Question 1

A company is transitioning its vital applications from a local data center to AWS. They currently utilize a Microsoft SQL Server Always On cluster on-premises and wish to switch to a managed database service provided by AWS. The solutions architect is tasked with orchestrating a heterogeneous migration of the database to AWS.

A. Transition the SQL Server database to Amazon RDS for MySQL using standard backup and restore methods.

B. Employ AWS Snowball Edge Storage Optimized to move the data to Amazon S3, set up Amazon RDS for MySQL, and utilize SQL Server capabilities such as BULK INSERT with S3.

C. Convert the database schema using the AWS Schema Conversion Tool for Amazon RDS for MySQL compatibility, then migrate the database contents using AWS Database Migration Service (AWS DMS).

D. Migrate the database content over the network using AWS DataSync to Amazon S3, establish an Amazon RDS for MySQL database, and incorporate SQL Server functionalities like BULK INSERT with S3.

The correct answer is: C

Convert the database schema using the AWS Schema Conversion Tool for Amazon RDS for MySQL compatibility, then migrate the database contents using AWS Database Migration Service (AWS DMS).

This solution is most suitable as it addresses the heterogeneous nature of the migration (from Microsoft SQL Server to MySQL). The AWS Schema Conversion Tool is specifically designed to facilitate schema conversion when migrating from one type of database to a different type, ensuring compatibility with the target AWS managed database service. After schema conversion, AWS DMS can be used to efficiently migrate the data. This approach reduces the complexity of the migration process and helps ensure a smooth transition to Amazon RDS.

Why the Other Answers are Incorrect:

A. Simply using backup and restore utilities may not account for differences between Microsoft SQL Server and MySQL databases, potentially resulting in compatibility issues with the database schema and data types.

B. While AWS Snowball Edge is an efficient way to transfer large amounts of data, it doesn't address schema conversion for a heterogeneous migration. Moreover, using BULK INSERT

with Amazon S3 is not a direct feature of Amazon RDS for MySQL and would require additional data transformation steps.

D. AWS DataSync is an effective tool for data transfer but, like option B, does not provide a solution for database schema conversion necessary for a heterogeneous migration. The additional step of using BULK INSERT with Amazon S3 also adds complexity and does not align with RDS for MySQL capabilities.

Question 2

A publishing company has a design team that frequently updates icons and static assets for an e-commerce web application. These assets are hosted on an Amazon S3 bucket within the company's production AWS account. The design team, which has access to a separate development AWS account, tests the static assets there before they need to be transferred to the S3 bucket in the production account. The solutions architect is tasked with ensuring that the design team can upload assets to the production account's S3 bucket without granting them access that could potentially alter other components of the web application. (Choose three)

A. Within the production account, formulate a new IAM policy granting permissions for both retrieving and uploading content to the specified S3 bucket.

B. Within the development account, establish a new IAM policy providing permissions for accessing and modifying the S3 bucket in the production account.

C. In the production account, set up an IAM role and associate it with the newly created policy. Designate the development account as a trusted entity for this role.

D. In the development account, generate an IAM role and attach it to the new policy. Specify the production account as a trusted entity for this role.

E. In the development account, form a group comprising the design team's IAM users. Assign an IAM policy to this group that permits the sts:AssumeRole action on the role created in the production account.

F. In the development account, assemble a group including the design team's IAM users. Attach an IAM policy to this group authorizing the sts:AssumeRole action on a role within the development account.

**The correct combination of steps is:

A, C, E.

These steps collectively provide the design team with the necessary access to the production S3 bucket without exposing other parts of the application to risk.

Step A creates the required IAM policy in the production account that specifically allows actions on the S3 bucket.

Step C involves creating an IAM role in the production account that is associated with the policy from Step A, which enables designated IAM entities from the development account to assume this role.

Finally, Step E establishes a group in the development account for the design team members and attaches a policy to this group that allows them to assume the production account role, thereby granting them the needed access to the S3 bucket.

Why the Other Answers are Incorrect:

B. There's no need to create an IAM policy in the development account that grants access to the production S3 bucket because access should be managed through assumed roles for better security and cross-account access management.

D. It's not necessary to create a role in the development account that trusts the production account. The trust relationship is typically set up in the opposite direction, with the production account allowing a role from the development account to assume it.

F. This step is incorrect because it specifies allowing sts:AssumeRole on a role within the development account, which is not relevant to the requirement of accessing the production account's S3 bucket. The need is to allow the development account's design team to assume a role in the production account, not in their own account.

Question 3

A company has created an initial version of an application using AWS Elastic Beanstalk and Java. To minimize expenses during the development phase, the application was set up in a single-instance environment. However, performance testing has shown that the application has high CPU usage, often exceeding 85%, leading to performance constraints. The solutions architect is tasked with addressing these performance issues efficiently before the application's production release.

A. Deploy a new Elastic Beanstalk application with a load-balanced environment. Include all available Availability Zones and implement a scaling action to add more instances when CPU usage exceeds 85% for a continuous period of 5 minutes.

B. Set up an additional Elastic Beanstalk environment and use the traffic splitting deployment strategy to distribute a specified portion of the traffic to this new environment when the CPU

usage goes beyond 85% for a duration of 5 minutes.

C. Adjust the current environment's settings to convert it into a load-balanced environment across all available Availability Zones and introduce a scaling trigger that adds more instances when the average CPU load remains above 85% for 5 minutes.

D. Use the "Rebuild environment" option in Elastic Beanstalk with load balancing enabled, ensure it encompasses an Availability Zone, and configure a scaling policy to add resources when the total CPU load surpasses 85% for a 5-minute window.

The correct answer is: C.

Adjust the current environment's settings to convert it into a load-balanced environment across all available Availability Zones and introduce a scaling trigger that adds more instances when the average CPU load remains above 85% for 5 minutes.

This solution offers the least operational overhead by simply modifying the existing environment rather than creating new environments or applications. It leverages Elastic Beanstalk's ability to switch from a single-instance environment to a load-balanced, auto-scaling environment, which is designed to handle increased load by distributing traffic and scaling horizontally as needed.

Why the Other Answers are Incorrect:

A. Creating a new Elastic Beanstalk application introduces unnecessary complexity and overhead. It requires setting up a new application and potentially migrating data or configurations from the existing one, which is more work than modifying the existing environment.

B. Introducing a second Elastic Beanstalk environment with traffic splitting doesn't directly address the high CPU utilization issue. This option is more appropriate for blue/green deployment strategies and not for scaling to meet performance demands.

D. The "Rebuild environment" action in Elastic Beanstalk is typically used to recover from an environment configuration issue or to return to a clean state. It is not meant for scaling purposes and would not provide the desired outcome of automatically scaling based on CPU utilization. Moreover, specifying "an" Availability Zone, as opposed to "all" Availability Zones, does not fully utilize the scaling and high availability potential of a load-balanced environment.

Question 4

A finance company operates a vital application on Linux EC2 instances in AWS, which includes a self-managed MySQL database with high I/O demands. The application performs

adequately under regular traffic conditions but experiences slowdowns during peak usage at the end of each month, despite using Elastic Load Balancers and Auto Scaling.

A. Activate Elastic Load Balancer pre-warming, switch to larger EC2 instances, and convert all attached EBS volumes to the GP2 type.

B. Migrate the database to Amazon RDS once and scale horizontally by adding more read replicas for handling the increased load at month-end.

C. Set up a CloudWatch and AWS Lambda system to modify the EC2 instances' EBS volume types, sizes, or IOPS based on specific CloudWatch metrics.

D. Swap out current EBS volumes for new Provisioned IOPS volumes with the highest available storage capacity and IOPS, by creating snapshots before the monthly peak and reverting once it's over.

The correct answer is: **B.

Migrate the database to Amazon RDS once and scale horizontally by adding more read replicas for handling the increased load at month-end.

This option is the most effective as it leverages Amazon RDS, a managed database service that can easily scale to meet increased demand. Creating additional read replicas can distribute the heavy read load during intensive reporting periods, thereby minimizing the impact on performance. This managed approach also reduces operational overhead associated with database maintenance and scaling.

Why the Other Answers are Incorrect:

A. While pre-warming the Elastic Load Balancers and using bigger EC2 instances may help to some extent, simply converting EBS volumes to GP2 doesn't ensure they can handle the I/O demands of month-end reporting. GP2 volumes have baseline performance and burst capabilities that might still fall short during peak times.

C. Configuring CloudWatch and Lambda to dynamically change EBS volume configurations might introduce complexity and risks of downtime due to the volume modifications. It also requires careful planning and testing to ensure the automated adjustments align with the performance needs without unintended consequences.

D. Using the highest available Provisioned IOPS (PIOPS) volumes for the duration of the peak period may improve I/O performance, but it's a costly solution that also involves the risk and operational overhead of swapping volumes and managing snapshots. This solution could be less cost-effective and might result in downtime during volume changes, which is not ideal for a business-critical application.

Question 5

A company is running a Java application on VMs in its data center. This application, characterized by its complex dependencies and stability, is due for modernization. The company aims to move this application to AWS while reducing the need for extensive server maintenance and keeping code modifications to a minimum.

A. Transition the application to Amazon Elastic Container Service (ECS) on AWS Fargate using AWS App2Container. Utilize Amazon Elastic Container Registry (ECR) for storing container images, ensuring the ECS task execution role has access to the ECR repository. Set up ECS with an Application Load Balancer (ALB) to facilitate interaction with the application.

B. Convert the application code into a containerized format compatible with AWS Lambda, and create a REST API using Amazon API Gateway with Lambda integration to communicate with the application.

C. Migrate the application to Amazon Elastic Kubernetes Service (EKS) on EKS managed node groups, again utilizing AWS App2Container. Store the container images in Amazon ECR, providing EKS nodes access to the ECR repository. Employ Amazon API Gateway for application interaction.

D. Containerize the application code for AWS Lambda and configure Lambda to operate with an Application Load Balancer (ALB), using the ALB as the interface for the application.

The correct answer is: A.

A. Transition the application to Amazon Elastic Container Service (ECS) on AWS Fargate using AWS App2Container. Utilize Amazon Elastic Container Registry (ECR) for storing container images, ensuring the ECS task execution role has access to the ECR repository. Set up ECS with an Application Load Balancer (ALB) to facilitate interaction with the application.

This solution is the most suitable as it enables a seamless migration of the existing Java application to a modern, containerized environment with minimal code changes. AWS App2Container helps in automating the containerization of existing applications, making it easier to shift to AWS with reduced effort. ECS on AWS Fargate offers a serverless compute engine for containers, eliminating the need for server maintenance. The use of ALB ensures efficient traffic management and load balancing.

Why the Other Answers are Incorrect:

B. AWS Lambda is designed for serverless, event-driven architectures and has limitations in terms of runtime and memory, making it less suitable for complex applications with heavy dependencies. Also, migrating to Lambda would likely require significant code changes, which contradicts the requirement to minimize code modifications.

C. While EKS is a robust solution for containerized applications, it involves more operational complexity compared to ECS on Fargate. Managing Kubernetes clusters, even with managed node groups, typically requires more administrative work than using a fully managed service like ECS on Fargate.

D. Running containerized applications directly on AWS Lambda is not a typical use case and would likely necessitate considerable changes to the application code. Lambda functions are designed for short-lived, event-driven processes, not for hosting traditional applications like a Java web application with complex dependencies.

Question 6

A company operates an asynchronous HTTP application using AWS Lambda, triggered by a public Amazon API Gateway endpoint, both located in the us-east-1 Region. To enhance the application's resilience, a solutions architect is tasked with implementing a failover mechanism to another AWS Region.

A. Set up a secondary API Gateway endpoint in the us-west-2 Region that redirects traffic to the existing Lambda function in us-east-1. Utilize Amazon Route 53 with a failover routing policy to manage traffic between the two API Gateway endpoints.

B. Implement an Amazon Simple Queue Service (SQS) queue and configure the API Gateway to send traffic to this SQS queue instead of directly to the Lambda function. Adjust the Lambda function to process messages from the SQS queue.

C. Replicate the Lambda function in the us-west-2 Region and create a corresponding API Gateway endpoint there to handle traffic for this new Lambda instance. Use AWS Global Accelerator and an Application Load Balancer to control traffic distribution between the two API Gateway endpoints.

D. Deploy the Lambda function along with an API Gateway endpoint in the us-west-2 Region. Configure Amazon Route 53 with a failover routing policy to manage traffic across both API Gateway endpoints in the two regions.

The correct answer is: D.

D. Deploy the Lambda function along with an API Gateway endpoint in the us-west-2 Region. Configure Amazon Route 53 with a failover routing policy to manage traffic across both API Gateway endpoints in the two regions.

This approach is the most effective for setting up regional failover. By deploying the Lambda function and its associated API Gateway endpoint in a second region (us-west-2), the application gains regional redundancy. Amazon Route 53 with failover routing can then intelligently route traffic to the primary region (us-east-1) under normal circumstances and

automatically switch to the secondary region (us-west-2) if the primary becomes unavailable, ensuring continuous availability.

Why the Other Answers are Incorrect:

A. While creating an additional API Gateway endpoint in another region could help with regional failover, redirecting traffic from us-west-2 to the Lambda function in us-east-1 does not provide true regional failover for the Lambda function itself. In the event of a regional issue in us-east-1, both the API Gateway and Lambda function would be impacted.

B. Introducing SQS and changing the architecture to a queue-based model adds unnecessary complexity and does not address the regional failover requirement for the Lambda function. This setup would still be vulnerable to regional issues in us-east-1.

C. While deploying the Lambda function in a second region and using AWS Global Accelerator with an Application Load Balancer offers a robust traffic management solution, it introduces more complexity and potential cost than necessary. Amazon Route 53 with a failover routing policy is a simpler and more cost-effective solution for managing regional failover between API Gateway endpoints.

Question 7

A retail company, structured under AWS Organizations, has established consolidated billing and organized its various departments into Organizational Units (OUs) such as Finance, Sales, HR, Marketing, and Operations. Each OU contains multiple AWS accounts for different environments like development, test, pre-production, and production. The HR department, anticipating the launch of a new system in three months, has acquired Reserved Instances (RIs) within its production AWS account for the upcoming application. The HR department needs to ensure that these RI discounts are exclusive to its account and not shared with other departments.

A. Disable RI sharing in the AWS Billing and Cost Management console of the HR department's production account.

B. Detach the HR department's production AWS account from the organization and include it solely within the consolidated billing arrangement.

C. Utilize the organization’s management account in the AWS Billing and Cost Management console to deactivate RI Sharing for the HR department's production AWS account.

D. Implement a Service Control Policy (SCP) within the organization to restrict access to the RIs, applying it across OUs of all departments except HR.

The correct answer is: C.

C. Utilize the organization’s management account in the AWS Billing and Cost Management console to deactivate RI Sharing for the HR department's production AWS account.

This option is most appropriate because the management of Reserved Instance sharing settings across an AWS Organization is centrally controlled through the organization’s management account. By turning off RI sharing for the HR department's production account within the management account's Billing and Cost Management console, the HR department can ensure that its RI discounts are not extended to other departments' accounts, thus meeting the requirement.

Why the Other Answers are Incorrect:

A. The ability to control RI sharing is not available at the individual account level within AWS Organizations. This setting is managed at the organization level through the management account. Therefore, turning off RI sharing in the HR department's production account is not feasible.

B. Removing the HR department's production account from the organization to manage RI sharing is not necessary and could lead to additional complexity and loss of organizational control benefits. It is more efficient to manage RI sharing through the organization’s management account without detaching the individual account.

D. Service Control Policies (SCPs) are used to manage permissions in AWS Organizations and cannot be used to control Reserved Instance sharing. RI sharing is a billing concept, not a permissions issue, so SCPs are not applicable in this context.

Question 8

A prominent web application operated by a large company is hosted on several Amazon EC2 Linux instances. These instances, part of an Auto Scaling group, are located in a private subnet and targeted by an Application Load Balancer. The setup includes AWS Systems Manager Session Manager, with the Systems Manager Agent active on all EC2 instances. After a recent application update, some instances are being flagged as unhealthy and subsequently terminated, reducing the application's capacity. The solutions architect, unable to pinpoint the issue through Amazon CloudWatch logs, needs to access an unhealthy EC2 instance for direct troubleshooting.

A. Temporarily halt the HealthCheck process of the Auto Scaling group. Then utilize Session Manager for accessing an EC2 instance that has been labeled unhealthy.

B. Activate termination protection on the EC2 instances. Proceed to log into an unhealthy instance using Session Manager.

C. Adjust the Auto Scaling group's termination policy to target the oldest instances first. Subsequently, access an unhealthy instance via Session Manager.

D. Pause the Terminate process in the Auto Scaling group. Employ Session Manager to log into an EC2 instance identified as unhealthy.

The correct answer is: D.

D. Pause the Terminate process in the Auto Scaling group. Employ Session Manager to log into an EC2 instance identified as unhealthy.

This approach is the most suitable for troubleshooting. By suspending the Auto Scaling group's Terminate process, the solutions architect can prevent the automatic termination of unhealthy instances. This allows sufficient time to use Session Manager for securely accessing an unhealthy instance without needing to open inbound ports or manage SSH keys, thus facilitating direct investigation of the issue.

Why the Other Answers are Incorrect:

A. Suspending the HealthCheck process of the Auto Scaling group does not prevent the termination of instances already marked as unhealthy. This action only stops the marking of new instances as unhealthy, which is not helpful for accessing and investigating instances already affected.

B. Enabling termination protection on EC2 instances is generally used to prevent instances from being terminated due to manual scaling activities or by certain Auto Scaling processes. However, it does not specifically target or protect instances that have already been marked as unhealthy, which are the focus for troubleshooting.

C. Changing the termination policy to 'OldestInstance' will influence which instances Auto Scaling selects for termination during scale-in activities. However, it does not directly help in accessing unhealthy instances for troubleshooting purposes. The policy change does not prevent the termination of instances already marked as unhealthy.

Question 9

A company, operating under AWS Organizations with accounts organized into different Organizational Units (OUs), seeks to implement an AWS WAF solution that can be centrally managed across these various accounts. The requirement is for a system where administrators can easily include or exclude specific accounts or OUs from the AWS WAF rule sets. Additionally, there should be a mechanism to automatically update and correct any AWS WAF rules that are not in compliance across all accounts.

A. Implement AWS Firewall Manager for centralized management of AWS WAF rules. Utilize AWS Systems Manager Parameter Store to maintain a list of account numbers and OUs. Modify this list as needed to manage account or OU inclusion. Set up an Amazon EventBridge rule to detect changes in the parameter store and trigger an AWS Lambda function, which then updates the security policy in the Firewall Manager's administrative account.

B. Establish an AWS Config rule across the organization, mandating that all resources in specified OUs adhere to the AWS WAF rules. Implement automated remediation using AWS Lambda for resources that don't comply. Deploy AWS WAF rules using AWS CloudFormation stack sets, targeting the same OUs as the AWS Config rule.

C. Define AWS WAF rules in the management account of AWS Organizations. Store and manage account numbers and OUs through AWS Lambda environment variables. Adjust these variables to modify account or OU participation. Implement cross-account IAM roles in member accounts and use AWS STS within a Lambda function for cross-account creation and updating of AWS WAF rules.

D. Utilize AWS Control Tower for the organization-wide management of AWS WAF rules. Employ AWS Key Management Service (KMS) to keep track of account numbers and OUs. Update AWS KMS for changes in account or OU participation. Create IAM users in member accounts, allowing AWS Control Tower in the management account to use the access keys for updating AWS WAF rules in member accounts.

The correct answer is: A.

A. Implement AWS Firewall Manager for centralized management of AWS WAF rules. Utilize AWS Systems Manager Parameter Store to maintain a list of account numbers and OUs. Modify this list as needed to manage account or OU inclusion. Set up an Amazon EventBridge rule to detect changes in the parameter store and trigger an AWS Lambda function, which then updates the security policy in the Firewall Manager's administrative account.

This solution is the most effective and involves the least operational overhead. AWS Firewall Manager is designed for centralized management of AWS WAF rules across multiple AWS accounts, making it ideal for organizations with multiple accounts and OUs. Using the Systems Manager Parameter Store to keep a dynamic list of accounts and OUs, combined with EventBridge and Lambda for automated updates, provides a streamlined and flexible approach to manage and enforce compliance of AWS WAF rules.

Why the Other Answers are Incorrect:

B. AWS Config can enforce compliance but is not specifically designed for the centralized management of AWS WAF rules. The use of CloudFormation stack sets for deploying WAF

rules adds unnecessary complexity compared to the direct management capabilities of AWS Firewall Manager.

C. Managing AWS WAF rules through Lambda environment variables and cross-account IAM roles introduces significant operational complexity and manual overhead. This method requires more maintenance and does not leverage the centralized management capabilities of AWS Firewall Manager.

D. AWS Control Tower is primarily used for setting up and governing a secure, multi-account AWS environment and is not specifically tailored for the management of AWS WAF rules. The use of AWS KMS for storing account numbers and OUs, along with creating IAM users for cross-account access, adds unnecessary complexity and does not align with the intended use of these services for centralized WAF rule management.

Question 10

A company utilizes an AWS Lambda function to fetch recent updates from an Amazon Aurora database within the same VPC. The Lambda function processes the data and stores it in an Amazon S3 bucket with server-side encryption using AWS KMS keys. Currently, the database credentials are provided to the Lambda function through environment variables. The company requires that the data never traverse the public Internet and seeks a method to reduce risks associated with potentially compromised database credentials.

A. Implement IAM database authentication for the Aurora database. Update the Lambda function's IAM role to enable database access via IAM authentication. Introduce a gateway VPC endpoint for Amazon S3 in the same VPC.

B. Activate IAM database authentication for the Aurora database. Modify the Lambda function's IAM role for database access through IAM authentication. Ensure that data transfers to Amazon S3 occur over HTTPS.

C. Store the database credentials in AWS Systems Manager Parameter Store and configure credential rotation. Update the Lambda function's IAM role for access to Parameter Store. Alter the Lambda function to obtain credentials from Parameter Store. Add a gateway VPC endpoint for Amazon S3 in the VPC.

D. Place the database credentials in AWS Secrets Manager and enable credential rotation. Amend the Lambda function's IAM role to access Secrets Manager. Update the Lambda function to retrieve credentials from Secrets Manager. Require HTTPS for data transfers to Amazon S3.

The correct answer is: A.

A. Implement IAM database authentication for the Aurora database. Update the Lambda function's IAM role to enable database access via IAM authentication. Introduce a gateway VPC endpoint for Amazon S3 in the same VPC.

This solution aligns with the requirements by eliminating the need to store database credentials in the Lambda environment, thereby minimizing the impact of any potential credential compromise. IAM database authentication provides a secure and direct method for the Lambda function to access the Aurora database. The addition of a gateway VPC endpoint for S3 ensures that data transfers to the S3 bucket occur within the AWS network, not traversing the public Internet.

Why the Other Answers are Incorrect:

B. While enabling IAM database authentication and ensuring HTTPS for S3 transfers enhances security, this option doesn't include a gateway VPC endpoint for S3. Without the VPC endpoint, data transfer to S3 might still traverse the public Internet, which contradicts the requirement.

C. Storing credentials in Systems Manager Parameter Store with rotation improves security, but using IAM database authentication (as in option A) is a more streamlined solution. Additionally, option C does not specify enforcing HTTPS for data transfers to Amazon S3, leaving potential for data to travel over the public Internet.

D. Like option C, storing credentials in AWS Secrets Manager with rotation is a secure approach, but it adds unnecessary complexity compared to using IAM database authentication. Also, this option does not include a gateway VPC endpoint for Amazon S3, potentially allowing data to traverse the public Internet.

Question 11

A mobile gaming company has transitioned its entire on-premises infrastructure to AWS. A solutions architect, tasked with ensuring that the cloud setup adheres to the Well-Architected Framework, notices from the monthly AWS Cost Explorer reports that frequent launches and terminations of large EC2 instances by developers have led to significant costs. These instances are not in line with the required instance types for their development testing. To address this, the solutions architect needs to establish a mechanism that restricts the developers to launching only specific, permitted EC2 instance types.

A. Implement a managed AWS Config rule named 'desired-instance-type' that lists the permissible instance types. Set this rule to execute whenever a new EC2 instance is initiated.

B. Create an EC2 launch template specifying the allowed instance types and associate this template with the developers' IAM accounts.

C. Draft a new IAM policy that explicitly states the allowed instance types. Attach this policy to an IAM group comprising the developer IAM accounts.

D. Utilize EC2 Image Builder to assist the developers in creating a standard image pipeline and a golden image for their use.

The correct answer is: C.

C. Draft a new IAM policy that explicitly states the allowed instance types. Attach this policy to an IAM group comprising the developer IAM accounts.

This approach is effective as it directly restricts the actions of the developers' IAM accounts based on the specified instance types in the IAM policy. By attaching the policy to an IAM group that includes all developer accounts, the solutions architect can enforce the appropriate instance type usage, thereby controlling costs and adhering to the planned infrastructure design.

Why the Other Answers are Incorrect:

A. While AWS Config can monitor and report on compliance with specified configurations, it is not designed to proactively prevent actions like launching EC2 instances of certain types. It can alert on non-compliance but cannot enforce restrictions at the time of instance creation.

B. Creating an EC2 launch template with specific instance types is a good practice, but it does not enforce restrictions. Developers could still launch instances outside of the template, so this approach wouldn't guarantee compliance.

D. EC2 Image Builder is a service for automating the creation of customized machine images, but it doesn't provide direct control over the types of instances that developers can launch. It’s more about building and maintaining images rather than enforcing instance type usage policies.

Question 12

A company operating several projects on AWS has them spread across multiple AWS accounts under a single AWS Organizations umbrella. The company needs to ensure cloud infrastructure costs are accurately attributed to the appropriate project. However, they've encountered an issue with some Amazon EC2 instances not being appropriately tagged with a 'Project' tag, essential for cost allocation. To address and prevent this tagging oversight, the solutions architect needs to take specific steps (choose three).

A. Implement an AWS Config rule within each AWS account to identify resources that are missing the 'Project' tag.

B. Establish a Service Control Policy (SCP) at the organization level that denies the action 'ec2:RunInstances' if the 'Project' tag is absent.

C. Utilize Amazon Inspector across the organization to detect resources lacking the necessary tags.

D. Formulate an IAM policy for each account that denies the action 'ec2:RunInstances' when the 'Project' tag is missing.

E. Set up an AWS Config aggregator at the organization level to compile a list of EC2 instances missing the 'Project' tag.

F. Employ AWS Security Hub to consolidate information about EC2 instances that are not tagged with the 'Project' tag.

The correct answer is: A, B, E.

These actions collectively provide a comprehensive approach to both identifying untagged resources and enforcing tagging compliance for future resource creation.

A. Using AWS Config rules in each account to identify missing tags is effective for ongoing monitoring and compliance assessment, ensuring all resources are correctly tagged.

B. Implementing an SCP that denies the launching of EC2 instances without the required 'Project' tag ensures compliance at the organization level, preventing future instances from being created without the necessary tagging.

E. Creating an AWS Config aggregator at the organization level allows for centralized visibility into tagging compliance across all accounts, making it easier to track and manage instances missing the 'Project' tag.

Why the Other Answers are Incorrect:

C. Amazon Inspector is a service designed for security vulnerability assessments and is not specialized in identifying missing tags on resources.

D. While an IAM policy in each account could theoretically enforce tagging, it's not as effective as using SCPs for organization-wide enforcement. SCPs provide centralized control and are more suitable for enforcing policies across multiple accounts within AWS Organizations.

F. AWS Security Hub is focused on security and compliance findings across AWS accounts and is not specifically tailored for managing tagging practices or identifying untagged resources like EC2 instances.

Question 13

A company currently operates an on-premises monitoring system with a PostgreSQL database that is struggling to scale and frequently encounters storage capacity issues. The company seeks to develop a hybrid monitoring solution integrated with AWS, utilizing a pre-existing VPN connection between their network and AWS. Their solution criteria include the use of managed AWS services to reduce complexity, an auto-scaling buffer for data throughput with minimal maintenance, a tool for near-real-time event visualization, and compatibility with semi-structured JSON data and dynamic schemas. (Choose two)

A. Implement Amazon Kinesis Data Firehose as a scalable buffer for events, and use an AWS Lambda function for event processing and transformation.

B. Set up an Amazon Kinesis data stream as a scalable buffer for events, and employ an AWS Lambda function to process and transform these events.

C. Deploy an Amazon Aurora PostgreSQL DB cluster for event storage, and utilize Amazon QuickSight for near-real-time data visualization and dashboard creation.

D. Utilize Amazon Elasticsearch Service (Amazon ES) for event storage, and leverage its accompanying Kibana endpoint for near-real-time visualization and dashboard creation.

E. Use an Amazon Neptune DB instance for event storage, and integrate Amazon QuickSight for the generation of near-real-time visualizations and dashboards.

The correct answer is: A, D.

A. Amazon Kinesis Data Firehose provides a fully managed, scalable solution for buffering events. It can handle high throughput with minimal administration, meeting the requirements for a scalable buffer. AWS Lambda can be used alongside Kinesis Data Firehose for event processing and transformation, providing flexibility and scalability.

D. Amazon Elasticsearch Service is well-suited for handling semi-structured data like JSON and supports dynamic schemas. It comes with a built-in Kibana endpoint, which is an effective tool for creating near-real-time visualizations and dashboards, aligning with the company's requirements for data visualization.

Why the Other Answers are Incorrect:

B. While Amazon Kinesis data streams are suitable for buffering events, they require more management and scaling configuration compared to Kinesis Data Firehose. Kinesis Data Firehose provides a more hands-off, managed solution, aligning better with the requirement for minimal operational complexity.

C. Amazon Aurora PostgreSQL is a scalable database solution, but it doesn't inherently meet the need for a managed buffer that auto-scales with minimal administration. Additionally, QuickSight, although capable of visualization, may not provide the near-real-time visualization capabilities required for this monitoring solution.

E. Amazon Neptune is designed primarily for graph database use cases and is not the best fit for a monitoring solution focused on semi-structured JSON data. Moreover, it doesn't meet the requirement for an auto-scaling buffer for event data ingestion.

Question 14

A team responsible for handling behavioral data in a company operates within a Multi-AZ VPC setup featuring public and private subnets, along with an internet gateway. Each public subnet is equipped with a NAT gateway. The company primarily utilizes Amazon Kinesis Data Streams for its applications, which predominantly run in private subnets. Tasked with reducing costs while keeping the applications functional, a solutions architect examines the infrastructure using Cost Explorer and identifies that expenses labeled as EC2-Other, particularly those related to NatGateway-Bytes, are notably high. The architect is looking for a strategy to lower these costs.

A. Implement VPC Flow Logs and utilize Amazon Athena to inspect the logs, identifying and eliminating non-essential traffic. Adjust security groups to block traffic contributing to high costs.

B. Incorporate an interface VPC endpoint for Kinesis Data Streams into the VPC. Confirm that the applications possess the necessary IAM permissions to utilize this interface VPC endpoint.

C. Activate VPC Flow Logs and Amazon Detective to investigate non-Kinesis Data Streams related traffic. Modify security groups to prevent this irrelevant traffic.

D. Introduce an interface VPC endpoint for Kinesis Data Streams in the VPC. Verify that the VPC endpoint policy permits traffic from the company's applications.

The correct answer is: D.

D. Introduce an interface VPC endpoint for Kinesis Data Streams in the VPC. Verify that the VPC endpoint policy permits traffic from the company's applications.

This solution directly addresses the high NAT Gateway costs by creating an interface VPC endpoint for Kinesis Data Streams. This endpoint allows applications in the private subnet to interact directly with Kinesis Data Streams without using the NAT Gateway, thus reducing the data transfer costs associated with the NAT Gateway. Ensuring the VPC endpoint policy allows application traffic is crucial for maintaining application functionality.

Why the Other Answers are Incorrect:

A. While analyzing VPC Flow Logs with Amazon Athena can help identify unnecessary traffic, it doesn't address the fundamental issue of reducing NAT Gateway costs associated with Kinesis Data Streams traffic. Blocking high-cost traffic through security groups might inadvertently disrupt essential application functionalities.

B. Adding an interface VPC endpoint for Kinesis Data Streams is a step in the right direction, but merely confirming IAM permissions for the applications doesn't ensure that traffic to Kinesis Data Streams will bypass the NAT Gateway. Ensuring the correct VPC endpoint policy is crucial for routing traffic correctly.

C. Using VPC Flow Logs and Amazon Detective provides insights into traffic patterns and potential security issues but does not offer a direct solution to reduce NAT Gateway costs related to Kinesis Data Streams traffic. The key is to provide a direct path for this traffic within the VPC, avoiding the NAT Gateway.

Question 15

A European retail company with an on-premises data center has AWS infrastructure spread across the eu-west-1 and us-east-1 Regions. The company's goal is to facilitate network traffic flow from its on-premises infrastructure to its VPCs in both AWS Regions and enable direct routing between the VPCs across these Regions. To achieve high availability, the company has two 1 Gbps AWS Direct Connect connections, each leading to different Direct Connect locations (DX-A and DX-B) in Europe. Additionally, each AWS Region has a single AWS Transit Gateway for managing inter-VPC traffic within the Region.

A. Set up a private Virtual Interface (VIF) connecting the DX-A Direct Connect link to a Direct Connect gateway, and similarly for the DX-B connection to the same gateway for redundancy. Associate both the eu-west-1 and us-east-1 Transit Gateways with this Direct Connect gateway. Peer the Transit Gateways to enable cross-Region routing.

B. Establish a transit VIF for the DX-A connection linked to a Direct Connect gateway and associate it with the eu-west-1 Transit Gateway. Then, create another transit VIF for the DX-B connection to a different Direct Connect gateway, associating it with the us-east-1 Transit Gateway. Peer these Direct Connect gateways for high availability and cross-Region routing.

C. Configure a transit VIF for the DX-A connection to a Direct Connect gateway, and do the same for the DX-B connection to the same gateway for failover. Link both the eu-west-1 and us-east-1 Transit Gateways to this Direct Connect gateway. Set up the Direct Connect gateway to handle traffic routing between the Transit Gateways.

D. Implement a transit VIF from the DX-A connection to a Direct Connect gateway, and similarly for the DX-B connection to the same gateway for high availability. Associate both the eu-west-1 and us-east-1 Transit Gateways with this Direct Connect gateway. Peer the Transit Gateways for cross-Region routing.

The correct answer is: D.

D. Implement a transit VIF from the DX-A connection to a Direct Connect gateway, and similarly for the DX-B connection to the same gateway for high availability. Associate both the eu-west-1 and us-east-1 Transit Gateways with this Direct Connect gateway. Peer the Transit Gateways for cross-Region routing.

This solution effectively meets the requirements by using a single Direct Connect gateway connected to both Transit Gateways in different regions. The setup uses transit VIFs for both Direct Connect connections (DX-A and DX-B), ensuring high availability. Peering the Transit Gateways allows for direct routing between VPCs in different AWS Regions, fulfilling the company's need for network traffic flow without single points of failure.

Why the Other Answers are Incorrect:

A. This option uses private VIFs instead of transit VIFs and does not support the required peering between the Transit Gateways for cross-Region routing. Private VIFs are used for connecting to VPCs but do not support the Transit Gateway associations needed for this scenario.

B. While this solution uses transit VIFs, it suggests creating two separate Direct Connect gateways and peering them. This approach is less efficient and more complex compared to using a single Direct Connect gateway associated with both Transit Gateways, as in option D.

C. This option, like D, correctly suggests using a transit VIF and a single Direct Connect gateway for both connections. However, it lacks the crucial step of peering the Transit Gateways, which is necessary for enabling cross-Region routing.

Question 16

A company operating its application on AWS has a policy where the security team must authorize the creation of any new IAM users. To comply with this policy, the procedure is such that whenever a new IAM user is created, their access is immediately revoked, followed by a notification to the security team for approval. This process is supported by a multi-Region AWS CloudTrail trail active in the company's AWS account.

The steps to meet these requirements involve (choose three):

A. Establish an Amazon EventBridge (formerly Amazon CloudWatch Events) rule, configured to detect the 'CreateUser' event as reported by CloudTrail. This rule should trigger when the 'CreateUser' API call is made.

B. Set up CloudTrail to directly alert an Amazon SNS topic whenever a 'CreateUser' event occurs.

C. Trigger an Amazon Elastic Container Service (Amazon ECS) container using AWS Fargate technology to execute the access removal process.

D. Initiate an AWS Step Functions state machine, which is designed to automate the access removal for the new IAM user.

E. Employ Amazon Simple Notification Service (SNS) to inform the security team about the new IAM user creation and pending approval.

F. Utilize Amazon Pinpoint to send a notification to the security team regarding the user creation.

The correct combination of steps is: **A, D, E.**

A. EventBridge rule: This rule will effectively identify the 'CreateUser' event in CloudTrail and initiate the required automated response.

D. AWS Step Functions: This service automates the removal of access for the newly created IAM user, ensuring immediate compliance with the company policy.

E. Amazon SNS: This service reliably delivers the notification to the security team, informing them of the new IAM user creation and prompting their approval action.

Why the Other Answers are Incorrect:

B. CloudTrail itself does not have a built-in mechanism to send notifications directly to SNS. Instead, the integration of CloudTrail with EventBridge is the appropriate approach to detect specific API calls and trigger subsequent actions.

C. Utilizing ECS with Fargate for removing user access is more complex and resource-intensive than necessary. AWS Step Functions offers a more streamlined and efficient solution for orchestrating the access removal process.

F. Amazon Pinpoint is primarily a marketing communications service and is not typically used for internal notification processes like informing a security team. Amazon SNS is better suited for this kind of internal communication due to its direct integration with AWS services and simplicity in sending notifications.

Question 17

A company planning to transition to AWS desires a structured approach encompassing multiple AWS accounts with centralized access control for all applications and accounts, while ensuring network traffic remains private. The structure requires multi-factor authentication (MFA) for user logins and specific roles aligned with user groups. The setup involves distinct accounts for development, staging, production, and a shared network. The production and shared network accounts must connect to all other accounts, whereas the development and staging accounts should only access each other.

The steps to achieve these requirements include (choose three):

A. Implement AWS Control Tower to create a landing zone environment. Utilize it to enroll new accounts and integrate existing accounts into an AWS Organizations structure.

B. Utilize AWS Security Hub across all accounts to oversee cross-account access, relying on AWS CloudTrail to enforce MFA at login.

C. Deploy transit gateways and create transit gateway VPC attachments in each account, setting up appropriate routing configurations via route tables.

D. Set up AWS IAM Identity Center (formerly AWS Single Sign-On) for centralized access management across accounts, creating permission sets with mandatory MFA.

E. Activate AWS Control Tower in all accounts for routing management and enforce MFA at login using CloudTrail findings.

F. Establish IAM users and groups with enforced MFA, and use Amazon Cognito user and identity pools for access management across accounts.

The correct combination of steps is: A, C, D.

A. AWS Control Tower is ideal for setting up a multi-account environment with centralized governance and compliance. It facilitates the creation of a landing zone and simplifies the process of enrolling and managing accounts within AWS Organizations.

C. Transit gateways provide a scalable way to connect different VPCs and accounts, supporting the requirement for a shared network and private connectivity between specific accounts.

D. AWS IAM Identity Center streamlines the process of central access management with MFA and role-based access control, aligning with the company's requirements for secure and managed access across all AWS accounts.

Why the Other Answers are Incorrect:

B. AWS Security Hub is primarily a security monitoring service and does not directly manage cross-account access or enforce MFA logins. Its role is more about aggregating and analyzing security findings rather than access management.

E. AWS Control Tower facilitates account management and governance but is not specifically used for routing management between accounts. The routing requirement is better addressed with transit gateways.

F. While IAM users and groups with MFA provide secure access, and Amazon Cognito offers identity management solutions, these services are more complex for the scenario described than using AWS Control Tower and IAM Identity Center. The latter provides a more integrated and simpler solution for managing multi-account environments with MFA and role assignments.

Question 18

A company operates its application across three separate AWS accounts, each dedicated to a specific environment: development, testing, and production. These environments, hosted in the eu-west-1 Region, utilize stateful Amazon EC2 instances and Amazon RDS for MySQL databases ranging from 500 GB to 800 GB. While the development and testing teams work during standard business hours on business days, the production environment is active continuously. To optimize costs, especially given that all resources are tagged by their respective environments, the company seeks a solution that requires minimal operational effort.

A. Implement an Amazon EventBridge rule to execute daily, triggering an AWS Lambda function. This function should start or stop instances based on the environment tag, day, and time.

B. Set up an Amazon EventBridge rule to activate every business day evening, calling an AWS Lambda function to stop instances based on their environment tag. Also, create another EventBridge rule for business day mornings to invoke a different Lambda function, restarting the instances.

C. Establish an Amazon EventBridge rule for every business day evening, linked to an AWS Lambda function that terminates instances per their tag. Then, configure a second EventBridge rule for business day mornings, associated with another Lambda function that restores instances from the last backup according to the tag.

D. Create an Amazon EventBridge rule scheduled hourly, connected to a single AWS Lambda function. This function should handle the termination or restoration of instances from their last backup, contingent on the tag, day, and time.

The correct answer is: B.

B. Set up an Amazon EventBridge rule to activate every business day evening, calling an AWS Lambda function to stop instances based on their environment tag. Also, create another EventBridge rule for business day mornings to invoke a different Lambda function, restarting the instances.

This solution effectively reduces operational costs by stopping the EC2 instances in the development and testing environments outside of business hours when they are not in use, and then restarting them when needed. This approach offers a balance between cost savings and operational simplicity, as it minimizes unnecessary running costs without the complexity of terminating and restoring instances.

Why the Other Answers are Incorrect:

A. Running a rule once every day may not align well with the specific working hours of the development and testing teams, potentially leading to instances being stopped or started at inappropriate times.

C. Terminating instances every evening and restoring them from backups every morning is operationally complex and potentially risky, especially for stateful instances. This approach could lead to data inconsistencies, longer recovery times, and overall operational inefficiency.

D. Running a rule every hour to terminate or restore instances is excessively frequent and operationally complex, leading to potential disruptions and inefficiencies. This method is more involved than necessary for achieving cost savings.

Question 19

A company offering a SaaS solution on AWS has set up an Amazon API Gateway REST API with AWS Lambda integration, operating in multiple regions within the same production account. The solution features tiered pricing, where the premium tier allows customers up to 3,000 API calls per second, with each customer identified by a unique API key. However, during peak hours, several premium tier customers across various regions are encountering 429 Too Many Requests error responses from multiple API methods, despite logs showing that the Lambda function is not being invoked.

Potential causes for these error messages could be:

A. The Lambda function has reached its concurrency limit.

B. The Lambda function has hit its regional concurrency limit.

C. The overall API Gateway account limit for calls per second has been exceeded.

D. The API Gateway has reached its default per-method limit for calls per second.

The correct answer is: C.

C. The overall API Gateway account limit for calls per second has been exceeded.

This scenario is likely due to the API Gateway account-level limit being surpassed. API Gateway enforces a default maximum rate limit on the number of API calls that can be made per second, regardless of the Lambda function's capacity or concurrency settings. When this threshold is exceeded, API Gateway returns a 429 Too Many Requests error, which aligns with the issue experienced by premium customers during peak usage.

Why the Other Answers are Incorrect:

A. & B. While Lambda functions have concurrency limits, both at the function level and regional level, the error message and logs indicate that the Lambda function is never invoked. This suggests that the issue is occurring at the API Gateway level before the request reaches the Lambda function.

D. The API Gateway's default per-method limit refers to a throttling limit set for individual API methods. However, since the issue is being reported across multiple methods and regions by premium customers who are within their allocated rate of 3,000 calls per second, it is less likely to be an issue with per-method limits and more indicative of a broader account-level limit being reached.

Question 20

A financial organization is in the process of transitioning its web application from an on-premises setup to AWS. This company has been reliant on a third-party security tool for 15 years to scrutinize inbound traffic. However, this tool lacks a cloud-based solution from its provider, raising concerns about its integration with AWS technologies. The organization's plan is to host its application on Amazon EC2 instances within an Auto Scaling group inside a dedicated VPC. It's imperative for the security tool to examine all network traffic to and from the VPC without impacting the application's real-time performance, and the architecture must ensure high availability within an AWS Region.

The potential steps for the solutions architect to consider for fulfilling these requirements are (choose two):

A. Implement the security tool on EC2 instances organized within a new Auto Scaling group, located in the current VPC. This setup allows the security tool to be scalable and responsive to varying network traffic.

B. Position the web application behind a Network Load Balancer. This could distribute the application's network traffic across multiple EC2 instances efficiently.

C. Establish an Application Load Balancer ahead of the EC2 instances running the security tool. This would manage the distribution of incoming application traffic.

D. Configure a Gateway Load Balancer in each Availability Zone. This load balancer would reroute incoming and outgoing VPC traffic through the security tool.

E. Set up a transit gateway to aid in the communication among different VPCs, which might be necessary for complex network architectures.

The correct answer is: A, D.

A. Placing the security tool on EC2 instances within a new Auto Scaling group in the same VPC enables the tool to adapt to varying loads, ensuring that it can inspect traffic effectively without hindering performance.

D. By deploying a Gateway Load Balancer in each Availability Zone, traffic to and from the VPC is consistently directed through the security tool. This approach guarantees thorough inspection while maintaining high availability across the region.

Reasons why other options are incorrect:

B. Using a Network Load Balancer focuses on distributing traffic among EC2 instances but does not directly address the need to route all traffic through the security tool for inspection.

C. An Application Load Balancer is better suited for handling HTTP/HTTPS traffic and does not cater to the specific requirement of routing all network traffic through the security tool for inspection.

E. A transit gateway is primarily used for streamlining inter-VPC communications and is not directly relevant to the specific need of integrating a third-party security tool for inspecting all VPC traffic.

Question 21

A company uses various appliances with IoT sensors from different vendors. These sensors transmit status information in unique formats to an existing application, which converts the data into JSON. The application processes the JSON data daily and stores it in a relational database for subsequent analysis. The company seeks a new, more efficient data analysis solution on AWS that enhances speed and cost-effectiveness.

Possible solutions include:

A. Integrate the IoT sensors with AWS IoT Core, using a rule to trigger an AWS Lambda function for parsing the data into a .csv file format and storing it on Amazon S3. Employ AWS Glue for data cataloging, and utilize Amazon Athena and Amazon QuickSight for the analysis.

B. Transition the application server to AWS Fargate, ensuring it continues to receive data from IoT sensors and converts it into a relational format. Store the processed data in Amazon Redshift for analysis.

C. Implement an AWS Transfer for SFTP server and modify the IoT sensors to transmit data as .csv files via SFTP. Use AWS Glue to catalog these files and Amazon Athena for data analysis.

D. Deploy AWS Snowball Edge to directly gather data from IoT sensors for local analysis, periodically aggregating the data into Amazon Redshift for broader analysis.

The correct answer is: A.

A. Integrate the IoT sensors with AWS IoT Core, using a rule to trigger an AWS Lambda function for parsing the data into a .csv file format and storing it on Amazon S3. Employ AWS Glue for data cataloging, and utilize Amazon Athena and Amazon QuickSight for the analysis.

This solution efficiently meets the company's needs by leveraging AWS IoT Core to directly manage data from various IoT sensors. AWS Lambda can parse this data into a standardized format (such as .csv) and store it in Amazon S3. AWS Glue then catalogs the data for analysis, while Amazon Athena and Amazon QuickSight provide powerful and flexible tools for analyzing the data, optimizing both speed and cost.

Reasons why other options are incorrect:

B. Migrating the server to AWS Fargate doesn't fundamentally change the data processing or analysis methods. While Fargate provides a serverless environment, it doesn't inherently optimize data parsing or analysis compared to the proposed integration with AWS IoT Core and associated AWS services.

C. Setting up an AWS Transfer for SFTP server to handle .csv file transfers adds unnecessary complexity. It requires additional steps to reconfigure the IoT sensors and doesn't take full advantage of the seamless integration and analysis capabilities offered by AWS IoT Core, Lambda, Glue, Athena, and QuickSight.

D. Using AWS Snowball Edge for local data collection and periodic aggregation into Amazon Redshift is more suited for edge computing scenarios where connectivity is limited. This approach doesn't fully leverage cloud-native services for real-time data processing and analysis, and might not be as cost-effective or efficient for the company's requirements.

Question 22

A company is transitioning several applications to AWS and plans a rapid migration post the establishment of networking and security frameworks. They have already established an AWS Direct Connect connection within a central network account. Anticipating the creation of hundreds of AWS accounts and VPCs, the company requires seamless network access between its corporate network and AWS resources, as well as the ability to manage internet routing for AWS resources via its on-premises data center.

To achieve these objectives, the company should consider the following steps (choose three):

A. Set up a Direct Connect gateway in the central account and initiate association proposals using the Direct Connect gateway and the account ID for each virtual private gateway in the various accounts.

B. Implement both a Direct Connect gateway and a transit gateway in the central network account, and connect the transit gateway to the Direct Connect gateway using a transit Virtual Interface (VIF).

C. Establish an internet gateway, attach it to the subnets, and configure it to manage internet traffic.

D. Make the transit gateway available to other accounts and link the VPCs to this transit gateway.

E. Organize VPC peering as needed for connectivity between VPCs.

F. Set up exclusively private subnets and configure the necessary routes on the transit gateway and customer gateway to facilitate outbound internet traffic from AWS through NAT services located in the data center.

The correct combination of steps is: B, D, F.

B. The creation of a transit gateway in the central network account, linked to the Direct Connect gateway, provides a centralized and scalable way to manage network connectivity across multiple VPCs and accounts, enabling seamless communication with the corporate network.

D. Sharing the transit gateway with other AWS accounts and attaching their VPCs to it simplifies network management and ensures consistent and efficient communication across all VPCs and the corporate network.

F. Configuring only private subnets and setting appropriate routes on the transit gateway and customer gateway allows the company to route outbound internet traffic from AWS through their on-premises data center, adhering to their requirement for centralized internet access management.

Reasons why other options are incorrect:

A. While creating a Direct Connect gateway in the central account is a step in the right direction, merely creating association proposals for each virtual private gateway is not sufficient for the scalable and efficient management of network connectivity that the company anticipates needing.

C. An internet gateway facilitates direct internet access for subnets in AWS, which does not align with the company's strategy to route cloud resources to the internet via their on-premises data center.

E. VPC peering is useful for establishing network connectivity between specific VPCs but is not scalable for hundreds of VPCs and accounts. The transit gateway approach is more efficient and manageable for the company's projected scale.

Question 23

A company with a vast number of AWS accounts has recently centralized its process for the acquisition and modification of Reserved Instances (RIs). Under the new system, individual business units must request these actions through a specialized team, rather than managing RIs independently within their own AWS accounts. The solutions architect is tasked with securely implementing and enforcing this new protocol.

The steps to achieve this are (choose two):

A. Consolidate all AWS accounts under an AWS Organizations structure with all features activated. This setup enables centralized management and policy application across the entire organization.

B. Implement AWS Config to monitor and report on IAM policies, particularly those that restrict access to RI purchase and modification actions.

C. Formulate an IAM policy within each AWS account to explicitly deny the 'ec2:PurchaseReservedInstancesOffering' and 'ec2:ModifyReservedInstances' actions.

D. Develop a Service Control Policy (SCP) that prohibits the 'ec2:PurchaseReservedInstancesOffering' and 'ec2:ModifyReservedInstances' actions. Apply this SCP across all Organizational Units (OUs) within AWS Organizations.

E. Ensure that all AWS accounts are part of an AWS Organizations setup that utilizes the consolidated billing feature.

The correct combination of steps is: **A, D.**

A. By placing all AWS accounts under an AWS Organizations framework with full features enabled, the company can efficiently manage policies and processes, such as RI purchasing, across all accounts.

D. Creating an SCP that disallows specific actions related to RI purchases and modifications and applying it to each OU in the organization ensures that these restrictions are uniformly enforced across all accounts. This approach is the most effective and secure way to maintain the centralized process.

Reasons why other options are incorrect:

B. While AWS Config can monitor compliance with specific policies, it does not actively enforce policies. In this scenario, the goal is to prevent the actions, not just report on them.

C. Implementing individual IAM policies in each account would require significant administrative effort and could lead to inconsistencies. SCPs (as in option D) provide a more streamlined and organization-wide approach.

E. While consolidated billing is a feature of AWS Organizations, it does not directly contribute to enforcing policies on RI purchases or modifications. The focus here is on enforcing specific actions (or restrictions thereof), which is better achieved through SCPs.

Question 24

A company's critical application relies on an Amazon RDS for MySQL database, configured in Multi-AZ mode, for data storage. However, a recent test of RDS database failover led to a 40-second downtime, which is problematic for the application's operations. The goal is to devise a solution that can reduce this outage duration to under 20 seconds.

The potential steps to achieve this reduced downtime include:

A. Implement Amazon ElastiCache for Memcached as a caching layer before the database. B. Implement Amazon ElastiCache for Redis as a caching layer in front of the database. C. Introduce RDS Proxy to serve as an intermediary between the application and the database. D. Transition the database from Amazon RDS for MySQL to Amazon Aurora MySQL. E. Set up an Amazon Aurora Replica for the database. F. Establish an RDS for MySQL read replica.

The correct combination of steps is: **C, D, E.**

C. Utilizing RDS Proxy can improve database failover times by maintaining established connections. This helps to ensure that the application can more quickly resume operations following a failover.

D. Migrating to Amazon Aurora MySQL can offer faster failover times compared to standard RDS for MySQL. Aurora is designed for higher availability and durability, which can be crucial for critical applications.

E. Creating an Amazon Aurora Replica can further enhance availability and failover capabilities. Aurora Replicas can be promoted to primary instances more quickly than traditional RDS read replicas, reducing downtime during failovers.

Reasons why other options are incorrect:

A & B. While Amazon ElastiCache, whether using Memcached or Redis, can significantly improve database performance by caching frequently accessed data, it does not directly contribute to reducing failover times for the RDS database.

F. Creating a read replica of RDS for MySQL can provide additional read capacity and is useful for scaling, but it does not significantly improve the failover time for the primary RDS instance. In this context, migrating to Amazon Aurora (option D) and using Aurora Replicas (option E) are more effective strategies for reducing failover time.

Question 25

An AWS partner, part of an organization named org1, is developing a service within AWS Organizations and needs to access AWS resources in a client's account, which belongs to a different organization (org2). For this service, the partner must establish a secure, least privilege method of access, using an API or command-line tool, to the client's AWS resources.

The potential methods for granting this access include:

A. The client provides their AWS account access keys to the partner, enabling them to log in and carry out the necessary operations.

B. The client creates an IAM user with the necessary permissions and shares the user's credentials with the partner for performing the required tasks.

C. The client establishes an IAM role with the appropriate permissions, and the partner uses the role's Amazon Resource Name (ARN) to request access for executing the necessary tasks.

D. The client sets up an IAM role with the relevant permissions and incorporates an external ID in the role's trust policy. The partner then uses the role's ARN, along with the external ID, to request access for the required operations.

The most secure method is: D.

D. The client sets up an IAM role with the relevant permissions and incorporates an external ID in the role's trust policy. The partner then uses the role's ARN, along with the external ID, to request access for the required operations.

This approach is the most secure because it leverages the AWS best practice of using IAM roles for cross-account access, which avoids sharing credentials. The addition of an external ID to the IAM role's trust policy provides an extra layer of security. It ensures that only the intended AWS partner can assume the role, mitigating the risk of the "confused deputy" problem, where a malicious entity could otherwise assume the role.

Reasons why other options are incorrect:

A. Sharing AWS account access keys is highly insecure and against AWS best practices. It gives full access to the account and poses significant security risks.

B. Creating an IAM user and sharing its credentials, while slightly more controlled than sharing access keys, is still insecure and not recommended. It lacks the finer control and auditing capabilities provided by assuming an IAM role, as in option D.

C. While using an IAM role for access is a secure approach, this option lacks the additional security measure of an external ID. The external ID prevents other AWS accounts from assuming the role even if they know the role's ARN, making option D more secure.

Question 26

A delivery company is aiming to shift its third-party route planning software, available as a Docker image from a public registry, to AWS. This application is designed to operate across multiple containers, each serving a specific geographic section to optimize route map generation. The company requires a solution that allows for efficient resource allocation, scaling based on the number of active containers, while keeping operational complexity to a minimum.

The options for implementing this solution include:

A. Establish an Amazon Elastic Kubernetes Service (Amazon EKS) cluster using Amazon EC2 instances. Deploy the route planning application in Kubernetes pods, applying custom tags to these pods through the Amazon EKS CLI.

B. Set up an Amazon Elastic Kubernetes Service (Amazon EKS) cluster using AWS Fargate. Deploy the application using the Amazon EKS CLI and tag the pods using the AWS CLI's tag-resource API.

C. Configure an Amazon Elastic Container Service (Amazon ECS) cluster with Amazon EC2 instances. Utilize the AWS CLI to run tasks for the planning application, assigning custom

tags to each task with the --tags option.

D. Create an Amazon Elastic Container Service (Amazon ECS) cluster on AWS Fargate. Use the AWS CLI's run-task command, enabling ECSManagedTags, and assign custom tags to each task using the --tags option.

The most suitable solution is: D.

D. Create an Amazon Elastic Container Service (Amazon ECS) cluster on AWS Fargate. Use the AWS CLI's run-task command, enabling ECSManagedTags, and assign custom tags to each task using the --tags option.

This approach provides the least operational overhead for the company's requirements. AWS Fargate abstracts away the underlying server infrastructure, allowing the company to focus on running containers without managing EC2 instances. The ability to tag tasks directly with ECS-managed tags facilitates efficient resource tracking and allocation based on the number of containers running for each section.

Reasons why other options are incorrect:

A. While Amazon EKS on EC2 provides a robust Kubernetes environment, it introduces more operational complexity due to the need for managing EC2 instances. This makes it less optimal compared to the serverless nature of Fargate in option D.

B. Amazon EKS on Fargate offers a serverless Kubernetes experience, but tagging Kubernetes pods directly is not as straightforward as tagging ECS tasks. Additionally, EKS generally involves more complex management compared to ECS, particularly for use cases that don't require specific Kubernetes features.

C. Amazon ECS on EC2 would require managing the underlying EC2 instances, increasing operational complexity. Fargate, as used in option D, offers a more streamlined, serverless experience.

Question 27

A software company, utilizing AWS, has deployed an application across multiple accounts and regions. Specifically, the application is hosted on Amazon EC2 instances in a VPC (referred to as the application VPC) in the us-east-1 region, with an assigned IPv4 CIDR block of 10.10.0.0/16. Separately, in another AWS account, there's a shared services VPC in the us-east-2 region, designated with an IPv4 CIDR block of 10.10.10.0/24. When a cloud engineer attempts to establish VPC peering between the application VPC and the shared services VPC using AWS CloudFormation, they encounter an error indicating a failure in the peering process.

Potential reasons for this peering error include (choose two):

A. The IPv4 CIDR ranges of the two VPCs (10.10.0.0/16 and 10.10.10.0/24) have overlapping addresses.

B. The VPCs are situated in different regions (us-east-1 and us-east-2).

C. Either one or both of the accounts lack an Internet gateway, which is required for VPC peering.

D. One of the VPCs was not appropriately shared through AWS Resource Access Manager (RAM).

E. The IAM role in the account accepting the VPC peering request lacks the necessary permissions to establish the peering connection.

The correct answers are: **A, E.**

A. Overlapping CIDR blocks are a common reason for VPC peering failures. In this case, the CIDR block 10.10.10.0/24 is a subset of the larger block 10.10.0.0/16, causing an overlap which is not permissible in VPC peering.

E. Insufficient IAM permissions can prevent the establishment of a VPC peering connection. If the IAM role in the accepting account does not have the necessary permissions to action the peering request, the process will fail.

Reasons why other options are incorrect:

B. AWS allows VPC peering across different regions, so the fact that the VPCs are in us-east-1 and us-east-2 does not inherently prevent peering.

C. An Internet gateway is not a requirement for VPC peering, as VPC peering connections are solely for private network traffic between VPCs and do not involve external internet access.

D. While AWS Resource Access Manager is used for sharing resources across AWS accounts, it is not a prerequisite for initiating VPC peering. The issue in this scenario is not related to resource sharing but to overlapping CIDR ranges and potential IAM permission issues.

Question 28

A company's serverless application, primarily using AWS Lambda functions, was recently audited and found to have Lambda execution roles with overly broad IAM permissions, including unrestricted access to Amazon S3 buckets and Amazon DynamoDB tables. The company seeks to refine these permissions, ensuring each Lambda function has only the

essential permissions required for its operation. The task at hand is to efficiently identify the necessary permissions for each Lambda function.

The strategies to achieve this include:

A. Utilize Amazon CodeGuru for profiling Lambda functions to identify AWS API calls. Compile a list of these API calls and resources used by each Lambda function. Based on this inventory, create tailored IAM policies for each function, ensuring alignment with the company's operational needs.

B. Enable AWS CloudTrail logging for the account, and employ AWS IAM Access Analyzer to automatically generate IAM policies based on activities recorded in CloudTrail logs. These policies should be reviewed to confirm they align with the company's business requirements.

C. Activate AWS CloudTrail logging, then write and run a script to analyze the CloudTrail logs. This script should identify AWS API calls made by each Lambda execution role and summarize these findings. Based on this analysis, create more restrictive IAM policies for each Lambda function.

D. Switch on AWS CloudTrail logging and direct the logs to an Amazon S3 bucket. Use Amazon EMR to process these CloudTrail logs stored in S3, generating a detailed report of API calls and resources utilized by each execution role. Subsequently, formulate a new IAM policy for each role, ensuring they are reviewed for business compliance and stored in an S3 bucket.

The most efficient solution is: B.

B. Enable AWS CloudTrail logging for the account, and employ AWS IAM Access Analyzer to automatically generate IAM policies based on activities recorded in CloudTrail logs. These policies should be reviewed to confirm they align with the company's business requirements.

This approach is the least labor-intensive and most effective. AWS IAM Access Analyzer can analyze CloudTrail logs to understand the actions and resources used by each Lambda function and then automatically generate appropriate IAM policies. These generated policies should be reviewed to ensure they meet the company's specific business needs.

Reasons why other options are less efficient:

A. Using Amazon CodeGuru for profiling involves a more manual process of reviewing API calls and creating policies, which is more time-consuming compared to the automated policy generation in option B.

C. Writing a custom script to parse CloudTrail logs and then manually creating IAM policies based on the script’s output is a much more labor-intensive process compared to using the

automated capabilities of IAM Access Analyzer in option B.

D. Processing CloudTrail logs with Amazon EMR and then manually creating and reviewing policies is a complex and time-consuming process. It lacks the streamlined efficiency of using IAM Access Analyzer's automated policy generation and analysis.

Question 29

A solutions architect is tasked with assessing the efficiency of a company's use of Amazon EC2 instances and Amazon Elastic Block Store (EBS) volumes. The company operates several large, high-memory EC2 instances for database clusters in active/passive configurations. However, the utilization of these instances varies, and no clear usage pattern has been established. The objective is to analyze the environment and make adjustments based on the analysis, prioritizing cost-effectiveness.

The options for carrying out this task include:

A. Set up a dashboard using AWS Systems Manager OpsCenter to visualize Amazon CloudWatch metrics related to the EC2 instances and EBS volumes. Periodically examine the dashboard to detect usage trends and adjust (rightsize) the EC2 instances according to the observed peak metrics.

B. Enable Amazon CloudWatch detailed monitoring for both the EC2 instances and EBS volumes. Create a dashboard based on these metrics, review it to determine usage patterns, and then adjust the EC2 instance sizes based on the highest observed metrics.

C. Install the Amazon CloudWatch agent on each EC2 instance. Activate AWS Compute Optimizer and allow it to analyze the environment for a minimum of 12 hours. Review the recommendations provided by Compute Optimizer and modify (rightsize) the EC2 instances accordingly.

D. Enroll in the AWS Enterprise Support plan. Activate AWS Trusted Advisor, wait for 12 hours, then review its recommendations and adjust the EC2 instances as suggested.

The most cost-effective solution is: C.

C. Install the Amazon CloudWatch agent on each EC2 instance. Activate AWS Compute Optimizer and allow it to analyze the environment for a minimum of 12 hours. Review the recommendations provided by Compute Optimizer and modify (rightsize) the EC2 instances accordingly.

AWS Compute Optimizer uses machine learning to analyze historical utilization data and provides recommendations for optimal EC2 instance types and sizes. This process ensures that resources are appropriately matched to workload requirements, leading to potential cost

savings. The use of the CloudWatch agent enhances the quality of the data used for these recommendations.

Reasons why other options are less efficient:

A. Creating and manually reviewing a dashboard in AWS Systems Manager OpsCenter can be time-consuming and may not provide as comprehensive and actionable insights as automated tools like AWS Compute Optimizer.

B. While Amazon CloudWatch detailed monitoring offers granular metrics, the process of manually creating and reviewing a dashboard to identify usage patterns and rightsize instances is less efficient and could lead to less accurate sizing decisions compared to the automated analysis provided by AWS Compute Optimizer.

D. Subscribing to the AWS Enterprise Support plan and using AWS Trusted Advisor involves additional costs and may not be necessary just for rightsizing EC2 instances. AWS Compute Optimizer, being a specialized tool for EC2 optimization, offers more targeted and cost-effective recommendations for this specific need.

Question 30

A company operating within a multi-account AWS setup, governed by AWS Control Tower and interconnected with AWS Transit Gateway, has deployed a web application utilizing AWS Lambda and Amazon RDS in one of its application accounts. Concurrently, the company's database administrators, operating from a distinct DBA account, centrally manage all organizational databases. They access an RDS database, located in the application account, via an Amazon EC2 instance in the DBA account. The application team securely stores database credentials as secrets in AWS Secrets Manager within the application account and currently shares these manually with the DBAs. These secrets are encrypted using the default AWS managed key for Secrets Manager in the application account. The objective is to find a solution that grants database administrators access to the database without the need for manual sharing of secrets.

Possible solutions include:

A. Leverage AWS Resource Access Manager (AWS RAM) to share the secrets from the application account with the DBA account. In the DBA account, create an IAM role named DBA-Admin with permissions to access the shared secrets, and attach this role to the EC2 instance for accessing cross-account secrets.

B. In the application account, create an IAM role named DBA-Secret with permissions to access the secrets. In the DBA account, create an IAM role named DBA-Admin with

permissions to assume the DBA-Secret role in the application account. Attach the DBA-Admin role to the EC2 instance to enable access to the cross-account secrets.

C. In the DBA account, create an IAM role named DBA-Admin with permissions to access the secrets and the default AWS managed key in the application account. In the application account, attach resource-based policies to the key to permit access from the DBA account, and attach the DBA-Admin role to the EC2 instance for accessing the cross-account secrets.

D. In the DBA account, create an IAM role named DBA-Admin with permissions to access the secrets in the application account. Apply a Service Control Policy (SCP) to the application account to authorize access to the secrets from the DBA account. Attach the DBA-Admin role to the EC2 instance for accessing the cross-account secrets.

The most suitable solution is: B.

B. In the application account, create an IAM role named DBA-Secret with permissions to access the secrets. In the DBA account, create an IAM role named DBA-Admin with permissions to assume the DBA-Secret role in the application account. Attach the DBA-Admin role to the EC2 instance to enable access to the cross-account secrets.

This approach is effective because it utilizes IAM roles to facilitate secure, cross-account access to the secrets stored in AWS Secrets Manager. By creating a role in the application account (DBA-Secret) that has access to the secrets and a role in the DBA account (DBA-Admin) that can assume the DBA-Secret role, the solution provides a secure mechanism for the DBAs to access the necessary credentials without manual sharing.

Reasons why other options are incorrect:

A. AWS RAM does not support sharing of AWS Secrets Manager secrets directly. Thus, this option would not fulfill the requirement.

C. While IAM roles can be used for cross-account access, AWS Secrets Manager does not support directly attaching resource-based policies to the secrets for cross-account access. The required permissions involve assuming a role in the application account, as described in option B.

D. SCPs are used for placing restrictions at the organization level in AWS Organizations, not for granting specific cross-account access permissions to resources like secrets in AWS Secrets Manager. The approach in option B is more direct and appropriate for this scenario.

Question 31

A company utilizing AWS Organizations has established two organizational units (OUs) under its root OU: Research and DataOps. To comply with regulatory demands, all resources

deployed by the company within AWS must be located exclusively in the ap-northeast-1 Region. Furthermore, for the DataOps OU, there is a requirement to restrict EC2 instances to a specific set of predefined instance types. The solutions architect is tasked with enforcing these restrictions in a way that is operationally efficient and requires minimal ongoing maintenance.

The strategies for achieving this include:

A. Set up an IAM role in one of the accounts within the DataOps OU and apply an inline policy utilizing the 'ec2:InstanceType' condition key to limit the role to certain EC2 instance types.

B. Establish an IAM user in every account under the root OU and attach an inline policy using the 'aws:RequestedRegion' condition key to restrict access to all regions except ap-northeast-1.

C. Formulate a Service Control Policy (SCP) with the 'aws:RequestedRegion' condition key to limit access to all regions except ap-northeast-1, and apply this SCP to the root OU.

D. Create an SCP using the 'ec2:Region' condition key to restrict access to all AWS Regions except ap-northeast-1, and implement this SCP across the root OU, DataOps OU, and Research OU.

E. Develop an SCP employing the 'ec2:InstanceType' condition key to restrict access to specific EC2 instance types, and apply this SCP to the DataOps OU.

The most appropriate solution is: **C, E.**

C. Creating an SCP with the 'aws:RequestedRegion' condition key effectively limits resource deployment to the ap-northeast-1 Region across the entire organization. By applying this SCP to the root OU, it ensures that all accounts under the root, including Research and DataOps OUs, adhere to the regional restriction, thus fulfilling the regulatory requirement.

E. Implementing an SCP with the 'ec2:InstanceType' condition key specifically for the DataOps OU allows for the enforcement of instance type restrictions within this OU. This targeted approach ensures that EC2 instances in the DataOps OU comply with the predefined list of instance types.

Reasons why other options are incorrect:

A. Using an IAM role with an inline policy in one account under the DataOps OU is not as efficient or broad-reaching as applying an SCP. This method would require similar roles and policies to be replicated across all accounts in the OU, increasing operational complexity.

B. Creating an IAM user with restrictive policies in every account is operationally inefficient and does not ensure organization-wide compliance as effectively as an SCP applied at the OU level.

D. The 'ec2:Region' condition key is not the correct approach for restricting access to specific AWS Regions. The correct key for region restrictions in SCPs is 'aws:RequestedRegion', as used in option C.

Question 32

A company operates a serverless application in a single AWS region, which processes external URLs to extract metadata. This application architecture involves an Amazon SNS (Simple Notification Service) topic that sends URLs to an Amazon SQS (Simple Queue Service) queue. An AWS Lambda function, triggered by this queue, processes these URLs and stores the results in an Amazon S3 bucket. The company now aims to expand this URL processing across multiple regions to analyze regional variations in site content. However, the URL publication must originate from the existing region, and the processing results must be stored in the original S3 bucket.

To achieve a multi-region deployment that satisfies these requirements, the following changes should be considered (choose two):

A. Set up the SQS queue and the Lambda function in additional regions.

B. Link the SNS topic in each region to the SQS queue.

C. Connect the SQS queues in each new region to the existing SNS topic.

D. Configure the SQS queue to distribute URLs to SNS topics in various regions.

E. Implement the SNS topic and the Lambda function in other regions.

The correct combination of changes is: **A, C.**

A. Deploying the SQS queue and the Lambda function in other regions enables the company to process URLs locally in each region. This setup allows for regional variations in content to be analyzed effectively.

C. By subscribing the SQS queues in the new regions to the original SNS topic, URLs can be distributed to all regional queues. This ensures that the same URLs are processed in each region, meeting the requirement of multi-region comparison.

Reasons why other options are incorrect:

B. Subscribing the SNS topic to the SQS queue is a reversal of the intended workflow. The SNS topic is the publisher, and SQS queues are the subscribers in the architecture.

D. Configuring the SQS queue to publish URLs to SNS topics in each region is unnecessary and complicates the architecture. The existing SNS topic can directly publish to SQS queues in multiple regions (as per option C).

E. Deploying the SNS topic in other regions is not required. A single SNS topic can publish messages to SQS queues in multiple regions. What's needed is the deployment of the SQS queue and the Lambda function in other regions (as per option A) and configuring them to subscribe to the original SNS topic.

Question 33

A company operates a custom, stateless Extract, Transform, Load (ETL) tool on an Amazon EC2 Linux instance. This tool, a Linux binary that can't be altered, is single-threaded, consumes 2 GB of memory, and is CPU-intensive. It is scheduled to execute every four hours and has a maximum runtime of 20 minutes. The solutions architect is tasked with reevaluating and updating the architecture of this solution.

Possible strategies for this revamp include:

A. Migrate the application to AWS Lambda, using Amazon CloudWatch Logs to trigger the Lambda function at four-hour intervals.

B. Implement AWS Batch for executing the application, with an AWS Step Functions state machine orchestrating the AWS Batch job at four-hour frequencies.

C. Utilize AWS Fargate to host the application, scheduling the Fargate task to launch every four hours via Amazon EventBridge (formerly Amazon CloudWatch Events).

D. Run the application on Amazon EC2 Spot Instances, employing AWS CodeDeploy to manage the deployment and execution of the application every four hours.

The most suitable solution is: C.

C. Utilize AWS Fargate to host the application, scheduling the Fargate task to launch every four hours via Amazon EventBridge.

AWS Fargate is an efficient solution for running containerized applications without the need to manage the underlying server infrastructure. It's particularly well-suited for tasks like the company's ETL application, which has specific resource requirements (CPU and memory intensive) and runs at regular intervals. Using Amazon EventBridge to schedule these tasks every four hours ensures that the application runs on time without manual intervention.

Reasons why other options are less suitable:

A. AWS Lambda might not be the optimal choice for this CPU-intensive application, especially considering Lambda's limitations in terms of execution time (15 min) and memory allocation. Additionally, the application's binary nature might make it incompatible with the Lambda execution environment.

B. AWS Batch is designed for batch processing workloads and could be used for this purpose. However, it's generally more suited for larger and more complex batch processing jobs. The overhead of setting up AWS Batch and Step Functions for a relatively simple, recurring task might be unnecessary compared to the straightforwardness of Fargate.

D. Using Amazon EC2 Spot Instances could be cost-effective but involves managing instances and ensuring availability, which adds complexity. AWS CodeDeploy adds another layer of complexity in deployment management. Fargate offers a simpler, serverless solution without the need for instance management.

Question 34

A company is developing a follow-up to a successful online game, anticipating a substantial global player base active from the first week of launch. The current game architecture, deployed in a single AWS region, includes an Amazon S3 bucket for game assets and an Amazon DynamoDB table for storing player scores. The solutions architect is tasked with designing a multi-region solution to enhance latency, reliability, and ease of implementation.

The potential strategies for this upgrade include:

A. Set up an Amazon CloudFront distribution for the S3 bucket assets, enable S3 Cross-Region Replication, create a new DynamoDB table in another region, and use this table as a replica in DynamoDB global tables.

B. Implement an Amazon CloudFront distribution for the S3 bucket assets, enable S3 Same-Region Replication, create a new DynamoDB table in a different region, and set up asynchronous replication between the DynamoDB tables using AWS Database Migration Service (AWS DMS) with Change Data Capture (CDC).

C. Establish a new S3 bucket in a different region and activate S3 Cross-Region Replication. Configure an Amazon CloudFront distribution with origin failover using two origins for the S3 buckets in each region. Enable DynamoDB global tables by turning on Amazon DynamoDB Streams, and add a replica table in a new region.

D. Create an additional S3 bucket in the same region, configure S3 Same-Region Replication, set up an Amazon CloudFront distribution with origin failover for the two S3 buckets, and

create a new DynamoDB table in another region to be used as a replica in DynamoDB global tables.

The most effective solution is: C.

C. Establish a new S3 bucket in a different region and activate S3 Cross-Region Replication. Configure an Amazon CloudFront distribution with origin failover using two origins for the S3 buckets in each region. Enable DynamoDB global tables by turning on Amazon DynamoDB Streams, and add a replica table in a new region.

This approach addresses the key requirements effectively. The use of Amazon CloudFront and S3 Cross-Region Replication ensures that game assets are delivered with low latency globally. The configuration of DynamoDB global tables provides a multi-region, fully managed database solution that automatically handles the replication across regions, ensuring high availability and fast access to player score data.

Reasons why other options are less suitable:

A. While using CloudFront and S3 Cross-Region Replication is beneficial, the manual creation of a new DynamoDB table and configuring it as a replica is less efficient compared to automatically managed replication in DynamoDB global tables.

B. S3 Same-Region Replication is not necessary when using CloudFront, and setting up replication between DynamoDB tables using AWS DMS with CDC adds complexity and is not as seamless as DynamoDB global tables.

D. S3 Same-Region Replication and a new S3 bucket in the same region are redundant when using CloudFront. Also, manually managing the DynamoDB replica as suggested here is less efficient than using the built-in global table functionality.

Question 35

A real estate company operating an on-premises web application, designed to offer property information to renters and buyers, is planning to migrate its infrastructure to AWS. The application currently utilizes a Java-based backend and a NoSQL MongoDB database to manage subscriber information. The company's objective is to replicate this setup on AWS without altering the application, ensuring high availability in the process.

Available migration options include:

A. Transition to an Amazon Aurora DB cluster for managing subscriber data, and deploy the Java backend on Amazon EC2 instances configured within an Auto Scaling group spread across multiple Availability Zones.

B. Maintain the MongoDB database on Amazon EC2 instances, and set up the Java backend on EC2 instances within an Auto Scaling group located in a single Availability Zone.

C. Implement Amazon DocumentDB (compatible with MongoDB) with suitably sized instances across multiple Availability Zones for the subscriber database. Deploy the Java backend on Amazon EC2 instances using an Auto Scaling group also spanning multiple Availability Zones.

D. Set up Amazon DocumentDB (compatible with MongoDB) in an on-demand capacity mode across multiple Availability Zones for the subscriber database. Use an Auto Scaling group of Amazon EC2 instances across multiple Availability Zones for the Java backend.

The most suitable solution is: C.

C. Implement Amazon DocumentDB (compatible with MongoDB) with suitably sized instances across multiple Availability Zones for the subscriber database. Deploy the Java backend on Amazon EC2 instances using an Auto Scaling group also spanning multiple Availability Zones.

This approach provides high availability and aligns closely with the company's current on-premises setup. Amazon DocumentDB offers MongoDB compatibility, facilitating a seamless migration of the NoSQL database. Deploying both the DocumentDB instances and the EC2 instances (for the Java backend) across multiple Availability Zones ensures high availability and resilience.

Reasons why other options are less suitable:

A. Amazon Aurora is a relational database and not a direct substitute for MongoDB, making it unsuitable for the company's needs without modifying the application.

B. Running MongoDB on EC2 instances and limiting the Java backend to a single Availability Zone does not provide the high availability required for the application.

D. While Amazon DocumentDB is a suitable choice for MongoDB compatibility, there is no "on-demand capacity mode" for DocumentDB. The correct approach is to configure DocumentDB with appropriately sized instances for scalability and high availability, as described in option C.

Question 36

A digital marketing company utilizes separate AWS accounts for different teams. The creative team, within its own AWS account, maintains an Amazon S3 bucket to securely store images and media for marketing campaigns. They intend to grant the strategy team, operating from another AWS account, access to view the contents of this S3 bucket. Despite setting up an IAM role named 'strategy_reviewer' in the strategy team's account and using a custom AWS Key Management Service (AWS KMS) key in the creative team's account linked to the S3

bucket, the strategy team encounters an 'Access Denied' error when trying to access the bucket's objects. The solutions architect is tasked with ensuring access for the strategy team, while restricting permissions to the minimum necessary.

The solutions architect should consider the following steps (choose three):

A. Implement a bucket policy granting read permissions for the S3 bucket, with the principal set to the account ID of the Strategy account.

B. Modify the 'strategy_reviewer' IAM role to grant full permissions for the S3 bucket and decrypt permissions for the custom KMS key.

C. Revise the policy of the custom KMS key in the Creative account to include decrypt permissions for the 'strategy_reviewer' IAM role.

D. Formulate a bucket policy granting read permissions for the S3 bucket, with the principal set as an anonymous user.

E. Adjust the policy of the custom KMS key in the Creative account to grant encrypt permissions to the 'strategy_reviewer' IAM role.

F. Amend the 'strategy_reviewer' IAM role to provide read permissions for the S3 bucket and decrypt permissions for the custom KMS key.

The correct combination of steps is: **A, C, F.**

A. Crafting a bucket policy that offers read permissions and designating the principal as the Strategy account ID ensures that the strategy team can access the S3 bucket.

C. Updating the KMS key policy to grant decrypt permissions to the 'strategy_reviewer' IAM role is crucial because the S3 objects are encrypted with the custom KMS key. Without this permission, the strategy team cannot decrypt the objects even if they have read access to the bucket.

F. Modifying the 'strategy_reviewer' IAM role to include read permissions for the S3 bucket and decrypt permissions for the KMS key ensures that the strategy team can access and decrypt the S3 objects.

Reasons why other options are incorrect:

B. Granting full permissions for the S3 bucket to the 'strategy_reviewer' role is excessive and contradicts the principle of least privilege.

D. Setting the principal to an anonymous user is a security risk and is not advisable for controlled access between AWS accounts.

E. Providing encrypt permissions for the KMS key to the 'strategy_reviewer' role is unnecessary for the purpose of viewing the contents of the S3 bucket.

Question 37

A life sciences company, specializing in genomics data analysis, is currently utilizing open-source tools and Docker containers on on-premises servers to process genomic data. This data, initially stored on a local SAN, is processed on-site. Facing capacity constraints and aiming to enhance scalability and reduce processing times, the company plans to transition their genomics analysis platform to AWS. With an established high-speed AWS Direct Connect link, each genome sequencing generates about 200 GB of data, and the company anticipates receiving 10-15 job requests daily. The processed data will ultimately be stored in Amazon S3.

The company seeks a solution that aligns with these specifications:

A. Utilize AWS Snowball Edge devices, scheduled regularly, to import sequencing data into AWS. Upon AWS receiving the Snowball Edge and uploading the data to Amazon S3, trigger AWS Lambda functions through S3 events for data processing.

B. Employ AWS Data Pipeline for transferring sequencing data to Amazon S3, followed by triggering an Amazon EC2 Auto Scaling group via S3 events. This group would launch EC2 instances with custom AMIs to run Docker containers for data processing.

C. Implement AWS DataSync for transferring sequencing data to Amazon S3. Subsequently, use S3 events to activate an AWS Lambda function, which initiates an AWS Step Functions workflow. Docker images, stored in Amazon ECR, would be used by AWS Batch for processing the sequencing data.

D. Use AWS Storage Gateway file gateway for transferring sequencing data to Amazon S3. Upon data arrival in S3, trigger AWS Batch jobs that operate on Amazon EC2 instances, running the necessary Docker containers for data processing.

The most effective solution is: C.

C. Implement AWS DataSync for transferring sequencing data to Amazon S3. Subsequently, use S3 events to activate an AWS Lambda function, which initiates an AWS Step Functions workflow. Docker images, stored in Amazon ECR, would be used by AWS Batch for processing the sequencing data.

This option effectively leverages AWS DataSync for efficient and accelerated data transfer to S3. The use of S3 events to trigger a Lambda function, which in turn initiates an AWS Step Functions workflow, provides a robust and scalable orchestration mechanism. AWS Batch, utilizing Docker images from Amazon ECR, offers a flexible and powerful solution for

processing large-scale genomic data, addressing the need for scalability and reduced processing time.

Reasons why other options are less suitable:

A. Regularly scheduled AWS Snowball Edge devices might not be the most efficient method for transferring large volumes of data frequently. This option could introduce delays inconsistent with the company's goal of reducing turnaround times.

B. AWS Data Pipeline is less optimized for handling large data transfers compared to AWS DataSync. Also, triggering EC2 Auto Scaling for processing lacks the workflow orchestration and container management capabilities provided by AWS Batch and Step Functions.

D. While AWS Storage Gateway file gateway is a viable option for data transfer, it might not be as efficient as DataSync for large genomic datasets. Additionally, the direct triggering of AWS Batch jobs from S3 events, without the workflow orchestration provided by Step Functions and Lambda, might be less effective in managing complex processing workflows.

Question 38

A company is currently operating a content management application on a Windows-based Amazon EC2 instance in a development setting. This application interacts with static content on a 2 TB Amazon EBS volume, used as the root device. The company is looking to transition this application into a production environment, with a focus on high availability and fault tolerance. The new setup should include at least three EC2 instances distributed across multiple Availability Zones. The solution should integrate all instances into an Active Directory domain and implement Windows Access Control Lists (ACLs) for file content access management. Additionally, it's imperative that all instances maintain identical content synchronously.

The solutions architect is tasked with designing a solution that accomplishes these goals with minimal management complexity. Potential solutions include:

A. Establish an Amazon Elastic File System (EFS) file share. Create an Auto Scaling group spanning three Availability Zones, maintaining a minimum of three instances. Use a user data script for application installation, Active Directory domain joining, and EFS file share mounting.

B. Generate a new AMI from the current EC2 instance. Set up an Amazon FSx for Lustre file system. Create an Auto Scaling group across three Availability Zones, keeping a minimum of three instances. Use a user data script for Active Directory domain joining and mounting the FSx for Lustre file system.

C. Implement an Amazon FSx for Windows File Server file system. Configure an Auto Scaling group across three Availability Zones with at least three instances. Utilize a user data script for application setup and FSx for Windows File Server file system mounting, and perform a seamless domain join.

D. Create a new AMI from the existing EC2 instance. Set up an Amazon Elastic File System (EFS) file system. Form an Auto Scaling group over three Availability Zones with a minimum of three instances. Use a seamless domain join for Active Directory integration.

The most suitable solution is: C.

C. Implement an Amazon FSx for Windows File Server file system. Configure an Auto Scaling group across three Availability Zones with at least three instances. Utilize a user data script for application setup and FSx for Windows File Server file system mounting, and perform a seamless domain join.

Amazon FSx for Windows File Server provides a fully managed native Windows file system with built-in Active Directory integration and support for Windows ACLs, making it ideal for the company's needs. It ensures consistent file storage across multiple instances and supports the necessary features for a Windows-based application. The Auto Scaling group ensures high availability across Availability Zones, and the user data script automates the instance setup.

Reasons why other options are less suitable:

A. Amazon EFS is designed for Linux-based systems and does not natively support Windows ACLs, which are required for this scenario.

B. Amazon FSx for Lustre is optimized for high-performance computing workloads and does not offer native Windows file system features, such as Active Directory integration and Windows ACLs.

D. Similar to option A, Amazon EFS is not suitable for Windows-based applications that require native Windows file system features.

Question 39

A SaaS company offering a case management solution currently utilizes a standalone SMTP server for sending email messages from its application, including acknowledgement emails to customers. These emails are based on a template that integrates customer data before dispatch. The company is transitioning this email functionality to AWS and seeks a solution that minimizes operational effort while being cost-effective.

The potential solutions are:

A. Deploy an SMTP server on Amazon EC2 using an Amazon Machine Image (AMI) from AWS Marketplace. Store the email template in an Amazon S3 bucket. Develop an AWS Lambda function to fetch the template, merge it with customer data, and use an SDK within the Lambda function to send the email.

B. Utilize Amazon Simple Email Service (Amazon SES) for sending emails. Keep the email template in an Amazon S3 bucket. Construct an AWS Lambda function to access the template, integrate customer data, and use an SDK in the Lambda function for email dispatch.

C. Install an SMTP server on Amazon EC2 with an AMI from AWS Marketplace. Save the email template in Amazon SES with placeholders for customer data. Design an AWS Lambda function to invoke the SES template, substituting customer data in the placeholders, and use the SMTP server from AWS Marketplace to send the email.

D. Implement Amazon Simple Email Service (Amazon SES) for email sending. Store the email template in SES with placeholders for customer data. Create an AWS Lambda function to employ the SendTemplatedEmail API operation, substituting customer data in the placeholders and specifying the email recipient.

The most appropriate solution is: D.

D. Implement Amazon Simple Email Service (Amazon SES) for email sending. Store the email template in SES with placeholders for customer data. Create an AWS Lambda function to employ the SendTemplatedEmail API operation, substituting customer data in the placeholders and specifying the email recipient.

This approach leverages Amazon SES, a managed service designed specifically for sending emails, thus reducing operational overhead compared to maintaining an SMTP server. Storing email templates directly in SES with customizable parameters simplifies the process and integrates seamlessly with AWS Lambda for dynamic content generation. The use of the SendTemplatedEmail API operation in Lambda automates email sending with minimal maintenance required.

Reasons why other options are less suitable:

A. Setting up an SMTP server on EC2 instances introduces unnecessary complexity and maintenance overhead compared to using a managed service like Amazon SES.

B. While using Amazon SES for sending emails and AWS Lambda for template retrieval is effective, storing the email template in an S3 bucket adds an additional step in the process, making it less efficient compared to storing the template directly in SES (as in option D).

C. This option also involves managing an SMTP server on EC2, which increases operational overhead. Additionally, the process is more convoluted than directly using Amazon SES's

built-in template and sending capabilities.

Question 40

A company processes videos in the AWS Cloud using Amazon EC2 instances within an Auto Scaling group. The video processing duration is approximately 30 minutes. EC2 instances are dynamically scaled based on the workload in an Amazon SQS (Simple Queue Service) queue. The SQS queue has a redrive policy directed to a dead-letter queue with a `maxReceiveCount` of 1, and the visibility timeout is set to 1 hour. The company uses Amazon CloudWatch alarms to alert the development team when messages appear in the dead-letter queue. Despite no apparent errors in the application logs, notifications about messages in the dead-letter queue and unprocessed videos occur several times a day.

To address this issue, the company could consider:

A. Activating termination protection for the EC2 instances.

B. Extending the SQS queue visibility timeout to 3 hours.

C. Implementing scale-in protection for the instances during processing.

D. Modifying the redrive policy to set `maxReceiveCount` to 0.

The most effective solution is: C.

C. Implementing scale-in protection for the instances during processing.

Scale-in protection prevents EC2 instances from being terminated by the Auto Scaling group while they are still processing videos. This is crucial because if an instance is terminated mid-process, the message (video) being processed returns to the queue and could be sent to the dead-letter queue after being received again (due to the `maxReceiveCount` of 1), despite no issues with the video or the processing application.

Reasons why other options are less suitable:

A. Enabling termination protection for EC2 instances can prevent them from being terminated, but it's a broader measure that might not directly address the issue of instances being terminated during processing. Scale-in protection is a more targeted approach for instances that are actively processing jobs.

B. Increasing the visibility timeout to 3 hours might reduce the chances of messages returning to the queue prematurely if processing is delayed. However, this does not address the potential termination of instances during processing, which is likely causing messages to move to the dead-letter queue.

D. Setting the `maxReceiveCount` to 0 in the redrive policy is not practical, as this would mean messages are never sent to the dead-letter queue, potentially causing problematic messages to remain in circulation indefinitely. This does not address the underlying issue of instances being terminated while processing.

Question 41

A company has created APIs using Amazon API Gateway with Regional endpoints, which invoke AWS Lambda functions. These APIs are secured with API Gateway's authentication mechanisms. Upon reviewing the design, a solutions architect identifies certain APIs that don't require public access. The goal is to restrict these specific APIs to be accessible only within a Virtual Private Cloud (VPC), while still ensuring that all API calls are made by authenticated users.

To achieve this objective with minimal effort, the following solutions are considered:

A. Establish an internal Application Load Balancer (ALB), create a target group, assign the Lambda function as the target, and use the ALB's DNS name for API calls from within the VPC.

B. Eliminate the DNS entry for the API in API Gateway, set up a hosted zone in Amazon Route 53, create a CNAME record, update the API in API Gateway with this CNAME record, and use the record for making API calls from the VPC.

C. Convert the API Gateway endpoint from Regional to private, create an interface VPC endpoint, attach a resource policy to the API, and utilize this VPC endpoint for API calls within the VPC.

D. Deploy Lambda functions within the VPC, provision an EC2 instance with an Apache server, use the server to invoke the Lambda functions, and rely on the EC2 instance's internal CNAME record for API calls from the VPC.

The most efficient solution is: C.

C. Convert the API Gateway endpoint from Regional to private, create an interface VPC endpoint, attach a resource policy to the API, and utilize this VPC endpoint for API calls within the VPC.

This approach directly aligns with the requirement to restrict API access to within the VPC. By converting the API Gateway endpoint to a private type and creating an interface VPC endpoint, the APIs can be securely accessed from within the VPC. The resource policy ensures that only authenticated and authorized requests are processed, maintaining the security requirement.

Reasons why other options are less suitable:

A. Configuring an internal ALB and targeting Lambda functions do not inherently restrict API access to the VPC, nor does it ensure that API Gateway's authentication mechanisms are used.

B. Removing the API Gateway DNS entry and using Route 53 with a CNAME record does not provide a straightforward solution to restrict access to the VPC. This method also adds unnecessary complexity without directly addressing the requirement.

D. Placing Lambda functions inside the VPC and using an EC2 instance with an Apache server to call these functions is an overly complex solution that introduces additional management overhead. It diverges from the goal of minimal effort and does not leverage the capabilities of API Gateway for secure API management.

Question 42

A weather service, operating a web application hosted on AWS in the eu-west-1 Region, provides high-resolution weather maps that are frequently updated and stored in an Amazon S3 bucket, along with static HTML content. The web application uses Amazon CloudFront for content delivery. Recently, the service expanded to the us-east-1 Region, where users are experiencing intermittent slow performance when accessing their weather maps.

To improve performance for users in the us-east-1 Region, the company is considering the following options (choose two):

A. Set up AWS Global Accelerator with an endpoint targeting the S3 bucket in eu-west-1, configuring endpoint groups for TCP ports 80 and 443 in us-east-1.

B. Create a new S3 bucket in us-east-1 and implement S3 cross-Region replication to synchronize data from the existing S3 bucket in eu-west-1.

C. Utilize Lambda@Edge to alter requests originating from North America to employ the S3 Transfer Acceleration endpoint in us-east-1.

D. Employ Lambda@Edge to modify requests coming from North America to access the S3 bucket in us-east-1.

E. Configure an AWS Global Accelerator endpoint for us-east-1 as a new origin in the CloudFront distribution and use Lambda@Edge to redirect requests from North America to this new origin.

The most suitable combination of steps to resolve the performance issues is: **B, D.**

B. Creating a new S3 bucket in us-east-1 and enabling cross-Region replication from eu-west-1 will place the content closer to users in the us-east-1 Region. This reduces latency as the data is stored within the same region as the users, ensuring faster access to the weather maps.

D. Using Lambda@Edge to modify requests from North America to the newly created S3 bucket in us-east-1 ensures that users in this region are served content from a geographically closer source, further improving performance.

Reasons why other options are less suitable:

A. AWS Global Accelerator improves performance by optimizing the path to the application, but it does not address the core issue of geographical distance between the S3 bucket in eu-west-1 and users in us-east-1.

C. S3 Transfer Acceleration optimizes file transfers to S3 buckets, but it doesn't address the issue of content being stored far from the end-users. It's more useful for speeding up uploads to S3, not for improving content delivery speeds from S3.

E. Configuring AWS Global Accelerator as a new origin in the CloudFront distribution is not necessary and adds complexity. The existing CloudFront setup, combined with a new S3 bucket in us-east-1 and request modifications via Lambda@Edge, is sufficient to enhance performance.

Question 43

A company is encountering a problem where new sessions cannot be established in their Amazon Workspaces environment. The user profiles are stored on an Amazon FSx for Windows File Server, which has a configured storage capacity of 10 TB. Upon investigation, the solutions architect discovers that the file system has reached its maximum capacity. The objective is to restore user access and implement a strategy to avoid similar issues in the future.

The possible solutions are:

A. Delete older user profiles to free up space, and then transfer the user profiles to an Amazon FSx for Lustre file system.

B. Expand the file system's capacity using the `update-file-system` command. Set up an Amazon CloudWatch metric to monitor available space and use Amazon EventBridge to trigger an AWS Lambda function that automatically increases capacity as needed.

C. Employ the FreeStorageCapacity metric in Amazon CloudWatch to keep track of the file system space. Use AWS Step Functions to automatically expand the file system's capacity

when necessary.

D. Clear out old user profiles to make room, create a second FSx for Windows File Server file system, and reconfigure 50% of the user profile redirections to use this new file system.

The most appropriate solution is: B.

B. Expand the file system's capacity using the `update-file-system` command. Set up an Amazon CloudWatch metric to monitor available space and use Amazon EventBridge to trigger an AWS Lambda function that automatically increases capacity as needed.

This option addresses the immediate problem by expanding the existing file system's capacity, ensuring users can regain access. The automated monitoring and capacity expansion through CloudWatch and Lambda provide a proactive approach to managing storage, preventing recurrence of the issue.

Reasons why other options are less suitable:

A. Migrating user profiles to an Amazon FSx for Lustre file system isn't ideal because FSx for Lustre is optimized for high-performance computing and not for profile storage, which typically requires Windows compatibility.

C. While monitoring with CloudWatch and using Step Functions can be effective, this approach lacks the direct integration and simplicity of using EventBridge with a Lambda function for automated capacity management.

D. Creating an additional FSx for Windows File Server and redistributing user profiles adds complexity and doesn't offer an automated solution to manage future storage capacity issues. It also involves significant administrative overhead in managing and maintaining two file systems.

Question 44

An international delivery company uses AWS to operate its delivery management system, which allows drivers to upload delivery confirmations, including signatures and photos, via FTP to an Amazon EC2 instance. The EC2 instance processes these files by adding metadata after querying a central database and then archives them in Amazon S3. However, as the company grows, drivers face issues with the system rejecting connections due to the FTP server's dropped connections and memory problems. Despite a system engineer setting up a cron task to reboot the EC2 instance every 30 minutes, there are reports of missing files in the archive and the central system not being consistently updated.

The solutions architect must devise a scalable solution to ensure reliable file archiving and system updates, without modifying the handheld devices. The proposed solutions are:

A. Create an Amazon Machine Image (AMI) of the existing EC2 instance and use it in an Auto Scaling group behind an Application Load Balancer, with a minimum of three instances.

B. Implement AWS Transfer Family to establish an FTP server that stores files in Amazon Elastic File System (EFS), then mount the EFS volume on the existing EC2 instance and redirect file processing to this new path.

C. Utilize AWS Transfer Family to set up an FTP server that directly uploads files to Amazon S3. Configure S3 event notifications to trigger an AWS Lambda function via Amazon Simple Notification Service (Amazon SNS). The Lambda function should add the required metadata and update the delivery system.

D. Modify the handheld devices to directly upload files to Amazon S3 and use S3 event notifications through Amazon Simple Queue Service (Amazon SQS) to trigger an AWS Lambda function, which then processes the metadata and updates the delivery system.

The most suitable solution is: C.

C. Utilize AWS Transfer Family to set up an FTP server that directly uploads files to Amazon S3. Configure S3 event notifications to trigger an AWS Lambda function via Amazon Simple Notification Service (Amazon SNS). The Lambda function should add the required metadata and update the delivery system.

This solution is scalable and ensures reliable archiving of files and system updates. AWS Transfer Family provides a managed FTP service that can scale to meet the demands of the growing number of drivers. The use of S3 for storage, coupled with Lambda for processing, eliminates the need for manual intervention and issues related to EC2 instance capacity.

Reasons why other options are less suitable:

A. While using an Auto Scaling group of EC2 instances could improve scalability, it doesn't address the fundamental issue of managing FTP connections and file processing. The overhead of maintaining and scaling EC2 instances remains.

B. AWS Transfer Family with EFS improves FTP server scalability, but it still requires the EC2 instance for file processing, which could become a bottleneck.

D. This option is not viable as it requires modifications to the handheld devices, which the company has specified cannot be changed.

Question 45

A company operates an application in the AWS Cloud, which utilizes containers on an Amazon Elastic Container Service (Amazon ECS) cluster with the Fargate launch type. The

application's relational data is stored in Amazon Aurora MySQL. To comply with regulatory requirements, the application needs a recovery strategy to switch to a different AWS Region in case of an application failure, ensuring no data loss occurs.

The proposed solutions for meeting these requirements with minimal operational effort are:

A. Set up an Aurora Replica in an alternate AWS Region.

B. Implement AWS DataSync for ongoing replication of the data to a different Region.

C. Utilize AWS Database Migration Service (AWS DMS) for continuous data replication to a different Region.

D. Employ Amazon Data Lifecycle Manager (Amazon DLM) to schedule snapshots every 5 minutes.

The most suitable solution is: A.

A. Set up an Aurora Replica in an alternate AWS Region.

Creating an Aurora Replica in a different Region offers a streamlined approach to meet the recovery objectives. It enables real-time replication of the database, ensuring data is up-to-date and available for a quick recovery in the event of a regional outage or application failure, with minimal operational overhead.

Reasons why other options are less appropriate:

B. AWS DataSync is generally used for transferring data between on-premises storage and AWS services or between different AWS services. It is not optimized for real-time database replication, and its setup and maintenance might introduce more operational complexity compared to an Aurora Replica.

C. AWS DMS provides continuous replication capabilities but is typically used for database migration or synchronization between different database platforms. For Aurora MySQL, an Aurora Replica is a more integrated and efficient solution, specifically designed for high availability and cross-region replication.

D. Amazon DLM automates the process of creating snapshots, but snapshots every 5 minutes might not guarantee zero data loss in case of a failure. Moreover, restoring from snapshots is generally slower and more operationally complex compared to switching to a standby replica.

Question 46

A financial services company regularly receives a data feed from its credit card servicing partner. Every 15 minutes, around 5,000 plaintext records are transmitted over HTTPS and stored in an Amazon S3 bucket with server-side encryption. These records contain sensitive credit card primary account number (PAN) data. The company's requirement is to automatically obscure the PAN data, eliminate and merge certain fields, and then convert the data into JSON format for further internal processing. The solution also needs to be scalable to accommodate additional data feeds in the future.

The potential solutions to address these requirements are:

A. Trigger an AWS Lambda function upon the arrival of files, which parses each record and places it in an Amazon SQS queue. Subsequently, another Lambda function processes these records from the SQS queue, stores them temporarily in S3, and then a final Lambda function transforms the records into JSON format and transfers them to another S3 bucket for further processing.

B. Use an AWS Lambda function to process file arrivals, extracting each record and sending it to an Amazon SQS queue. Utilize an AWS Fargate container application, configured to scale based on the SQS queue messages, to process the records and transform them into JSON format. Once processed, the data is moved to a different S3 bucket, and the Fargate instance is scaled down.

C. Implement an AWS Glue crawler with a custom classifier matching the data feed formats to create a corresponding table definition. On file arrival, trigger an AWS Lambda function to initiate an AWS Glue ETL job, which processes and transforms the entire record batch as per the requirements and outputs the data in JSON format to another S3 bucket for internal processing.

D. Set up an AWS Glue crawler and a custom classifier to match the data feed formats and create a table definition. Trigger an Amazon Athena query upon file arrival, which initiates an Amazon EMR ETL job to process and transform the records according to the specified requirements and output them in JSON format to another S3 bucket. After processing, scale down the EMR cluster.

The most suitable solution is: C.

C. Implement an AWS Glue crawler with a custom classifier matching the data feed formats to create a corresponding table definition. On file arrival, trigger an AWS Lambda function to initiate an AWS Glue ETL job, which processes and transforms the entire record batch as per the requirements and outputs the data in JSON format to another S3 bucket for internal processing.

This option offers a scalable, managed ETL service that can handle the processing requirements with minimal operational overhead. AWS Glue is designed to easily manage transformations of large batches of data, making it ideal for the company's current and future needs.

Reasons why other options are less appropriate:

A. Using multiple Lambda functions with SQS may introduce complexity and potential limitations in handling large volumes of data, especially given the Lambda execution time limits.

B. While AWS Fargate provides scalability, the setup described involves more operational complexity and might not be as efficient in processing large batches of data compared to a dedicated ETL service like AWS Glue.

D. The combination of Amazon Athena, Amazon EMR, and AWS Glue is unnecessarily complex for the described requirement. EMR is typically used for more complex data processing tasks that might be overkill for this scenario.

Question 47

A company aims to establish a business continuity plan for its main on-premises application, which currently runs on physical servers alongside other applications. This critical application uses a MySQL database for data storage, and the operating systems used are compatible with Amazon EC2. The company is seeking an AWS-based solution that minimizes operational complexity to ensure continuity in the event of an application failure.

The proposed solutions for achieving this goal with minimal operational overhead are:

A. Implement the AWS Replication Agent on all source servers, including those hosting the MySQL database. Set up a replication process for these servers, periodically launch test instances for drills, and use these instances for failover during an actual failure event.

B. Install the AWS Replication Agent on the source servers, including the MySQL servers. Utilize AWS Elastic Disaster Recovery in the designated AWS Region, configure the launch settings, and regularly conduct failover and fallback exercises using the latest data points.

C. Set up AWS Database Migration Service (AWS DMS) replication instances and an Amazon Aurora MySQL DB cluster as the target database. Initiate a DMS task for initial data transfer to Aurora MySQL and employ AWS Schema Conversion Tool (AWS SCT) for ongoing data synchronization. For other software components, deploy on EC2 instances using suitable base AMIs.

D. Deploy AWS Storage Gateway Volume Gateway on-premises, and mount its volumes on all servers. Migrate the application and MySQL database to these volumes, regularly creating snapshots. Prepare software on EC2 instances using compatible AMIs. In case of failure, launch a Volume

Gateway on EC2, restore volumes from the latest snapshot, and mount them on the EC2 instances.

The most suitable solution is: B.

B. Install the AWS Replication Agent on the source servers, including the MySQL servers. Utilize AWS Elastic Disaster Recovery in the designated AWS Region, configure the launch settings, and regularly conduct failover and fallback exercises using the latest data points.

This option provides a streamlined and integrated approach for disaster recovery with minimal operational overhead. AWS Elastic Disaster Recovery automates the replication and recovery processes, ensuring that the company can quickly switch to AWS in the event of an on-premises failure, with the ability to regularly test the failover mechanism.

Reasons why other options are less appropriate:

A. While this option involves replication, it may require significant manual intervention for regular testing and during actual failover events, increasing operational complexity.

C. Using AWS DMS and AWS SCT for continuous data synchronization, along with setting up the rest of the software on EC2 instances, introduces more operational steps and complexity compared to a dedicated disaster recovery solution like AWS Elastic Disaster Recovery.

D. Implementing AWS Storage Gateway and managing snapshots require more manual configuration and operational maintenance. This approach is less streamlined and might not be as rapid or reliable for failover compared to an automated disaster recovery solution.

Question 48

A company needs to provide external auditors, who operate from a single AWS account, with secure, read-only access to its own AWS account for regulatory financial audits. The solution must align with AWS security best practices.

The potential solutions for this requirement are:

A. Implement resource policies for all resources in the company's AWS account to grant access to the auditors' AWS account, incorporating a unique external ID for each policy.

B. Establish an IAM role within the company's AWS account that establishes trust with the auditors' AWS account. Attach an IAM policy granting the necessary permissions to this role

and assign a unique external ID to the role's trust policy.

C. Create an IAM user in the company's AWS account, attach the necessary IAM policies for read-only access, and generate API access keys for this user. Share these access keys with the auditors.

D. Formulate an IAM group in the company's AWS account with the requisite permissions. Create individual IAM users for each auditor in the company's account and add these users to the IAM group.

The most appropriate solution is: B.

B. Establish an IAM role within the company's AWS account that establishes trust with the auditors' AWS account. Attach an IAM policy granting the necessary permissions to this role and assign a unique external ID to the role's trust policy.

This approach adheres to AWS security best practices by using IAM roles for cross-account access. It allows the auditors to assume the role with read-only permissions without the need to manage individual user credentials in the company's account. The external ID adds an additional layer of security to prevent the unauthorized assumption of the role.

Reasons why other options are less suitable:

A. Configuring resource policies for all resources to grant access can be operationally complex and difficult to manage, especially in a large AWS environment with multiple resources.

C. Creating IAM users and sharing access keys is not considered a best practice for cross-account access due to the security risks associated with distributing access keys.

D. Creating individual IAM users for each auditor in the company's account increases administrative overhead and does not align with the best practice of least privilege, as it could grant more access than necessary.

Question 49

A company operating a latency-sensitive trading platform uses Amazon DynamoDB for storage. They seek to enhance the platform's performance while maintaining high availability. The current DynamoDB table is configured in on-demand capacity mode. The task is to design a solution that optimally reduces latency for the trading platform.

The proposed solutions are:

A. Set up a two-node DynamoDB Accelerator (DAX) cluster and adjust the application to perform both read and write operations via DAX.

B. Establish a three-node DAX cluster, configure the application to read from DAX, and write directly to the DynamoDB table.

C. Implement a three-node DAX cluster, direct application reads from the DynamoDB table, and writes through DAX.

D. Create a single-node DAX cluster and configure the application to read from DAX, while writes are made directly to the DynamoDB table.

The optimal solution is: B.

B. Establish a three-node DAX cluster, configure the application to read from DAX, and write directly to the DynamoDB table.

This approach ensures the least latency for reads by utilizing a three-node DAX cluster, which provides caching to reduce read latency significantly. Writing directly to the DynamoDB table is typically more efficient as DAX primarily enhances read performance. The three-node cluster configuration enhances high availability and fault tolerance, crucial for a trading platform.

Reasons why other options are less appropriate:

A. A two-node DAX cluster might not offer the same level of high availability and fault tolerance as a three-node cluster, which is essential for a latency-sensitive trading platform.

C. Configuring the application to read directly from DynamoDB and write through DAX is not the most latency-efficient approach. DAX is designed to improve read latency, so it's best used for read operations.

D. A single-node DAX cluster does not provide the same level of high availability and fault tolerance as a three-node cluster. In a latency-sensitive and high-availability environment, having more nodes in the DAX cluster is preferable.

Question 50

A company has transitioned its application to AWS, where the frontend is a static site on two Amazon EC2 instances behind an Application Load Balancer (ALB), and the backend is a Python application on three EC2 instances behind another ALB. Originally sized for peak demand based on on-premises criteria, these are large, general-purpose On-Demand Instances. The application experiences high usage during lunchtime but otherwise has low traffic.

The task is to reduce infrastructure costs while maintaining application availability. The potential steps are (choose two):

A. Switch all EC2 instances to compute-optimized instances with a core count matching the current setup.

B. Host the application frontend as a static site on Amazon S3.

C. Utilize AWS Elastic Beanstalk for deploying the application frontend with the same instance type.

D. Convert all backend EC2 instances to Spot Instances.

E. Migrate the backend Python application to general-purpose, burstable EC2 instances with a core count matching the current setup.

The most cost-effective and suitable solution is: **B, E.**

B. Host the application frontend as a static site on Amazon S3.

E. Migrate the backend Python application to general-purpose, burstable EC2 instances with a core count matching the current setup.

Reasons for these choices:

B. Hosting a static website on Amazon S3 is more cost-effective than using EC2 instances. It provides high availability and scalability without the overhead of managing servers.

E. Using general-purpose, burstable EC2 instances for the backend caters to the variable traffic patterns, providing sufficient performance during peak times while reducing costs during low-traffic periods.

Reasons why other options are less appropriate:

A. Switching to compute-optimized instances may not yield significant cost savings since they are designed for compute-bound applications and might be overkill for this use case.

C. Using AWS Elastic Beanstalk with the same instance type doesn't directly address the cost optimization need, as it maintains the current instance sizing and type.

D. While Spot Instances can be cost-effective, they may not provide the necessary availability for the application backend, especially if the application cannot handle interruptions effectively.

Question 51

A company operates an event ticketing platform on AWS, leveraging Amazon Elastic Kubernetes Service (Amazon EKS) with Amazon EC2 and an Amazon RDS for MySQL DB instance. The platform is expanding to incorporate new features on Amazon EKS using AWS

Fargate. Demand for the platform fluctuates, with sporadic high peaks tied to event dates. The goal is to enhance the platform's cost-efficiency.

The potential solutions are:

A. Secure Standard Reserved Instances for the EKS cluster's EC2 instances to cover the baseline load. Use Spot Instances for peak demand. Opt for 1-year All Upfront Reserved Instances for the RDS database, aligning with the highest anticipated yearly demand.

B. Commit to Compute Savings Plans for the EKS cluster's medium-level predicted load. Use On-Demand Capacity Reservations for peak times based on event schedules. Select 1-year No Upfront Reserved Instances for the RDS database to cater to the expected base load and scale out read replicas during peak periods.

C. Opt for EC2 Instance Savings Plans for the EKS cluster's base load. Utilize Spot Instances for peak demand surges. Acquire 1-year All Upfront Reserved Instances for the RDS database, based on the base load, and manually scale up the DB instance during peak times.

D. Invest in Compute Savings Plans for the EKS cluster's base load. Use Spot Instances for demand peaks. Secure 1-year All Upfront Reserved Instances for the RDS database, targeting the base load, and manually scale up the DB instance as needed during peaks.

The most cost-effective solution is: **B.**

B. Commit to Compute Savings Plans for the EKS cluster's medium-level predicted load. Use On-Demand Capacity Reservations for peak times based on event schedules. Select 1-year No Upfront Reserved Instances for the RDS database to cater to the expected base load and scale out read replicas during peak periods.

Reasons for this choice:

- Compute Savings Plans offer flexibility and cost savings for a medium load, aligning with varying demands.
- On-Demand Capacity Reservations provide the necessary resources for predictable peak times without long-term commitment.
- 1-year No Upfront Reserved Instances for the RDS database offer cost savings for the base load with the flexibility of scaling out read replicas during peaks, ensuring performance without overcommitting resources.

Reasons why other options are less suitable:

A. Standard Reserved Instances might not offer the necessary flexibility for fluctuating demands, and All Upfront payments lack cost flexibility.

C. EC2 Instance Savings Plans are less flexible than Compute Savings Plans. Scaling the DB instance manually during peaks might not be as effective as using read replicas.

D. This option is similar to B but involves All Upfront payments, which lack cost flexibility compared to No Upfront options.

Question 52

A company has implemented an application on AWS Elastic Beanstalk with Amazon Aurora for database operations, and Amazon CloudFront for handling web requests. This CloudFront setup uses the Elastic Beanstalk domain as the main server and includes an alternate domain name for user access. The company performs routine maintenance weekly, during which the application is temporarily offline. To prevent showing CloudFront error messages to visitors in these periods, they plan to display an informative message instead. An Amazon S3 bucket has already been established for this purpose.

The solutions architect needs to determine the subsequent steps to fulfill this requirement. The options are (choose three):

A. Upload static informational content to the S3 bucket.

B. Create a new CloudFront distribution with the S3 bucket as the origin.

C. Add the S3 bucket as a secondary origin in the existing CloudFront distribution, using an origin access identity (OAI) for both the distribution and the S3 bucket.

D. Temporarily switch the default cache behavior in the current CloudFront distribution to the S3 origin during maintenance, and revert after maintenance is complete.

E. During maintenance, add a cache behavior for the S3 origin in a new CloudFront distribution, adjust the path pattern and precedence, and remove it post-maintenance.

F. Configure Elastic Beanstalk to direct traffic to the S3 bucket during maintenance.

The correct choices are **A, C, and D,** for the following reasons:

A. **Uploading static content to the S3 bucket**: This is necessary to provide the informational message during maintenance. The S3 bucket will host the static message or web page for visitors.

C. **Setting the S3 bucket as a second origin in the current CloudFront distribution, with OAI**: This step is crucial for smoothly alternating between the main application and the maintenance message. By adding the S3 bucket as an additional origin, the CloudFront distribution can redirect requests to the S3 bucket when necessary, using the origin access identity for secure and distinct access to both the application and the S3 content.

D. Modifying the default cache behavior to the S3 origin during maintenance: This allows the CloudFront distribution to serve the static content from the S3 bucket instead of the application during maintenance. Reverting to the original settings post-maintenance ensures normal application delivery resumes.

The reasons the other options are not suitable:

B. Creating a new CloudFront distribution for the S3 bucket: This adds unnecessary complexity, as the existing distribution can already be configured to serve both the application and the maintenance message.

E. Setting up a new cache behavior on a new distribution during maintenance: Similar to B, creating an entirely new distribution for this purpose is redundant and adds unnecessary complexity.

F. Configuring Elastic Beanstalk to serve traffic from the S3 bucket: This approach is impractical, as Elastic Beanstalk is meant for managing application environments, not for directly serving static content from an S3 bucket.

Question 53

A business has an application that allows users to upload images. These uploads trigger an AWS Lambda function, which then processes and saves the images in an Amazon S3 bucket. The application calls this Lambda function using a specific function version ARN (Amazon Resource Name). The function uses environment variables to take in parameters for image processing. The business frequently alters these variables to fine-tune the image processing results. After testing various parameters, the business publishes a new version of the Lambda function with the updated environment variables. However, this necessitates changes in the application to point to the new function version ARN, leading to user disruptions.

The task for a solutions architect is to streamline this process, reducing user interruptions and minimizing operational overhead. The proposed solutions are:

A. Alter the environment variables of the already published Lambda function version. Utilize the $LATEST version for testing image processing parameters.

B. Set up an Amazon DynamoDB table to keep the image processing parameters. Adjust the Lambda function to fetch these parameters from the DynamoDB table.

C. Embed the image processing parameters directly in the Lambda function code, eliminating the need for environment variables. Release a new function version whenever there's an update to these parameters.

D. Implement a Lambda function alias and modify the client application to use the ARN of this alias. Update the Lambda alias to point to new function versions following successful testing.

The most suitable solution is D, for these reasons:

- Using a Lambda function alias: This approach allows the business to update the Lambda function without needing to change the application's Lambda function ARN each time. The alias acts as a reference point that can be reconfigured to point to different function versions as needed. This method significantly reduces operational overhead and user disruptions since the application does not require frequent updates.

The other options have notable drawbacks:

A. Modifying environment variables of the published version: This method does not adhere to best practices for version control and can lead to unpredictable behaviors in a production environment. Using the $LATEST version for testing is risky as it might affect live operations.

B. Using a DynamoDB table for parameters: While this approach centralizes parameter management, it adds complexity and potential latency since the Lambda function must query DynamoDB each time it runs. This also introduces additional costs and maintenance requirements for the DynamoDB table.

C. Hardcoding parameters in the Lambda function: This method eliminates the flexibility provided by environment variables. Every change would require updating and publishing a new version of the Lambda function, followed by thorough testing, which is time-consuming and less efficient.

Question 54

A leading global media company is in the process of setting up a multi-regional application deployment, leveraging Amazon DynamoDB global tables to ensure consistent user experiences across two main continents where their user base is concentrated. The deployment in each region will feature a public Application Load Balancer (ALB). The company currently manages its public DNS in-house and intends to make the application accessible via an apex domain. The goal is to find a solution that fulfills these requirements with minimal effort.

The proposed solutions to this scenario are:

A. Transition the public DNS management to Amazon Route 53. Establish CNAME records for the apex domain to redirect to the ALB, and apply a geolocation routing policy for directing traffic based on the user's location.

B. Implement a Network Load Balancer (NLB) ahead of the ALB. Transfer public DNS management to Amazon Route 53. Create a CNAME record for the apex domain that points to the NLB's static IP address, utilizing a geolocation routing policy for traffic routing based on user location.

C. Set up an AWS Global Accelerator with multiple endpoint groups targeting the appropriate AWS Regions. Use the static IP address provided by the accelerator to create a record in the public DNS for the apex domain.

D. Develop an Amazon API Gateway API backed by AWS Lambda in one of the AWS Regions. Configure a Lambda function to distribute traffic to the application deployments using a round-robin method. Establish CNAME records for the apex domain to direct to the API's URL.

The most efficient solution is C, for the following reasons:

- Using AWS Global Accelerator: This service simplifies the management of global traffic through static IP addresses that are network border-independent. By creating endpoint groups in relevant AWS Regions and pointing them to the respective ALBs, the Global Accelerator automatically routes traffic to the nearest optimal region, enhancing performance and reliability. This approach eliminates the need for complex DNS configurations and ensures apex domain compatibility with minimal setup and maintenance efforts.

The other options are less suitable due to these drawbacks:

A. Migrating to Route 53 with CNAME records and geolocation policy: While Route 53 offers advanced DNS management, using CNAME records for apex domains is not natively supported. Additionally, managing geolocation routing policies can be more complex and less efficient compared to the automated routing provided by AWS Global Accelerator.

B. Implementing NLB and migrating DNS to Route 53: Like option A, this also involves complex DNS management. Moreover, NLBs are typically used for non-HTTP/S traffic and might not be the optimal choice for a media company primarily dealing with web-based content.

D. Creating an API Gateway API with Lambda for routing: This setup introduces unnecessary complexity and potential performance bottlenecks. Managing traffic through Lambda and API Gateway is less straightforward and efficient compared to using AWS Global Accelerator, which is designed for high-performance global traffic management.

Question 55

A business is in the process of creating a new serverless API using Amazon API Gateway and AWS Lambda. Their Lambda functions, integrated with the API Gateway, utilize several shared libraries and custom classes. The task at hand for a solutions architect is to streamline the deployment of this system and enhance code reusability.

Here are the rephrased options for achieving this objective:

A. Create a Docker image containing the shared libraries and custom classes, then store this image in an Amazon S3 bucket. Subsequently, construct a Lambda layer sourcing from this Docker image. The Lambda functions for the API should be deployed as Zip packages configured to utilize this Lambda layer.

B. Build a Docker image with the shared libraries and custom classes and upload it to Amazon Elastic Container Registry (Amazon ECR). Formulate a Lambda layer that derives from this Docker image. Deploy the API's Lambda functions in Zip packages, setting them up to leverage the Lambda layer.

C. Deploy the shared libraries and custom classes within a Docker container on Amazon Elastic Container Service (Amazon ECS) using the AWS Fargate launch type. Then, deploy the API's Lambda functions as Zip packages, configuring them to use this container as a source for a Lambda layer.

D. Package the shared libraries, custom classes, and the API's Lambda function code into a Docker image. Upload this image to Amazon Elastic Container Registry (Amazon ECR) and set up the API's Lambda functions to directly use this Docker image as their deployment package.

The most suitable solution is D, for these reasons:

- Using a Docker image for complete deployment: By packaging all necessary components—shared libraries, custom classes, and Lambda function code—into a single Docker image and storing it in Amazon ECR, the deployment process becomes more streamlined and efficient. This approach ensures that all elements of the Lambda functions are bundled together, reducing the complexity of managing separate layers or dependencies.

The other options have specific limitations:

A. Storing a Docker image in S3 and creating a Lambda layer: Lambda layers sourced from Docker images cannot directly utilize images stored in S3. This approach is technically unfeasible in the AWS ecosystem.

B. Creating a Lambda layer from a Docker image in ECR: While storing Docker images in ECR is a valid approach, using these images to create Lambda layers adds unnecessary

complexity. Lambda layers are more suited for lightweight dependencies and libraries, not full Docker images.

C. Using ECS with Fargate for Lambda layers: Deploying shared libraries and classes in an ECS container and then using them as a Lambda layer is not a straightforward or standard approach. Lambda layers don't directly support usage of ECS containers, and this method complicates the deployment process unnecessarily.

Question 56

A manufacturing firm is developing a system to inspect products at its factory. The factory is equipped with IP cameras at the end of each assembly line. The company has trained a machine learning (ML) model using Amazon SageMaker to recognize common defects in products from still images. To enhance the process, the company aims to give immediate, on-site feedback to factory workers when defects are identified. Crucially, this feedback system must operate even during internet outages. The company maintains a local Linux server that runs an API, which delivers this direct feedback to the workers. The challenge is to determine the most effective way to deploy the ML model to fulfill these specific needs.

The possible deployment strategies are:

A. Utilize Amazon Kinesis to stream video from each IP camera to AWS. Use Amazon EC2 instances to capture still images from these streams and upload them to an Amazon S3 bucket. Set up a SageMaker endpoint with the ML model. Trigger an AWS Lambda function to call the inference endpoint whenever new images are uploaded, and program the Lambda function to interact with the local API if a defect is detected.

B. Install AWS IoT Greengrass on the local server and deploy the ML model there. Create a Greengrass component to capture still images from the cameras and perform inference. Configure this component to activate the local API if it identifies a defect.

C. Acquire an AWS Snowball device. Deploy the SageMaker endpoint with the ML model and an Amazon EC2 instance on this device. Capture still images from the cameras and run inference using the EC2 instance, setting it up to communicate with the local API upon detecting defects.

D. Set up Amazon Monitron sensors on each IP camera and an Amazon Monitron Gateway locally. Deploy the ML model to the Monitron devices. Utilize Amazon Monitron health state alarms to trigger an AWS Lambda function, which then calls the local API when defects are identified.

The most appropriate solution is B, for these reasons:

- Using AWS IoT Greengrass for Local Deployment: This option allows for the deployment of the ML model directly on the local server, ensuring that the system can operate independently of internet connectivity. The Greengrass component can process images and run the ML inference locally, and immediately trigger the local API upon detecting defects. This setup is efficient, reduces latency, and fulfills the requirement of functioning during internet outages.

The other options are less suitable for these reasons:

A. Streaming via Amazon Kinesis and using Lambda: This solution is heavily reliant on internet connectivity to stream video to AWS and to trigger Lambda functions, which does not meet the requirement of operating during internet outages.

C. Utilizing AWS Snowball with EC2 and SageMaker: While this approach supports local processing, it's an overly complex and costly solution compared to IoT Greengrass. Snowball is typically used for large-scale data transport, not for continuous, real-time inference.

D. Employing Amazon Monitron with Lambda: This method is designed more for equipment monitoring and not specifically for real-time image processing and defect detection. It also relies on Lambda functions, which would require internet connectivity, contradicting the requirement of offline functionality.

Question 57

A solutions architect is tasked with developing a proposal for transitioning a company's physical data center to the AWS Cloud. The primary tool at their disposal is an export from a configuration management database (CMDB) that details all of the company's servers. The goal is to identify a method that analyzes this data in a way that is both effective and economical.

The potential solutions for this situation are:

A. Utilize the AWS Well-Architected Tool to input the CMDB data, conduct an analysis, and generate relevant recommendations.

B. Employ the Migration Evaluator to conduct an analysis, using a specific data import template to incorporate the data from the CMDB export.

C. Apply resource matching rules, utilizing the CMDB export alongside the AWS Price List Bulk API to compare CMDB data against AWS services in a comprehensive manner.

D. Make use of the AWS Application Discovery Service to import and analyze the CMDB data.

The most cost-effective solution is B:

- Using the Migration Evaluator with a Data Import Template: The Migration Evaluator (formerly known as TSO Logic) is specifically designed for analyzing on-premises environments and projecting costs for moving to AWS. It can efficiently process CMDB exports, making it a suitable choice for creating a detailed, accurate business case for cloud migration. The use of a tailored data import template simplifies the process of integrating CMDB data into the analysis.

The other options have certain drawbacks:

A. AWS Well-Architected Tool: While this tool is valuable for reviewing AWS workloads against best practices, it is not specifically designed for detailed cost analysis or migration planning from on-premises environments. It might not efficiently handle CMDB exports for this purpose.

C. Resource Matching with AWS Price List Bulk API: This approach involves a more manual process of matching resources and analyzing costs, which could be more time-consuming and less accurate compared to using a specialized tool like Migration Evaluator.

D. AWS Application Discovery Service: This service is more focused on discovering and collecting data about on-premises infrastructure. While it could be used to import CMDB data, it is not primarily geared towards cost analysis and migration planning, which are key aspects of creating a business case for cloud migration.

Question 58

A company operates a website hosted on Amazon EC2 instances, which are managed by an Auto Scaling group and fronted by an Application Load Balancer (ALB). An AWS WAF web ACL is linked to this ALB. The website frequently experiences application layer attacks, leading to abrupt and substantial spikes in traffic. These attacks are characterized by their origination from various IP addresses, as observed in the access logs. The company seeks a solution to effectively counter these attacks with minimal operational complexity.

The proposed strategies to address this challenge are:

A. Implement an Amazon CloudWatch alarm to track server access, setting a threshold based on the frequency of access by each IP address. Configure the alarm to automatically add IP addresses exceeding this threshold to the web ACL's deny list.

B. Incorporate AWS Shield Advanced alongside the existing AWS WAF, and designate the ALB as a protected resource under Shield Advanced.

C. Establish an Amazon CloudWatch alarm to oversee user IP addresses, creating a threshold based on access frequency per IP address. Set up this alarm to trigger an AWS Lambda

function, which then implements a deny rule in the application server's subnet route table for the IPs causing the alarm.

D. Analyze the access logs to identify patterns in the attacking IP addresses. Utilize Amazon Route 53's geolocation routing policy to block traffic from countries where these IP addresses are located.

The most appropriate solution is B:

- **Deploying AWS Shield Advanced with AWS WAF**: AWS Shield Advanced provides additional protections against more sophisticated and larger scale DDoS attacks, particularly at the application layer. By integrating it with AWS WAF and protecting the ALB, the company can benefit from enhanced security without significant increases in operational management. AWS Shield Advanced automatically mitigates large-scale and sophisticated attacks, minimizing the need for manual intervention or complex configurations.

The other options are less suitable due to these reasons:

A. **Creating a CloudWatch alarm with auto-update to web ACL**: This approach might not be effective against distributed attacks from multiple IP addresses and could result in a high number of false positives. Managing and updating the ACL based on IP addresses would also entail significant operational overhead.

C. **CloudWatch alarm with Lambda function for subnet deny rules**: This method introduces complexity in managing IP-based deny rules at the subnet level. It could lead to unintentional network access issues and does not scale well for distributed attacks.

D. **Route 53 geolocation policy for blocking countries**: Blocking traffic based on country of origin might not be effective if the attack sources are globally distributed. Additionally, this approach could inadvertently block legitimate users and does not address the problem of sophisticated application-layer attacks.

Question 59

A company is operating a vital application with its data layer hosted in a single AWS Region. This layer comprises an Amazon DynamoDB table and an Amazon Aurora MySQL DB cluster, with the Aurora MySQL version supporting global databases. The application layer is already functioning across two Regions. According to the company's policy, key applications must have both their application and data tiers spread over two Regions, with a Recovery Time Objective (RTO) and Recovery Point Objective (RPO) of just a few minutes each. The task for the solutions architect is to propose a strategy to align the data tier with this policy.

The potential actions to achieve this are (choose two):

A. Extend the Aurora MySQL DB cluster to include an additional Region.

B. Incorporate an additional Region for each table within the Aurora MySQL DB cluster.

C. Implement routine cross-Region backups for both the DynamoDB table and the Aurora MySQL DB cluster.

D. Transform the existing DynamoDB table into a global table by integrating an additional Region into its configuration.

E. Employ Amazon Route 53 Application Recovery Controller for automating database backups and recovery processes to a secondary Region.

The correct choices are A and D, for these reasons:

- A. Extending the Aurora MySQL DB Cluster to Another Region: By adding another Region to the Aurora MySQL DB cluster, the company can achieve cross-Region replication. This aligns with the policy of having data tier components in multiple Regions and helps in meeting the low RTO and RPO requirements, as the data is continuously replicated.
- D. Converting the DynamoDB Table to a Global Table: Adding another Region to the existing DynamoDB table configuration to create a global table ensures real-time replication across Regions. This step is essential for meeting the company's policy for critical applications and ensures minimal RTO and RPO.

The other options are less effective:

- B. Adding an Additional Region for Each Table in the Cluster: Aurora does not replicate at the table level but at the cluster level. Therefore, adding another Region for each table is not a feasible approach in Aurora's architecture.
- C. Setting Up Scheduled Cross-Region Backups: While backups are crucial for disaster recovery, they do not support the requirement of a few minutes of RTO and RPO, as restoring from backups can be time-consuming.
- E. Using Route 53 Application Recovery Controller: This tool is more focused on orchestrating application recovery rather than providing real-time data replication. It wouldn't be as effective in achieving the low RTO and RPO requirements for the data tier as the options A and D.

Question 60

A telecommunications firm is utilizing AWS to run an application, with an established AWS Direct Connect link bridging their on-premises data center to AWS. The application is hosted on Amazon EC2 instances, distributed across multiple Availability Zones, and managed by an

internal Application Load Balancer (ALB). The setup includes multiple target groups and path-based routing for URL path-based request forwarding. Clients from the on-premises network access the application via HTTPS, with the ALB handling TLS termination.

The company is now introducing an on-premises firewall appliance that operates on an IP address-based allow list. The challenge is to devise a strategy that ensures uninterrupted client access to the AWS-hosted application, in line with the new firewall configuration.

The proposed solutions to address this requirement are:

A. Adjust the existing ALB to use fixed IP addresses, assigning IP addresses in various Availability Zones to the ALB. Incorporate these IP addresses into the firewall appliance's allow list.

B. Implement a Network Load Balancer (NLB), associating it with static IP addresses across multiple Availability Zones. Set up an ALB-type target group for the NLB, including the existing ALB. Register the NLB IP addresses in the firewall appliance's allow list and reconfigure clients to connect through the NLB.

C. Establish a Network Load Balancer (NLB) with static IP addresses in different Availability Zones. Link the current target groups to the NLB. Shift client connections to use the NLB and remove the ALB. Add the NLB IP addresses to the firewall appliance.

D. Deploy a Gateway Load Balancer (GWLB), assigning it static IP addresses in several Availability Zones. Create an ALB-type target group for the GWLB, adding the current ALB. Include the GWLB IP addresses in the firewall appliance's allow list and update clients to connect via the GWLB.

The most suitable option is B:

- Implementing a Network Load Balancer with Static IP Addresses: NLBs support static IP addresses, making them ideal for integration with the firewall's IP address-based allow list. By establishing an ALB-type target group within the NLB and including the existing ALB, the company can maintain its current routing and load balancing configuration. This setup allows for seamless integration with the firewall appliance, ensuring continuous application accessibility for clients without significant changes to the existing infrastructure.

The other options are less effective:

A. Static IP Addresses for ALB: ALBs do not support static IP addresses. This option is technically unfeasible in the AWS environment.

C. NLB with Target Groups and Removing ALB: This approach unnecessarily complicates the existing setup by removing the ALB, which is already configured for path-based routing and target group management.

D. Using GWLB with Static IP Addresses: While GWLBs support static IP addresses, they are primarily designed for integrating and managing third-party virtual appliances in the cloud. This solution would introduce unnecessary complexity and is not the best fit for the company's requirements of simple client access and path-based routing.

Question 61

A business operates an application on a group of Amazon EC2 instances, which are located in private subnets and accessed via an internet-facing Application Load Balancer (ALB). This ALB also serves as the origin for an Amazon CloudFront distribution. The CloudFront distribution is safeguarded by an AWS WAF web ACL equipped with several AWS-managed rules. The company seeks a method to restrict direct internet access to the ALB, aiming for a solution with minimal operational complexity.

The proposed solutions to achieve this goal are:

A. Develop a new web ACL mirroring the rules of the existing one and link this new ACL with the ALB.

B. Link the current web ACL with the ALB.

C. Implement a security group rule for the ALB that permits only traffic originating from the AWS-managed prefix list for CloudFront.

D. Configure a security group rule on the ALB to exclusively allow traffic from CloudFront's IP address ranges.

The most appropriate solution is C:

- C. Using a Security Group Rule with AWS Managed Prefix List for CloudFront: This option effectively restricts the ALB to only accept traffic coming from CloudFront. By utilizing the AWS-managed prefix list, which automatically updates to include CloudFront's current IP ranges, the need for manual updates is eliminated. This approach ensures that the ALB is accessible only through CloudFront, fulfilling the requirement with the least operational overhead.

The other options have certain drawbacks:

A. Creating and Associating a New Web ACL with the ALB: While associating a web ACL with the ALB provides additional security, it doesn't directly prevent internet traffic from

accessing the ALB. This option also involves duplicating the existing ACL, which adds unnecessary complexity.

B. Associating the Existing Web ACL with the ALB: Like option A, this method adds security but does not prevent direct internet access to the ALB. The purpose of the web ACL is to filter unwanted traffic, not to restrict the source of traffic.

D. Adding a Security Group Rule for CloudFront IP Ranges: Manually configuring a security group to allow only CloudFront IP ranges would require constant updates to reflect changes in CloudFront's IP addresses. This approach is less efficient and involves more operational overhead compared to using the managed prefix list.

Question 62

A business is utilizing Amazon ElastiCache for Redis as a caching layer in their application. A recent security review found that while the company has enabled encryption at rest for ElastiCache, it has not activated encryption in transit. Moreover, the cache is accessible without any form of user authentication. To enhance security, the company needs to implement a solution that not only mandates user authentication but also ensures complete encryption, both at rest and in transit.

The potential solutions for enhancing the security of the ElastiCache setup are:

A. Generate an AUTH (authentication) token and store it as an encrypted parameter in AWS System Manager Parameter Store. Establish a new ElastiCache cluster with AUTH enabled and encryption in transit activated. Modify the application to access the AUTH token from Parameter Store as needed and utilize it for authentication.

B. Produce an AUTH token and keep it in AWS Secrets Manager. Adapt the existing ElastiCache cluster to employ this AUTH token and enable encryption in transit. Update the application to fetch the AUTH token from Secrets Manager when required and use it for authentication.

C. Create an SSL (Secure Socket Layer) certificate and save it in AWS Secrets Manager. Form a new ElastiCache cluster with encryption in transit configured. Revise the application to retrieve the SSL certificate from Secrets Manager as needed and apply it for authentication.

D. Develop an SSL certificate and store it as an encrypted advanced parameter in AWS Systems Manager Parameter Store. Modify the current ElastiCache cluster to support encryption in transit. Adjust the application to obtain the SSL certificate from Parameter Store when necessary and use it for authentication.

The most suitable option is B:

- B. Using AWS Secrets Manager for Storing AUTH Token: This approach effectively addresses both the lack of authentication and the absence of encryption in transit. By storing the AUTH token in AWS Secrets Manager, a secure and managed environment for storing secrets is utilized. The existing ElastiCache cluster can be configured to require this AUTH token for access, thereby adding authentication. Additionally, enabling encryption in transit on the existing cluster ensures that data is encrypted while moving between the application and the cache. This solution is efficient as it avoids the need to create a new cluster and makes use of the existing infrastructure with minimal changes.

The reasons why other options are less appropriate:

A. Creating a New Cluster with System Manager Parameter Store: This solution involves creating a new cluster, which might be unnecessary and could lead to additional overhead. Also, using System Manager Parameter Store, while secure, is not as specialized for managing secrets as AWS Secrets Manager.

C. Creating a New Cluster with SSL Certificate in Secrets Manager for Authentication: SSL certificates are used for encryption, not for user authentication. This approach is a misunderstanding of how SSL certificates and authentication mechanisms work in ElastiCache.

D. Updating Existing Cluster with SSL Certificate in Parameter Store for Authentication: Similar to option C, this misunderstands the role of SSL certificates. They do not serve as an authentication mechanism for accessing the cache.

Question 63

A business is operating a computational workload on Amazon EC2, utilizing Spot Instances within an Auto Scaling group. The configuration for launching these instances involves two placement groups and a single instance type as specified in the launch template. Recently, issues with launching instances from the Auto Scaling group have been observed, as indicated by the monitoring system. These issues have been linked to increased wait times for users. The company is looking for a strategy to enhance the reliability of their workload.

The potential solutions to address this challenge are:

A. Switch out the current launch template for a launch configuration that enables the Auto Scaling group to select instances based on their attributes.

B. Introduce a new version of the launch template that incorporates attribute-based instance type selection. Set this updated template version as the one for the Auto Scaling group to use.

C. Modify the existing launch template for the Auto Scaling group by adding more placement groups.

D. Alter the launch template to specify a larger instance type.

The best solution is B:

- B. Creating a New Launch Template Version with Attribute-Based Instance Type Selection: This option enhances the flexibility and reliability of the workload by allowing the Auto Scaling group to choose from a broader range of instance types, rather than being restricted to just one. By using attribute-based instance type selection, the Auto Scaling group can launch instances that are similar but not identical to the primary type, thereby reducing the likelihood of instance launch failures due to availability issues with a specific instance type.

The other options are less effective for these reasons:

A. Replacing Launch Template with Launch Configuration: Although a launch configuration could allow for some flexibility, it is generally less versatile than a launch template. Launch templates offer more advanced features, including versioning and attribute-based instance type selection.

C. Increasing Number of Placement Groups in Launch Template: Adding more placement groups doesn't address the core issue of instance type availability. Placement groups are more about controlling the physical location of instances for performance optimization, not about improving instance availability.

D. Changing to a Larger Instance Type in Launch Template: Merely switching to a larger instance type may not solve the problem of instance availability. It could potentially worsen the situation if the larger instances are less readily available or more expensive. This approach does not address the underlying issue of relying on a single instance type.

Question 64

A business is transitioning its document processing workload to AWS. Several of their applications have been updated to directly interact with Amazon S3 for storing, retrieving, and altering documents. The processing server, producing around 5 documents per second, needs to make these documents available for customer download from Amazon S3 within 30 minutes of processing completion. However, during migration, it was identified that the processing server, which requires quick local file access and runs on Linux, cannot be immediately updated to support the S3 API.

The task is to find a solution that aligns with these needs while requiring minimal effort. The proposed options are:

A. Shift the application to an AWS Lambda function, utilizing the AWS SDK for Java for creating, modifying, and accessing files directly in Amazon S3.

B. Implement an Amazon S3 File Gateway, setting up a file share connected to the document store. Use NFS to mount this file share on an Amazon EC2 instance. Employ the RefreshCache API call to update the S3 File Gateway when modifications are made in Amazon S3.

C. Set up Amazon FSx for Lustre with an import/export policy and link this new file system to an S3 bucket. Install the Lustre client and use NFS to mount the document store on an Amazon EC2 instance.

D. Use AWS DataSync to establish a connection with an Amazon EC2 instance, and configure a task to sync the generated files to and from Amazon S3.

The most suitable solution is B:

- B. Amazon S3 File Gateway: This solution offers a seamless way to integrate the existing server setup with Amazon S3. S3 File Gateway acts as a bridge, allowing the server to interact with the S3 storage using familiar file system protocols like NFS. Files written to the NFS mount are automatically synchronized to S3, meeting the requirement for the files to be available for download within 30 minutes. This approach does not require major changes to the existing server or applications, thus involving the least amount of effort.

The reasons why the other options are less appropriate:

A. Migrating to AWS Lambda: This would involve significant changes to the application architecture and may not be feasible for a workload that produces documents at a high rate and requires fast local file access.

C. Configuring Amazon FSx for Lustre: FSx for Lustre is optimized for high-performance computing scenarios and might be overkill for this situation. It also requires the installation and configuration of the Lustre client, adding complexity.

D. Using AWS DataSync: While DataSync could synchronize files to and from S3, it's typically used for data transfer tasks like data migration or periodic syncing, rather than real-time file access and synchronization. This makes it less ideal for a setup where immediate access to files in S3 is required.

Question 65

A logistics company operates a serverless architecture in the AWS Cloud to manage various aspects of its operations, including user data, delivery specifics, and historical purchase

information. The architecture is composed of multiple microservices. A key microservice stores sensitive user data in an Amazon DynamoDB table, while other microservices hold duplicates of parts of this sensitive data across different storage solutions.

The company is required to promptly erase user data upon request. This means that when the primary user service deletes a user's record from the DynamoDB table, all other microservices must immediately remove their copies of the related data.

To fulfill these needs, the following options are considered:

A. Enable DynamoDB Streams for the DynamoDB table. Set up an AWS Lambda function triggered by these streams to send deletion events to an Amazon Simple Queue Service (Amazon SQS) queue. Each microservice should then regularly check this queue and act to delete the user data from the DynamoDB table.

B. Implement DynamoDB event notifications for the table. Establish an Amazon Simple Notification Service (Amazon SNS) topic as a destination for these notifications. Each microservice should subscribe to this SNS topic and respond by removing the user data from the DynamoDB table upon notification.

C. Modify the central user service to broadcast an event on a custom Amazon EventBridge bus when a user is deleted. Create individual EventBridge rules for each microservice that recognize this specific event pattern and trigger the necessary logic in each microservice to delete the user data from the DynamoDB table.

D. Adjust the central user service to place a message in an Amazon SQS queue whenever a user is deleted. Direct each microservice to monitor this queue and respond by deleting the user data from the DynamoDB table.

The most effective solution is B:

- B. DynamoDB Event Notifications with SNS: This option leverages the integration of DynamoDB with SNS to disseminate user deletion notifications. When a user is deleted, an event notification is sent to an SNS topic. As microservices are subscribers to this topic, they receive these notifications immediately and can promptly act to delete the corresponding user data. This setup offers a real-time, push-based mechanism to ensure timely data deletion across all microservices.

The reasons other options are less suitable:

A. DynamoDB Streams with Lambda and SQS: While DynamoDB Streams accurately capture table changes, relying on microservices to continuously poll an SQS queue introduces potential delays in data deletion, as well as increased complexity and resource usage.

C. EventBridge for Inter-Service Communication: Although EventBridge is a robust solution for event-driven architectures, it involves more setup and configuration compared to the direct integration of DynamoDB with SNS. This method might be more complex than necessary for this specific use case.

D. Central User Service Posting to SQS: Similar to option A, this approach relies on the polling mechanism of SQS, which could lead to delays in data deletion. It also adds an extra layer of complexity by requiring the central user service to manage queue messages.

Question 66

A business operates a web application within a Virtual Private Cloud (VPC) on AWS. This application is served by a group of Amazon EC2 instances, which are managed by an Application Load Balancer (ALB) equipped with AWS WAF for added security. An external client must establish a connection to this web application, and for this purpose, the company needs to supply specific IP addresses to all external clients.

To address this requirement while minimizing operational complexity, the following solutions are proposed:

A. Switch the ALB to a Network Load Balancer (NLB) and assign an Elastic IP address to the NLB. Provide this Elastic IP address to the external customer.

B. Allocate an Elastic IP address and associate it with the ALB. Share this Elastic IP address with the customer.

C. Implement an AWS Global Accelerator standard accelerator, designating the ALB as the endpoint of this accelerator. Distribute the accelerator's IP addresses to the customer.

D. Set up an Amazon CloudFront distribution with the ALB as its origin. Determine the distribution's public IP address by pinging its DNS name, then give this IP address to the customer.

The most suitable solution is C:

- C. AWS Global Accelerator with ALB as Endpoint: AWS Global Accelerator provides static IP addresses that can be used by clients to access the web application. By configuring the ALB as the endpoint for the Global Accelerator, the static IP addresses assigned to the accelerator can be shared with the external customer. This approach offers a scalable and reliable solution with minimal overhead, as it doesn't require replacing the existing ALB or managing Elastic IP addresses.

The reasons other options are less suitable:

A. Replacing ALB with NLB and Assigning Elastic IP: While NLBs support Elastic IPs, replacing the ALB with an NLB might involve significant architectural changes and potential feature loss, especially if the ALB's advanced routing capabilities or integration with AWS WAF are being utilized.

B. Assigning Elastic IP to ALB: Elastic IPs cannot be directly associated with ALBs. ALBs are designed to be accessed via their DNS name and do not support static IP addresses directly.

D. Using CloudFront Distribution: CloudFront distributions use a network of edge locations and do not have a dedicated, static public IP address. Pinging the CloudFront distribution's DNS name to find a public IP address is not a reliable method as the IP address can change and is not meant to be static. This approach also adds unnecessary complexity and potential latency, especially for a straightforward web application hosting scenario.

Question 67

A business currently operates several AWS accounts for development purposes and plans to transition its production application to AWS. The objective is to ensure that Amazon Elastic Block Store (Amazon EBS) encryption at rest is mandated in both existing and future production accounts. The company seeks a solution that leverages predefined templates and compliance controls.

To achieve this, a combination of steps involving organizational structuring and policy enforcement is required. The potential steps are choose three:

A. Implement AWS CloudFormation StackSets to distribute AWS Config rules across production accounts.

B. Establish a new AWS Control Tower landing zone within an existing developer account, create Organizational Units (OUs) for different account types, and categorize production and development accounts into their respective OUs.

C. Set up a new AWS Control Tower landing zone within the company's central management account. Organize production and development accounts into corresponding OUs.

D. Incorporate existing accounts into the organization using AWS Organizations. Formulate Service Control Policies (SCPs) to uphold compliance standards.

E. Design a guardrail from the management account to monitor EBS encryption status.

F. Create a guardrail specifically for the production OU to monitor compliance with EBS encryption standards.

The optimal solution comprises steps C, D, and F:

- C. Creating a Control Tower Landing Zone in Management Account: Initiating a Control Tower landing zone in the central management account is a strategic move. It facilitates the structured setup of the AWS environment with the distinction between production and development environments, enabling centralized governance and easier management.
- D. Inviting Existing Accounts to Join AWS Organizations: By incorporating existing accounts into AWS Organizations, the company can efficiently manage policies across all accounts. This step is crucial for centralized governance and compliance enforcement.
- F. Creating a Guardrail for Production OU: Establishing a guardrail specifically for the production OU ensures that all accounts within this OU adhere to the EBS encryption requirement. This guardrail acts as a compliance check, enforcing the encryption policy only in production environments.

The reasons other options are less appropriate:

A. Using CloudFormation StackSets: While StackSets is a powerful tool for deploying resources across multiple accounts, it does not inherently provide the governance or compliance monitoring capabilities that are central to the company's requirements.

B. Control Tower in Developer Account: Setting up the Control Tower landing zone in a developer account could complicate the distinction between development and production environments and might not be optimal for governance purposes.

E. Guardrail from Management Account for EBS Encryption: While creating a guardrail from the management account could be useful, it's more effective to apply it specifically to the production OU, as in option F, to ensure targeted enforcement of policies.

Question 68

A business operates a vital, stateful web application on two Linux Amazon EC2 instances behind an Application Load Balancer (ALB), with its data managed by an Amazon RDS for MySQL database. The DNS for this application is handled by Amazon Route 53. The goal is to enhance the application's resilience without significantly altering its existing architecture, ensuring optimal latency post-failover. The Recovery Point Objective (RPO) and Recovery Time Objective (RTO) targets are as follows: for the application tier, an RPO of 2 minutes and an RTO of 30 minutes; for the database tier, an RPO of 5 minutes and an RTO of 30 minutes.

To meet these objectives, the following solutions are proposed:

A. Utilize AWS Elastic Disaster Recovery for the EC2 instances. Create a cross-region read replica of the RDS DB instance. Establish an ALB in a secondary AWS Region. Deploy an AWS Global Accelerator endpoint, linking it to the ALBs, and update the DNS records to direct to the Global Accelerator endpoint.

B. Implement Amazon Data Lifecycle Manager (Amazon DLM) for snapshotting the EBS volumes of the EC2 instances. Use RDS automated backups with cross-region replication. Set up an ALB in a different AWS Region. Configure an AWS Global Accelerator endpoint, associate it with the ALBs, and update the DNS records to point to the Global Accelerator endpoint.

C. Formulate a backup plan in AWS Backup for both the EC2 instances and the RDS DB instance, with replication to a secondary AWS Region. Create an ALB in this secondary region. Establish an Amazon CloudFront distribution in front of the ALB and modify the DNS records to point to CloudFront.

D. Employ Amazon Data Lifecycle Manager (Amazon DLM) for EBS volume snapshots of the EC2 instances. Create a cross-region read replica for the RDS DB instance. Set up an ALB in an alternate AWS Region. Implement an AWS Global Accelerator endpoint and associate it with the ALBs.

The most appropriate solution is A:

- A. AWS Elastic Disaster Recovery and Cross-Region Read Replica: This option effectively addresses the RPO and RTO requirements for both the application and database tiers. AWS Elastic Disaster Recovery is designed for minimal RPO and RTO, meeting the application tier's needs. The cross-region read replica for RDS ensures database resilience and quick recovery, aligning with the specified RPO and RTO for the database tier. Using Global Accelerator helps in maintaining optimal latency after a failover and simplifies DNS management by pointing to a single, static endpoint.

The reasons why the other options are less appropriate:

B. Amazon DLM for EBS and RDS Automated Backups: While DLM is effective for EBS snapshots and RDS backups ensure data safety, the recovery process might not meet the specified RTOs, especially for the application tier. Additionally, DLM snapshots are typically not as frequent as needed for a 2-minute RPO.

C. AWS Backup and CloudFront: AWS Backup is a comprehensive solution, but it may not meet the tight RPO and RTO requirements for the application. Using CloudFront does not directly contribute to the application's resiliency in terms of recovery objectives.

D. Amazon DLM and Cross-Region Read Replica: Similar to option B, relying on DLM for the application tier might not achieve the required RPO and RTO. While the read replica addresses the database tier's needs, the overall solution may fall short for the application tier's recovery objectives.

Question 69

An AWS solutions architect aims to optimize costs and properly scale Amazon EC2 instances within a single AWS account. The focus is on ensuring that the instances are efficiently utilized based on key performance metrics like CPU, memory, and network usage.

To achieve these goals, the following steps are considered choose two:

A. Acquiring either AWS Business Support or AWS Enterprise Support for the account.

B. Activating AWS Trusted Advisor and examining its “Low Utilization Amazon EC2 Instances” suggestions.

C. Installing the Amazon CloudWatch agent on the EC2 instances and setting up the agent to gather memory utilization metrics.

D. Enabling AWS Compute Optimizer in the account to receive analysis results and recommendations for optimization.

E. Setting up an EC2 Instance Savings Plan targeting specific AWS Regions, instance families, and operating systems.

The most effective steps are C and D:

- C. Install Amazon CloudWatch Agent for Memory Metrics: Implementing the CloudWatch agent on the EC2 instances is crucial for detailed performance monitoring, particularly for metrics like memory usage, which are not natively provided by EC2. This data is vital for making informed decisions about instance sizing and utilization.
- D. Configure AWS Compute Optimizer for Recommendations: AWS Compute Optimizer analyzes the historical utilization metrics of EC2 instances and provides recommendations on appropriate instance sizes. This service can suggest optimal configurations based on actual usage patterns, helping in cost optimization and ensuring that the instances are correctly sized according to CPU, memory, and network requirements.

The reasons why the other options are less suitable:

A. AWS Business or Enterprise Support: While these support plans offer a range of benefits, including guidance and technical support, they don't directly contribute to instance sizing or cost optimization based on specific utilization metrics.

B. Using AWS Trusted Advisor: Trusted Advisor can provide useful insights into underutilized resources. However, its recommendations are generally less comprehensive compared to the detailed analysis and specific suggestions offered by AWS Compute Optimizer, especially for memory and network metrics.

E. EC2 Instance Savings Plan: While Savings Plans can offer cost reductions, they are financial instruments that do not directly assist in performance-based sizing or optimization of EC2 instances. They are more about cost management after deciding on instance types and sizes.

Question 70

A business is utilizing an AWS CodeCommit repository and has a requirement to maintain a backup of this repository's data in a secondary AWS region. This process needs to ensure that the repository data is consistently and securely backed up to another location for redundancy and disaster recovery purposes.

The proposed solutions for creating a backup of the CodeCommit repository in a different region are:

A. Implement AWS Elastic Disaster Recovery to mirror the CodeCommit repository data to the alternate region.

B. Utilize AWS Backup with an hourly schedule to back up the CodeCommit repository and create a cross-region copy of the backup in the second region.

C. Establish an Amazon EventBridge rule that triggers AWS CodeBuild upon code pushes to the repository. Configure CodeBuild to clone the repository, package its content into a .zip file, and then transfer this file to an Amazon S3 bucket located in the second region.

D. Set up an AWS Step Functions workflow to execute on an hourly basis for capturing a snapshot of the CodeCommit repository. Arrange for this workflow to move the snapshot to an S3 bucket in the secondary region.

The most appropriate solution is C:

- C. EventBridge, CodeBuild, and S3 Backup: This method involves setting up an automated process where any code push to the CodeCommit repository triggers an EventBridge rule. This rule invokes AWS CodeBuild, which is configured to clone the repository, package its contents, and then copy the package to an S3 bucket in the desired secondary region. This approach offers a practical and customizable way to create backups of the repository data in another region, ensuring that the backup is up-to-date with the latest changes.

The reasons why the other options are less suitable:

A. AWS Elastic Disaster Recovery: While this service is designed for server and workload replication for disaster recovery, it is not specifically tailored for backing up CodeCommit repositories. It's more aligned with EC2-based applications and environments.

B. AWS Backup for CodeCommit: As of my last update in April 2023, AWS Backup does not support AWS CodeCommit. AWS Backup is generally used for backing up EBS volumes, RDS databases, DynamoDB tables, and other supported AWS services.

D. AWS Step Functions for Repository Snapshots: AWS Step Functions is a workflow service that could theoretically be used for automating backup tasks. However, there is no direct, native functionality in AWS to take "snapshots" of CodeCommit repositories like you would with EBS volumes or databases. This would require a custom implementation, making it a more complex and less efficient solution compared to option C.

Question 71

A company is currently using an AWS CodeCommit repository for its operations and has identified the need to maintain a redundant copy of the repository's data in a different AWS region for robustness and disaster recovery. The aim is to ensure that the repository data is reliably and securely replicated to an alternative location while maintaining the integrity and consistency of the data.

To achieve this, several strategies are under consideration:

A. Employ AWS Elastic Disaster Recovery for mirroring the data from the CodeCommit repository to another region.

B. Schedule regular backups of the CodeCommit repository using AWS Backup, and then replicate these backups to a secondary AWS region.

C. Set up an automated process with Amazon EventBridge, which triggers AWS CodeBuild upon new code commits. CodeBuild can then clone the repository, create a .zip file of its contents, and upload this file to an Amazon S3 bucket in the target region.

D. Implement a regular process using AWS Step Functions to create snapshots of the CodeCommit repository and transfer these snapshots to an S3 bucket in the alternative region.

The most fitting solution is C:

- C. EventBridge, CodeBuild, and S3 for Repository Backup: This option effectively meets the requirement of keeping a backup copy of the CodeCommit repository data in another region. With each code commit, EventBridge triggers a CodeBuild process that clones the repository, packages it into a .zip file, and then copies this package to an S3 bucket in the second region. This approach ensures that the backup is consistent with the latest repository state and leverages AWS services for an automated and reliable backup process.

The reasons why the other options are not as suitable:

A. AWS Elastic Disaster Recovery: This service is primarily designed for disaster recovery of server and workload operations, rather than specifically for CodeCommit repositories. It is more applicable to environments like EC2 instances.

B. AWS Backup for CodeCommit: As of the last update in April 2023, AWS Backup does not support AWS CodeCommit. The service is typically used for other AWS resources like EBS volumes, RDS databases, and DynamoDB tables.

D. AWS Step Functions for Repository Snapshots: Step Functions is a service for orchestrating workflows and could be used to automate backup processes. However, there isn't a native feature in AWS to create "snapshots" of CodeCommit repositories, making this approach more complex and less direct compared to using EventBridge and CodeBuild.

Question 72

A company has recently acquired a new AWS account and is focused on assessing its security, particularly regarding Amazon S3 buckets. The primary concern is to be alerted only when an S3 bucket becomes publicly accessible. The company already has an Amazon Simple Notification Service (Amazon SNS) topic set up, to which the data security team's email is subscribed, for receiving such notifications.

To address this specific requirement, the following solutions are proposed:

A. Implement S3 event notifications for all S3 buckets to detect the 'isPublic' event and configure these notifications to be sent to the established SNS topic.

B. Set up an analyzer in AWS Identity and Access Management Access Analyzer. Then, create an Amazon EventBridge rule tailored to the event type “Access Analyzer Finding” with a filter criterion of “isPublic: true.” Designate the SNS topic as the target for this EventBridge rule.

C. Establish an Amazon EventBridge rule to monitor “Bucket-Level API Call via CloudTrail” events, focusing specifically on the “PutBucketPolicy” action. Route these notifications to the SNS topic.

D. Activate AWS Config and implement the 'cloudtrail-s3-dataevents-enabled' rule. Then, create an Amazon EventBridge rule for “Config Rules Re-evaluation Status” events, filtering for “NON_COMPLIANT” status. Configure this rule to send notifications to the SNS topic.

The most suitable solution is B:

- B. IAM Access Analyzer with EventBridge and SNS: IAM Access Analyzer is a tool specifically designed to identify resources in your AWS environment that are shared with

an external entity. Setting up an analyzer and creating an EventBridge rule to filter for "Access Analyzer Finding" events where `isPublic` is true directly addresses the requirement. When an S3 bucket becomes publicly accessible, Access Analyzer generates a finding, which is then captured by EventBridge and sent to the SNS topic, thus alerting the data security team.

The reasons why the other options are not as suitable:

A. S3 Event Notifications for 'isPublic' Events: Amazon S3 does not natively generate an 'isPublic' event. S3 event notifications are typically used for object-level operations like PUT, POST, DELETE, etc., not for bucket policy changes.

C. EventBridge Rule for 'PutBucketPolicy' via CloudTrail: Monitoring 'PutBucketPolicy' API calls could indicate changes to bucket policies, but it doesn't specifically identify when a bucket becomes publicly accessible. This approach could result in notifications for policy changes that do not relate to public exposure.

D. AWS Config with 'cloudtrail-s3-dataevents-enabled' Rule: This AWS Config rule checks whether data event logging is enabled for S3 buckets, but it does not directly relate to whether a bucket is publicly exposed. The focus of this rule is on CloudTrail logging, not on bucket accessibility.

Question 73

A solutions architect is tasked with analyzing the application and database portfolio of a recently acquired company, which currently operates in an on-premises data center that lacks comprehensive documentation. The goal is to develop a business case for migrating this portfolio to AWS. The applications in question exhibit variable traffic patterns, with some being batch processes executed monthly. The architect's immediate challenge is to gain a detailed understanding of the existing portfolio, including the number of applications and databases, as well as their interdependencies, before proceeding with the migration.

To address this challenge, the following solutions are proposed:

A. Utilize AWS Server Migration Service (AWS SMS) and AWS Database Migration Service (AWS DMS) for migration evaluation, alongside AWS Service Catalog for understanding application and database dependencies.

B. Implement AWS Application Migration Service to deploy agents on the on-premises infrastructure, managing these agents via AWS Migration Hub. Employ AWS Storage Gateway for assessing local storage needs and database dependencies.

C. Engage Migration Evaluator to create an inventory of servers and generate a report for the business case. Utilize AWS Migration Hub for portfolio visualization and AWS Application

Discovery Service for discerning application dependencies.

D. Apply AWS Control Tower in the destination AWS account to create an application portfolio, using AWS SMS for in-depth reporting and business case development. Establish a landing zone for core accounts and resources.

The most fitting solution is C:

- C. Migration Evaluator, Migration Hub, and Application Discovery Service: Migration Evaluator (formerly known as TSO Logic) is designed for analyzing on-premises environments to project costs for moving to AWS, making it suitable for initial assessment and business case creation. AWS Migration Hub provides a centralized view of the migration process, including the portfolio of applications and databases. AWS Application Discovery Service is specifically tailored to identify and understand application dependencies in on-premises environments. This combination offers a comprehensive approach to assess, document, and plan the migration efficiently.

The reasons why the other options are less suitable:

A. AWS SMS, AWS DMS, and Service Catalog: While AWS SMS and DMS are useful for the actual migration process, they are not primarily designed for the initial assessment phase. AWS Service Catalog helps manage catalogs of IT services but does not provide insights into existing on-premises infrastructure.

B. Application Migration Service and Storage Gateway: The Application Migration Service, focusing on migrating applications to AWS, is more about the execution phase of migration rather than the initial assessment. AWS Storage Gateway is used for integrating on-premises environments with cloud storage but does not offer comprehensive discovery or assessment capabilities for applications and databases.

D. AWS Control Tower and SMS: AWS Control Tower is used for setting up and governing a multi-account AWS environment and does not directly contribute to the assessment of on-premises applications and databases. While AWS SMS is essential for server migration, it does not offer pre-migration assessment functionalities.

Question 74

A company is operating an application deployed as a ReplicaSet in an Amazon Elastic Kubernetes Service (Amazon EKS) cluster, with nodes distributed across multiple Availability Zones. The application produces numerous small files, which need to be accessible to all instances of the application running within the cluster. Additionally, the company has a requirement to back up these files and maintain these backups for a duration of one year.

The solutions under consideration for storing and backing up these files are:

A. Set up an Amazon Elastic File System (Amazon EFS) with a mount target in each subnet hosting the EKS cluster nodes. Configure the ReplicaSet to access the EFS file system, directing the application to store its files there. Use AWS Backup to handle the file backups, retaining the data for one year.

B. Create an Amazon Elastic Block Store (Amazon EBS) volume and enable the EBS Multi-Attach feature. Adjust the ReplicaSet to mount this EBS volume, and guide the application to save files on this volume. Utilize AWS Backup to manage backups of this data, keeping them for a year.

C. Establish an Amazon S3 bucket and configure the ReplicaSet to mount this bucket. Instruct the application to write files to the S3 bucket. Enable S3 Versioning to maintain copies of the data and set an S3 Lifecycle policy to automatically delete objects after one year.

D. Direct the ReplicaSet to use the local storage available on each application pod for file storage. Employ a third-party tool to back up the EKS cluster, ensuring data retention for one year.

The most appropriate solution is A:

- A. Amazon EFS with AWS Backup: Amazon EFS provides a scalable file storage solution that can be accessed concurrently by multiple EKS nodes across different Availability Zones, making it ideal for applications that generate files needing to be shared across instances. EFS integrates seamlessly with EKS and supports the required file access pattern. Using AWS Backup for EFS, the company can manage backups effectively and retain them for the specified duration of one year. This solution offers the fastest storage performance for the described use case while meeting backup and retention requirements.

The reasons why the other options are less suitable:

B. Amazon EBS with Multi-Attach: While EBS Multi-Attach allows an EBS volume to be attached to multiple EC2 instances, it is typically limited to instances within the same Availability Zone. This could be a constraint given that the EKS cluster spans multiple Availability Zones. Additionally, EBS is block storage, not typically used for file sharing scenarios as EFS is.

C. Amazon S3 for File Storage: S3 is an object storage service and is not natively designed for file system operations or to be mounted directly by applications for file read/write operations as required by the ReplicaSet in the EKS cluster. While S3 is highly durable and supports versioning and lifecycle policies, it doesn't fit the primary use case as efficiently as EFS.

D. Local Storage on Pods with Third-Party Backup: Relying on local storage of pods is not ideal for sharing files across multiple instances of an application in a distributed environment like EKS. Furthermore, using a third-party tool for backup adds additional complexity and may not provide the same level of integration and convenience as AWS-native solutions.

Question 75

A company operates a customer service center that automatically dispatches interactive surveys via text messages to customers following their calls. The supporting applications for this service are hosted on aging hardware in the company's on-premises data center, leading to system downtime issues. The company is seeking to transition this system to AWS to enhance its reliability, while also aiming to minimize the ongoing operational overhead.

The proposed solutions for migrating this customer service center system to AWS are:

A. Implement Amazon Connect as a replacement for the outdated call center hardware, and utilize Amazon Pinpoint for managing and sending the text message surveys to customers.

B. Adopt Amazon Connect for modernizing the call center infrastructure, and use Amazon Simple Notification Service (Amazon SNS) for the delivery of text message surveys to customers.

C. Migrate the existing call center software to Amazon EC2 instances within an Auto Scaling group, and rely on these EC2 instances to handle the sending of text message surveys to customers.

D. Replace the old call center hardware with Amazon Pinpoint and leverage Pinpoint for both managing the call center interactions and sending text message surveys to customers.

The most suitable solution is A:

- A. Amazon Connect and Amazon Pinpoint: Amazon Connect is a cloud-based contact center service that offers a reliable, scalable solution to replace traditional call center hardware. It can manage customer calls efficiently and integrates well with other AWS services. Amazon Pinpoint is specifically designed for customer engagement through various channels, including text messages. This combination offers a comprehensive, low-overhead solution, where Amazon Connect handles the call center operations and Amazon Pinpoint manages the survey distribution via text messages.

The reasons why the other options are less suitable:

B. Amazon Connect with Amazon SNS: While Amazon SNS is capable of sending text messages, it is more of a broad notification service and less specialized in customer engagement compared to Amazon Pinpoint. Pinpoint offers more features for managing

customer interactions and analyzing engagement data, making it a better fit for the survey aspect of the solution.

C. Migrating to EC2 Instances with Auto Scaling: This approach involves migrating the existing software to EC2, which could require significant effort and maintenance. It might not address the reliability issues effectively and adds complexity in managing the infrastructure.

D. Using Amazon Pinpoint for Call Center and Surveys: Amazon Pinpoint is not a replacement for call center hardware; its primary function is customer engagement and analytics, not managing call center operations. Amazon Connect is the appropriate service for replacing call center hardware.

Question 76

A company is developing a call center using Amazon Connect and is focused on establishing a disaster recovery (DR) strategy across different AWS Regions. The call center is comprised of multiple contact flows, a significant number of users, and several claimed phone numbers. The objective is to devise a DR solution that minimizes the Recovery Time Objective (RTO).

The proposed solutions for this DR strategy are:

A. Implement an AWS Lambda function to periodically check the Amazon Connect instance's availability and alert the operations team if it's down, using an Amazon EventBridge rule for regular checks. In case of an outage, the operations team would manually set up a new Amazon Connect instance in a backup region using the AWS Management Console and apply the contact flows, users, and phone numbers via an AWS CloudFormation template.

B. Set up a secondary Amazon Connect instance in another region with all current users pre-configured. Use a Lambda function to monitor the primary Amazon Connect instance's availability, again with EventBridge for scheduling. Should an outage occur, the Lambda function is designed to automatically execute a CloudFormation template that sets up contact flows and phone numbers in the backup region.

C. Establish a secondary Amazon Connect instance in a different region, pre-loaded with all existing contact flows and claimed phone numbers. Utilize an Amazon Route 53 health check for the primary instance's URL and create an Amazon CloudWatch alarm for health check failures. Have a Lambda function ready to deploy a CloudFormation template that sets up all users, triggered by the CloudWatch alarm.

D. Pre-provision a new Amazon Connect instance in an alternative region with all users and contact flows already in place. Use a Route 53 health check for the primary instance's URL, coupled with a CloudWatch alarm for monitoring health check failures. Prepare a Lambda

function to execute a CloudFormation template for setting up claimed phone numbers, with the alarm triggering the Lambda function.

The most effective solution is D:

- **D. Pre-Provisioned Connect Instance with Users and Flows, Route 53 Health Checks, and Lambda for Phone Numbers**: This option proactively addresses potential downtime by having a secondary Amazon Connect instance ready with all users and contact flows. The health check and alarm system promptly identify any issues with the primary instance. The Lambda function, triggered by the CloudWatch alarm, quickly sets up the claimed phone numbers, ensuring a rapid transition to the DR site. This approach significantly reduces RTO by automating the switchover process and having most of the setup pre-configured.

The reasons why the other options are less suitable:

A. **Manual Provisioning Post-Notification**: This method relies on manual intervention to set up a new Connect instance after being notified of an outage, which could significantly delay the recovery process.

B. **Automated Deployment Post-Outage Detection**: While this option automates the deployment of contact flows and phone numbers upon detecting an outage, the initial step of having users pre-configured but not the contact flows or numbers might lead to delays in making the backup instance fully operational.

C. **Pre-Loaded Flows and Numbers, User Provisioning Post-Outage**: This approach has the contact flows and numbers pre-configured, but the delay in provisioning users after an outage is detected could still result in a slower RTO compared to option D.

Question 77

A business running an application on AWS processes and aggregates data from various sources using proprietary algorithms. Post-ETL processes, this data is stored in Amazon Redshift tables. The company then sells the data, which involves downloading it from Redshift and sending it to numerous clients via FTP. As the client base has expanded, managing these data transactions has become increasingly complex. To streamline this process, the company plans to utilize AWS Data Exchange for distributing data to its customers. A key requirement is to verify customer identities before sharing data, and customers need access to the latest data upon publication.

The potential solutions for sharing data with verified customers through AWS Data Exchange are:

A. Implement AWS Data Exchange for APIs to distribute data, enabling subscription verification. In the company's AWS account, establish an Amazon API Gateway Data API service integrated with Amazon Redshift. Clients would need to subscribe to the data product for access.

B. In the company's AWS account, set up an AWS Data Exchange datashare by linking it to the Redshift cluster, incorporating subscription verification. This requires clients to subscribe to the data product to gain access.

C. Periodically transfer data from the Amazon Redshift tables to an Amazon S3 bucket. Use AWS Data Exchange for S3 to share this data, configuring subscription verification. Clients must subscribe to the data product for access.

D. Publish the Redshift data to Open Data on AWS Data Exchange. Clients are required to subscribe to the data product in AWS Data Exchange. In the company's AWS account, use IAM resource-based policies on the Amazon Redshift tables to restrict access to verified AWS accounts only.

The most fitting solution is B:

- **B. AWS Data Exchange Datashare with Redshift and Subscription Verification**: This option directly connects AWS Data Exchange to the Amazon Redshift cluster, enabling efficient and streamlined data sharing. Subscription verification ensures that only verified customers can access the data. By allowing clients to subscribe to the data product, they can access the most recent data immediately upon publication, meeting the company's requirements for customer verification and data timeliness with minimal operational overhead.

The reasons why the other options are less suitable:

A. **Data Exchange for APIs with API Gateway and Redshift**: While this approach could work, it introduces additional complexity by requiring the setup and management of an API Gateway. This is not as straightforward as directly connecting Data Exchange to Redshift.

C. **Data Exchange for S3 with Periodic Data Transfer**: This method involves extra steps of periodically moving data to S3, which could delay customers' access to the most recent data and increase operational overhead.

D. **Open Data on AWS Data Exchange with IAM Policies**: Publishing data to Open Data on AWS Data Exchange and using IAM policies for access control might not provide the same level of straightforward customer verification and management as a dedicated AWS Data Exchange datashare. This approach could be more cumbersome to manage and may not align with the goal of reducing operational overhead.

Question 78

A solutions architect is tasked with developing a system for event processing that can dynamically adjust its capacity based on incoming event volumes. Additionally, the system needs to segregate events that encounter processing errors into a different queue for further examination.

The proposed solutions for this event processing system are:

A. Utilize Amazon Simple Notification Service (Amazon SNS) for receiving event details. Set up an AWS Lambda function as a subscriber to this SNS topic to handle event processing. In cases of processing failures, configure the Lambda function to redirect failed events to a designated Amazon Simple Queue Service (Amazon SQS) queue.

B. Direct events to an Amazon Simple Queue Service (Amazon SQS) queue. Implement an Amazon EC2 Auto Scaling group that scales based on the 'ApproximateAgeOfOldestMessage' metric of the queue. Design the application to reroute messages that fail processing to a specified dead-letter queue.

C. Store events in an Amazon DynamoDB table and activate a DynamoDB stream for this table. Configure the stream to trigger an AWS Lambda function responsible for processing these events.

D. Route events to an Amazon EventBridge event bus. Develop an application running on Amazon EC2 instances within an Auto Scaling group, placed behind an Application Load Balancer (ALB). Assign the ALB as the target for the event bus. Configure the event bus for event retries and establish a mechanism for writing unprocessable messages to a dead-letter queue.

The most suitable solution is B:

- B. SQS Queue with EC2 Auto Scaling and Dead-Letter Queue: This option effectively meets the requirements of scalability and error handling. By using an SQS queue, the system can buffer incoming events. The EC2 Auto Scaling group can dynamically scale based on the queue's 'ApproximateAgeOfOldestMessage' metric, ensuring adequate processing capacity. For events that fail processing, the application can move them to a dead-letter queue, allowing for separate review and handling. This solution offers a balance of scalability, reliability, and error segregation with relatively low complexity.

The reasons why the other options are less suitable:

A. SNS with Lambda and On-Failure Destination: While this setup uses Lambda for processing and can handle failures by moving events to an SQS queue, it lacks the intrinsic

scalability based on event volume that is provided by an SQS queue with an Auto Scaling group.

C. DynamoDB with Streams and Lambda Function: This approach does not inherently provide the required scalability based on the number of events. DynamoDB streams trigger Lambda functions, but this setup might not efficiently handle fluctuating event volumes.

D. EventBridge with EC2 and ALB: While this configuration includes EC2 Auto Scaling, using EventBridge and an ALB adds complexity and may not be as efficient for queuing and processing events as a direct SQS and EC2 Auto Scaling setup. Additionally, the mechanism for error handling and dead-letter queuing is less straightforward compared to option B.

Question 79

A company operates a data processing engine in the AWS Cloud, designed to calculate a sustainability index using environmental data from its numerous logistics centers across Europe. The logistics centers have millions of devices that transmit data to the processing engine via a RESTful API. This API encounters unpredictable and intense traffic spikes. The primary objective is to ensure that all data sent by these devices is reliably processed without any data loss.

The proposed solutions to manage this data influx and process it effectively are:

A. Deploy an Application Load Balancer (ALB) for the RESTful API and create an Amazon Simple Queue Service (Amazon SQS) queue. Set up an ALB listener and a target group, with the SQS queue as the target. Process the messages in the queue using a container running on Amazon Elastic Container Service (Amazon ECS) with the Fargate launch type.

B. Set up an Amazon API Gateway HTTP API for the RESTful API and create an Amazon SQS queue. Integrate the API Gateway directly with the SQS queue and use an AWS Lambda function to process the messages from the queue.

C. Implement the RESTful API through an Amazon API Gateway REST API, backed by a fleet of Amazon EC2 instances in an Auto Scaling group. Configure an API Gateway Auto Scaling group proxy integration and utilize the EC2 instances for processing incoming data.

D. Establish an Amazon CloudFront distribution for the RESTful API, create a data stream in Amazon Kinesis Data Streams, and designate this data stream as the CloudFront distribution's origin. Utilize an AWS Lambda function to consume and process data from the data stream.

The most effective solution is B:

- B. Amazon API Gateway with SQS and Lambda: This solution leverages the scalability and flexibility of Amazon API Gateway to manage the RESTful API, combined with the reliability of Amazon SQS for queuing the incoming data. The integration of API Gateway with SQS ensures that all data sent by devices is queued without loss, even during traffic bursts. The AWS Lambda function then processes the data from the queue. This setup provides a scalable, serverless architecture capable of handling variable traffic loads without risking data loss.

The reasons why the other options are less suitable:

A. ALB with SQS and ECS Fargate: While this setup includes queuing with SQS, using an ALB as a direct interface to SQS is not a standard practice. Additionally, managing containers with ECS Fargate might introduce more complexity compared to the serverless approach in option B.

C. API Gateway with EC2 Auto Scaling: This approach requires managing and scaling EC2 instances, which increases operational complexity. It lacks the direct queueing mechanism to handle bursts of traffic, potentially risking data loss.

D. CloudFront with Kinesis Data Streams and Lambda: Using CloudFront and Kinesis for this use case is unconventional. CloudFront is typically used for content delivery rather than as an interface for a data ingestion API. While Kinesis can handle high-throughput data, the setup might be more complex and less efficient for this specific scenario compared to the direct API Gateway and SQS integration.

Question 80

A company managing its cloud infrastructure with AWS Organizations has a multi-account structure divided into three Organizational Units (OUs). In each OU, there are over 100 AWS accounts, and each of these accounts contains a single Virtual Private Cloud (VPC). All VPCs within an OU are in the same AWS Region, and their CIDR ranges do not overlap. The company needs a network setup where VPCs within the same OU can communicate with each other, but VPCs in different OUs are isolated from one another. The goal is to achieve this with minimal operational complexity.

The solutions proposed for enabling VPC communication within OUs while maintaining isolation between OUs are:

A. Implement an AWS CloudFormation stack set to establish VPC peering connections between accounts in each OU. Deploy this stack set across each OU.

B. In each OU, set up a dedicated networking account with a single VPC. Use AWS Resource Access Manager (AWS RAM) to share this VPC with all other accounts in the OU. Establish

VPC peering connections between this central networking account and each account in the OU.

C. Deploy a transit gateway in one account within each OU. Utilize AWS Resource Access Manager (AWS RAM) to share the transit gateway across the entire OU. Set up transit gateway VPC attachments for every VPC in the OU.

D. For each OU, create a dedicated networking account that includes one VPC. Form VPN connections between the networking account and the other accounts in the OU. Employ third-party routing software to manage transitive traffic between the VPCs.

The most suitable solution is C:

- C. Transit Gateway with AWS RAM: By deploying a transit gateway in one account per OU and sharing it using AWS RAM, the company can efficiently manage inter-VPC communication within each OU. Transit gateways offer a scalable and centralized way to connect multiple VPCs, allowing for simplified network management and reduced operational complexity. This approach aligns with the requirement for VPCs within the same OU to communicate while remaining isolated from VPCs in other OUs.

The reasons why the other options are less suitable:

A. CloudFormation Stack Set for VPC Peering: While this could technically enable VPC-to-VPC communication, managing peering connections between a large number of VPCs can become operationally complex. It doesn't scale as well as a transit gateway, especially with over 100 accounts in each OU.

B. Central Networking Account with VPC Peering: This method involves sharing a central VPC via AWS RAM and setting up peering with each account's VPC. It would require a significant number of VPC peering connections, leading to increased complexity and potential limitations in scalability.

D. Dedicated Networking Account with VPN and Third-Party Routing: Establishing VPN connections and using third-party routing software introduces additional complexity and potential points of failure. It also requires the management of third-party software, which increases operational overhead.

Question 81

A company is transitioning an application to AWS, intending to leverage fully managed services for the migration. A key requirement is the storage of large, crucial documents associated with the application. The storage solution must meet specific criteria: high durability and availability, encryption of data both at rest and in transit, and the ability for the company to manage and periodically rotate the encryption key.

The potential storage solutions being considered are:

A. Implement AWS Storage Gateway in file gateway mode, utilizing Amazon EBS volume encryption with an AWS Key Management Service (KMS) key to encrypt the storage gateway volumes.

B. Utilize Amazon S3, applying a bucket policy to enforce HTTPS for data in transit and server-side encryption with AWS KMS for data at rest.

C. Opt for Amazon DynamoDB, ensuring SSL connections to DynamoDB, and use an AWS KMS key for encrypting the stored data objects.

D. Set up instances with attached Amazon EBS volumes for data storage, using EBS volume encryption with an AWS KMS key.

The most appropriate solution is B:

- B. Amazon S3 with HTTPS and AWS KMS: Amazon S3 is a highly durable and available storage service, well-suited for storing large documents. By enforcing HTTPS (using a bucket policy), data in transit remains encrypted. S3's server-side encryption with AWS KMS meets the requirement for data encryption at rest, with the added benefit of allowing the company to manage and rotate the encryption key. This solution adheres to all specified requirements while offering a fully managed, scalable, and secure storage option.

The reasons why the other options are less suitable:

A. Storage Gateway in File Gateway Mode: While this could provide durable storage and the ability to use AWS KMS for encryption, it's a more complex solution compared to using Amazon S3 directly. It involves additional overhead in managing the storage gateway and doesn't offer the same simplicity and scalability as S3.

C. Amazon DynamoDB with SSL and AWS KMS: DynamoDB is primarily a NoSQL database service, optimized for storing structured data, not large documents. While it offers encryption capabilities, it's not the ideal choice for the specified use case of storing large files.

D. Instances with EBS Volumes: This approach requires managing EC2 instances and EBS volumes, which adds operational complexity and does not leverage fully managed services to the extent that S3 does. While it offers encryption with AWS KMS, it's not as straightforward or scalable for document storage as S3.

Question 82

A company operates a public API, hosted as tasks on Amazon Elastic Container Service (Amazon ECS) using AWS Fargate, with traffic managed by an Application Load Balancer (ALB). The ECS tasks utilize Service Auto Scaling based on CPU utilization. Although the service has been functioning effectively for several months, the API's performance recently deteriorated significantly, rendering the application unusable. This issue was attributed to a large number of SQL injection attacks, which led to the API service scaling to its maximum limit. A solutions architect is tasked with devising a strategy to block these SQL injection attacks from reaching the ECS API service, ensuring that only legitimate traffic is allowed, and operational efficiency is maximized.

The proposed solutions for enhancing the security of the API are:

A. Implement a new AWS WAF web Access Control List (ACL) to monitor both HTTP and HTTPS requests directed to the ALB that routes traffic to the ECS tasks.

B. Set up a new AWS WAF Bot Control system, including a managed rule group in the Bot Control configuration to monitor traffic and allow only legitimate requests to reach the ALB serving the ECS tasks.

C. Establish a new AWS WAF web ACL with a rule specifically designed to block requests matching patterns typical of SQL database attacks, while permitting all other traffic. Associate this web ACL with the ALB in front of the ECS tasks.

D. Create a new AWS WAF web ACL and an empty IP set within AWS WAF. Add a rule to the web ACL to deny requests from IP addresses listed in the IP set. Develop an AWS Lambda function to analyze the API logs, identify IP addresses associated with SQL injection attacks, and add these IPs to the set. Attach the web ACL to the ALB handling the ECS tasks.

The most appropriate solution is C:

- **C. AWS WAF Web ACL with SQL Database Attack Rule**: This approach directly addresses the issue of SQL injection attacks. By creating a web ACL in AWS WAF and adding a rule to specifically block patterns indicative of SQL injection, the solution effectively filters out malicious traffic while allowing legitimate requests to proceed. Attaching this web ACL to the ALB ensures that the filtering occurs before the traffic reaches the ECS tasks, thus preventing the service from scaling unnecessarily due to malicious activity. This solution is operationally efficient as it relies on predefined rules for common attack patterns, reducing the need for extensive manual configuration or ongoing maintenance.

The reasons why the other options are less suitable:

A. Monitoring HTTP/HTTPS Requests with AWS WAF: While monitoring requests is a critical component of security, this option does not specify how the SQL injection attacks will be identified and blocked. It lacks the focused approach needed to address the specific issue of SQL injection.

B. AWS WAF Bot Control: Bot Control is designed to manage bot traffic, which may not be effective against SQL injection attacks specifically. This solution might not sufficiently address the particular security threat faced by the company.

D. AWS WAF with IP Blocking and Lambda Function: While blocking IPs of known attackers can be part of a security strategy, this reactive approach (identifying and then blocking) may not be fast enough to prevent SQL injection attacks, especially new ones from previously unknown IPs. It also introduces additional operational overhead in maintaining the IP list and requires continuous log analysis, which may not be the most efficient use of resources.

Question 83

An environmental organization is implementing a network of sensors across major cities in a country to monitor air quality. These sensors are integrated with AWS IoT Core for transmitting timeseries data readings, which are then stored in Amazon DynamoDB. For ensuring business continuity, the organization requires a system that supports data ingestion and storage across two different AWS Regions.

The potential solutions for this multi-regional data handling requirement are:

A. Utilize Amazon Route 53 with an alias failover routing policy pointing to AWS IoT Core data endpoints in both regions. Transition the data storage to Amazon Aurora global tables.

B. Establish domain configurations for AWS IoT Core in each region and set up Amazon Route 53 with a latency-based routing policy. Direct the policy to the AWS IoT Core data endpoints in both regions. Migrate the data to Amazon MemoryDB for Redis with cross-region replication enabled.

C. Configure domain settings for AWS IoT Core in each region and create an Amazon Route 53 health check to assess the health of these domain configurations. Implement a failover routing policy in Route 53 using the domain names from AWS IoT Core domain configurations. Convert the existing DynamoDB table into a global table.

D. Set up an Amazon Route 53 latency-based routing policy using AWS IoT Core data endpoints in both regions. Implement DynamoDB streams and establish cross-region data replication.

The most effective solution is C:

- C. AWS IoT Core Domain Configurations with Route 53 Failover and DynamoDB Global Table: This option involves setting up domain configurations for AWS IoT Core in both regions, ensuring data ingestion can occur regardless of the region. The Amazon Route 53 health check monitors the health of these domains, and the failover routing policy ensures that data is sent to an operational region if one becomes unavailable. By upgrading the DynamoDB table to a global table, the solution provides consistent data availability and synchronization across regions. This approach meets the requirement for business continuity with minimal complexity and operational overhead.

The reasons why the other options are less suitable:

A. Route 53 Alias Failover with Aurora Global Tables: While the Route 53 configuration could manage failover, migrating from DynamoDB to Aurora global tables would involve significant changes to the data storage architecture. This is more complex and may not align with the specific needs of timeseries data storage.

B. Latency-Based Routing with MemoryDB for Redis: This solution introduces MemoryDB for Redis, which requires a shift from DynamoDB and adds complexity. Additionally, MemoryDB is more suited for caching rather than long-term timeseries data storage.

D. Route 53 Latency-Based Routing with DynamoDB Streams: Using latency-based routing does not guarantee business continuity in case of a regional outage. Furthermore, DynamoDB streams for cross-region replication add complexity and might not be as efficient or straightforward as using DynamoDB global tables for multi-regional synchronization.

Question 84

In a company using AWS Organizations for managing multiple AWS accounts, the finance team operates a data processing application utilizing AWS Lambda and Amazon DynamoDB. The marketing team, operating in a separate AWS account, needs access to certain data in the DynamoDB table managed by the finance team. However, this table contains confidential information, and the marketing team should only be able to view specific attributes. The company requires a solution to grant the marketing team appropriate and restricted access to this DynamoDB table.

The proposed solutions for providing selective access to the DynamoDB table are:

A. Implement a Service Control Policy (SCP) to permit the marketing team's AWS account access to specific attributes in the DynamoDB table, and associate this SCP with the Organizational Unit (OU) of the finance team.

B. In the finance team's account, create an IAM role with a policy that utilizes IAM conditions for fine-grained access control to specific DynamoDB attributes. Establish trust with the

marketing team's AWS account. Then, in the marketing team's account, create an IAM role with the necessary permissions to assume the IAM role in the finance team's account.

C. Formulate a resource-based IAM policy with conditions targeting specific attributes in the DynamoDB table (fine-grained access control) and attach this policy to the DynamoDB table. In the marketing team's account, create an IAM role authorized to access the DynamoDB table in the finance team's account.

D. Generate an IAM role in the finance team's account for accessing the DynamoDB table, applying an IAM permissions boundary to restrict access to only certain attributes. In the marketing team's account, create an IAM role with permissions to assume the IAM role in the finance team's account.

The most suitable solution is B:

- **B. IAM Role with Fine-Grained Access Control and Cross-Account Trust**: This approach effectively addresses the requirement. By creating an IAM role in the finance team's account with specific conditions to allow access only to certain attributes of the DynamoDB table, the solution ensures that only the necessary data is accessible. Establishing a trust relationship with the marketing team's account allows the marketing team to assume this role, thereby gaining the appropriate level of access. This method leverages AWS's IAM capabilities for fine-grained access control and cross-account access, providing a secure and controlled way to share specific data.

The reasons why the other options are less suitable:

A. **Using SCP for Attribute-Level Access**: SCPs are used to manage permissions at the account level within AWS Organizations and are not suitable for attribute-level access control within DynamoDB or any other specific AWS service.

C. **Resource-Based IAM Policy on DynamoDB Table**: DynamoDB does not support attaching resource-based policies directly to tables for cross-account access. Instead, cross-account access is typically managed through IAM roles.

D. **IAM Role with Permissions Boundary**: While an IAM role with a permissions boundary can restrict access within an account, it's not the most straightforward method for enabling cross-account access with attribute-level restrictions in DynamoDB. The proposed method in option B is more direct and suitable for this requirement.

Question 85

A solutions architect is developing an application that needs to store objects in Amazon S3 buckets and operate simultaneously in two different AWS Regions. A crucial requirement is

that the objects stored in S3 buckets across these regions are kept in sync. The objective is to achieve this synchronization with minimal operational effort.

The potential solutions to ensure synchronized objects across S3 buckets in two regions are (choose three):

A. Implement an S3 Multi-Region Access Point and update the application to use this Multi-Region Access Point for object storage.

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

C. Adjust the application to simultaneously store objects in both S3 buckets.

D. Create S3 Lifecycle rules for each bucket to handle object copying from one bucket to the other.

E. Enable S3 Versioning on both S3 buckets.

F. Configure event notifications for each S3 bucket to trigger an AWS Lambda function that copies objects from one bucket to the other.

The optimal solution consists of steps A, B, and E:

- A. S3 Multi-Region Access Point: Utilizing a Multi-Region Access Point simplifies the application's interaction with S3 by providing a single endpoint for object storage across multiple regions. This access point abstracts the regional complexity and aids in the operational management of the buckets.
- B. Two-Way Cross-Region Replication (CRR): Configuring CRR between the two S3 buckets ensures that objects are automatically replicated in both directions, keeping the buckets in sync. This replication is managed by AWS and happens seamlessly in the background, providing a low-overhead solution for maintaining data consistency across regions.
- E. Enable S3 Versioning: Versioning is a prerequisite for setting up CRR. It keeps track of and safeguards against unintended overwrites or deletions, maintaining the integrity of objects across both buckets.

The reasons why the other options are less suitable:

C. Manual Dual Storage: Modifying the application to store objects in both buckets introduces complexity and potential errors. It's more efficient to let AWS services like CRR handle the synchronization automatically.

D. Lifecycle Rules for Object Copying: Implementing lifecycle rules for copying objects between buckets is a less efficient method for synchronization. This approach can introduce

delays and doesn't offer the same level of integration and automation as CRR.

F. Event Notifications with Lambda Function: While using Lambda to copy objects could work, it introduces additional complexity and potential points of failure. Managing a Lambda function for this purpose is operationally more intensive compared to the automated CRR process.

Question 86

A company operates an IoT platform in an on-premises setup, involving a server that communicates with IoT devices via the MQTT protocol. This platform frequently collects telemetry data (at least every 5 minutes) from the IoT devices and also maintains device metadata in a MongoDB cluster. Additionally, there is an application installed on an on-premises machine that performs periodic data aggregation and transformation tasks, generating reports that are accessible through a web application running on the same machine. These periodic jobs vary in duration (120-600 seconds), while the web application is constantly active. The company is now transitioning this platform to AWS and aims to minimize the operational overhead of managing this infrastructure.

The potential solutions for migrating and optimizing this IoT platform on AWS with minimal operational overhead are (choose three):

A. Utilize AWS Lambda functions to establish connections with IoT devices.

B. Configure IoT devices to send data to AWS IoT Core.

C. Store device metadata in a self-managed MongoDB database hosted on an Amazon EC2 instance.

D. Use Amazon DocumentDB (with MongoDB compatibility) for storing device metadata.

E. Implement AWS Step Functions state machines with AWS Lambda tasks for report generation, storing the reports in Amazon S3, and serving the reports using Amazon CloudFront with an S3 origin.

F. Deploy an Amazon Elastic Kubernetes Service (Amazon EKS) cluster with Amazon EC2 instances to handle report preparation and use an ingress controller in the EKS cluster to serve the reports.

The most suitable combination of steps is B, D, and E:

- B. Publishing to AWS IoT Core: Configuring IoT devices to send data directly to AWS IoT Core simplifies data ingestion and management. IoT Core is a fully managed service that

efficiently handles MQTT communications, reducing the complexity of maintaining MQTT connections.

- D. Storing Metadata in Amazon DocumentDB: Replacing the on-premises MongoDB cluster with Amazon DocumentDB, which offers MongoDB compatibility, significantly reduces operational overhead. DocumentDB is a fully managed service, eliminating the need for managing database servers and infrastructure.
- E. Using AWS Step Functions and Lambda with Amazon S3 and CloudFront: For report generation, utilizing AWS Step Functions and Lambda automates the aggregation and transformation process. Storing reports in S3 and distributing them via CloudFront provides a scalable and efficient solution for report accessibility, minimizing the management required for web hosting and report distribution.

The reasons why the other options are less suitable:

A. AWS Lambda for IoT Connections: Directly using Lambda functions to connect with IoT devices could introduce unnecessary complexity, as IoT Core is specifically designed for this purpose and offers better integration with AWS services.

C. Self-Managed MongoDB on EC2: Managing MongoDB on EC2 instances involves significant overhead in terms of database management, infrastructure scaling, and maintenance, which contradicts the goal of reducing operational effort.

F. Amazon EKS for Report Preparation and Serving: Setting up an EKS cluster with EC2 instances for report generation and distribution introduces higher complexity compared to using serverless solutions like Step Functions, Lambda, and CloudFront. This approach requires more hands-on management of Kubernetes infrastructure.

Question 87

A global manufacturing company is gearing up to shift most of its applications to AWS but faces specific challenges. Some applications need to stay within a particular country or at the company's central on-premises data center due to data regulatory demands or the necessity for extremely low latency (single-digit milliseconds). Additionally, there are concerns about hosting applications at factory sites with limited network infrastructure. The company seeks a unified development environment to enable its developers to build applications once and deploy them either on-premises, in the cloud, or in a hybrid setting, using familiar tools, APIs, and services.

The proposed solutions to create a consistent hybrid environment that addresses these requirements are:

A. Migrate all applications to the nearest AWS Region that meets compliance standards and establish an AWS Direct Connect link between the on-premises data center and AWS. Use a

Direct Connect gateway for the connection.

B. Implement AWS Snowball Edge Storage Optimized devices for applications with specific data regulatory or low latency requirements, keeping these devices on-premises. Use AWS Wavelength at factory sites to host the workloads.

C. Deploy AWS Outposts for applications requiring adherence to data regulations or low latency. For workloads at factory sites with limited network infrastructure, use AWS Snowball Edge Compute Optimized devices.

D. Move applications that must comply with data regulations or have low latency requirements to an AWS Local Zone. Employ AWS Wavelength for hosting workloads at the factory sites.

The most fitting solution is C:

- C. AWS Outposts and Snowball Edge Compute Optimized Devices: AWS Outposts is a fully managed service that extends AWS infrastructure, services, APIs, and tools to virtually any data center, co-location space, or on-premises facility for a truly consistent hybrid experience. It is ideal for meeting specific latency or data residency requirements. For factory sites with limited network infrastructure, AWS Snowball Edge Compute Optimized devices provide a way to run applications in environments with connectivity constraints, offering a consistent set of tools and services as used in AWS. This combination effectively addresses the company's varied requirements across different locations.

The reasons why the other options are less suitable:

A. AWS Direct Connect with the Nearest AWS Region: While Direct Connect provides a dedicated network connection to AWS, it does not address on-premises requirements for specific applications or those at factory sites with limited network infrastructure.

B. Snowball Edge Storage Optimized and AWS Wavelength: Snowball Edge Storage Optimized is focused more on data transfer and storage, not on providing a full range of compute capabilities. AWS Wavelength is designed to bring AWS services to the edge of the 5G network, which may not be applicable or available in factory site scenarios.

D. AWS Local Zone and AWS Wavelength: AWS Local Zones extend AWS services to specific geographic locations, but they might not be available in all the regions where the company operates. Wavelength focuses on 5G networks and may not suit factory sites with limited networking infrastructure.

Question 88

A company is updating its online ordering application and facing an increase in cyber attacks. The revised application will be hosted on Amazon Elastic Container Service (ECS) and will use Amazon DynamoDB for data storage. Access to the application will be provided through a public Application Load Balancer (ALB). The company's primary objectives are to prevent these attacks and maintain business continuity with minimal disruption during an attack, all while ensuring cost-effectiveness.

The proposed solutions for enhancing security and ensuring continuity are:

A. Set up an Amazon CloudFront distribution with the ALB as the origin. Implement a custom header with a unique value for the CloudFront domain and configure the ALB to allow traffic only when this header and value are present.

B. Deploy the application across two AWS Regions and use Amazon Route 53 to distribute traffic evenly between these regions.

C. Enable auto-scaling for the Amazon ECS tasks and create a DynamoDB Accelerator (DAX) cluster.

D. Implement Amazon ElastiCache to reduce the load on DynamoDB.

E. Establish an AWS WAF web Access Control List (ACL) with a suitable rule group and associate this web ACL with the Amazon CloudFront distribution.

The most suitable combination of steps is **A and E**:

- **A. CloudFront Distribution with Custom Header Verification on ALB**: This method provides an additional layer of security by utilizing CloudFront as the first point of contact for incoming requests, effectively acting as a buffer against direct attacks on the ALB. The custom header mechanism adds an extra check, making it more difficult for bad actors to directly target the ALB, thereby enhancing security.
- **E. AWS WAF Web ACL Associated with CloudFront**: Deploying AWS WAF with a well-configured web ACL in conjunction with the CloudFront distribution offers a robust defense against common web exploits and attacks. This setup allows for the filtering of malicious traffic before it reaches the application infrastructure, thereby ensuring business continuity even during ongoing attacks.

The reasons why the other options are less suitable:

B. **Route 53 with Multi-Region Deployment**: While multi-region deployment provides high availability, it might not be the most cost-effective solution for protecting against attacks. It doesn't directly address the security concerns and could significantly increase operational costs.

C. ECS Auto-Scaling and DAX Cluster: Auto-scaling ECS tasks can help manage load, and DAX can improve DynamoDB performance, but neither directly addresses the issue of preventing attacks or ensuring business continuity during attacks.

D. ElastiCache to Reduce DynamoDB Load: While ElastiCache can enhance database performance, it doesn't contribute to preventing cyber attacks or improving security, which is the primary concern in this scenario.

Question 89

A company operates a web application on AWS, which delivers static content via an Amazon S3 bucket through an Amazon CloudFront distribution. Dynamic content is served using an Application Load Balancer (ALB) that routes requests to a group of Amazon EC2 instances within Auto Scaling groups. The application utilizes a domain configured in Amazon Route 53. During peak hours, some users have encountered intermittent access issues, specifically receiving HTTP 503 Service Unavailable errors from the ALB. The company seeks a solution to display a custom error message page for these errors, aiming for immediate response with minimal operational effort.

The proposed solutions to present a custom error page during HTTP 503 errors are:

A. Implement a Route 53 failover routing policy, with a health check to monitor the ALB's status and failover to an S3 bucket endpoint hosting the error page.

B. Set up a secondary CloudFront distribution and an S3 static website for the custom error page. Configure Route 53 with a failover routing policy between the two CloudFront distributions in an active-passive setup.

C. Establish a CloudFront origin group with the ALB as the primary origin and an S3 bucket configured for a static website as the secondary origin. Enable origin failover in the CloudFront distribution and update the S3 static website to include the custom error page.

D. Develop a CloudFront function to check each HTTP response code from the ALB. Create an S3 static website in a bucket to host the error page. Modify the function to fetch the error page from the S3 bucket and present it to users in case of a 503 error.

The best solution is C:

- C. CloudFront Origin Group with Origin Failover: By creating an origin group in CloudFront with the ALB as the primary origin and an S3 bucket (hosting the static error page) as the secondary origin, the company can utilize CloudFront's origin failover feature. In case the ALB returns a 503 error, CloudFront automatically serves the custom error page from the S3 bucket. This setup provides a seamless and automatic failover

with minimal overhead, ensuring that users see the custom error page immediately during ALB outages or overloads.

The reasons why the other options are less suitable:

A. **Route 53 Failover Routing Policy**: While Route 53 failover can redirect traffic to a secondary endpoint, it's based on health checks and not on specific HTTP response codes. This might not be as immediate or responsive to individual 503 errors compared to CloudFront's origin failover.

B. **Secondary CloudFront Distribution and Route 53 Failover**: Configuring a secondary CloudFront distribution and using Route 53 failover introduces additional complexity. This approach is less streamlined compared to having a single CloudFront distribution with origin failover.

D. **CloudFront Function with S3 Static Website**: Implementing a CloudFront function to evaluate HTTP response codes and fetch the error page from an S3 bucket adds operational complexity. This method requires custom programming and maintenance, whereas origin failover provides a built-in, low-maintenance solution.

Question 90

A company is preparing to transition an application to AWS. This application is currently run in a Docker container and utilizes NFS version 4 for file sharing. The solutions architect is tasked with designing a containerized solution on AWS that is both secure and scalable, and importantly, does not require the management of underlying infrastructure.

The proposed solutions for deploying this containerized application on AWS are:

A. Use Amazon Elastic Container Service (ECS) with the Fargate launch type for deploying the application containers. Employ Amazon Elastic File System (EFS) as the shared storage solution. In the ECS task definition, include details such as the EFS file system ID, the container mount point, and an IAM role for EFS authorization.

B. Deploy the application containers on Amazon ECS with the Fargate launch type. Utilize Amazon FSx for Lustre as the shared storage service. In the ECS task definition, specify the FSx for Lustre file system ID, the container mount point, and an IAM role for FSx for Lustre authorization.

C. Opt for Amazon ECS with the Amazon EC2 launch type and enable auto-scaling. Use Amazon EFS for shared storage. Mount the EFS file system on the ECS container instances and associate an EFS authorization IAM role with the EC2 instance profile.

D. Implement the application containers on Amazon ECS with the Amazon EC2 launch type, ensuring auto-scaling is active. Use Amazon Elastic Block Store (EBS) volumes with the Multi-Attach feature for shared storage. Attach these EBS volumes to the ECS container instances and assign an EBS authorization IAM role to the EC2 instance profile.

The best solution is A:

- A. ECS with Fargate and EFS: This option aligns with the requirements for a scalable and secure containerized solution without the need for managing underlying infrastructure. Amazon ECS with the Fargate launch type abstracts away the server and cluster management tasks. Amazon EFS is compatible with NFS version 4, making it a suitable choice for the shared file system. By integrating EFS with ECS and referencing the necessary details in the task definition, the solution ensures seamless operation and security with minimal overhead.

The reasons why the other options are less suitable:

B. ECS with Fargate and FSx for Lustre: Amazon FSx for Lustre is optimized for high-performance computing scenarios and might not be necessary or as cost-effective for standard file-sharing needs compared to EFS. It also does not directly support NFS version 4.

C. ECS with EC2, Auto-scaling, and EFS: Although this option uses EFS, choosing ECS with the EC2 launch type introduces the overhead of managing EC2 instances and scaling, which contradicts the requirement to avoid infrastructure management.

D. ECS with EC2, Auto-scaling, and EBS Multi-Attach: EBS with Multi-Attach does not provide a native NFS interface and adds complexity in managing volumes and instances. Like option C, this approach also involves managing EC2 instances.

Question 91

A company has an application in the AWS Cloud, with the main functionality running on Amazon EC2 instances grouped in an Auto Scaling group. An Application Load Balancer (ALB) manages the distribution of traffic among these EC2 instances. The domain api.example.com, set up in Amazon Route 53, directs traffic to the ALB. The development team has made significant updates to the application's business logic. The company has a policy to expose only 10% of its customers to new updates during a testing phase, ensuring that a customer consistently interacts with the same version of the business logic throughout this period. The company needs a deployment strategy for these updates that adheres to these criteria.

The proposed methods for deploying the updates are:

A. Set up a second ALB and deploy the updated logic to EC2 instances within a new Auto Scaling group. Configure the new ALB to handle traffic to these instances. Modify the Route

53 record to implement weighted routing, directing traffic to both ALBs.

B. Introduce a second target group for the existing ALB and deploy the new business logic on EC2 instances in this group. Adjust the ALB listener rule to incorporate weighted target groups and enable target group stickiness on the ALB.

C. Create a new launch configuration for the existing Auto Scaling group, specifying it to use the AutoScalingRollingUpdate policy with a MaxBatchSize option set to 10. Substitute the current launch configuration in the Auto Scaling group with this new one and roll out the updates.

D. Establish a new Auto Scaling group linked to the current ALB. Deploy the updated business logic to EC2 instances in this new group. Modify the ALB to utilize the least outstanding requests (LOR) routing algorithm and enable session stickiness on the ALB.

The most suitable solution is B:

- B. Second Target Group with Weighted Target Groups and ALB Stickiness: This approach allows for the controlled rollout of the new business logic. By creating a second target group for the ALB and deploying the updated application to EC2 instances within this group, the company can manage the distribution of traffic between the old and new versions. The use of weighted target groups in the ALB listener rule facilitates directing only a portion of the traffic (10%) to the new logic. Enabling stickiness on the ALB ensures that customers maintain consistent interaction with the same version of the business logic during the testing window.

The reasons why the other options are less suitable:

A. Second ALB with Weighted Routing in Route 53: While this could technically work, it introduces unnecessary complexity by managing a second ALB. It is more efficient to handle the traffic distribution through a single ALB with multiple target groups.

C. New Launch Configuration with AutoScalingRollingUpdate: This method doesn't provide the precise control over traffic distribution needed to ensure that only 10% of customers are exposed to the new logic. It's more suited for a complete, gradual rollout rather than a controlled testing phase.

D. New Auto Scaling Group with LOR Routing and ALB Stickiness: Creating an entirely new Auto Scaling group for the updated application increases complexity and doesn't offer as straightforward a mechanism for controlling traffic distribution to the specific 10% as the weighted target groups method. Additionally, changing the ALB's routing algorithm to LOR doesn't directly contribute to achieving the desired testing strategy.

Question 92

A large education company has adopted Amazon WorkSpaces to facilitate access to internal applications across multiple universities. User profiles are stored on an Amazon FSx for Windows File Server file system, which is configured with a DNS alias and linked to a self-managed Active Directory. With an increasing number of users utilizing WorkSpaces, the company is experiencing a rise in login times, which have become unacceptably long. Investigations point to a decline in the file system's performance, attributed to its configuration on HDD storage with a throughput of 16 MBps. To address this issue, the company needs to enhance the file system's performance, and a solutions architect is tasked with this improvement, ideally to be executed during a set maintenance window with minimal administrative effort.

The solutions proposed to enhance the file system's performance are:

A. Create a backup of the current file system using AWS Backup. Restore this backup to a new FSx for Windows File Server file system configured with SSD storage and a throughput capacity of 32 MBps. After completing the backup and restore process, update the DNS alias and delete the original file system.

B. Temporarily disconnect users from the file system, then increase the throughput capacity to 32 MBps and switch the storage type to SSD through the Amazon FSx console. Once updated, reconnect the users to the file system.

C. Set up an AWS DataSync agent on a new Amazon EC2 instance. Create a DataSync task, specifying the existing file system as the source and a new FSx for Windows File Server file system with SSD storage and 32 MBps throughput as the target. Execute the task as scheduled. Upon completion, update the DNS alias accordingly and remove the original file system.

D. Enable shadow copies on the current file system using Windows PowerShell. Schedule shadow copy jobs to create backups of the file system. Then, create a new FSx for Windows File Server file system with SSD storage and 32 MBps throughput. After completing the copy job, update the DNS alias and eliminate the original file system.

The most suitable solution is A:

- A. Backup and Restore with AWS Backup: This approach leverages AWS Backup to create a point-in-time backup of the existing file system and then restore it to a new FSx file system with improved performance parameters (SSD storage and 32 MBps throughput). This method ensures a smooth transition with minimal administrative effort, as AWS handles the backup and restore processes. Once the new file system is in place and verified, the DNS alias can be updated, and the original file system can be decommissioned.

The reasons why the other options are less suitable:

B. Directly Updating FSx Configuration: Amazon FSx does not currently allow direct in-place updates to switch storage types from HDD to SSD or to modify the throughput capacity of an existing file system.

C. Using AWS DataSync for Migration: While DataSync can transfer data between file systems, it introduces additional complexity and overhead in setting up and managing the DataSync agent and task. This approach requires more steps and resources compared to the streamlined backup and restore process.

D. Shadow Copies and Manual File System Creation: Using shadow copies and manually creating a new FSx file system with enhanced parameters is more complex and time-consuming. This method also requires additional steps to manage the shadow copy process and subsequent data transfer, making it less efficient.

Question 93

A company is operating an application on AWS, which interacts with a single Amazon S3 bucket for reading and writing objects. The company now needs to adapt this application for deployment in two different AWS Regions. The objective is to implement this change while incurring the least operational overhead.

The proposed solutions for enabling application deployment across two AWS Regions with S3 storage are:

A. Implement an Amazon CloudFront distribution using the existing S3 bucket as the origin. Modify the application for deployment in a second AWS Region to utilize the CloudFront distribution. Use AWS Global Accelerator for accessing the data in the S3 bucket.

B. Create a new S3 bucket in another AWS Region. Establish bidirectional S3 Cross-Region Replication (CRR) between the original and the new S3 bucket. Set up an S3 Multi-Region Access Point covering both buckets. Modify and deploy the application in both AWS Regions.

C. Set up a new S3 bucket in a different AWS Region. Deploy the application in this second Region and configure it to use the new S3 bucket. Configure S3 Cross-Region Replication (CRR) from the original S3 bucket to the new one.

D. Configure an S3 gateway endpoint with the existing S3 bucket as the origin. Deploy the application in a second Region and adjust it to use the new S3 gateway endpoint. Implement S3 Intelligent-Tiering on the S3 bucket.

The most appropriate solution is B:

- B. New S3 Bucket, Bidirectional CRR, and S3 Multi-Region Access Point: This option creates a seamless and efficient environment for the application to operate in two AWS Regions. By creating a new S3 bucket in the second Region and setting up bidirectional Cross-Region Replication, data is synchronized between both buckets, ensuring consistency. The S3 Multi-Region Access Point simplifies the process of connecting to these buckets from different Regions, providing a single endpoint for the application to interact with S3, regardless of the Region. This approach minimizes operational overhead by automating data synchronization and simplifying application configuration.

The reasons why the other options are less suitable:

A. CloudFront Distribution and AWS Global Accelerator: While CloudFront can effectively distribute static content, this setup doesn't address the need for active data synchronization between two AWS Regions. Global Accelerator improves access but doesn't solve the multi-region deployment challenge for the S3 data.

C. New S3 Bucket with Unidirectional CRR: This solution only provides one-way replication and does not ensure that changes made in the second Region are reflected back to the original bucket, potentially leading to data inconsistencies.

D. S3 Gateway Endpoint and Intelligent-Tiering: Using an S3 gateway endpoint and Intelligent-Tiering doesn't address the core requirement of deploying the application in two Regions with synchronized S3 storage. This setup also doesn't facilitate easy replication or access to S3 data across Regions.

Question 94

An online gaming company plans to transfer its gaming platform to AWS. The gaming application, requiring high-performance computing (HPC) capabilities, includes a frequently updating leaderboard. The application, built on Node.js, currently runs on a compute-optimized Ubuntu instance and utilizes an on-premises Redis instance to track game state. The company seeks a migration approach that enhances the application's performance on AWS.

The proposed solutions for migrating and optimizing the gaming platform on AWS are:

A. Set up an Auto Scaling group of m5.large Amazon EC2 Spot Instances, managed by an Application Load Balancer (ALB). Use Amazon ElastiCache for Redis to handle the leaderboard.

B. Configure an Auto Scaling group of c5.large Amazon EC2 Spot Instances, also overseen by an ALB. Maintain the leaderboard using Amazon OpenSearch Service.

C. Establish an Auto Scaling group with c5.large Amazon EC2 On-Demand Instances behind an ALB. Utilize Amazon ElastiCache for Redis for the leaderboard.

D. Implement an Auto Scaling group of m5.large Amazon EC2 On-Demand Instances, coordinated through an ALB. Keep track of the leaderboard using an Amazon DynamoDB table.

The most appropriate solution is C:

- C. Auto Scaling Group of c5.large EC2 On-Demand Instances with ElastiCache for Redis: This configuration aligns well with the company's requirements. The c5.large instances are compute-optimized, suitable for HPC processing needs of the gaming application. Using On-Demand Instances ensures consistent availability and performance, which is critical for gaming applications. Amazon ElastiCache for Redis is a natural choice for managing the leaderboard, providing a managed, in-memory data store with high throughput and low latency, similar to the company's existing Redis setup.

The reasons why the other options are less suitable:

A. m5.large EC2 Spot Instances with ElastiCache for Redis: While ElastiCache for Redis is a good fit for the leaderboard, the use of m5.large instances, which are general-purpose and not compute-optimized, may not provide the required HPC performance. Additionally, Spot Instances, while cost-effective, can be interrupted, potentially affecting the game's availability.

B. c5.large EC2 Spot Instances with OpenSearch Service: c5.large instances are compute-optimized, but using Spot Instances might risk interruptions. OpenSearch Service, designed for search and analytics, is not the best fit for a frequently updated leaderboard compared to an in-memory data store like Redis.

D. m5.large EC2 On-Demand Instances with DynamoDB: m5.large instances are not as suitable for HPC tasks compared to c5.large instances. While DynamoDB offers high performance, it may not match the speed and suitability of ElastiCache for Redis for leaderboard scenarios, where in-memory processing is advantageous.

Question 95

A solutions architect is tasked with creating an application that allows employees to submit their timesheets through mobile devices. The submissions are expected to peak weekly, primarily on Fridays, and the data needs to be stored in a way that facilitates monthly report generation by payroll administrators. The solution must ensure high availability, scalability to meet the influx of data submissions and reporting demands, and should aim to reduce operational complexity.

The proposed solutions for this application are (choose two):

A. Utilize Amazon EC2 On-Demand Instances with load balancing across multiple Availability Zones, and implement scheduled Amazon EC2 Auto Scaling to increase capacity in anticipation of the higher submission volume on Fridays.

B. Implement the application in containers on Amazon Elastic Container Service (ECS) with load balancing across multiple Availability Zones. Use scheduled Service Auto Scaling to enhance capacity before the anticipated high submission volume on Fridays.

C. Host the application front end in an Amazon S3 bucket, served by Amazon CloudFront, and deploy the backend using Amazon API Gateway with AWS Lambda proxy integration.

D. Store the timesheet data in Amazon Redshift and employ Amazon QuickSight for generating reports, using Redshift as the data source.

E. Save the timesheet data in Amazon S3 and utilize Amazon Athena along with Amazon QuickSight to create the reports, with Amazon S3 serving as the data source.

The most appropriate combination of steps is **C and E**:

- **C. S3 and CloudFront for Front End, API Gateway and Lambda for Backend**: This serverless approach offers high availability and scalability. Hosting the front end on S3 and serving it via CloudFront ensures fast, global access with minimal management overhead. Using API Gateway and Lambda for the backend allows for handling timesheet submissions efficiently, scaling automatically to meet demand, particularly during peak times.
- **E. Store Data in S3, Use Athena and QuickSight for Reporting**: Storing timesheet data in Amazon S3 offers a highly scalable and durable solution. Using Amazon Athena for querying this data directly from S3 and QuickSight for reporting provides a flexible, serverless, and scalable way to generate the needed monthly reports with minimal infrastructure management.

The reasons why the other options are less suitable:

A. & B. **EC2 Instances or ECS with Scheduled Scaling**: Both these options involve more operational overhead compared to a serverless architecture. Managing EC2 instances or ECS, even with Auto Scaling, requires more effort in terms of setup, maintenance, and scaling, compared to the fully managed services used in options C and E.

D. **Redshift for Data Storage and QuickSight for Reporting**: While Amazon Redshift is a powerful data warehousing solution, it may introduce unnecessary complexity and cost for this use case, especially considering the serverless and straightforward alternatives provided by S3, Athena, and QuickSight.

Question 96

A company has sensitive data stored in an Amazon S3 bucket and needs to ensure robust logging and monitoring measures. Specifically, the company requires that all activities involving objects in the S3 bucket be logged, with these logs preserved for a duration of five years. Additionally, the security team needs to be alerted via email whenever there is an attempt to delete data from this S3 bucket. The company is seeking a cost-effective solution to fulfill these requirements.

The potential solutions for meeting these requirements are (choose three):

A. Enable AWS CloudTrail to log S3 data events, which provides detailed records of actions taken on S3 objects, including object deletion attempts.

B. Activate S3 server access logging for the S3 bucket, which logs requests made to the S3 bucket but may not provide the detailed data event logging that CloudTrail offers.

C. Set up Amazon S3 to send object deletion event notifications to Amazon Simple Email Service (Amazon SES), which is not a direct native functionality provided by S3.

D. Configure Amazon S3 to trigger object deletion events to an Amazon EventBridge event bus, which then publishes these events to an Amazon Simple Notification Service (Amazon SNS) topic. Subscribers to this SNS topic, like the security team's email, can receive notifications.

E. Arrange for Amazon S3 to forward logs to Amazon Timestream with data storage tiering, which is an overcomplicated solution for simple logging needs and not cost-effective for this specific requirement.

F. Create a new S3 bucket specifically for storing logs, and apply an S3 Lifecycle policy to manage log retention effectively and cost-efficiently, ensuring logs are kept for the required 5-year period.

The most appropriate combination of steps is A, D, F:

- A. AWS CloudTrail for S3 Data Events: CloudTrail is well-suited for logging detailed data events in S3, including object access and deletion attempts. It's a more comprehensive solution for logging compared to S3 server access logging.
- D. S3 to EventBridge to SNS for Deletion Alerts: This setup allows S3 to send object deletion events to EventBridge, which then can relay these events to an SNS topic. The security team can subscribe to this topic to receive immediate email notifications of deletion attempts, fulfilling the real-time alert requirement.

- F. Dedicated S3 Bucket with Lifecycle Policy for Logs: Storing CloudTrail logs in a dedicated S3 bucket with a lifecycle policy ensures that the logs are retained for the specified duration of 5 years in a cost-effective manner. This policy can automate the transition of logs to more cost-effective storage classes over time and eventually expire them after the retention period.

The reasons why the other options are less suitable:

B. Configure S3 server access logging for the S3 bucket: S3 server access logging provides basic access information such as requester, bucket name, request time, etc. However, it doesn't provide the detailed level of data event logging (such as read and write operations on objects) that CloudTrail offers. While it can capture some access information, it isn't as comprehensive for monitoring and auditing purposes, especially for sensitive data. It's also less effective for capturing specific events like object deletions in detail.

C. Configure Amazon S3 to send object deletion events to Amazon SES: Amazon S3 does not natively support direct integration with Amazon SES to send notifications. S3 can publish events to SNS, SQS, and Lambda, but not directly to SES. Therefore, this approach is not feasible for the requirement to send email notifications upon object deletion. The typical method to achieve email notifications would be through SNS (as described in option D).

E. Configure Amazon S3 to send the logs to Amazon Timestream with data storage tiering: Amazon Timestream is a time-series database service optimized for IoT and operational applications, not for storing access logs. Using Timestream for storing S3 access logs is not only an over-engineered solution but also likely to be more expensive and complex compared to using a dedicated S3 bucket with a lifecycle policy. Timestream is designed for different use cases and does not align well with the simple logging and long-term storage requirements stated.

Question 97

A company is creating a hybrid setup involving servers in both its on-premises data center and in the AWS Cloud. The company has already deployed Amazon EC2 instances across three Virtual Private Clouds (VPCs), each located in a different AWS Region. Additionally, there's an existing AWS Direct Connect connection linking the data center to the AWS Region geographically closest to it. The company's objective is to enable its on-premises servers to access the EC2 instances in all three VPCs and to utilize AWS public services. The solution should be as cost-effective as possible.

The potential solutions for achieving this connectivity are (choose two):

A. Implement a Direct Connect gateway in the closest AWS Region, attach it to the existing Direct Connect connection, and use this gateway to link the VPCs in the other two Regions.

B. Establish additional Direct Connect connections from the on-premises data center to the other two AWS Regions.

C. Set up a private Virtual Interface (VIF) and create an AWS Site-to-Site VPN connection over this private VIF to the VPCs in the other two Regions.

D. Create a public VIF and establish an AWS Site-to-Site VPN connection over this public VIF to access the VPCs in the other two Regions.

E. Implement VPC peering to connect the VPCs across different Regions, and use a private VIF with the existing Direct Connect connection to link to the peered VPCs.

The most suitable combination of steps is A and D:

- A. Direct Connect Gateway in Closest Region: By deploying a Direct Connect gateway in the nearest region and attaching it to the existing Direct Connect connection, the company can efficiently connect its on-premises data center to the VPCs in all three AWS Regions. This approach leverages the existing Direct Connect connection, thus minimizing costs and complexity.
- D. Public VIF for AWS Public Services: Setting up a public VIF on the existing Direct Connect connection allows the on-premises servers to access AWS public services directly. This approach is cost-effective as it utilizes the existing Direct Connect link and does not require additional physical connections.

The reasons why the other options are less suitable:

B. Additional Direct Connect Connections: Creating more Direct Connect connections to the other two AWS Regions would significantly increase costs and complexity. This is not necessary when the existing connection can be leveraged through a Direct Connect gateway.

C. Private VIF with Site-to-Site VPN: While a private VIF can be used for VPN connections, in this scenario, it would add complexity and may not be necessary since the Direct Connect gateway (option A) can provide the required connectivity to the VPCs in different Regions.

E. VPC Peering and Private VIF: VPC peering across Regions would involve complex routing and configuration. Moreover, using a private VIF for this purpose doesn't add value when the Direct Connect gateway can provide simpler and more efficient inter-region connectivity.

Question 98

A company managing numerous AWS accounts under AWS Organizations aims to establish baseline protection against the Open Web Application Security Project (OWASP) top 10 web application vulnerabilities. They plan to use AWS WAF across all existing and future Amazon

CloudFront distributions within their organization. To implement this protection effectively, a solutions architect is considering a combination of AWS services and features.

The proposed steps to provide this baseline protection are (choose three):

A. Activate AWS Config across all AWS accounts in the organization.

B. Turn on Amazon GuardDuty in every account within the organization.

C. Enable all available features within AWS Organizations.

D. Utilize AWS Firewall Manager to centrally deploy AWS WAF rules across all accounts for CloudFront distributions.

E. Employ AWS Shield Advanced to apply AWS WAF rules across all accounts for CloudFront distributions.

F. Use AWS Security Hub to implement AWS WAF rules in all accounts for all CloudFront distributions.

The most suitable combination of steps is A, C, D:

- A. Activate AWS Config: AWS Config helps in assessing, auditing, and evaluating the configurations of AWS resources, including AWS WAF rules. It can be instrumental in ensuring that WAF rules are correctly set up and maintained across all accounts.
- C. Enable All Features in AWS Organizations: Enabling all features in AWS Organizations allows for advanced management and governance capabilities across all accounts. This includes service control policies (SCPs) which can enforce certain compliance and security standards like the implementation of AWS WAF rules.
- D. Use AWS Firewall Manager: Firewall Manager simplifies the management of AWS WAF rules across multiple AWS accounts and resources, including CloudFront distributions. It ensures that WAF rules are consistently applied, making it an ideal tool for deploying and maintaining the OWASP top 10 protections across the organization's CloudFront distributions.

The reasons why the other options are less suitable:

B. Enable Amazon GuardDuty: While GuardDuty is a useful security service for threat detection, it does not directly contribute to the management or deployment of AWS WAF rules for protecting against OWASP vulnerabilities.

E. Use AWS Shield Advanced: AWS Shield Advanced provides additional protection against DDoS attacks but does not specifically manage AWS WAF rules or provide broad protection against OWASP vulnerabilities.

F. Use AWS Security Hub: Security Hub is a tool for aggregating and prioritizing security findings from various AWS services. However, it is not a tool for deploying or managing AWS WAF rules across multiple accounts or CloudFront distributions.

Therefore, A, C, and D collectively offer a comprehensive approach to deploy and manage AWS WAF rules across an organization, ensuring baseline protection against OWASP top 10 web application vulnerabilities.

Question 99

A solutions architect at a company has set up a SAML 2.0 federated identity solution with the company's on-premises identity provider (IdP) for authenticating user access to their AWS environment. While the architect can access AWS successfully through the federated identity web portal, test users are unable to do so. To troubleshoot this issue, the architect needs to verify certain aspects of the identity federation configuration to ensure it's properly set up.

The potential areas to check for proper configuration of identity federation are:

A. The permissions policy of the IAM user allows the use of SAML federation for that user.

B. The IAM roles intended for federated users or groups have a trust policy in place that designates the SAML provider as the principal.

C. The test users are members of the AWSFederatedUsers group in the company's IdP.

D. The federated identity web portal is correctly invoking the AWS STS AssumeRoleWithSAML API call with the ARN of the SAML provider, the ARN of the IAM role, and the SAML assertion from the IdP.

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

F. The company's IdP correctly defines SAML assertions that map users or groups to IAM roles with the appropriate permissions in AWS.

The most appropriate items to check are B, D, and F:

- B. IAM Roles Trust Policy with SAML Provider as Principal: It's crucial to ensure that IAM roles for federated access are properly configured with a trust policy that recognizes the SAML provider as a principal. This setup allows users authenticated via the IdP to assume these roles.
- D. Correct AWS STS AssumeRoleWithSAML API Call: The web portal must correctly call the AssumeRoleWithSAML API, providing the necessary SAML provider ARN, IAM role

ARN, and SAML assertion. This process is key to granting federated users the appropriate access to AWS resources.

- F. Appropriate SAML Assertions Defined in IdP: The IdP must correctly define SAML assertions to map users or groups to the corresponding IAM roles in AWS. These mappings are critical for ensuring that users receive the correct permissions upon accessing AWS.

The reasons why the other options are less suitable:

A. IAM User's Permissions for SAML Federation: SAML federation works at the role level, not the user level. Therefore, the permissions of an individual IAM user are not relevant to SAML federation access issues.

C. Membership in AWSFederatedUsers Group in IdP: While group memberships in the IdP can dictate access, the existence of a specific 'AWSFederatedUsers' group is not a standard requirement for SAML federation with AWS.

E. IdP's DNS Hostname Accessibility from AWS VPCs: The accessibility of the IdP's DNS hostname from AWS VPCs is typically not a factor in federated identity authentication, as the authentication process is handled externally from the AWS network.

Question 100

A solutions architect is tasked with enhancing an application hosted on AWS, which utilizes an Amazon Aurora MySQL database. The database is currently facing issues with overloaded connections, primarily due to the application's operations that involve inserting records. The application currently stores its database credentials in a text-based configuration file. The architect's goal is to devise a solution that not only handles the increased connection load but also secures the database credentials and enables automatic rotation of these credentials periodically.

The proposed solutions for addressing these requirements are:

A. Introduce an Amazon RDS Proxy in front of the Aurora MySQL DB instance and use AWS Secrets Manager to manage and store the database connection credentials.

B. Implement an Amazon RDS Proxy in front of the Aurora MySQL DB instance and keep the connection credentials in AWS Systems Manager Parameter Store.

C. Establish an Aurora Replica and utilize AWS Secrets Manager for managing and storing the database connection credentials.

D. Create an Aurora Replica and store the database connection credentials in AWS Systems Manager Parameter Store.

The most suitable solution is A:

- A. RDS Proxy with Secrets Manager: Using Amazon RDS Proxy can effectively manage the overloaded connections by pooling and sharing database connections, thereby improving the application's ability to handle a high number of connections. Storing the credentials in AWS Secrets Manager enhances security and provides the capability for automatic credential rotation. This combination addresses both the connection load issue and the requirement for secure and manageable credentials.

The reasons why the other options are less suitable:

B. RDS Proxy with Parameter Store: While an RDS Proxy would help with the connection issues, using AWS Systems Manager Parameter Store for credentials does not offer the same level of security and automated rotation capabilities as Secrets Manager.

C. Aurora Replica with Secrets Manager: Creating an Aurora Replica would enhance read capacity but wouldn't address the overloaded connections caused by write operations (inserts). While Secrets Manager secures and rotates credentials, this option doesn't resolve the primary issue of connection overloads.

D. Aurora Replica with Parameter Store: Similar to option C, an Aurora Replica wouldn't alleviate the overloaded write operation connections. Additionally, using Parameter Store doesn't provide the same level of credential management and automatic rotation as Secrets Manager.