service, attach the security group with customer IP addresses to the tasks that
run the SFTP server Associate the Elastic IP address with the NLB Sync all files from the SFTP server to
the S3 bucket.

D. Disassociate the Elastic IP address from the EC2 instance. Create a multi-attach Amazon Elastic
Block Store (Amazon EBS) volume to be used for SFTP file hosting. Create a Network Load Balancer
(NLB) with the Elastic IP address attached. Create an Auto Scaling group with EC2 instances that run
an SFTP server. Define in the Auto Scaling group that instances that are launched should attach the
new multi-attach EBS volume Configure the Auto Scaling group to automatically add instances behind the NLB. configure the Auto Scaling group to use the security group that allows customer IP
addresses for the EC2 instances that the Auto Scaling group launches Sync all files from the SFTP
server to the new multi-attach EBS volume.

Answer:

B

Explanation:

The optimal solution is to use AWS Transfer Family, a fully managed service that supports SFTP and significantly reduces infrastructure management overhead. By creating a VPC-hosted, internet-facing endpoint, the company can associate its existing Elastic IP address with the Transfer Family server. This approach is crucial for meeting the requirement of no customer-facing changes, as clients can continue connecting to the same IP address. Using Amazon S3 as the backend storage provides high availability and durability for the files. Attaching the existing security group to the VPC endpoint maintains the established IP-based access controls, ensuring a seamless and secure migration with minimal disruption.

Why Incorrect Options are Wrong:

A. A "publicly accessible" AWS Transfer Family endpoint is assigned an AWS-managed public DNS hostname and does not support associating a pre-existing Elastic IP address.
C. This solution using AWS Fargate, an NLB, and Amazon EFS significantly increases infrastructure complexity, which contradicts the requirement to minimize management.
D. This option, using an Auto Scaling group of EC2 instances and a multi-attach EBS volume,

dramatically increases management complexity compared to the original single-instance setup.

References:

1. AWS Transfer Family User Guide - Create an internet-facing endpoint in a virtual private cloud: This document explicitly states the capability to use a pre-existing Elastic IP address with a VPC-hosted endpoint. It says, "For VPC hosted endpoints, you can associate one or more Elastic IP addresses with the endpoints of your SFTP server." This directly supports option B and refutes option A. (AWS Transfer Family User Guide, "Working with server endpoints" section).
2. AWS Transfer Family User Guide - Identity and access management for AWS Transfer Family: This guide details how Transfer Family integrates with identity providers and supports SSH keys for authentication, confirming that the customer connection method can remain unchanged. (AWS Transfer Family User Guide, "Security in AWS Transfer Family" section).
3. AWS Transfer Family Product Page: The official product page describes the service as "a fully managed service," which directly addresses the requirement to "minimize the complexity of infrastructure management." (aws.amazon.com/aws-transfer-family/).
4. AWS Transfer Family User Guide - Control access using security groups: This documentation confirms that security groups can be associated with the VPC endpoint for a Transfer Family server to control traffic, fulfilling the requirement to maintain the existing IP-based access controls. (AWS Transfer Family User Guide, "Security in AWS Transfer Family" - "Network and access control" subsection).

Question: 24

A solutions architect is creating an application that stores objects in an Amazon S3 bucket The solutions architect must deploy the application in two AWS Regions that will be used simultaneously
The objects in the two S3 buckets must remain synchronized with each other.
Which combination of steps will meet these requirements with the LEAST operational overhead? (Select THREE)

A. Create an S3 Multi-Region Access Point. Change the application to refer to the Multi-Region Access Point

B. Configure two-way S3 Cross-Region Replication (CRR) between the two S3 buckets

C. Modify the application to store objects in each S3 bucket.

D. Create an S3 Lifecycle rule for each S3 bucket to copy objects from one S3 bucket to the other S3
bucket.

E. Enable S3 Versioning for each S3 bucket

F. Configure an event notification for each S3 bucket to invoke an AVVS Lambda function to copy objects from one S3 bucket to the other S3 bucket.

Answer:

A, B, E

Explanation:

This solution creates a robust, active-active, multi-region architecture with minimal operational overhead using managed AWS services. S3 Cross-Region Replication (CRR) is the native feature for automatically synchronizing objects between buckets. For an active-active setup where writes occur in both regions, two-way (or bidirectional) CRR is required (B). A mandatory prerequisite for S3 CRR is that versioning must be enabled on both the source and destination buckets (E). To simplify the application architecture and provide a single global endpoint that routes traffic to the lowest latency S3 bucket, an S3 Multi-Region Access Point is used (A). This combination offloads the complexity of replication and regional routing from the application to AWS-managed services.

Why Incorrect Options are Wrong:

C. Modify the application to store objects in each S3 bucket.
This approach, known as dual-writing, increases application complexity and operational overhead, as the application must manage writes, retries, and potential inconsistencies.
D. Create an S3 Lifecycle rule for each S3 bucket to copy objects from one S3 bucket to the other S3 bucket.

S3 Lifecycle rules are used for managing object lifecycle (e.g., storage class transitions, expiration), not for active data synchronization between buckets.
F. Configure an event notification for each S3 bucket to invoke an AWS Lambda function to copy objects from one S3 bucket to the other S3 bucket.
This is a custom solution that incurs significantly more operational overhead (development, deployment, monitoring) compared to the managed S3 CRR service.

References:

1. Amazon S3 User Guide - Replicating objects: "To set up replication, you create a replication configuration that includes... the source bucket that you want to replicate objects from... the destination bucket or buckets where you want Amazon S3 to replicate objects." For a two-way setup, this is configured in both directions.
2. Amazon S3 User Guide - Requirements for replication: "Both your source and destination buckets must have versioning enabled." This directly supports option E as a prerequisite for option B.
3. Amazon S3 User Guide - Using a Multi-Region Access Point to route requests: "Amazon S3 Multi-Region Access Points provide a global endpoint that applications can use to fulfill requests from S3 buckets that are located in multiple AWS Regions... S3 Multi-Region Access Points route your request over the AWS global network to the S3 bucket with the lowest latency." This supports option A for simplifying application access.

4. Amazon S3 User Guide - Replication configuration overview: This section details how S3 replication handles objects in a bidirectional configuration, noting that "S3 replication does not replicate an object that is already a replica," which prevents infinite replication loops in a two-way setup. This confirms the viability of option B.

Question: 25

A solutions architect is designing an application to accept timesheet entries from employees on their
mobile devices. Timesheets will be submitted weekly, with most of the submissions occurring on Friday. The data must be stored in a format that allows payroll administrators to run monthly reports
The infrastructure must be highly available and scale to match the rate of incoming data and reporting requests.
Which combination of steps meets these requirements while minimizing operational overhead? (Select TWO

A. Deploy the application to Amazon EC2 On-Demand Instances with load balancing across multiple
Availability Zones. Use scheduled Amazon EC2 Auto Scaling to add capacity before the high volume
of submissions on Fridays

B. Deploy the application in a container using Amazon Elastic Container Service (Amazon ECS) with
load balancing across multiple Availability Zones Use scheduled Service Auto Scaling to add capacity

before the high volume of submissions on Fridays

C. Deploy the application front end to an Amazon S3 bucket served by Amazon CloudFront Deploy
the application backend using Amazon API Gateway with an AWSLambda proxy integration

D. Store the timesheet submission data in Amazon Redshift Use Amazon QuickSight to generate the
reports using Amazon Redshift as the data source

E. Store the timesheet submission data in Amazon S3. Use Amazon Athena and Amazon QuickSight
to generate the reports using Amazon S3 as the data source.

Answer:

C, E

Explanation:

This scenario requires a highly available, scalable solution with minimal operational overhead to handle a variable workload (peak on Fridays). A serverless architecture is the best fit.
Option C describes a classic serverless web application backend. Using Amazon API Gateway

and AWS Lambda provides a highly available and automatically scaling compute layer that precisely matches the rate of incoming timesheet submissions without any need for managing servers or scheduling capacity. Hosting the static front-end on Amazon S3 with Amazon CloudFront for distribution is also a best practice for performance, scalability, and low operational overhead.

Option E provides a serverless data analytics and reporting solution. Storing raw timesheet data in Amazon S3 offers a durable, scalable, and cost-effective data store. Amazon Athena can then be used to run standard SQL queries directly on the data in S3 for the monthly reports, without needing to provision or manage any data warehouse infrastructure. Amazon QuickSight integrates seamlessly with Athena to visualize the data.

Why Incorrect Options are Wrong:

A. Deploying on EC2 instances involves significant operational overhead for managing servers, patching, and security. Scheduled scaling is less efficient than the automatic, on-demand scaling of a serverless approach.

B. While Amazon ECS abstracts some infrastructure management, it still requires more operational overhead than a fully serverless Lambda-based approach. Scheduled scaling has the same drawbacks as in option A.

D. Amazon Redshift is a provisioned data warehouse, which introduces significant operational overhead for cluster management and scaling. It is also not cost-effective for this use case compared to a serverless S3/Athena solution.

References:

1. AWS Whitepapers & Guides, "Serverless Architectures with AWS Lambda" (Page 5): This document outlines the pattern of using Amazon S3 for static content, Amazon API Gateway for API endpoints, and AWS Lambda for backend logic. It states, "These services allow you to build and deploy applications and services with high availability, and scalability, and you are not responsible for any of the operational overhead of managing the infrastructure." This directly supports option C.
2. AWS Documentation, "What is Amazon Athena?": The documentation states, "Amazon Athena is an interactive query service that makes it easy to analyze data directly in Amazon Simple Storage Service (Amazon S3) using standard SQL... Athena is serverless, so there is no infrastructure to set up or manage, and you pay only for the queries you run." This supports the choice of Athena in option E for minimizing operational overhead.
3. AWS Documentation, "Data lakes and analytics on AWS": This guide describes the common pattern of using Amazon S3 as a central data lake. It highlights using services like Amazon Athena for interactive queries and Amazon QuickSight for visualization, which aligns perfectly with the architecture described in option E.
4. AWS Well-Architected Framework, "Operational Excellence Pillar" (Design Principles section): This framework advocates for performing operations as code and using managed services to

reduce the burden of infrastructure management. The serverless combination of C and E aligns with these principles, whereas options A, B, and D, which require more manual infrastructure management, do not.

Question: 26

A company wants to send data from its on-premises systems to Amazon S3 buckets. The company
created the S3 buckets in three different accounts. The company must send the data privately without the data traveling across the internet The company has no existing dedicated connectivity to
AWS
Which combination of steps should a solutions architect take to meet these requirements? (Select TWO.)

A. Establish a networking account in the AWS Cloud Create a private VPC in the networking account.
Set up an AWS Direct Connect connection with a private VIF between the on-premises environment
and the private VPC.

B. Establish a networking account in the AWS Cloud Create a private VPC in the networking account.
Set up an AWS Direct Connect connection with a public VIF between the on-premises environment
and the private VPC.

C. Create an Amazon S3 interface endpoint in the networking account.

D. Create an Amazon S3 gateway endpoint in the networking account.

E. Establish a networking account in the AWS Cloud Create a private VPC in the networking account.
Peer VPCs from the accounts that host the S3 buckets with the VPC in the network account.

Answer:

A, C

Explanation:

To meet the requirement of sending data privately from an on-premises environment to Amazon S3 without using the internet, a dedicated private connection must be established first. An AWS Direct Connect connection with a private virtual interface (VIF) creates a private, dedicated network link between the on-premises data center and a VPC in AWS.
Once this private path is established, an Amazon S3 interface endpoint must be created in the VPC. An interface endpoint is a network interface with a private IP address that serves as an entry point for traffic destined for S3. This endpoint is accessible from the on-premises network over the Direct Connect connection, ensuring that all data transfer to S3 remains on the private

network.

Why Incorrect Options are Wrong:

B. A public VIF is used to access AWS public services via their public IP addresses over a Direct Connect link; it does not provide private connectivity into a VPC.
D. An S3 gateway endpoint provides private access to S3 only for resources within the VPC. It cannot be accessed from an on-premises network over Direct Connect or a VPN.
E. VPC peering connects different VPCs within AWS. It does not solve the fundamental requirement of establishing the initial private connectivity from the on-premises environment to AWS.

References:

1. AWS Direct Connect User Guide, "Virtual interfaces": This document distinguishes between VIF types. It states, "A private virtual interface is used to access an Amazon VPC using private IP addresses." This supports the choice of a private VIF (Option A) for connecting to a private VPC.
2. AWS PrivateLink for Amazon S3, "Accessing S3 from on-premises": The documentation explicitly states, "With AWS PrivateLink for Amazon S3, you can provision interface VPC endpoints... in your virtual private cloud (VPC). These endpoints are directly accessible from applications that are on-premises over VPN and AWS Direct Connect." This confirms that an interface endpoint (Option C) is required for on-premises access.

3. Amazon VPC User Guide, "Gateway VPC endpoints": This guide clarifies the limitation of gateway endpoints: "Gateway endpoints do not allow access from an on-premises network, from a peered VPC, or from a transit gateway." This directly invalidates Option D for this scenario.
4. AWS Whitepaper, "Hybrid Connectivity" (Page 10): This whitepaper discusses connectivity patterns. It details how AWS Direct Connect with a private VIF establishes a private connection to a VPC, and how VPC endpoints are then used to privately access AWS services like S3 from both the VPC and connected on-premises networks.

Question: 27

A company operates quick-service restaurants. The restaurants follow a predictable model with high
sales traffic for 4 hours daily Sales traffic is lower outside of those peak hours.
The point of sale and management platform is deployed in the AWS Cloud and has a backend that is
based on Amazon DynamoDB. The database table uses provisioned throughput mode with 100.000
RCUs and 80.000 WCUs to match known peak resource consumption.
The company wants to reduce its DynamoDB cost and minimize the operational overhead for the IT
staff.
Which solution meets these requirements MOST cost-effectively?

A. Reduce the provisioned RCUs and WCUs

B. Change the DynamoDB table to use on-demand capacity.

C. Enable Dynamo DB auto scaling tor the table

D. Purchase 1-year reserved capacity that is sufficient to cover the peak load for 4 hours each day.

Answer:

C

Explanation:

The scenario describes a workload with predictable daily peaks. DynamoDB Auto Scaling is the ideal solution for this pattern. It automatically adjusts the table's provisioned read and write capacity in response to actual traffic, scaling up for the 4-hour peak and scaling down during periods of low traffic. This approach ensures that the application has sufficient capacity during high-demand periods while minimizing costs by not overprovisioning during off-peak hours. This directly addresses the requirements for cost reduction and minimal operational overhead, as the scaling process is fully managed by AWS.

Why Incorrect Options are Wrong:

A. Reducing the provisioned RCUs and WCUs permanently would cause performance degradation and request throttling during the daily 4-hour peak, failing to meet application requirements.
B. On-demand capacity is best suited for new or unpredictable workloads. For a predictable, high-traffic workload like this, provisioned capacity with auto scaling is more cost-effective.
D. Reserved capacity provides a billing discount on provisioned capacity but does not solve the

underlying issue of overprovisioning for 20 hours a day. You would still pay for peak capacity 24/7.

References:

1. AWS DynamoDB Developer Guide, "Managing throughput capacity on DynamoDB tables": "If your application traffic is variable or unpredictable, you can enable DynamoDB auto scaling... With auto scaling, you create a scaling policy that defines a target utilization... DynamoDB auto scaling then creates Amazon CloudWatch alarms that track your table's consumed capacity. When the table's utilization deviates from your target, a CloudWatch alarm is triggered and Amazon EC2 Auto Scaling initiates a scaling event." This describes the mechanism that perfectly fits the described scenario.
2. AWS DynamoDB FAQs, "Q: When should I use on-demand mode vs. provisioned mode with auto scaling?": "You should use on-demand mode when you have unpredictable application traffic... You should use provisioned mode with auto scaling if you have predictable application traffic, traffic that increases or decreases gradually, and if you want to control your DynamoDB costs." This directly contrasts the use cases, placing the question's scenario firmly in the auto scaling category for cost optimization.
3. AWS DynamoDB Developer Guide, "Reserved capacity": "With DynamoDB reserved capacity, you pay a one-time upfront fee and commit to a minimum provisioned usage level over a period of time... Reserved capacity is a billing feature. It does not affect DynamoDB performance." This clarifies that reserved capacity is a financial instrument applied to provisioned capacity, not a mechanism for handling variable loads.

Question: 28

A company manages hundreds of AWS accounts centrally in an organization in AWS Organizations.
The company recently started to allow product teams to create and manage their own S3 access points in their accounts. The S3 access points can be accessed only within VPCs not on the internet.
What is the MOST operationally efficient way to enforce this requirement?

A. Set the S3 access point resource policy to deny the s3 CreateAccessPoint action unless the s3:
AccessPointNetworkOngm condition key evaluates to VPC.

B. Create an SCP at the root level in the organization to deny the s3 CreateAccessPoint action unless
the s3 AccessPomtNetworkOngin condition key evaluates to VPC.

C. Use AWS CloudFormation StackSets to create a new 1AM policy in each AVVS account that allows
the s3: CreateAccessPoint action only if the s3 AccessPointNetworkOrigin condition key evaluates to
VPC.

D. Set the S3 bucket policy to deny the s3: CreateAccessPoint action unless the s3AccessPointNetworkOrigin condition key evaluates to VPC.

Answer:

B

Explanation:

Service Control Policies (SCPs) are the most operationally efficient mechanism for enforcing permission guardrails across all accounts in an AWS Organization. By applying an SCP at the organization's root, you can centrally mandate that any attempt to create an S3 access point (s3:CreateAccessPoint action) will fail unless its network origin is restricted to a VPC. This is achieved by using the s3:AccessPointNetworkOrigin condition key with a value of VPC. This single policy is inherited by all accounts, providing a non-overridable, scalable, and efficient governance solution without requiring configuration in individual accounts.

Why Incorrect Options are Wrong:

A. An S3 access point resource policy controls access permissions to an existing access point, not the permissions required for its creation.
C. While CloudFormation StackSets can deploy IAM policies, this is less efficient and secure than an SCP. It requires managing policy attachments and can be overridden by administrators within

member accounts.

D. An S3 bucket policy governs access to the bucket and its objects. It cannot be used to control the account-level s3:CreateAccessPoint action.

References:

1. AWS Organizations User Guide, Service Control Policies (SCPs): "SCPs are 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." (See the "Service Control Policies (SCPs)" section).
2. AWS Identity and Access Management User Guide, Actions, resources, and condition keys for Amazon S3: The documentation for the s3:AccessPointNetworkOrigin condition key states, "Filters access by the network origin of the access point. This key can be used to allow access point creation only if the network origin is VPC." (See the "Condition keys for Amazon S3" section).
3. Amazon S3 User Guide, Example - Requiring VPC-only S3 access points: This guide provides a direct example of the required SCP: "To require that all S3 access points created in the organization are VPC-only access points, you can use a service control policy (SCP) in AWS Organizations." The example policy provided is a direct solution to the question's scenario. (See the "Managing access to S3 access points" section).

Question: 29

A company is collecting a large amount of data from a fleet of IoT devices Data is stored as Optimized
Row Columnar (ORC) files in the Hadoop Distributed File System (HDFS) on a persistent Amazon EMR
cluster. The company's data analytics team queries the data by using SQL in Apache Presto deployed
on the same EMR cluster Queries scan large amounts of data, always run for less than 15 minutes,
and run only between 5 PM and 10 PM.
The company is concerned about the high cost associated with the current solution A solutions architect must propose the most cost-effective solution that will allow SQL data queries
Which solution will meet these requirements?

A. Store data in Amazon S3 Use Amazon Redshift Spectrum to query data.

B. Store data in Amazon S3 Use the AWS Glue Data Catalog and Amazon Athena to query data

C. Store data in EMR File System (EMRFS) Use Presto in Amazon EMR to query data

D. Store data in Amazon Redshift. Use Amazon Redshift to query data.

Answer:

B

Explanation:

The most cost-effective solution is to decouple storage from compute. Storing the ORC files in Amazon S3 provides a durable, scalable, and low-cost storage layer, eliminating the need for HDFS on an expensive, persistent EMR cluster.
Amazon Athena is a serverless query service that allows you to run standard SQL queries on data directly in Amazon S3. Because the workload is intermittent (running only between 5 PM and 10 PM) and queries are short-lived (15 minutes), Athena's pay-per-query pricing model is ideal. You are only charged for the data scanned by your queries, with no costs for idle time or cluster management. The AWS Glue Data Catalog serves as the central metadata repository, allowing Athena to understand the schema of the ORC files in S3.

Why Incorrect Options are Wrong:

A. Amazon Redshift Spectrum requires a running Amazon Redshift cluster, which incurs costs even when not actively querying. This makes it less cost-effective than the fully serverless Athena for this intermittent workload.
C. Using Amazon EMR, even with EMRFS on S3, still requires provisioning and paying for a cluster during the query window. This is generally more expensive and complex to manage for

short, ad-hoc queries compared to a serverless solution like Athena.
D. Storing data in Amazon Redshift requires a continuously running data warehouse cluster and an ETL process to load the data. This is the most expensive option and is not suitable for querying files directly or for intermittent workloads.

References:

1. Amazon Athena Documentation: "Amazon Athena is an interactive query service that makes it easy to analyze data in Amazon S3 using standard SQL. Athena is serverless, so there is no infrastructure to manage, and you pay only for the queries that you run."
Source: AWS Documentation, What is Amazon Athena?, Section: "How Athena works".
2. AWS Glue Documentation: "The AWS Glue Data Catalog is a central repository to store structural and operational metadata for all your data assets... Once your data is cataloged, it is immediately searchable, queryable, and available for ETL. It integrates with Amazon EMR, Amazon Redshift, Amazon Athena..."
Source: AWS Documentation, AWS Glue Developer Guide, Section: "Populating the AWS Glue Data Catalog".
3. AWS Big Data Blog: In a comparison of query engines, Athena is highlighted for its serverless nature and cost-effectiveness for ad-hoc analysis. "With Athena, you pay per query and are charged based on the amount of data scanned by the query... This makes Athena a good choice for interactive, ad hoc data exploration..."

Source: AWS Big Data Blog, Amazon Redshift Spectrum vs. Athena: A detailed feature and cost comparison, May 2023.
4. Amazon EMR Documentation: "With Amazon EMR you pay a per-second rate for every instance you launch." A persistent cluster, as described in the problem, incurs costs 24/7, making it inefficient for a workload that runs only five hours a day.
Source: AWS Documentation, Amazon EMR Pricing.

Question: 30

An environmental company is deploying sensors in major cities throughout a country to measure air
quality The sensors connect to AWS IoT Core to ingest timesheets data readings. The company stores
the data in Amazon DynamoDB
For business continuity the company must have the ability to ingest and store data in two AWS Regions
Which solution will meet these requirements?

A. Create an Amazon Route 53 alias failover routing policy with values for AWS IoT Core data endpoints in both Regions Migrate data to Amazon Aurora global tables

B. Create a domain configuration for AWS IoT Core in each Region Create an Amazon Route 53 latency-based routing policy Use AWS IoT Core data endpoints in both Regions as values Migrate the
data to Amazon MemoryDB for Radis and configure Cross-Region replication

C. Create a domain configuration for AWS IoT Core in each. Region Create an AmazonRoute 53 health
check that evaluates domain configuration health Create a failover routing policy with values for the

domain name from the AWS IoT Core domain configurations Update the DynamoDB table to a global
table

D. Create an Amazon Route 53 latency-based routing policy. Use AWS IoT Core data endpoints in
both Regions as values. Configure DynamoDB streams and Cross-Region data replication

Answer:

C

Explanation:

This solution correctly addresses both multi-region ingestion and storage for business continuity. For ingestion, creating custom domain configurations for AWS IoT Core in each region and using an Amazon Route 53 failover routing policy with health checks provides a robust, automated failover mechanism. If the primary region's IoT endpoint becomes unhealthy, Route 53 will automatically redirect sensor traffic to the secondary region. For storage, converting the Amazon DynamoDB table to a global table is the standard, fully managed solution for maintaining a multi-region, multi-active database. It automatically handles data replication between the specified regions, ensuring data is available in both locations as required.

Why Incorrect Options are Wrong:

A. Migrating to Amazon Aurora is an unnecessary architectural change; the requirement is to make the existing DynamoDB-based solution multi-region.
B. Latency-based routing optimizes for performance, not for failover, which is the core business continuity requirement. It also incorrectly suggests migrating to MemoryDB.
D. Latency-based routing is inappropriate for a failover scenario. While DynamoDB Streams can be used for replication, global tables are the superior, managed solution for this use case.

References:

1. AWS IoT Core Multi-Region with Failover: The AWS IoT Blog describes this exact pattern: "To achieve this, you can use Amazon Route 53 with a failover routing policy. You can create a health check that monitors the health of your primary region's endpoint. If the health check fails, Route 53 automatically routes traffic to your secondary region's endpoint."
Source: AWS IoT Blog, "Implementing multi-region AWS IoT Core with disaster recovery", Section: "Solution overview".
2. DynamoDB Global Tables: The official documentation states that global tables are the recommended solution for multi-region, multi-active workloads. "Amazon DynamoDB global tables provide a fully managed solution for deploying a multi-region, multi-active database...".
Source: AWS DynamoDB Developer Guide, "Global Tables", Introduction section.
3. Route 53 Failover Routing Policy: The documentation clarifies the use case for failover routing. "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." This is distinct from latency routing.
Source: Amazon Route 53 Developer Guide, "Choosing a routing policy", Section: "Failover routing".
4. AWS IoT Core Custom Domains: Using custom domains is a prerequisite for managing endpoints with Route 53 for failover. "Domain configurations for AWS IoT Core allow you to create a custom domain and use it for your device data endpoint."
Source: AWS IoT Core Developer Guide, "Custom domains", Introduction section.

Question: 31

A company is migrating an application to AWS. It wants to use fully managed services as much as
possible during the migration The company needs to store large, important documents within the application with the following requirements
1 The data must be highly durable and available
2. The data must always be encrypted at rest and in transit.
3 The encryption key must be managed by the company and rotated periodically
Which of the following solutions should the solutions architect recommend?

A. Deploy the storage gateway to AWS in file gateway mode Use Amazon EBS volume encryption
using an AWS KMS key to encrypt the storage gateway volumes

B. Use Amazon S3 with a bucket policy to enforce HTTPS for connections to the bucket and to enforce
server-side encryption and AWS KMS for object encryption.

C. Use Amazon DynamoDB with SSL to connect to DynamoDB Use an AWS KMS key to encrypt DynamoDB objects at rest.

D. Deploy instances with Amazon EBS volumes attached to store this data Use EBS volume encryption
using an AWS KMS key to encrypt the data.

Answer:

B

Explanation:

Amazon S3 is a fully managed object storage service designed for high durability (99.999999999%) and availability, making it ideal for storing large, important documents. The solution meets all requirements directly:
1. Encryption in transit: A bucket policy can enforce the use of HTTPS (TLS) for all data transfers.
2. Encryption at rest: Server-Side Encryption with AWS Key Management Service (SSE-KMS) encrypts objects using a KMS key.
3. Key Management: Using SSE-KMS with a customer-managed key gives the company full control over the key's lifecycle, including periodic rotation, which can be automated within AWS KMS. This approach leverages fully managed services to meet all security and operational requirements efficiently.

Why Incorrect Options are Wrong:

A. Storage Gateway is a hybrid cloud service for connecting on-premises infrastructure to AWS storage. It is not the most direct, fully managed solution for a new cloud-native application.
C. Amazon DynamoDB is a NoSQL database with a maximum item size of 400 KB. It is unsuitable for storing "large, important documents" and is designed for structured key-value data, not files.
D. Using EC2 instances with EBS volumes is an Infrastructure as a Service (IaaS) approach, not a fully managed service. It requires managing the operating system and instances, and EBS has lower native durability than S3.

References:

1. Amazon S3 Durability and Availability: AWS. (n.d.). Amazon S3 FAQs. "Amazon S3 is designed for 99.999999999% (11 9's) of durability... Amazon S3 Standard... is designed for 99.99% availability over a given year."
2. S3 Encryption with Customer-Managed Keys: AWS. (2023). Protecting data using server-side encryption with AWS Key Management Service keys (SSE-KMS). S3 User Guide. "When you use SSE-KMS, you can use the AWS managed key... or you can create and manage your own symmetric customer managed key."
3. Enforcing Encryption in Transit: AWS. (2023). Managing access with policies. S3 User Guide. A bucket policy can use the aws:SecureTransport c o n d i t i o n key to deny any requests that are not sent over HTTPS.
4. KMS Key Rotation: AWS. (2023). Rotating AWS KMS keys. AWS Key Management Service Developer Guide. "You can enable automatic key rotation for a customer managed key... AWS KMS rotates the KMS key 365 days after the enable date and every 365 days thereafter."
5. DynamoDB Item Size Limit: AWS. (2023). Service, account, and table quotas in Amazon DynamoDB. Amazon DynamoDB Developer Guide. "The maximum item size in DynamoDB is 400 KB, which includes both attribute name and value lengths (binary and text)."

Question: 32

A solutions architect must update an application environment within AWS Elastic Beanstalk using a
blue/green deployment methodology The solutions architect creates an environment that is identical
to the existing application environment and deploys the application to the new environment.
What should be done next to complete the update?

A. Redirect to the new environment using Amazon Route 53

B. Select the Swap Environment URLs option

C. Replace the Auto Scaling launch configuration

D. Update the DNS records to point to the green environment

Answer:

B

Explanation:

The most direct and recommended method to complete a blue/green deployment within AWS Elastic Beanstalk is to use the "Swap Environment URLs" feature. This action atomically swaps the CNAME records of the blue (existing) and green (new) environments. Production traffic is seamlessly redirected to the new application version with zero downtime. This approach is the standard practice for blue/green deployments in Elastic Beanstalk because it is managed, instantaneous, and allows for a simple rollback by performing the swap again if issues are discovered in the new environment.

Why Incorrect Options are Wrong:

A. While Route 53 can manage DNS for blue/green deployments, the native "Swap Environment URLs" feature is the specific, integrated mechanism provided by Elastic Beanstalk for this purpose.
C. Replacing the Auto Scaling launch configuration is a method for an in-place update within a single environment, not a blue/green deployment which involves two separate environments.
D. This is a generic description of the outcome. "Swap Environment URLs" is the specific Elastic Beanstalk console action that performs the underlying DNS CNAME record update.

References:

1. AWS Elastic Beanstalk Developer Guide: In the section on deployment strategies, it states, "To complete the blue/green deployment, you swap the CNAMEs of the two environments, which redirects traffic from the old version of your application to the new one. You do this using the Elastic Beanstalk console's Swap Environment URLs action..."

Source: AWS Documentation, AWS Elastic Beanstalk Developer Guide, "Blue/green deployments with Elastic Beanstalk".
2. AWS Elastic Beanstalk API Reference: The console action "Swap Environment URLs" corresponds to the SwapEnvironmentCNAMEs API call. The documentation for this API call confirms its function: "Swaps the CNAMEs of two environments."
Source: AWS Documentation, AWS Elastic Beanstalk API Reference, "SwapEnvironmentCNAMEs".
3. AWS Whitepapers & Guides: The "Blue/Green Deployments on AWS" guide outlines this pattern, stating, "AWS Elastic Beanstalk provides a feature to swap the environment URLs, which makes it easy to implement the blue/green deployment technique."
Source: AWS Prescriptive Guidance, Blue/Green Deployments on AWS, "Blue/green deployments using AWS Elastic Beanstalk" section.

Question: 33

A company has a complex web application that leverages Amazon CloudFront for global scalability
and performance Over time, users report that the web application is slowing down
The company's operations team reports that the CloudFront cache hit ratio has been dropping steadily. The cache metrics report indicates that query strings on some URLs are inconsistently ordered and are specified sometimes in mixed-case letters and sometimes in lowercase letters.
Which set of actions should the solutions architect take to increase the cache hit ratio as quickly as
possible?

A. Deploy a Lambda@Edge function to sort parameters by name and force them lo be lowercase Select the CloudFront viewer request trigger to invoke the function

B. Update the CloudFront distribution to disable caching based on query string parameters.

C. Deploy a reverse proxy after the load balancer to post-process the emitted URLs in the application
to force the URL strings to be lowercase.

D. Update the CloudFront distribution to specify casing-insensitive query string processing.

Answer:

A

Explanation:

Amazon CloudFront uses the entire URL, including the query string, to form the cache key. Inconsistent parameter ordering and case variations (e.g., ?a=1&B=2 vs. ?b=2&a=1) result in different cache keys for logically identical requests, causing cache misses and a low cache hit ratio.
By deploying a Lambda@Edge function triggered by the viewer request event, the query string can be normalized before CloudFront checks its cache. The function can programmatically sort the query parameters alphabetically and convert them to a consistent case (e.g., lowercase). This ensures that all equivalent requests generate the same cache key, significantly increasing the cache hit ratio and improving performance.

Why Incorrect Options are Wrong:

B. Disabling caching based on query strings would likely break the application's functionality, as it would serve the same cached content for requests that should be different (e.g., search results).
C. A reverse proxy placed after the load balancer (at the origin) would process requests only after a CloudFront cache miss has already occurred, so it cannot improve the cache hit ratio.
D. CloudFront does not have a native feature for case-insensitive query string processing.

Furthermore, this would not address the problem of inconsistently ordered query string parameters.

References:

1. AWS Documentation - Lambda@Edge Example Functions: The "Cache-key normalization" example provides a specific Lambda@Edge function to "change the order of query strings into a fixed order (for example, alphabetical order) and to make them all lowercase." This directly matches the solution in option A.
Source: AWS Documentation, Lambda@Edge example functions, Section: "Cache-key normalization".
2. AWS Documentation - Invoking Lambda@Edge functions in response to events: This document explains the different CloudFront events. The viewer request event is explicitly for modifying the request before it's checked against the CloudFront cache, which is essential for normalizing the cache key.
Source: AWS Documentation, Amazon CloudFront Developer Guide, Section: "Using Lambda@Edge with CloudFront Invoking Lambda@Edge functions in response to events".
3. AWS Documentation - Controlling the cache key: This documentation details the components of the default cache key, which includes "Query strings in the viewer request." This confirms why variations in the query string lead to cache misses.
Source: AWS Documentation, Amazon CloudFront Developer Guide, Section: "Working with cache policies Controlling the cache key".
4. AWS Compute Blog - How to normalize query string parameters to improve your cache hit ratio on Amazon CloudFront: This official blog post presents the exact problem described in the question and provides a step-by-step guide to implementing the solution using a Lambda@Edge function on the viewer-request trigger to sort and lowercase query parameters.
Source: AWS Compute Blog, "How to normalize query string parameters to improve your cache hit ratio on Amazon CloudFront", Nov 12, 2021.

Question: 34

A solutions architect has implemented a SAML 2 0 federated identity solution with their company's
on-premises identity provider (IdP) to authenticate users' access to the AWS environment. When the
solutions architect tests authentication through the federated identity web portal, access to the AWS
environment is granted However when test users attempt to authenticate through the federated identity web portal, they are not able to access the AWS environment
Which items should the solutions architect check to ensure identity federation isproperly configured?
(Select THREE)

A. The 1AM user's permissions policy has allowed the use of SAML federation for that user

B. The 1AM roles created for the federated users' or federated groups' trust policy have set the SAML
provider as the principal

C. Test users are not in the AWSFederatedUsers group in the company's IdP

D. The web portal calls the AWS STS AssumeRoleWithSAML API with the ARN of the SAML provider,
the ARN of the 1AM role, and the SAML assertion from IdP

E. The on-premises IdP's DNS hostname is reachable from the AWS environment VPCs

F. The company's IdP defines SAML assertions that properly map users or groups in the company to
1AM roles with appropriate permissions

Answer:

B, D, F

Explanation:

The scenario describes a user-specific failure in a SAML 2.0 federation setup. Since the architect's access works, the fundamental connection between the on-premises Identity Provider (IdP) and AWS is functional. The issue likely lies in the configuration specific to the test users. The three critical checkpoints for this scenario are:
1. IdP Assertion Mapping (F): The on-premises IdP must be configured to generate a SAML assertion that includes claims mapping the authenticated test users (or their groups) to a specific IAM role. A failure here means AWS never receives the instruction to grant the test users a role.
2. IAM Role Trust Policy (B): The destination IAM role must have a trust policy that explicitly

designates the SAML provider as a trusted principal. This allows the role to be assumed by users authenticated by that IdP.
3. STS API Call (D): The federated web portal must correctly use the SAML assertion from the IdP to call the AssumeRoleWithSAML API, passing the correct role ARN and provider ARN. This is the action that exchanges the assertion for temporary AWS credentials.

Why Incorrect Options are Wrong:

A. SAML federation is for authenticating external identities, not for granting permissions to existing IAM users. This concept is misapplied.
C. The specific group name "AWSFederatedUsers" is arbitrary. The core issue is the correct mapping of any group to a role, not membership in a specifically named group.
E. The authentication flow does not require AWS to connect to the on-premises IdP. The user's browser mediates the communication between the IdP and AWS endpoints.

References:

1. AWS Identity and Access Management User Guide. "Configuring a role for SAML 2.0 federation." This guide specifies that the role's trust policy must identify the SAML provider as the principal. This supports option B. The example policy shows: "Principal": "Federated": "arn:aws:iam::ACCOUNT-ID-WITHOUT-HYPHENS:saml-provider/PROVIDER-NAME" .
2. AWS Identity and Access Management User Guide. "Configuring SAML assertions for the authentication response." This section details the required claims within the SAML assertion, such as https://aws.amazon.com/SAML/Attributes/Role, which maps the user to an IAM role and SAML provider. This directly supports option F.
3. AWS Security Token Service API Reference. "AssumeRoleWithSAML." The documentation for this API action confirms that it is the mechanism used to obtain temporary security credentials using a SAML assertion. It requires the RoleArn, PrincipalArn (the SAML provider ARN), and the SAMLAssertion as parameters, which supports the check described in option D.
4. AWS Security Blog. "How to Implement Federated API and CLI Access Using SAML 2.0 and AD FS." The blog post outlines the end-to-end flow, stating, "The application then calls the AWS STS AssumeRoleWithSAML API and passes the SAML assertion, the ARN of the SAML provider, and the ARN of the IAM role to assume." This confirms the process described in options B, D, and F.

Question: 35

A company has automated the nightly retraining of its machine learning models by using AWS Step
Functions. The workflow consists of multiple steps that use AWS Lambda Each step can fail for various reasons and any failure causes a failure of the overall workflow
A review reveals that the retraining has failed multiple nights in a row without the company noticing
the failure A solutions architect needs to improve the workflow so that notifications are sent for all types of failures in the retraining process
Which combination of steps should the solutions architect take to meet these requirements? (Select
THREE)

A. Create an Amazon Simple Notification Service (Amazon SNS) topic with a subscription of type "Email" that targets the team's mailing list.

B. Create a task named "Email" that forwards the input arguments to the SNS topic

C. Add a Catch field all Task Map. and Parallel states that have a statement of "Error Equals": "States. ALL" and "Next": "Email".

D. Add a new email address to Amazon Simple Email Service (Amazon SES). Verify the email address.

E. Create a task named "Email" that forwards the input arguments to the SES email address

F. Add a Catch field to all Task Map, and Parallel states that have a statement of "Error Equals": "states. Runtime" and "Next": "Email".

Answer:

A, B, C

Explanation:

This solution establishes a complete and robust failure notification system using native AWS Step Functions capabilities. First, an Amazon SNS topic is created with an email subscription to act as the notification endpoint (A). Next, a Task state is defined within the Step Functions workflow to publish a message to this SNS topic (B). Finally, the core error handling is implemented by adding a Catch field to the relevant states. Using the reserved name States.ALL as the error type ensures that any type of error is caught. The Catch block then transitions the workflow to the notification Task state, ensuring a notification is sent for any failure (C).

Why Incorrect Options are Wrong:

D. Using Amazon SES is less idiomatic for simple notifications; Amazon SNS is the purpose-built service for pub/sub messaging and fan-out notifications to various endpoints.
E. This option proposes using SES, which is less direct than the optimized integration Step Functions has with SNS for sending notifications upon state changes or failures.
F. This Catch statement is too specific. States.Runtime would only catch a subset of possible errors, failing to meet the requirement to notify on all types of failures.

References:

1. AWS Step Functions Developer Guide, "Error handling in Step Functions": This document explains the use of Retry and Catch fields. It explicitly states, "To catch all errors, use the reserved name States.ALL." This supports option C as the correct method for catching all failures.
2. AWS Step Functions Developer Guide, "Service integration patterns" and "Manage Amazon SNS with Step Functions": This section details how to integrate Step Functions with other AWS services. It shows the syntax for a Task state that calls the sns:Publish API action, which directly supports the implementation described in option B.
3. Amazon Simple Notification Service Developer Guide, "Subscribing to an Amazon SNS topic": This guide describes how to create subscriptions for an SNS topic. It lists "Email" as a supported protocol, confirming that an SNS topic can directly send notifications to a team's mailing list, as required by option A.